



**UNIVERSIDAD DE GUAYAQUIL
FACULTAD DE INGENIERÍA INDUSTRIAL
CARRERA DE INGENIERÍA EN TELEINFORMÁTICA**

**TRABAJO DE TITULACIÓN
PREVIO A LA OBTENCIÓN DEL TÍTULO DE
INGENIERO EN TELEINFORMÁTICA**

**ÁREA
TECNOLOGÍAS APLICADAS**

**TEMA
“SISTEMA DE RECONOCIMIENTO FACIAL, MEDICIÓN
DE TEMPERATURA Y USO DE MASCARILLAS,
UTILIZANDO RASPBERRY PI Y MACHINE LEARNING.”**

**AUTOR
MOLINA MOREIRA LUIS JONATHAN**

**DIRECTORA DEL TRABAJO
ING. ELECT. GALLEGOS ZURITA DIANA ERCILIA, MSC.**

GUAYAQUIL, ABRIL 2022



ANEXO XI.- FICHA DE REGISTRO DE TRABAJO DE TITULACIÓN



FACULTAD DE INGENIERÍA INDUSTRIAL CARRERA INGENIERÍA EN TELEINFORMÁTICA

REPOSITORIO NACIONAL EN CIENCIA Y TECNOLOGÍA			
FICHA DE REGISTRO DE TRABAJO DE TITULACIÓN			
TÍTULO Y SUBTÍTULO:	SISTEMA DE RECONOCIMIENTO FACIAL, MEDICIÓN DE TEMPERATURA Y USO DE MASCARILLAS, UTILIZANDO RASPBERRY PI Y MACHINE LEARNING		
AUTOR(ES) (apellidos/nombres):	MOLINA MOREIRA LUIS JONATHAN		
REVISOR(ES)/TUTOR(ES) (apellidos/nombres):	ING. ELECT. GALLEGOS ZURITA DIANA ERCILIA, MG / ING. PARRA LÓPEZ RODOLFO ANTONIO, MG.		
INSTITUCIÓN:	UNIVERSIDAD DE GUAYAQUIL		
UNIDAD/FACULTAD:	FACULTAD DE INGENIERÍA INDUSTRIAL		
MAESTRÍA/ESPECIALIDAD:			
GRADO OBTENIDO:	INGENIERO EN TELEINFORMÁTICA		
FECHA DE PUBLICACIÓN:	21 DE ABRIL DEL 2022	No. DE PÁGINAS:	78
ÁREAS TEMÁTICAS:	TECNOLOGÍAS APLICADAS		
PALABRAS CLAVES/ KEYWORDS:	Prototipo, Reconocimiento facial, Machine Learning, Raspberry PI, Covid-19 / Prototype, Facial recognition, Machine Learning, Raspberry PI, Covid-19.		
RESUMEN/ABSTRACT (150-200 palabras):			
<p>El presente trabajo de titulación está orientado al diseño de un prototipo de control del correcto uso de la mascarilla por medio de reconocimiento facial y a su vez la medición de la temperatura corporal. En la elaboración del prototipo fueron utilizados diversos sensores en conjunto a una cámara web y una tarjeta de adquisición y procesamiento de datos como el caso de la Raspberry Pi que utilizan software libre en su programación y utilizando la herramienta tecnología Machine Learning de tal manera la cámara pueda obtener la imagen que llevara a cabo el reconocimiento facial verificando el correcto uso de la mascarilla. Es importante en esta nueva realidad, debido a que todavía estamos enfrentando una crisis de salud global por la pandemia seguir tomando</p>			

medidas de bioseguridad, el prototipo a desarrollar ayudara a disminuir la propagación del virus del COVID-19 por factor del mal uso de mascarilla o temperatura elevada.

The present titling work is oriented to the design of a prototype of control of the correct use of the mask by means of facial recognition and also, the measurement of body temperature. In the elaboration of the prototype, various sensors were used in conjunction with a webcam and a data acquisition and processing card as in the case of the Raspberry Pi that uses free software in its programming and using the Machine Learning technology tool so that the camera can obtain the image with which will be carried out the facial recognition, verifying the correct use of the mask. It is important in this new reality, because we are still facing a global health crisis due to the pandemic to continue taking biosecurity measures, the prototype to be developed will help to reduce the spread of the COVID-19 virus due to the misuse of a mask or elevated temperature.

ADJUNTO PDF:	SI X	NO
CONTACTO CON AUTOR/ES:	Teléfono: 0998024054	E-mail: ljonathan.molinam@ug.edu.ec
CONTACTO CON LA INSTITUCIÓN:	Nombre: Ing. Ramón Maquilón Nicola	
	Teléfono: 593-2658128	
	E-mail: direccionTi@ug.edu.ec	



**ANEXO XII.- DECLARACIÓN DE AUTORÍA Y DE AUTORIZACIÓN DE
LICENCIA GRATUITA
INTRANSFERIBLE Y NO EXCLUSIVA PARA EL USO
NO COMERCIAL DE LA OBRA CON FINES NO ACADÉMICOS**



**FACULTAD DE INGENIERÍA INDUSTRIAL
CARRERA INGENIERÍA EN TELEINFORMÁTICA**

**LICENCIA GRATUITA INTRANSFERIBLE Y NO COMERCIAL DE LA OBRA CON
FINES NO ACADÉMICOS**

Yo, **MOLINA MOREIRA LUIS JONATHAN**, con C.C. No. **0950843029**, certifico que los contenidos desarrollados en este trabajo de titulación, cuyo título es “**SISTEMA DE RECONOCIMIENTO FACIAL, MEDICIÓN DE TEMPERATURA Y USO DE MASCARILLAS, UTILIZANDO RASPBERRY PI Y MACHINE LEARNING**” son de mi absoluta propiedad y responsabilidad, en conformidad al Artículo 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN*, autorizo la utilización de una licencia gratuita intransferible, para el uso no comercial de la presente obra a favor de la Universidad de Guayaquil.

A handwritten signature in blue ink that reads "Luis Jonathan".

MOLINA MOREIRA LUIS JONATHAN
C.C. No. 0950843029



ANEXO VII.- CERTIFICADO PORCENTAJE DE SIMILITUD

FACULTAD DE INGENIERÍA INDUSTRIAL CARRERA INGENIERÍA EN TELEINFORMÁTICA



Habiendo sido nombrada **ING. GALLEGOS ZURITA DIANA ERCILIA**, tutor del trabajo de titulación certifico que el presente trabajo de titulación ha sido elaborado por **MOLINA MOREIRA LUIS JONATHAN**, C.C.: 0950843029, con mi respectiva supervisión como requerimiento parcial para la obtención del título de INGENIERO EN TELEINFORMÁTICA.

Se informa que el trabajo de titulación: **“SISTEMA DE RECONOCIMIENTO FACIAL, MEDICIÓN DE TEMPERATURA Y USO DE MASCARILLAS, UTILIZANDO RASPBERRY PI Y MACHINE LEARNING”**, ha sido orientado durante todo el periodo de ejecución en el programa Antiplagio (URKUND) quedando el 3% de coincidencia.



Document Information

Analyzed document	Molina Moreira Luis Jonathan.docx (D130712968)
Submitted	2022-03-17T22:43:00.0000000
Submitted by	
Submitter email	ljonathan.molinam@ug.edu.ec
Similarity	3%
Analysis address	diana.gallegosz.ug@analysis.arkund.com

Sources included in the report

SA	DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE SEGURIDAD VEHICULAR MEDIANTE RECONOCIMIENTO FACIAL A TRAVÉS DE VISIÓN ARTIFICIAL.pdf Document DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE SEGURIDAD VEHICULAR MEDIANTE RECONOCIMIENTO FACIAL A TRAVÉS DE VISIÓN ARTIFICIAL.pdf (D25440757)	88	1
SA	UNIVERSIDAD DE GUAYAQUIL / PLUAS LINDAO DORA.docx Document PLUAS LINDAO DORA.docx (D64895053) Submitted by: jairo.veintimillaa@ug.edu.ec Receiver: jairo.veintimillaa.ug@analysis.arkund.com	88	8
SA	FINAL tesis final lector alexis.pdf Document FINAL tesis final lector alexis.pdf (D29823490)	88	1
SA	UNIVERSIDAD DE GUAYAQUIL / PLUAS LINDAO DORA - URKUND.docx Document PLUAS LINDAO DORA - URKUND.docx (D65019568) Submitted by: jairo.veintimillaa@ug.edu.ec Receiver: jairo.veintimillaa.ug@analysis.arkund.com	88	1

<https://secure.arkund.com/view/124858680-816628-431235>



Firmado electrónicamente por:
**DIANA ERCILIA
GALLEGOS
ZURITA**

Ing. Gallegos Zurita Diana, Mg.
TUTOR DE TRABAJO DE TITULACIÓN
C.C. 1204926313

FECHA: 17 de marzo del 2022



ANEXO VI. - CERTIFICADO DEL DOCENTE-TUTOR DEL TRABAJO DE TITULACIÓN

**FACULTAD DE INGENIERÍA INDUSTRIAL
CARRERA INGENIERÍA EN
TELEINFORMÁTICA**



Guayaquil, 17 de marzo del 2022

Sr (a).

Ing. Annabelle Lizaraburu Mora, Mg.

Director (a) de Carrera Ingeniería en Teleinformática / Telemática

FACULTAD DE INGENIERÍA INDUSTRIAL

UNIVERSIDAD DE GUAYAQUIL

Ciudad. -

De mi consideración:

Envío a Ud. el Informe correspondiente a la tutoría realizada al Trabajo de Titulación **“SISTEMA DE RECONOCIMIENTO FACIAL, MEDICIÓN DE TEMPERATURA Y USO DE MASCARILLAS, UTILIZANDO RASPBERRY PI Y MACHINE LEARNING”**, del estudiante **MOLINA MOREIRA LUIS JONATHAN**, indicando que ha cumplido con todos los parámetros establecidos en la normativa vigente:

- El trabajo es el resultado de una investigación.
- El estudiante demuestra conocimiento profesional integral.
- El trabajo presenta una propuesta en el área de conocimiento.
- El nivel de argumentación es coherente con el campo de conocimiento.

Adicionalmente, se adjunta el certificado de porcentaje de similitud y la valoración del trabajo de titulación con la respectiva calificación.

Dando por concluida esta tutoría de trabajo de titulación, **CERTIFICO**, para los fines pertinentes, que el (los) estudiante (s) está (n) apto (s) para continuar con el proceso de revisión final.

Atentamente,



Firmado electrónicamente por:
**DIANA ERCILIA
GALLEGOS
ZURITA**

Ing. Gallegos Zurita Diana, Mg.
DOCENTE TUTOR
C.C. 1204926313



ANEXO VIII.- INFORME DEL DOCENTE REVISOR
FACULTAD DE INGENIERÍA INDUSTRIAL
CARRERA INGENIERÍA EN
TELEINFORMÁTICA



Guayaquil, 4 de abril de 2022.

Sr (a).

Ing. Annabelle Lizarzaburu Mora, MG

Director (a) de Carrera Ingeniería en Teleinformática / Telemática

FACULTAD DE INGENIERÍA INDUSTRIAL DE LA UNIVERSIDAD DE GUAYAQUIL

Ciudad. -

De mis consideraciones:

Envío a Ud. el informe correspondiente a la REVISIÓN FINAL del Trabajo de Titulación **“SISTEMA DE RECONOCIMIENTO FACIAL, MEDICIÓN DE TEMPERATURA Y USO DE MASCARILLAS, UTILIZANDO RASPBERRY PI Y MACHINE LEARNING”** del estudiante Molina Moreira Luis Jonathan. Las gestiones realizadas me permiten indicar que el trabajo fue revisado considerando todos los parámetros establecidos en las normativas vigentes, en el cumplimiento de los siguientes aspectos:

Cumplimiento de requisitos de forma:

- El título tiene un máximo de 17 palabras.
- La memoria escrita se ajusta a la estructura establecida.
- El documento se ajusta a las normas de escritura científica seleccionadas por la Facultad.
- La investigación es pertinente con la línea y sub-líneas de investigación de la carrera.
- Los soportes teóricos son de máximo 5 años.
- La propuesta presentada es pertinente.

Cumplimiento con el Reglamento de Régimen Académico:

- El trabajo es el resultado de una investigación.
- El estudiante demuestra conocimiento profesional integral.
- El trabajo presenta una propuesta en el área de conocimiento.
- El nivel de argumentación es coherente con el campo de conocimiento.

Adicionalmente, se indica que fue revisado, el certificado de porcentaje de similitud, la valoración del tutor, así como de las páginas preliminares solicitadas, lo cual indica que el trabajo de investigación cumple con los requisitos exigidos.

Una vez concluida esta revisión, considero que el estudiante está apto para continuar el proceso de titulación. Particular que comunicamos a usted para los fines pertinentes.

Atentamente,



Firmado electrónicamente por:

RODOLFO
ANTONIO PARRA
LÓPEZ

Ing. Parra López Rodolfo, Mg.

Docente Revisor

C.C.: 0909770448

FECHA: 4 de abril de 2022

Índice general

N.º	Descripción	Pág.
	Introducción	1

Capítulo I

El Problema

N.º	Descripción	Pág.
1.1	Planteamiento del problema	2
1.2	Delimitación del problema	3
1.3	Formulación del problema	3
1.4	Justificación e importancia	3
1.5	Objetivos	4
1.5.1	Objetivo general	4
1.5.2	Objetivos específicos	4
1.6	Hipótesis prospectiva	4
1.6.1	Conceptualización y operacionalización de las variables	5
1.6.2	Indicadores	6
1.7	Preguntas de investigación	6
1.8	Alcance del proyecto	7

Capítulo II

Marco Teórico

		8
2.1	Antecedentes del estudio	8
2.2	Fundamentación teórica	10
2.2.1	Control de acceso	10
2.2.1.1	Biometría	10

N.º	Descripción	Pág.
2.2.2	Reconocimiento facial	12
2.2.2.1	Método de trabajo	12
2.2.2.2	Algoritmos	13
2.2.3	Control de temperatura corporal	17
2.2.4	Sistema de control de acceso RIFD	17
2.2.5	Adecuaciones y controles en un local comercial	18
2.3	Definiciones conceptuales	19
2.3.1	Prototipo	19
2.3.2	Python	19
2.3.3	Raspberry Pi	19
2.3.4	Sensores	20
2.3.5	Sensor de temperatura sin contacto Módulo MLX90614GY-906	20
2.3.6	Buzzer	21
2.3.7	COVID-19	21
2.3.8	Coronavirus	21
2.3.9	Reconocimiento facial	21
2.3.10	Dataset	22
2.3.11	Procesamiento de imágenes	22
2.3.12	Redes neuronales	22
2.3.13	Machine Learning	22
2.4	Fundamentación legal	22
2.4.1	Constitución de la República del Ecuador	23

N.º	Descripción	Pág.
Capítulo III		
La Propuesta		
3.2	Metodología del proyecto	24
3.2.1	Fase de inicio	25
3.2.2	Fase de planificación	25
3.2.3	Fase de ejecución	25
3.2.4	Fase de seguimiento y control	25
3.2.5	Fase de cierre	25
3.3	Descripción	26
3.3.1	Factibilidad técnica	26
3.3.2	Factibilidad legal	29
3.3.3	Factibilidad económica	29
3.3.4	Factibilidad operacional	30
3.4	Esquema general del proyecto	31
3.5	Diseño y construcción	32
3.5.1	Diseño del prototipo	32
3.5.2	Desarrollo del prototipo	32
3.5.2.1	Instalación de OpenCV Y dependencias	32
3.5.2.2	Entrenamiento del sistema de reconocimiento de mascarilla	37
3.5.3	Conexiones del sistema de reconocimiento de mascarilla	40
3.5.4	Configuraciones de pines gpio	40
3.6	Pruebas de funcionamiento	42
3.7	Conclusiones	43

N.º	Descripción	Pág.
3.8	Recomendaciones	44
	Bibliografía	61

Índice de figuras

N.º	Descripción	Pág.
1	Ejemplos de rasgos del cuerpo humano	10
2	Fases de un sistema biométrico	11
3	Arquitectura unidireccional con tres capas	15
4	Ejemplo de una red neuronal para reconocimiento facial	15
5	Esquema general de un sistema de reconocimiento de patrones	16
6	Sistema RIDF	18
7	Imagen de una Raspberry Pi	20
8	Sensor de temperatura IR	19
9	Buzzer	19
10	Etapas de la metodología	22
11	Esquema de los pines GPIO	24
12	Sensor de temperatura infrarrojo	25
13	Diagrama de flujo sobre funcionamiento del prototipo.	29
14	Componentes del prototipo	30
15	Actualización de Raspberry Pi	31
16	Instalación de paquetes de desarrollo	31
17	Descarga del código fuente de OpenCV	32
18	Copelación de OpenCV para Python 3	33
19	Verificación de la compilación de OpenCV	33
20	Verificación de la versión de OpenCV instalada	34
21	Elaboración de dataset	35
22	Archivo de entrenamiento	35

23	Gráfico de precisión de los datos en el entrenamiento	36
24	IDE de Python de última fase	36
N.º	Descripción	Pág.
24	Reconocimiento de la mascarilla y l toma de temperatura	37
25	Conexiones de los sensores	38
26	Puerto GPIO	38
27	Conexión del sensor de temperatura	39
28	Conexión del Buzzer	39
29	Conexiones de los sensores del sistema	39
30	Prueba # 1	40
31	Prueba # 2	41
32	Prueba # 3	41
33	Página de inicio del sitio web www.raspberrypi.org	44
34	Finalización de la instalación	45
35	Raspberry Pi imager instalado correctamente	46
36	Selección de tipo y arquitectura del SO	46
37	Selección de la versión del SO	47
38	Selección del almacenamiento del SO	47
39	Escritura del SO en la SD	47
40	Iniciado la instalación de Raspberry Pi OS	48
41	Pestaña de inicio Raspberry Pi Os	48

Índice de tablas

N.º	Descripción	Pág.
1	Terminología aplicada a la identificación del problema	3
2	Conceptualización y operacionalización de las variables	5
3	Indicadores	7
4	Hardware y Software utilizados durante el desarrollo del proyecto	28
5	Costo de inversión en Software	29
6	Costo de inversión en Hardware	30
7	Costo total del proyecto	30
8	Pruebas técnicas del sistema	42

Índice de anexos

Nº	Descripción	Pág.
1	Instalación del sistema operativo	46
2	Instalación de Python, Open CV	51
3	Líneas de código del entrenador del detector de mascarilla	52
4	Líneas de código del detector de mascarilla	56



**ANEXO XIII.- RESUMEN DEL TRABAJO DE
TITULACIÓN (ESPAÑOL)
FACULTAD DE INGENIERÍA INDUSTRIAL
CARRERA INGENIERÍA EN TELEINFORMÁTICA**



**“SISTEMA DE RECONOCIMIENTO FACIAL, MEDICIÓN DE TEMPERATURA
Y USO DE MASCARILLAS, UTILIZANDO RASPBERRY PI Y MACHINE
LEARNING”**

Autor: Molina Moreira Luis Jonathan

Tutor: ING. ELECT. Gallegos Zurita Diana Ercilia, Mg.

Resumen

El presente trabajo de titulación está orientado al diseño de un prototipo de control del correcto uso de la mascarilla por medio de reconocimiento facial y a su vez la medición de la temperatura corporal. En la elaboración del prototipo fueron utilizados diversos sensores en conjunto a una cámara web y una tarjeta de adquisición y procesamiento de datos como el caso de la Raspberry Pi que utilizan software libre en su programación y utilizando la herramienta tecnología Machine Learning de tal manera la cámara pueda obtener la imagen que llevara a cabo el reconocimiento facial verificando el correcto uso de la mascarilla. Es importante en esta nueva realidad, debido a que todavía estamos enfrentando una crisis de salud global por la pandemia seguir tomando medidas de bioseguridad, el prototipo a desarrollar ayudara a disminuir la propagación del virus del COVID-19 por factor del mal uso de mascarilla o temperatura elevada.

Palabras Claves: Prototipo, Reconocimiento facial, Machine Learning, Raspberry PI, Covid-19.



**ANEXO XIV.- RESUMEN DEL TRABAJO DE
TITULACIÓN (INGLÉS)
FACULTAD DE INGENIERÍA INDUSTRIAL
CARRERA INGENIERÍA EN TELEINFORMÁTICA**



**FACIAL RECOGNITION SYSTEM, TEMPERATURE MEASUREMENT AND
USE OF MASKS, USING RASPBERRY PI AND MACHINE LEARNING**

Author: Molina Moreira Luis Jonathan

Advisor: EE. Gallegos Zurita Diana Ercilia, Mg

Abstract

The present titling work is oriented to the design of a prototype of control of the correct use of the mask by means of facial recognition and also, the measurement of body temperature. In the elaboration of the prototype, various sensors were used in conjunction with a webcam and a data acquisition and processing card as in the case of the Raspberry Pi that uses free software in its programming and using the Machine Learning technology tool so that the camera can obtain the image with which will be carried out the facial recognition, verifying the correct use of the mask. It is important in this new reality, because we are still facing a global health crisis due to the pandemic to continue taking biosecurity measures, the prototype to be developed will help to reduce the spread of the COVID-19 virus due to the misuse of a mask or elevated temperature.

Keywords: Prototype, Facial recognition, Machine Learning, Raspberry PI, Covid-19.

Introducción

En las circunstancias en que actualmente el mundo todavía está pasando por los acontecimientos de la pandemia del COVID-19 que afecta a todos los países, estos han implementado varias restricciones y medidas de bioseguridad para reducir la propagación y contagio de este virus. Solo en Ecuador desde su aparición hasta la presente fecha, según datos de la Organización Mundial de la Salud (OMS) se han registrado 849.386 personas infectadas con este virus y 35.347 personas fallecidas. (ORGANIZACION MUNDIAL DE LA SALUD, 2022)

Tomando en cuenta recomendaciones de la (OMS) el uso de la mascarilla junto con un distanciamiento prudente entre personas de al menos 1.5 metros es una combinación imprescindible para evitar riesgos de contagios de COVID-19, por lo cual muchos países alrededor del mundo han impuesto el uso obligatorio de mascarilla como medida de protección y prevención al portador ante una exposición directa de otras personas que pueden estar contagiada. (ORGANIZACION MUNDIAL DE LA SAULD, 2020)

Para la seguridad de los ciudadanos, es importante asegurarse de que las personas tengan buenas condiciones de salud y para eso, se al ingreso de un lugar concurrido usan medidas de bioseguridad, uno de esos procedimientos es el de medir la temperatura y el uso correcto de la mascarilla al ingreso de un establecimiento, en las mayorías de establecimientos esta labor es realizada por una persona encargada.

El planteamiento de la propuesta tecnológica permite realizar el proceso de detección del reconocimiento facial para el uso correcto de la mascarilla y la medición de la temperatura corporal, con el uso de un sistema biométrico y el uso de la herramienta tecnológica Machine Learning, la cual permite al dispositivo aprenda las variaciones faciales, también se hará uso de un sensor de temperatura infrarrojo el cual identificara la temperatura corporal del usuario y que esta no pase de los 37. 5°

Capítulo I

El Problema

1.1. Planteamiento del problema

Hoy, el mundo entero está siendo afectado por el virus COVID-19. Este presenta síntomas tales como fiebre, tos, dificultad para respirar, dolor de cabeza y dolores musculares. Este virus se propaga de poco a poco a la mayoría de los países, provocando muchas pérdidas de vidas humanas y económicas, desencadenando una crisis sanitaria y económica mundial, donde fue necesario mantener el aislamiento en los hogares para evitar la infección y muerte de más personas. (OMS, 2020)

A medida que avanza la pandemia, se han realizado muchos esfuerzos por parte del personal médico para responder a la emergencia. Sin embargo, los esfuerzos del personal médico por sí solos no pueden responder adecuadamente a la crisis debido a los numerosos errores en el ámbito sanitario.

Además de las deficiencias en el ámbito sanitario en Ecuador, uno de los problemas que enfrenta el país es el acceso mal controlado a personas contagiadas o potencialmente contagiadas a las instalaciones comerciales. Esto ha provocado un rebrote, especialmente en las ciudades más afectadas, como Guayaquil. Según el Ministerio de Salud Pública hasta el mes de mayo del 2021 se ha reportado a nivel nacional 433.870 casos confirmados y 15.361 personas fallecidas. Guayas es la segunda provincia en el Ecuador más afectada con 58,400 casos confirmados y 2679 fallecidos. (COE NACIONAL, 2021)

En muchas instalaciones comerciales de Guayaquil cuentan con un proceso manual de toma de temperatura y el personal encargado verifica también que ingrese con la mascarilla puesta al momento de acceder al local, lo cual no se respeta el distanciamiento mínimo que recomienda el COE (Comité de Operaciones de Emergencia) nacional el cual es de 2 metros entre el empleado y clientes responsables del procedimiento para evitar la posibilidad de contagio.

1.2. Delimitación del problema

Tabla 1.

Esta tabla describe el enfoque de la terminología analítica aplicada a la identificación del problema en donde se desarrolla la problemática.

Delimitador	Descripción
Campo	Tecnología
Área	Bioseguridad
Aspecto	Control de ingreso de uso de mascarilla y medición de temperatura

Elaboración: el autor

1.3. Formulación del problema

¿Cómo evitar la propagación del virus del COVID-19 al ingreso a un establecimiento?

Dada que en la actualidad todavía esta presente el posible contagio de COVID-19, es imprescindible tomar medidas de bioseguridad en lugares concurridos.

1.4. Justificación e importancia

Debido al riesgo que se expone el personal en los diferentes establecimientos al momento de recibir a los clientes con la toma de temperatura y observar que tenga colocada correctamente la mascarilla, se desarrolla un sistema que permita visualizar la temperatura corporal de una persona y a su vez mostrara si esta está usando correctamente la mascarilla, para que el usuario pueda ingresar al establecimiento. Esto con el fin de prevenir posibles contagios de COVID-19.

Hay varios estudios que corroboran la importancia del uso de la mascarilla como medida de prevención y control de la pandemia del COVID-19 ya que el virus se propaga de la boca o la nariz de una persona infectada en partículas pequeñas de líquidos cuando la persona tose, estornuda, canta, respira fuertemente o habla. por ello una de las principales normas de ingreso a cualquier empresa, establecimiento comercial, escuelas entre otras es el uso de la mascarilla junto con ello la desinfección de manos y corroborar de que el sujeto no tenga fiebre ya que este es un síntoma común de este virus. (ORGANIZACION MUNDIAL DE LA SALUD, 2020)

La importancia de este proyecto es que ayudará a que los clientes de los locales comerciales puedan asegurarse de su nivel de temperatura sea el adecuado antes del ingreso de un establecimiento y que estos usen su mascarilla, además los usuarios no tendrán

que hacer contacto con el termómetro en la entrada de cualquier establecimiento y así evitar contagios por dicha herramienta.

1.5. Objetivos

1.5.1 Objetivo general.

Monitorear el uso correcto de la mascarilla y toma de temperatura a través del desarrollo de un sistema de reconocimiento facial de acceso, utilizando Raspberry pi y Machine Learning, para evitar la propagación del virus de SARS-CoV-2.

1.5.2 Objetivos específicos.

- Revisar los fundamentos teóricos sobre los sistemas de reconocimiento facial con Machine Learning,
- Desarrollar un sistema de reconocimiento facial con las herramientas de OpenCV, Keras y Tensor Flow que pueda identificar el correcto uso de mascarillas y realice la toma de temperatura corporal, haciendo uso de Machine Learning.
- Implementar los sistemas de control y potencia utilizando una Raspberry Pi 4 junto con un sensor infrarrojo, y realizar pruebas de funcionalidad.

1.6. Hipótesis prospectiva

Si se implementa un prototipo de reconocimiento facial que reconozca el uso de la mascarilla y tome la temperatura corporal, por ende, reducirá la propagación del virus del SARS-CoV-2.

Variable independiente: Desarrollo de un prototipo de sistema de reconocimiento de uso de mascarilla y toma de temperatura

Causa: desarrollar un prototipo de reconocimiento facial y toma de temperatura para reducir el índice de contagios del SARS-CoV-2.

Variable dependiente: Proceso de control de protocolos y/o medidas de bioseguridad.

Consecuencia: Mejora en el control del proceso de control de cumplimiento de las medidas de bioseguridad a través del uso de una Raspberry Pi.

1.6.1 Conceptualización y operacionalización de las variables

Tabla 2.

Conceptualización y operacionalización de las variables

Variables	“Desarrollo de un prototipo de reconocimiento de uso de mascarilla y toma de temperatura”	“Proceso de control de protocolos y/o medidas de bioseguridad.”
Definiciones conceptuales	El desarrollo de un prototipo es una etapa previa a la obtención de un producto final, es decir un modelo de trabajo o una versión no pulida de un producto, en este caso un sistema de reconocimiento facial del uso de la mascarilla.	Al referirnos a un proceso de control, hablamos de normas en este caso hablamos de las medidas de bioseguridad, como el uso de la mascarilla y el distanciamiento que debe tener una persona de otra.
Definiciones operacionales	El desarrollo de este prototipo engloba aspectos específicos que puntualizan su correcto funcionamiento. Así como la distancia correcta de operación a la que debe estar una persona para que el prototipo pueda monitorear si está usando la mascarilla y si este presenta fiebre.	Para el desarrollo del prototipo se utiliza el microcontrolador Raspberry Pi, con sensor infrarrojo IR_MLX90614, encargado de medir la temperatura corporal del usuario, brindando señales de alerta mediante una alarma si el usuario presenta fiebre.

Elaboración: el autor

1.6.2 Indicadores

Tabla 3.

Si se implementa un prototipo de reconocimiento facial que reconozca el uso de la mascarilla y tome la temperatura corporal, por ende, evitara la propagación del virus del SARS-CoV-2.

Variables	Indicadores
-----------	-------------

“Desarrollo de un prototipo de reconocimiento facial y toma de temperatura para reducir el índice de contagios del SARS-CoV-2.”

- Diseño del prototipo
- Leguaje de programación (Python)
- Herramientas de programación (keras, Tensor Flow)
- Tipo de tecnología (Machine Learning)
- Posicionamiento de la cámara y sensor.
- Altura del sujeto
- Precio

“Proceso de control de medidas de bioseguridad al entrar a un establecimiento.”

- Sistema de control (Raspberry Pi 4)
 - Reconocimiento de la temperatura (sensor infrarrojo y buzzer)
 - Reconocimiento del uso de la mascarilla (cámara web)
-

Elaboración: el autor

1.7.Preguntas de investigación

¿Qué tipos de sistemas de reconocimiento facial existen en el mercado?

¿Qué tipo de herramienta de software se necesita para desarrollar un sistema de reconocimiento facial?

¿Cuáles son las alertas al usuario para que identifique su estado?

¿Cuán asequible sería el prototipo si surgiera en el mercado?

1.8.Alcance del proyecto

Este proyecto es desarrollado con el fin de garantizar el cumplimiento de las medidas de bioseguridad en los locales comerciales y de esta forma reducir el riesgo de contagio de COVID-19 mediante el uso de una Raspberry Pi en conjunto a la ayuda de unos sensores para monitorear la temperatura corporal de una persona el cual tendrá las siguientes funcionalidades:

- Medición de temperatura corporal mediante un sensor infrarrojo.
- Se muestra temperatura en pantalla.
- Identificación automática si la temperatura del usuario es normal o alta, si es alta se emitirá una alarma sonora.
- Margen de error de temperatura ± 0.5 °C.
- Reconocer si el usuario está usando su mascarilla.

Capítulo II

Marco teórico

2.1. Antecedentes del estudio

A lo largo de la historia, han surgido en el mundo muchas enfermedades mortales que han causado la muerte de millones personas. Algunas de estas enfermedades continúan propagándose, siendo aun virus que no ha tenido fin. Algunas de las enfermedades que han dejado gran cantidad de muertes a su paso tales como la peste, el VIH SIDA, la viruela, la gripe española, la gripe porcina, la fiebre amarilla y la fiebre hemorrágica del Ébola. Dichas plagas han cobrado la vida de millones de personas y ha provocado una crisis mundial. (El expreso, 2020)

En diciembre de 2019, apareció un nuevo virus mortal en el mercado mayorista de productos del mar en China (Wuhan). al comienzo se le denominó como 2019 Nuevo Coronavirus (2019n-CoV). El 11 de febrero del 2020 el Comité Internacional sobre Taxonomía de Virus (ICTV) le designo como síndrome respiratorio agudo severo o Coronavirus 2 (SARSCov2) mientras que el cuadro clínico le fue oficialmente llamado Coronavirus Disease – 2019 (COVID-19) (Inca & Inca, 2020)

En Ecuador el primer caso de COVID-19 se dio a conocer el 29 de febrero del 2020, anunciado por la ministra de Salud Pública de esa época, Catalina Andramuño, tratándose de un caso importado de una ciudadana ecuatoriana que regresó de España el 14 de Febrero de ese año. A partir de esa fecha las personas del país entraron en tención debido al contagio que podría provocarse, muchos ciudadanos a pesar de ser el primer contagio estos ya optaban por protegerse. (El Universo , 2020)

El Comité Provincial de Operaciones de Emergencia (COE) junto con el Ministerio de Salud Pública (MSP) diseñaron protocolos de bioseguridad para evitar contagios tales como uso de mascarilla, guantes, alcohol, lavarse correctamente las manos, lavar los alimentos, utilizar alfombra desinfectante en la entrada del hogar, mantener distanciamiento social de mínimo 2 metros entre personas, entre otros.

A partir de mayo del 2020 algunos locales o negocios fueron abriendo gradualmente sus puertas. Estos deben utilizar protocolos de bioseguridad para prevenir la propagación del virus, Las medidas de bioseguridad para ingresar a la instalación son la medición de la temperatura el uso de la mascarilla y la desinfección de las manos. La temperatura permitida de una persona es de hasta 37 ° C, pero si la temperatura supera los 37.5° C, esta puede estar infectado por el virus.

Actualmente se han diseñado varios prototipos de bioseguridad que ayudan a reportar casos de COVID-19 en instituciones o toma de temperatura en domicilios o negocios sin exponer al personal del local, contribuyendo a la protección de los ciudadanos.

Un primer trabajo desarrollado por (TOALA & OMAR, 2020) se basa en el Desarrollo de un Prototipo de Aplicación móvil que permita autoevaluar y reportar casos de COVID-19 en la Unidad Educativa Instituto Británico, en el cual desarrollaron una herramienta tecnológica cuyo fin es autoevaluar al estudiante si tiene una sintomatología similar al de un caso probable de COVID-19 y junto con la ayuda de un representante del Departamento de Consejería Estudiantil (DECE); desde la interfaz del móvil el estudiante contestará un test virtual que será evaluado por medio de la aplicación móvil; desde el lado del DECE, un representante podrá controlar y reportar a las autoridades del plantel. Para el desarrollo del prototipo utilizaron tecnología como Google Firebase, Java e Iconic. En la presente se concluyó que el 92% de los encuestados estuvieron de acuerdo de la propuesta tecnológica de tener una aplicación móvil que permita autoevaluar y reportar posibles casos de COVID-19, lo que significó que la propuesta podrá solucionar la problemática de la unidad educativa Instituto Británico.

Un segundo trabajo desarrollado por (TORRES & TRIANA, 2020) se diseñó un prototipo de sistema de control y medición de temperatura corporal. Esto ayuda a informar a las personas de su temperatura corporal y, por lo tanto, aumenta la seguridad al momento de ingresar a sus hogares. Para el desarrollo del sistema de control de acceso se utilizaron Arduino IDE. El proceso de pruebas se determina que el prototipo de control de acceso ayudara a reducir el número de personas infectadas con el virus del COVID-19 a través de herramientas de medición de temperatura. Este sistema mejora la seguridad de los residentes cuando ingresan a su hogar ya que no permite que nadie sin la tarjeta o contraseña ingrese.

Se puede concluir que hoy en día gracias al avance de la tecnología se pueden desarrollar herramientas tecnológicas para prevenir posibles contagios de COVID-19, puesto a que se puede prevenir posible contagios con la ayuda de estas herramientas y de esta manera proteger la salud de las personas.

2.2. Fundamentación teórica

2.2.1. Control de acceso

Un control de acceso es un sistema automatizado que permite aprobar o denegar de manera efectiva el acceso a una persona o grupos de personas a zonas restringidas en función de los parámetros de seguridad establecida por una empresa, organización u otra organización. (SISCA, 2015)

2.2.1.1 Biometría

Es la tecnología que analiza las características físicas de un individuo utilizando algoritmos capaces de medir patrones específicos, tales como huellas dactilares, retina, rasgos faciales, la voz, etc.

2.2.1.2 Rasgos biométricos

Los rasgos biométricos, son atributos o características fisiológicas del cuerpo humano. Deben ser factores que puedan ser registrados, cuantificados y procesados por un ordenador. La figura 1 muestra algunas características o rasgos del cuerpo humano (Díaz, 2016)



Figura 1. Ejemplos de rasgos del cuerpo humano

Fuente: tomada de alimenconet

2.2.1.3 Propiedades de los rasgos biométricos

Los rasgos usados en los sistemas biométricos requieren los siguientes atributos:

- **Universalidad:** Todos los usuarios del sistema deben de poseer los rasgos que el sistema solicita.
- **Singularidad:** Los rasgos deben ser únicos y no deben repetirse entre usuarios.
- **Cuantificable:** Las propiedades o rasgos deben ser elementos que se puedan registrar, cuantificar y procesar.
- **Rendimiento.** El nivel de precisión del sistema debe satisfacer las restricciones impuestas en el diseño de esta.
- **Aceptabilidad:** El usuario debe confiar en el uso del sistema.
- **Evasión o usurpación:** Establece la resistencia del sistema a las técnicas de fraude, como el robo de información personal y los cambios en las propiedades para evadir el sistema.

2.2.1.4 Diseño de un sistema biométrico

En general, el funcionamiento de un sistema biométrico se basa en dos etapas, registro y reconocimiento de usuario, como se muestra en la figura 2.

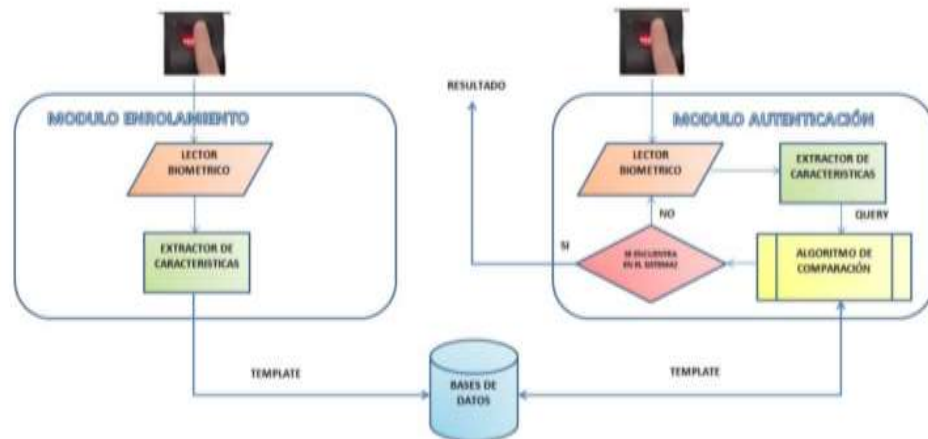


Figura 2. Fases de un sistema biométrico

Fuente: tomada de “Gestión de la identidad biométrica en las organizaciones”, 2020

Elaboración: Julián Felipe Micolta-López, 2015

Durante la fase de registro, el sistema toma la biometría a través del sensor, procesa los datos y obtiene el descriptor o la característica del rasgo para finalmente almacenar el ID de usuario y el descriptor en la base de datos (BD).

En la fase de identificación, por otro lado, el sistema recolecta datos biométricos y luego extrae los descriptores, y los compara con los descriptores almacenados en la base de datos para validar los mismos. Aquí determina si él está registrado en el sistema.

El sistema biométrico es esencialmente un problema de reconocimiento de patrones, por lo que el consta de los módulos que componen el sistema de reconocimiento de patrones. (GEPAR, 2007)

2.2.2 Reconocimiento facial

Es una tecnología desarrollada en el campo de la biometría que utiliza algoritmos computacionales que utilizan procesos informáticos capaces de detectar automáticamente los rostros humanos a partir de imágenes digitales utilizando un análisis profundo y detallado de los rasgos faciales de la persona comparándolos con una serie de imágenes almacenadas en el sistema. (kimaldi, 2017)

2.2.2.1 Método de trabajo

Como ya se ha comentado con anterioridad, un sistema de reconocimiento facial trabaja de igual manera que un sistema biométrico y este puede dividirse en 2 etapas o fases tales como son la fase de detección y la fase de reconocimiento.

- Detección: Para realizar la detección de rostros, primero debe obtener la imagen de entrada. Puede ser una sola imagen tomada con un sensor, cámara, etc. o múltiples imágenes desde diferentes ángulos para permitir una búsqueda más precisa. La detección de rostros se puede realizar de forma estática. Esta encuentra la posición del rostro contenida en cualquier imagen fija o de manera dinámica o en tiempo real donde dada una secuencia de video arbitraria, esta halla la localización del o los rostros que contengan en cada fotograma o para una secuencia de estos
- Reconocimiento: El proceso de reconocimiento primero implica extraer características relevantes del rostro detectado, luego normalizarlo y ajustarlo para una mejor visualización en la base de datos y almacenamiento posterior. Luego se debe hacer una comparación de los rostros normalizados con un conjunto o subconjunto de los resultados de las facetas de normalización anteriores en la base de datos del sistema, proporcionando porcentajes similares o valor equivalente. Debe definir un umbral para definirlo. Si el rostro pertenece o no a la base de datos.

2.2.2.2 Algoritmos

Actualmente, existen muchos algoritmos y métodos de reconocimiento facial, que se describen brevemente a continuación los más usados para una comparación rápida de sus características, ventajas e inconvenientes.

2.2.2.2.1 Correlación y FFT

Consiste en correlacionar la imagen con el rostro reconocido por cada imagen de la base de datos. Este es un proceso muy intensivo desde el punto de vista computacional ya que requiere un alto nivel computacional, así que se aprovechan las propiedades FFT (Fast Fourier Transform) para realizar la correlación, especialmente si tiene una gran cantidad de píxeles por rostro. (Lopez, 2017)

- Ventajas: Algoritmo muy estudiado y fácil de implementar.
- Desventajas: Algoritmo robusto para imágenes dispares pero muy impreciso para rostros, sobre todo donde las diferencias relativas entre las imágenes son pequeñas.

2.2.2.2.2 *PCA y Eigenfaces*

Este método se basa en una herramienta matemática llamada Análisis de componentes principales (PCA), que es útil para la compresión de imágenes y el desarrollo de sistemas de reconocimiento facial. “Es la técnica básica para entender el funcionamiento de esta tecnología y además forma parte importante de técnicas más avanzadas. Está creado a partir de dos fases, una de entrenamiento y otra de clasificación. En la primera, y por medio del PCA, se forma un espacio de facciones, más conocido como eigenspace, a partir de imágenes faciales para entrenamiento”. El espacio característico es una matriz formada por una serie de auto vectores (eigenvectores o eigenfaces) que contienen información sobre la evolución del valor de gris de cada píxel en el conjunto de imágenes utilizadas durante la ejecución del PCA. (knepublishing, 2020)

- Ventajas: Tiempo de cómputo pequeño y elevado decrecimiento de condiciones óptimas tales como iluminación, orientación y posición.
- Desventajas: Es muy dependiente a condiciones de iluminación, posición y orientación de los rostros en la imagen inicial.

2.2.2.2.3 *Basado en parámetros (LFA)*

Este es un derivado del método anterior, solventa el problema de los cambios en posturas y otros problemas del método anterior.

- Ventajas: Adaptación a cambios de expresión.
- Desventajas: Es computacionalmente costoso.

2.2.2.2.4 *Basado en redes neuronales*

Es un algoritmo de aprendizaje profundo que se originó de la computación cognitiva y evolutiva y pertenece a la rama de la inteligencia artificial que representa la función en torno a la arquitectura de aprendizaje. Por otro lado, la característica es que cuantos más elementos use para la representación de la función, más muestras realizara el entrenamiento para tener una

mejor generalización. Dada una definición de la ruta más larga entre el nodo de entrada del sistema y el nodo de salida del sistema.

La información se almacena en las interconexiones y es distribuida, detallada y resuelta por el contenido. Las operaciones se realizan en paralelo y de forma asincrónica. La operación se basa en el aprendizaje, no en algoritmos. Con una serie de ejemplos de capacitación, aprenden los pasos a seguir. Las redes neuronales están formadas por neuronas y estas a la vez se agrupan en capas.

Una de las características fundamentales es el aprendizaje o entrenamiento al que hay que someter a la red neuronal antes de poder aplicarla a la resolución de algún tipo de problema. (Julian, 2016)

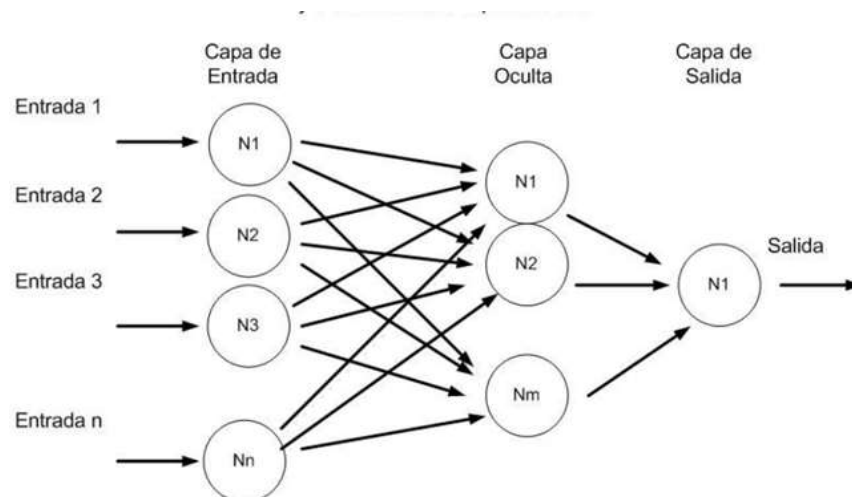


Figura 3. Arquitectura Unidireccional con tres capas entrada, oculta (procesamiento) y de salida

Fuente: Tomada de rtdibermatica.com

El reconocimiento facial desde el punto de vista de la red neuronal no sigue siendo un caso especial de reconocimiento y clasificación de patrones. Una red neuronal se entrena para este propósito memorizando una serie de patrones (en este caso, rostros individuales y rasgos faciales). Si ve una nueva, distorsionada o ruidosa, debe proporcionar la voz original o la voz más reciente.

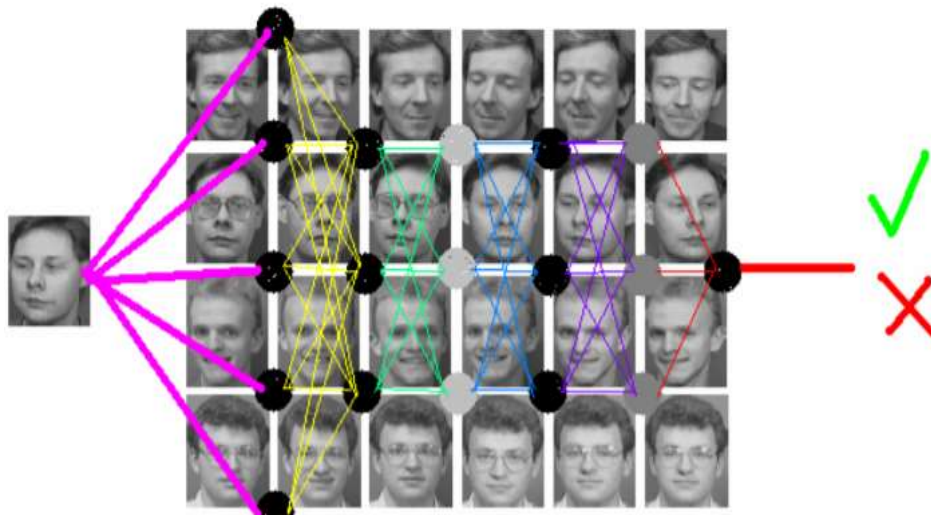


Figura 4. Ejemplo de una Red Neuronal para Reconocimiento Facial

Fuente: tomada de portal.amelica.org

Debido a que la distribución es en forma distribuida y colectiva, por lo que, si el entrenamiento es lo suficientemente bueno, teniendo en cuenta cambios de luz, posición, etc., entonces el resultado es óptimo. Todo a ello a con un elevado costo computacional si se realiza en computadores convencionales donde solo se dispone un microprocesador relativamente complejo que trabaja de manera secuencial, donde habría que emular el paralelismo. (Julian, 2016)

2.2.2.2.5 Diseño de un sistema de reconocimiento de patrones

El sistema de reconocimiento de patrones consta de varios módulos que funcionan sistemáticamente en toda plantilla, sea cual sea el modelo que se implemente. En la figura 5 se ilustra el proceso típico de un sistema de reconocimiento de patrones.



Figura 5. Esquema general de un sistema de reconocimiento de patrones

Fuente: tomada de maginvent.org/

2.2.2.2.5.1 Módulo de adquisición de datos

El proceso comienza con el registro de variables físicas en este caso rostros en un dataset en este se capturan los datos utilizando los sensores para su análisis y procesamiento en Raspberry Pi. La calidad de los datos que se capture dependerá de las características de la cámara o sensor que se use, características tales como la resolución y el diseño y también depende de los factores ambientales como la luminosidad o variaciones en la interacción del usuario.

2.2.2.2.5.2 Módulo de preprocesamiento de datos

Esta actividad se realiza para obtener una mejor la calidad de datos recuperados. En esta se realiza tareas tales como la normalización, eliminación de ruido aleatorio y de datos sin importancia de los dataset.

2.2.2.2.5.3 Módulo de extracción de características

La finalidad de este módulo es generar información relevante y significativa a partir de los datos guardados. La información relacionada se almacena en los vectores de que extraen las características de los rostros almacenados en los dataset (con mascarilla y sin mascarilla).

2.2.2.2.5.4 Módulo de clasificación de datos

En el desarrollo de clasificación los vectores de características se analizan utilizando el enfoque RP (reconocimiento de patrones) para identificar clases (con mascarilla y sin mascarilla) y posteriormente asignar un objeto a cada una de ellas.

2.2.2.2.5.5 Módulo de post-procesamiento de datos

El objetivo en esta etapa es evaluar los resultados de la calificación de las clases y observar si un patrón fue asignado a la clase correcta.

2.2.3 Control de Temperatura corporal

La temperatura corporal es una medida de la capacidad del organismo para la generación y eliminación de calor. Según la Organización Mundial de la Salud si los valores de temperatura están entre los 36,5 y los 37,5 se considera como normal, si los valores de

temperatura están más allá de este rango se puede considerar a la persona con fiebre. (Europapress, 2020)

2.2.4. Sistema de control de acceso RIFD

Antes del inicio de la pandemia había una creciente en los sistemas de control de acceso biométricos con detección de huella digital ya que era un medio rápido y confiable, ahora en tiempo de pandemia se usan más los sistemas con RFID ya que no requieren necesariamente contacto con los dispositivos de lectura que en algunos casos son susceptibles a la transmisión de patógenos peligrosos para la salud en este caso como el Coronavirus. (SEGURILATAM, 2020)



Figura 6. Imagen de un sistema RIFD

Fuente: tomada de *eltecnic.net*

2.2.5. Adecuaciones y controles en un local comercial

Según la Cámara comercial de Quito (CCQ, 2020) se deben cumplir normativas de control para acceder a un local comercial. Algunas adecuaciones y controles recomendados para prevenir la propagación del COVID-19, son los siguientes:

1. Medición de temperatura con un termómetro infrarrojo (mediciones sin contacto) de colaboradores, visitantes o clientes.
2. Siempre que sea posible se deberá implementar llaves con sensores de apertura automática en los servicios higiénicos u otros mecanismos, con el fin de no tener contacto al momento de abrir y cerrar el grifo durante el lavado de manos de los empleados del local comercial.
3. Contar con dispensadores de jabón automático en baterías sanitarias de uso del personal que trabaja para el local comercial.
4. Disponer de gel hidro-alcohólico al 70% al alcance de trabajadores, clientes o visitante.

5. Aumentar los índices de ventilación en espacios cerrados, abrir ventanas para que la circulación del aire se mantenga dentro del local comercial.

6. Implementar dispensadores de papel toalla desechables en todas las baterías sanitarias de uso del personal que trabaja en el local comercial.

7. Para garantizar el distanciamiento social considerar la adecuación de espacios, mediante la reorganización de estanterías, muebles, mostradores u otros.

8. En la medida de lo posible se recomienda colocar barrear acrílicas de protección en las cajas o servicio al cliente.

2.3 Definiciones conceptuales

2.3.1 Prototipo

Según (Pérez & Merino, 2015) nos dice que “Este término se emplea para nombrar al primer dispositivo que se desarrolla de algo y que sirve como modelo para la fabricación de los siguientes o como muestra”.

2.3.2. Python

Python es un lenguaje de programación orientado a objetos de alto nivel con semántica dinámica integradas principalmente para la web y el desarrollo de aplicaciones. En términos de dificultad, es un idioma de programación relativamente simple y fácil de aprender. Esto se debe a que el necesita una sintaxis centrada en la legibilidad. Tiene muchos usos en la actualidad entre ellos están: Big data, Inteligencia artificial, Data science, Desarrollo web entre otras (Capacitarte.org, 2020)

2.3.3. Raspberry Pi

Raspberry es una placa que se puede considerar un microcomputador de bajo costo, también se la puede considerar a la Raspberry como una placa de desarrollo de hardware libre, que permite la elaboración y programación de prototipos. La placa de Raspberry puede hacer todas las funciones de un computador y también funciones de un controlador como leer datos de entrada como la medición de un sensor o la pulsación de un botón y luego convertirlo en instrucciones de salida, por ejemplo, mover un servomotor, encender un diodo LED, etc. (Delgado, 2020)

La última versión de Raspberry Pi en este caso es la 4 cuenta con las siguientes características:

- Procesador ARM Cortex-A72

- Frecuencia del reloj de 1,5 GHz
- GPU: Video Core VI (con soporte para OpenGL ES 3.x)
- Memoria: 1 GB / 2 GB / 4 GB LPDDR4 SD RAM
- Conectividad: Bluetooth 5.0, Wi-Fi 802.11ac, Gigabit Ethernet
- Puertos: GPIO 40 pines 2 x micro HDMI 2 x USB 2.0 2 x USB 3.0 CSI (cámara Raspberry Pi)
- Micro SD
- Conector de audio Jack
- USB-C (alimentación)



Figura 7. Imagen de una Raspberry Pi.

Fuente: tomada de dynamoelectronics.com

2.3.4. Sensores

Un sensor es un componente eléctrico que actúa como un dispositivo de entrada y utiliza un transductor para funcionar, incluyendo que la salida del sensor es un dato útil para el sistema que está midiendo. Estos suelen utilizarse para medir estímulos externos en el entorno en el que se encuentra. Los datos obtenidos del sensor se basan en la medición porque el sensor actúa como un intermediario entre la variable física y el sistema de medición. (Corona, Abarca, & Mares, 2019)

2.3.5. Sensor de temperatura sin contacto Módulo MLX90614 GY-906

El MLX90614 es un sensor de temperatura digital IR sin contacto sensor que se puede utilizar para medir la temperatura de un objeto particular que va desde -70°C a 382.2°C con una precisión de 0.5°C . El sensor utiliza rayos IR (Infrarrojos) para medir la temperatura del objeto sin ningún contacto físico. Este sensor utilizado para obtener mediciones de temperatura sin contacto de alta precisión y es aplicado en sistemas de control de aire acondicionado móvil,

control de temperatura en piezas industriales o en electrodomésticos, sector salud, medición de temperatura corporal, etc. (Shudhanshu & Ankit, 2020)



Figura 8. *Imagen de un sensor de temperatura IR*

Fuente: tomada de dynamoelectronics.com

2.3.6. Buzzer

Un buzzer o altavoz es un dispositivo que convierte señales eléctricas en ondas sonoras. Dado que estos dispositivos no tienen componentes electrónicos internos, deben proporcionar una señal eléctrica para lograr el sonido deseado. Pese a tener la complejidad de proporcionar y el control de las señales eléctricas los buzzer tienen la ventaja de poder cambiar el timbre resultante cambiando la señal aplicada a los altavoces, lo que nos permite generar melodías. (Llamas, 2016)



Figura 9. *Imagen de un Buzzer*

Fuente: tomada de avelectronics.cc

2.3.7 COVID-19

Es una enfermedad infecciosa causada por el coronavirus descubierto a finales del 2019. La enfermedad fue descubierta durante un brote en Wuhan (China). El Covid-19 provocó una pandemia que actualmente sigue afectando muchos países en el mundo. (OMS, 2020)

2.3.8 Coronavirus

Son una extensa familia de virus causante de enfermedades en animales y humanos. En el caso de los humanos, provocan severas infecciones respiratorias. (OMS, 2020)

2.3.9 Reconocimiento facial

Es un sistema computacional que permite detectar o reconocer los rostros de las personas de manera automática en tiempo real

2.3.10 Dataset

Es la información almacenada o conjunto de datos relacionados a una misma característica o contexto destinada a su posterior uso.

2.3.11 Procesamiento de imágenes

Es un conjunto de procesos o técnicas que permite descubrir o interpretar las imágenes por medio de una herramienta de visión computacional.

2.3.12 Redes neuronales

Es aquel algoritmo de aprendizaje profundo dentro de la rama de Inteligencia Artificial, está relacionada al camino más extenso comprendido entre el nodo de ingreso del sistema y el nodo de salida del sistema.

2.3.13 Machine Learning

Esta es una tecnología que permite la automatización de una amplia gama de actividades para reducir la necesidad de intervención humana. Esta es una gran ventaja a la hora de gestionar grandes cantidades de información de forma mucho más eficaz. (APD, 2019)

2.4 Fundamentación legal

La Constitución de la República del Ecuador vigente, garantizan la educación y desarrollo profesional sin restricciones de los estudiantes que se encuentran culminando su etapa académica profesional, cuyos objetivos es respaldar los derechos legítimos a la educación con los que cuenta cada ciudadano ecuatoriano o extranjero que está en su pleno proceso de formación integral como ser humano y que brindará apoyo en el crecimiento del país.

Por tanto, el presente proyecto de titulación cumple con todos los requisitos que la ley establece para su adecuado desarrollo y fortalece los principios de justicia social. A continuación, se detallan cada uno de los artículos que respaldan esta propuesta.

2.4.1 Constitución de la República del Ecuador

Dentro de la Constitución de la Republica del Ecuador en el Capítulo Segundo Tercera Sección en los derechos del buen vivir se menciona en el artículo 16 literales 1,2 y 4 que:

Toda persona, individual o colectivamente, tiene derecho al acceso universal a las tecnologías de la información y comunicación, y por tanto al uso a todas las diversas formas de comunicación ya sea esta visual, auditiva, sensorial entre otras.

En los artículos se pueden enfatizar que el estado Ecuatoriano garantiza el derecho a la libertad de expresión sin excepción ni discriminación en donde se respete y no se niegue el derecho de acceso a la tecnología a todos sus ciudadanos.

A su vez en la Constitución Política del Ecuador en el Capítulo Segundo en los Derechos del Buen Vivir en su Sección Cuarta el Artículo 22 nos señala que las personas tienen derecho a desarrollar sus capacidades creativas, a realizar actividades culturales y artísticas de manera digna y sostenible, y también poder beneficiarse de la protección de sus respectivos derechos para la producciones científicas, literarias o artísticas. (Constitución de la República del Ecuador, 2008)

El Gobierno Ecuatoriano fomenta e impulsa el desarrollo tecnológico y este facilita a la población la búsqueda de mejores formas de desarrollar productos tecnológicos para el beneficio a la población.

Capítulo III

Propuesta

El presente proyecto pretende desarrollar un prototipo de sistema de control de acceso a personas, que permita monitorear el uso correcto de la mascarilla y toma de temperatura utilizando Raspberry pi y Machine Learning, para evitar la propagación del virus de SARS-CoV-2. Dicho sistema de control de acceso tendrá integrado un sensor de temperatura corporal infrarrojo con el cual se podrá tomar las temperaturas de los usuarios, esto junto a un buzzer que sonara si una persona presenta una temperatura superior a los 37.5°.

Hoy en día el mundo entero sigue atravesando una pandemia por el virus letal del COVID-19 lo que hace necesario cumplir protocolos de bioseguridad en instituciones educativas, instituciones públicas como privadas, domicilios y locales comerciales, donde hay mucha concurrencia de personas. Las instituciones o establecimientos optan por seleccionar a un empleado para tomar temperatura y ver que la otra persona use la mascarilla siendo peligroso porque no cumple con el distanciamiento mínimo de 2 metros. Este sistema de control de acceso al ser automatizado evita un posible contagio entre el empleado y el cliente, permitiendo cuidar la salud de las personas.

3.2 Metodología del proyecto

En este proyecto se desarrolló un prototipo que se encarga del reconocimiento de la mascarilla y toma de temperatura para prevenir posibles contagios de Covid-19. Al estar centrado en un campo tecnológico existen diferentes maneras de realizar este tipo de proyectos, en este caso la metodología a utilizar en el proyecto es la metodología PMI que incluye cinco fases desde el inicio del proyecto hasta su finalización, que se enfoca en el seguimiento y control del proyecto en caso de que se requieran cambios o modificaciones. Esto se hace para lograr las metas previamente establecidas para el producto o servicio desarrollado en el proyecto.



Figura 10. Etapas de la metodología

Fuente: tomada de www.coordinate.si

3.2.1 Fase de inicio

En esta etapa se especificara y se determinara los puntos importantes a cubrir, por consiguiente, se realizara un análisis previo con el fin definir la problemática para poder brindar una solución como se encuentra especificado en el primer capítulo.

3.2.2 Fase de planificación

Durante esta fase se estableció la investigación referente a los componentes (hardware y software) que permitirán la realización del prototipo y a su vez los requerimientos que estos requieran.

3.2.3 Fase de ejecución

Una vez ya definida la problemática y los componentes a utilizar para el diseño y construcción que permitirá solucionar el problema, en esta etapa se debe cumplir con el funcionamiento del prototipo en este caso el sistema deberá reconocer si una persona usa correctamente su mascarilla o no y así mismo deberá medir la temperatura corporal.

3.2.4 Fase de seguimiento y control

Una vez ya realizado el prototipo en esta fase se observa y se mide el rendimiento para así tener un buen funcionamiento y poder tener el control de anticiparse a corregir aspectos que no estén optimizados.

3.2.5 Fase de cierre

Ya terminadas las observaciones y pruebas se tienen que corregir aquellos aspectos que no estén del todo optimizado, se debe analizar aquellos factores que puedan influir en el funcionamiento del prototipo como posicionamiento de la cámara y sensor.

3.3 Descripción

3.2.1 Factibilidad técnica

Para el presente proyecto, se ha realizado un análisis de los componentes necesarios tanto de hardware como de software, con la finalidad de asegurar un funcionamiento correcto. Los componentes de hardware utilizados para el prototipo se pueden encontrar fácilmente en tiendas de electrónica. En cuanto a las herramientas de desarrollo, para la programación se utilizará el IDE de Jupyter que nos facilitara el desarrollo de la programación.

Recursos de hardware

Materiales para la elaboración del prototipo

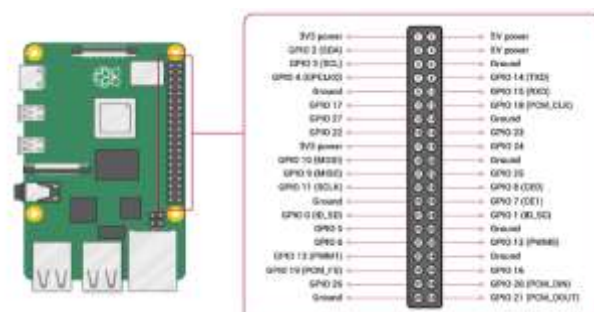
Los componentes de hardware que constan en el prototipo son los siguientes:

- Raspberry Pi 4
- Sensor de Temperatura MLX90614
- Cámara web
- Cables Jumper (Macho-Macho y Macho-Hembra)
- Cable Ethernet Rj45 5 Metros
- Fuente de alimentación de 5V - 3ª
- Tarjeta SD de 32GB
- Lector de tarjeta USB MicroSD
- Cable micro HDMI a HDMI (opcional)
- Case de la Raspberry Pi 4
- Disipadores de calor
- Buzzer

A continuación, se detallan los componentes que se requerirán para desarrollar el prototipo.

Raspberry Pi 4 Model B

Se utiliza este controlador para enviar instrucciones a los sensores conectados y estos puedan realizar las acciones que se les indica, como se muestra en la figura 11, estos se conectan por medio de los pines GPIO (General Purpose Input Output) que sirven para enviar y recibir señales de dispositivos externos en este caso los diferentes sensores.



Sensor de Temperatura MLX90614

El MLX90614 es un termómetro infrarrojo diseñado para medir temperatura de un objeto sin contacto, este provee dos métodos de salida. PWM y SMBus. La salida PWM 10-bits provee una resolución de 0.14°C , mientras que la interfaz TWI una resolución de 0.02°C . El MLX90614 está calibrado de fábrica en un rango de temperaturas amplio: -40 a 85°C para temperatura ambiente y -70 a 382.2°C para temperatura de objetos. El valor medido es el promedio de la temperatura de todos los objetos en el rango de visión del sensor. El MLX90614 ofrece una precisión estándar de 0.5°C en temperaturas ambiente.

Este sensor permite detectar la temperatura del objetivo para posteriormente interpretar si el objetivo tiene o no tiene fiebre.

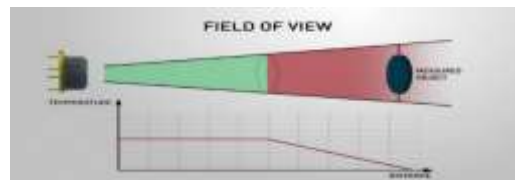


Figura 12. Sensor de temperatura infrarrojo

Fuente: tomada de uelectronics.com

Características del sensor infrarrojo MLX90614	
Voltaje de alimentación	5v o 3v
Distancia	De 3 a 5 centímetros
Rango de temperatura	-40°C a $+85^{\circ}\text{C}$
Pines	Vss, Scl, Vdd, SDA/PWM
Dimensiones	$1.7 \times 1.1 \times 0.6$ cm

Cámara web

Las imágenes se pueden describir por la intensidad o la reflectividad de la luz, dependiendo de su posición en el plano definido. En cambio, una cámara es un dispositivo que captura y registra imágenes. En otro termino es convertir información contenida en una imagen que se pueda registrar de manera reproducible. (cristhian, 2019)

Recursos de Software

Los componentes de software que constan en el prototipo son los siguientes:

Tabla 4. Tecnologías utilizadas durante el desarrollo del proyecto

Software utilizado	
Sistema operativo	Raspbian
Lenguaje de programación	Python
IDE	Jupyter
	Notebook
Librerías	Keras
	Tensor Flow
	Open CV

Elaboración: el autor

3.2.2 Factibilidad legal

Para la ejecución del presente proyecto, se han aplicado los artículos 350, 355, 385, 386, 387 y 388 de la constitución de la República del Ecuador, los artículos 3 y 8 de la Ley Orgánica de Educación Superior (LOES). Comprobando de esta manera, el cumplimiento de las leyes mencionadas. Las tecnologías utilizadas para la implementación del proyecto son de código abierto y de libre acceso, por lo que no tiene restricciones para el uso de estos.

3.2.3 Factibilidad económica

Para la elaboración del prototipo, se necesitan materiales electrónicos, equipos informáticos y software de desarrollo cuyo objetivo es proporcionar a las instituciones o locales un sistema que permita mostrar información sobre las temperaturas tomadas de las personas y si estas están usando la mascarilla al ingreso.

Tabla 5.

Costo de inversión en Software

Software	Precio unitario	Total
Python	\$0,00	\$0,00
Raspbian	\$0,00	\$0,00
Jupyter Notebook	\$0,00	\$0,00
Librerías de desarrollo	\$0,00	\$0,00
Total	\$0,00	\$0,00

Elaboración: el autor**Tabla 6.**

Costo de inversión en Hardware

Hardware	Cantidad	Costo unitario	Total
Kit Raspberry Pi 4	1	\$120,00	\$120,00
Sensor de Temperatura MLX90614	1	\$12,00	\$12,00
Espadines 1x40	1	\$0,40	\$0,40
Cables Jumper (Hembra-Hembra)	40	\$0,058	\$2,32
Cable Ethernet Rj45 5 Metros	1	\$4,00	\$4,00
Buzzer	1	\$0,75	\$0,75
Cámara web	1	\$10,00	\$10,00
Total		\$143,21	\$145,47

Elaboración: el autor

Tabla 7. Costo total del proyecto

Costo de hardware y software	
Costo de Software	\$0,00
Costo de Hardware	\$145,47
Total	\$145,47

Elaboración: el autor

3.2.4 Factibilidad operacional

El análisis operacional de este proyecto se considera factible, debido a que el prototipo planteado ayuda a mejorar el proceso de toma de temperatura corporal y uso de la mascarilla contribuyen a prevenir un posible contagio entre el empleado y el cliente. La institución o negocio resulta beneficiado, puesto que, ya no llevará un proceso manual para la toma de temperatura y uso de la mascarilla, los clientes podrán conocer su temperatura actual cuando hagan uso del prototipo de medición antes de ingreso.

3.4 Esquema general del proyecto

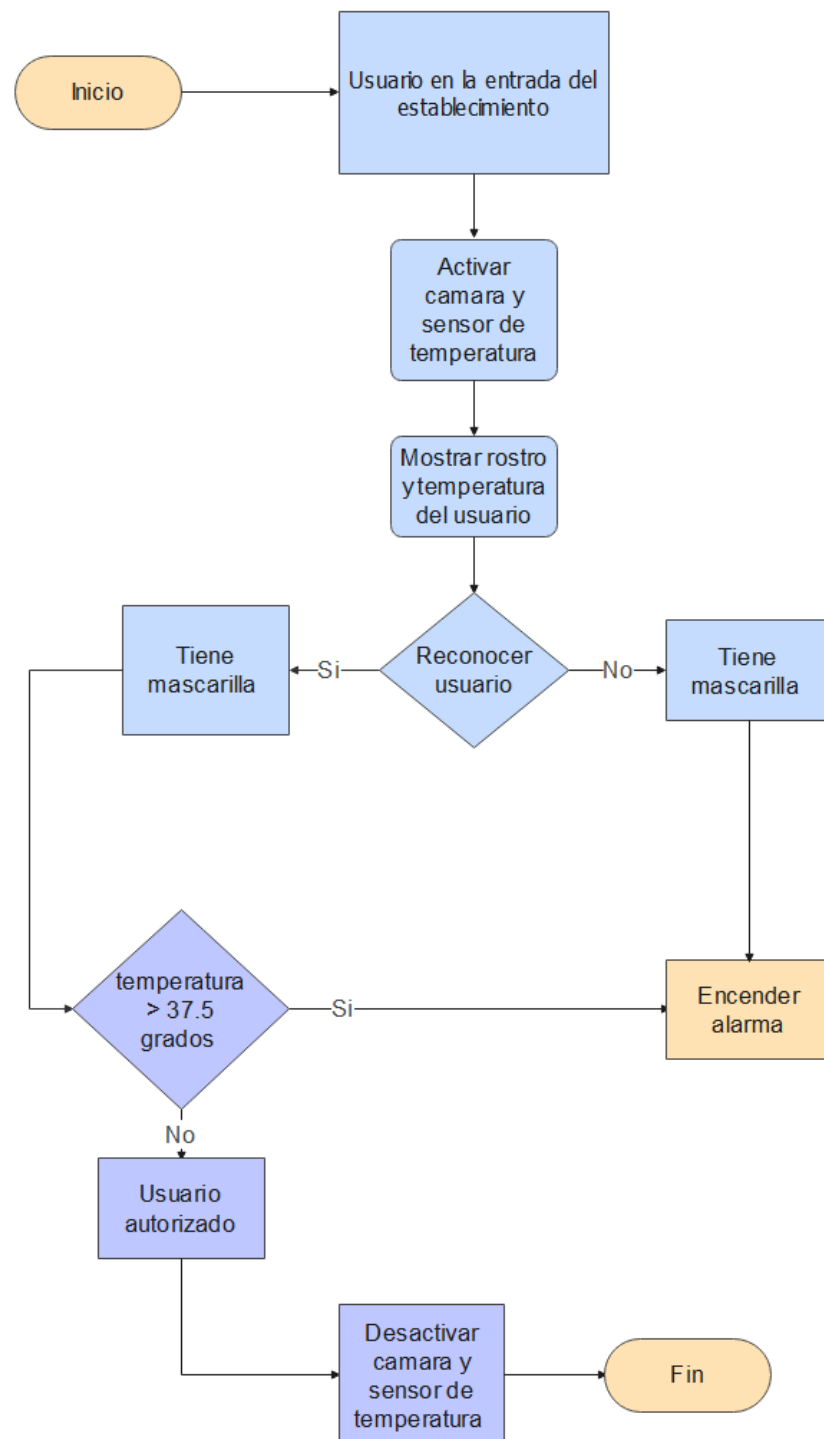


Figura 13. Diagrama de flujo sobre funcionamiento del prototipo.

Elaboración: El autor.

3.5 Diseño y construcción

3.5.1 Diseño del prototipo

En esta se explica el desarrollo del sistema de reconocimiento facial de uso correcto de mascarilla y toma de temperatura y a su vez comprobar la factibilidad del prototipo.

Se realiza un diseño o esquema del prototipo propuesto y definimos los componentes (hardware y software) adecuados para cada una de las etapas del desarrollo del sistema y por ultimo las pruebas de funcionalidad pertinentes. Para el desarrollo se contamos con la ayuda de un computador o un monitor para el uso de la Raspberry Pi en este caso una Raspberry Pi 4, para capturar los rostros para realizar el reconocimiento facial se usa una cámara web genérica junto con un sensor de temperatura infrarrojo y un buzzer que nos notificara si una persona no está usando la mascarilla o si tiene temperatura alta. A continuación, se muestra un esquema base para el proyecto.



Figura 14. Componentes del prototipo.

Fuente: investigacion directa

Elaboración: el autor

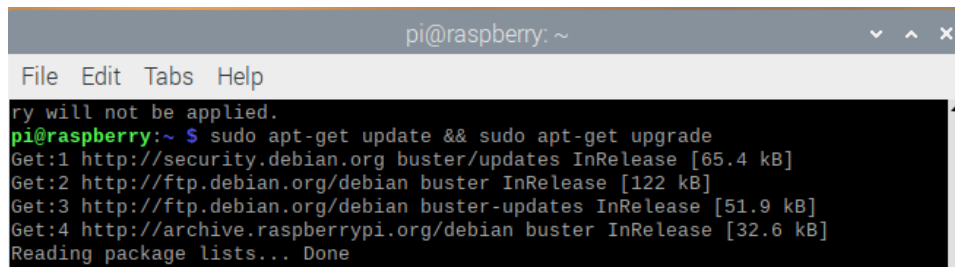
3.5.2 Desarrollo del prototipo

3.5.2.1 Instalación de OpenCV Y dependencias

Para el desarrollo del sistema se utiliza biblioteca de OpenCV, este reúne todos los recursos necesarios para lograr la más amplia variedad de aplicaciones de visión artificial que pueda imaginar, este incluye: adquisición/recepción de imágenes de cámaras digitales, procesamiento de imágenes y procesamiento de imágenes fijas y video (video transmitido y grabado) y algoritmos de inteligencia artificial.

- Para instalar OpenCV primeramente se recomienda actualizar la Raspberry con ello se actualizan los repositorios para tener una mejor optimización y trabajar

con librerías actualizas. Para ello usamos el siguiente comando **sudo apt-get update && sudo apt-get upgrade** tal y como en la figura 14.



```

pi@raspberrypi: ~
File Edit Tabs Help
ry will not be applied.
pi@raspberrypi:~ $ sudo apt-get update && sudo apt-get upgrade
Get:1 http://security.debian.org buster/updates InRelease [65.4 kB]
Get:2 http://ftp.debian.org/debian buster InRelease [122 kB]
Get:3 http://ftp.debian.org/debian buster-updates InRelease [51.9 kB]
Get:4 http://archive.raspberrypi.org/debian buster InRelease [32.6 kB]
Reading package lists... Done

```

Figura 15. Actualizacion de raspberry pi

Fuente: tomada de Raspberry Pi

Elaboración: el autor

Luego se procede a la instalacion de unos paquetes y herramientas de desarrollo que sirven para configurar los procesos de copilacion de OpenCV con los siguientes comandos:

sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev sudo apt-get install libxvidcore-dev libx264-dev

y proseguimos a la instalación de las demás librerías y dependencias necesarias:

sudo apt-get install libgtk2.0-dev libgtk-3-dev

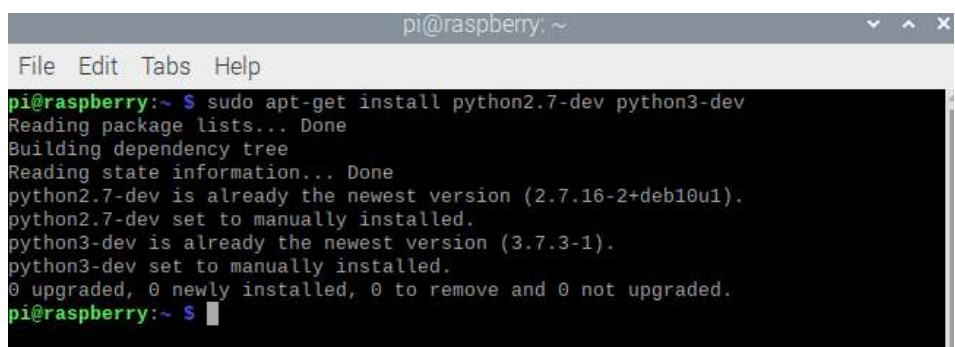
sudo apt-get install libatlas-base-dev gfortran

y por ultimo procedemos a instalar los archivos de Python 2.7 junto a los de Python 3 para la posterior copilacion de OpenCV con Python:

sudo apt-get install python2.7-dev python3-dev

he instalamos numpy con

sudo pip3 install numpy



```

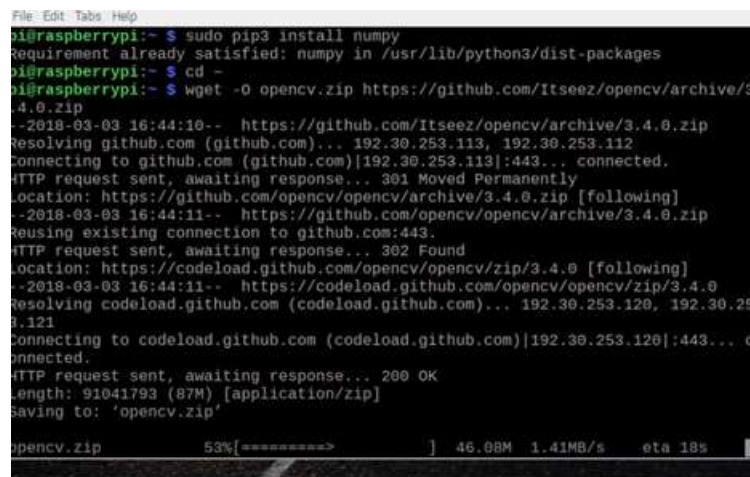
pi@raspberrypi: ~
File Edit Tabs Help
pi@raspberrypi:~ $ sudo apt-get install python2.7-dev python3-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
python2.7-dev is already the newest version (2.7.16-2+deb10u1).
python2.7-dev set to manually installed.
python3-dev is already the newest version (3.7.3-1).
python3-dev set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
pi@raspberrypi:~ $

```


Figura 16. Instalacion de paquetes de desarrollo**Fuente:** tomada de Raspberry Pi**Elaboración:** el autor

- Ahora se descarga el código fuente de OpenCV este puede ser la versión más reciente o una versión en la que queremos trabajar que vaya acorde con los requerimientos que se estén usando, la mayoría de bibliotecas están disponibles en repositorios Github o simplemente lo descargamos de la misma Raspberry, ahora precedemos a insertar la siguiente línea de comando:

cd ~

wget -O opencv.zip <https://github.com/Itseez/opencv/archive/3.4.0.zip>


```

pi@raspberrypi:~$ sudo pip3 install numpy
Requirement already satisfied: numpy in /usr/lib/python3/dist-packages
pi@raspberrypi:~$ cd ~
pi@raspberrypi:~$ wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.4.0.zip
--2018-03-03 16:44:10-- https://github.com/Itseez/opencv/archive/3.4.0.zip
Resolving github.com (github.com)... 192.30.253.113, 192.30.253.112
Connecting to github.com (github.com)[192.30.253.113]:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://github.com/opencv/opencv/archive/3.4.0.zip [following]
--2018-03-03 16:44:11-- https://github.com/opencv/opencv/archive/3.4.0.zip
Reusing existing connection to github.com:443.
HTTP request sent, awaiting response... 302 Found
Location: https://codeload.github.com/opencv/opencv/zip/3.4.0 [following]
--2018-03-03 16:44:11-- https://codeload.github.com/opencv/opencv/zip/3.4.0
Resolving codeload.github.com (codeload.github.com)... 192.30.253.120, 192.30.253.121
Connecting to codeload.github.com (codeload.github.com)[192.30.253.120]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 91041793 (87M) [application/zip]
Saving to: 'opencv.zip'

opencv.zip          53%[=====>] 46.08M  1.41MB/s  eta 18s

```

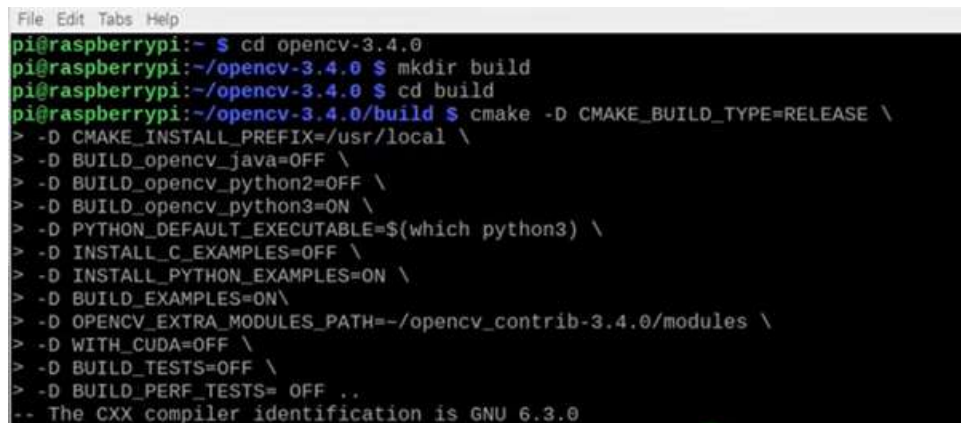
Figura 17. Descarga del código fuente de OpenCV**Fuente:** tomada de Raspberry Pi**Elaboración:** el autor

- Descomprimos el archivo
unzip opencv.zip
wget -O opencv_contrib.zip
https://github.com/Itseez/opencv_contrib/archive/3.4.0
unzip opencv_contrib.zip
- Empezamos la siguiente compilación usando CMake:
mkdir build
cd build
**cmake -D CMAKE_BUILD_TYPE=RELEASE **
**-D CMAKE_INSTALL_PREFIX=/usr/local **
**-D BUILD_opencv_java=OFF **

```

-D BUILD_opencv_python2=OFF \
-D BUILD_opencv_python3=ON \
-D PYTHON_DEFAULT_EXECUTABLE=$(which python3) \
-D INSTALL_C_EXAMPLES=OFF \
-D INSTALL_PYTHON_EXAMPLES=ON \
-D BUILD_EXAMPLES=ON\
-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-
3.4.0/modules \
-D WITH_CUDA=OFF \
-D BUILD_TESTS=OFF \
-D BUILD_PERF_TESTS= OFF ..

```



```

File Edit Tabs Help
pi@raspberrypi:~ $ cd opencv-3.4.0
pi@raspberrypi:~/opencv-3.4.0 $ mkdir build
pi@raspberrypi:~/opencv-3.4.0 $ cd build
pi@raspberrypi:~/opencv-3.4.0/build $ cmake -D CMAKE_BUILD_TYPE=RELEASE \
> -D CMAKE_INSTALL_PREFIX=/usr/local \
> -D BUILD_opencv_java=OFF \
> -D BUILD_opencv_python2=OFF \
> -D BUILD_opencv_python3=ON \
> -D PYTHON_DEFAULT_EXECUTABLE=$(which python3) \
> -D INSTALL_C_EXAMPLES=OFF \
> -D INSTALL_PYTHON_EXAMPLES=ON \
> -D BUILD_EXAMPLES=ON\
> -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.4.0/modules \
> -D WITH_CUDA=OFF \
> -D BUILD_TESTS=OFF \
> -D BUILD_PERF_TESTS= OFF ..
-- The CXX compiler identification is GNU 6.3.0

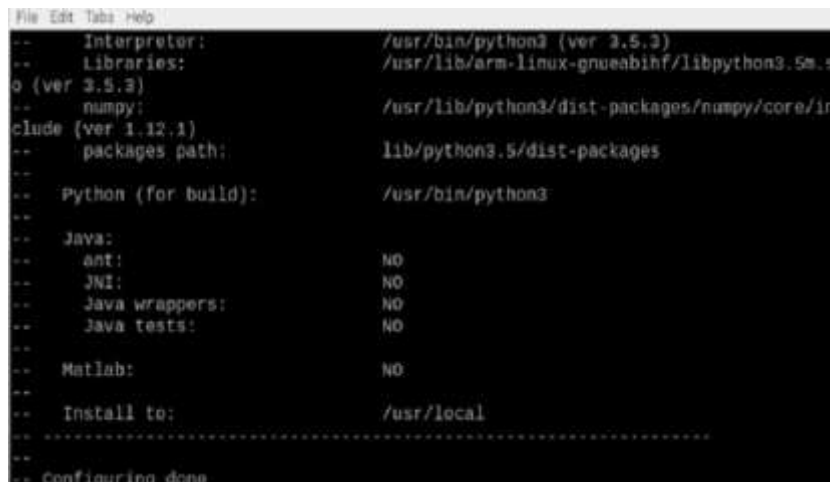
```

Figura 18. Copilacion de OpenCV para Python 3

Fuente: tomada de Raspberry Pi

Elaboración: el autor

Para verificar si se esta el proceso compilando en OpenCV para Python3 se examina la salida de CMake como en la figura.



```

File Edit Tabs Help
-- Interpreter: /usr/bin/python3 (ver 3.5.3)
-- Libraries: /usr/lib/arm-linux-gnueabi/libpython3.5m.so
o (ver 3.5.3)
-- numpy: /usr/lib/python3/dist-packages/numpy/core/in
clude (ver 1.12.1)
-- packages path: lib/python3.5/dist-packages
-- Python (for build): /usr/bin/python3
-- Java:
-- ant: NO
-- JNI: NO
-- Java wrappers: NO
-- Java tests: NO
-- Matlab: NO
-- Install to: /usr/local
-----
-- Configuring done

```

Figura 19. Verificación de la compilación de OpenCV**Fuente:** tomada de Raspberry Pi**Elaboración:** el autor

- Para empezar el proceso de compilación en OpenCV lo recomendable es hacerlo con un solo núcleo para ello escribimos el comando **make -j** esta instalación es larga puede tener una duración de aproximadamente 2 horas.

Al terminar el proceso de compilación se instala OpenCV en nuestra Raspberry Pi usando las siguientes líneas de comando:

sudo make install

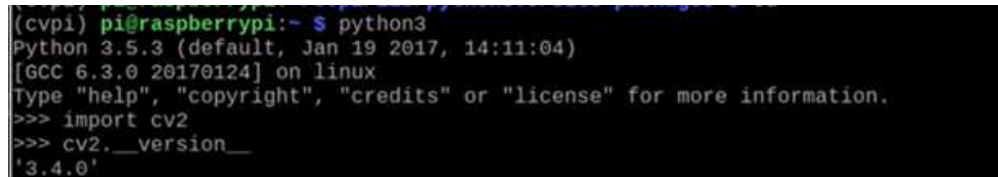
sudo ldconfig

- Ya por último si ha salido bien todo el proceso sin errores solo nos queda verificar la prueba de la instalación del paquete OpenCV listo para el proyecto, para ello ejecutamos las siguientes líneas de comandos :

python 3

Import cv2

Cv2.__version__



```
(cvpi) pi@raspberrypi:~ $ python3
Python 3.5.3 (default, Jan 19 2017, 14:11:04)
[GCC 6.3.0 20170124] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'3.4.0'
```

Figura 20. Verificación de la versión de OpenCV instalada**Fuente:** tomada de Raspberry Pi**Elaboración:** el autor

La captura de la Raspberry no muestra ningún error, con ello ya damos terminada la instalación de OpenCV versión 3.4.0 se instaló correctamente en un entorno virtual Python para la versión 3.5 de este.

3.5.2.2 Entrenamiento del sistema de reconocimiento de mascarilla

Una vez ya instalado OpenCV ya se puede empezar con las configuraciones y el entrenamiento facial, eso con la ayuda de una cámara para reconocer los rostros en tiempo real y alimentar el dataset para que este reconozca a las personas que tengan u no tengan puestas su mascarilla. Para ello se debe trabajar en diferentes fases

- Recopilación de datos y detección facial
- Entrenar el sistema de reconocimiento
- Reconocimiento facial

1) Recopilación de datos

Para empezar a desarrollar el sistema primero hay que desarrollar un dataset con la información que necesitemos para que el sistema pueda comenzar a identificar y distinguir los rostros que posean o no la mascarilla, en este caso necesitamos 2 carpetas una donde se almacena rostros de personas con mascarilla y otra con personas sin su mascarilla.

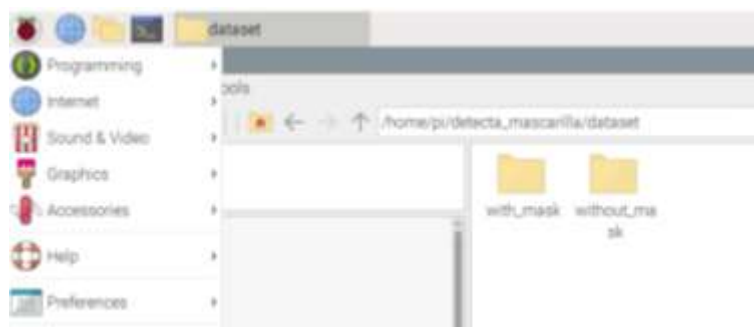


Figura 21. Elaboracion de dataset

Elaboración: el autor

2) Entrenamiento del reconocedor

En OpenCV ya cuenta con su propio entrenador y detector, tambien contiene una gran variedad de clasificadores ya entrenados previamente para el reconocimiento facial que se pueden descargar directamente desde el directorio (haarcascades). En esta parte se tiene que tomar todos los datos del usuario guardados en la base de datos que se realizo previamente y entrenarlos con el entrenador de OpenCV. Para ello se crea un IDE de Python para escribir el codigo del entrenamiento dentro de la carpeta del proyecto (detecta_mascarilla) .

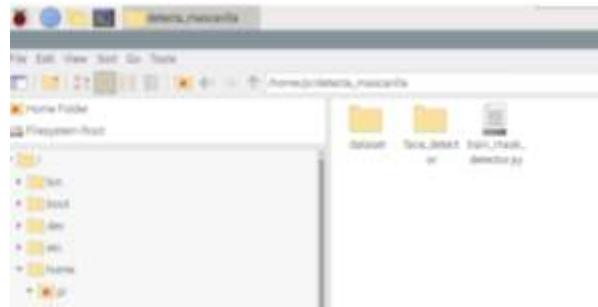


Figura 22. Archivo de entrenamiento

Elaboración: el autor

Para el reconocedor se emplea la técnica de reconocimiento facial basado en redes neuronales que viene incluido en el paquete de OpenCV. Luego de esto se ejecuta el script de Python `train_mask_detector.py` esto en el entorno creado dentro de la terminal de la Raspberry Pi y si el código es correcto comienza el entrenamiento dependiendo de cuantas épocas se determinó en este caso 20, hay que tomar en cuenta que entre más épocas determinamos en el entrenamiento tendrá mejores resultados el reconocedor.

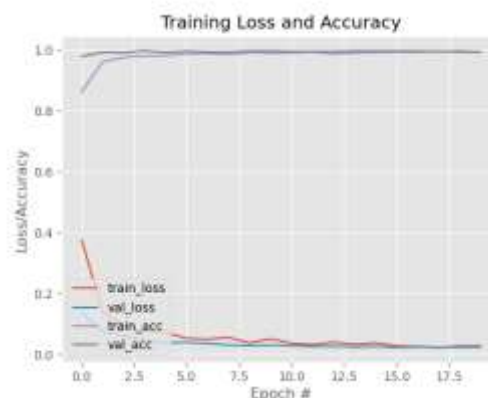


Figura 23. Grafico de precision de los datos en el entrenamiento

Fuente: tomada de Raspberry Pi

Elaboración: el autor

3) Reconocimiento facial

En esta última fase del reconocimiento facial, se captura el rostro del usuario en la cámara y el reconocedor devolverá su respectiva clasificación con una etiqueta que muestra de manera porcentual que tan seguro está el reconocedor con esta coincidencia con respecto al dataset que ya se entrenó anteriormente. Se ejecuta el script de Python **`detectar_mascarilla_temp_webcam.py`** dentro del entorno virtual ya creado dentro de la terminal.



Figura 24. IDE de Python de última fase

Elaboración: el autor

Entonces la cámara se activa y detecta el rostro y comprara con la el dataset de imágenes existente ya predeterminada previamente. Se puede identificar el rostro colocando una etiqueta sobre el rostro detectado y además se puede mostrar una etiqueta de confianza con la probabilidad en % de la coincidencia sea esta correcta o no en este casi así se usó y se colocó otra etiqueta que muestra la temperatura del usuario sobre el rostro como se muestra en la figura 24.

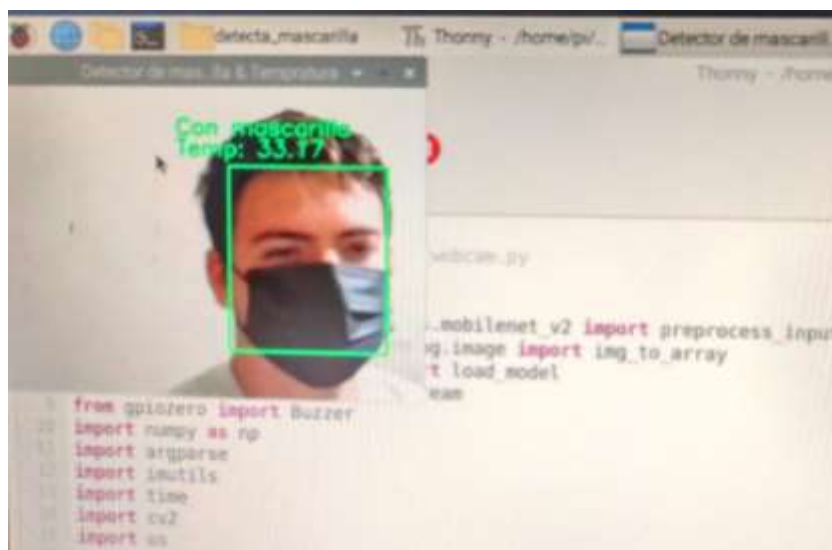


Figura 25. Reconocimiento de la mascarilla y l toma de tempertatura

Elaboración: el autor

3.5.3 Conexiones del sistema de reconocimiento de mascarilla

El prototipo consta de un circuito no tan complejo, los sensores trabajan con una placa programable como es el caso de la Raspberry Pi 4, una cámara web genérica, un sensor de temperatura infrarrojo y un buzzer.

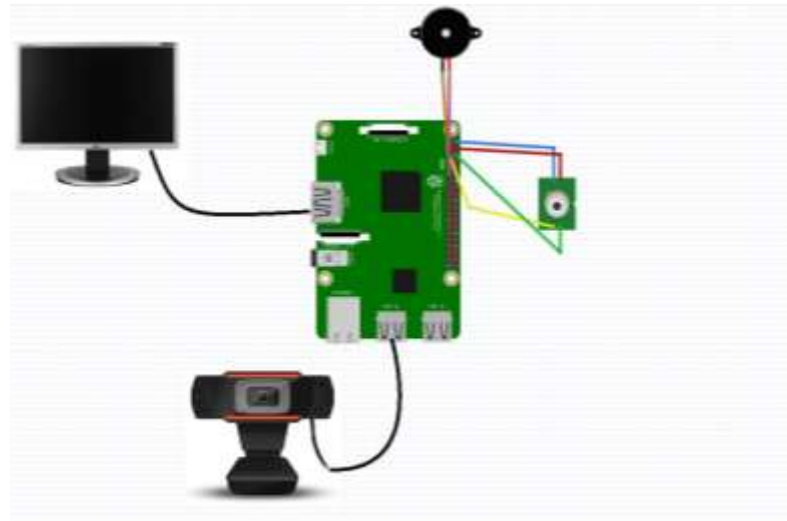


Figura 26. Conexiones de los sensores

Elaboración: el autor

3.5.4 Configuraciones de pines GPIO

General Purpose Input Output (GPIO) Es un sistema de entrada y salida de propósito general, lo que significa que consta de una serie de pines que se pueden usar como entradas o salidas para una variedad de usos. ya sea entrada y salida de información o señales de los sensores. Estos pines están incluidos en todos los modelos de Raspberry Pi. El puerto consta de dos pines de 5V y dos pines de 3,3V están presentes en la placa, así como varios pines de tierra GND: 0V, que no son configurables., la corriente máxima por cada pin es de 16 mA y una total en los pines de 50 mA.

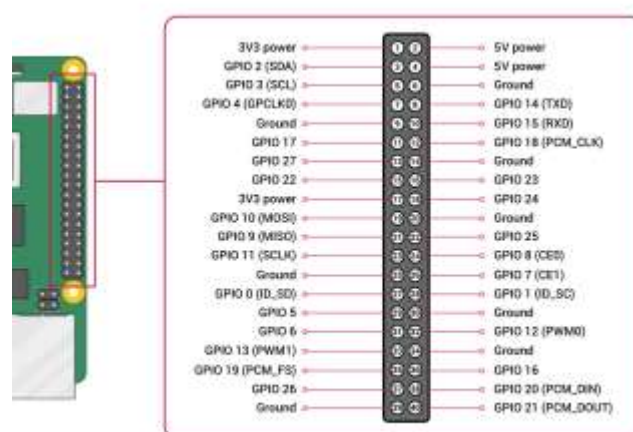


Figura 27. Puerto GPIO

Fuente: tomada de hwlibre.com

Elaboración: el autor

La conexión entre el tablero GPIO y el sensor de temperatura infrarrojo son: el pin de 5v aquí puede ser el pin 2 o 4 conectado a la entrada de la señal Vin del sensor, el pin 6 conectado al GROUND del sensor, mientras las conexiones de datos el pin 5 conectado a la entrada SDL, y el pin 3 a la SDA.

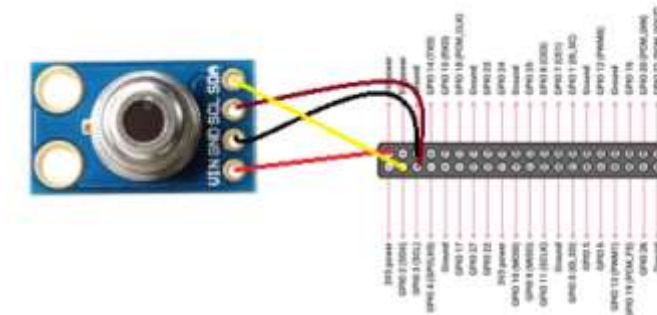


Figura 28. Conexión del sensor de temperatura

Elaboración: el autor

Para el buzzer este solo cuenta con 2 patas o pines 1 que sera de datos y el otro conectado a tierra o ground, para ello se utilizo el pin 9 para Ground y el 9 pin 11 que es el puerto GPIO 17 que sera designado para puerto de datos.

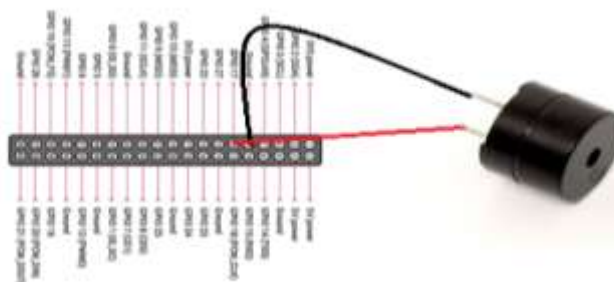


Figura 29. Conexión del Buzzer

Elaboración: el autor

3.6 Descripción

La función del sistema es poder identificar si una persona está usando correctamente la mascarilla y si esta presenta fiebre, esto con la ayuda de una base de datos que servirán de para el identificador pueda aprender las variaciones faciales de los usuarios.

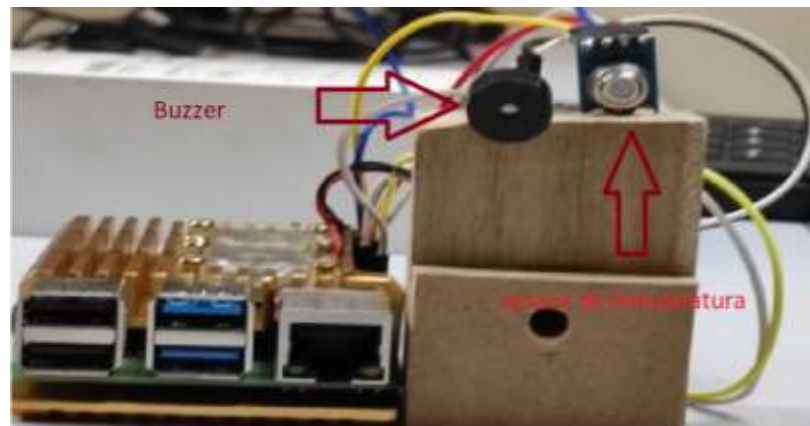


Figura 30. Conexiones de los sensores del sistema

Elaboración: el autor

Para que el sistema funcione de manera correcta hay que tomar en cuenta lo siguiente:

- El sistema debe colocarse a una altura considerable a la que pueda captar el rostro, la altura recomendable es a 1.5 metros a 1.65 metros.
- la distancia recomendable para que el sistema reconozca si el usuario lleva o no mascarilla es de 0.5 metros a 1 metro.
- Hay que tomar en cuenta que la distancia del sensor de temperatura trabaje de manera correcta debe ser menor a 10 centímetros.

3.7 Pruebas de funcionamiento

Durante el desarrollo del sistema de reconocimiento de mascarilla fueron necesarias pruebas del funcionamiento del prototipo con la finalidad de determinar la confiabilidad de este, tomado en cuenta que el mismo realiza un procesamiento de imágenes que se receptan por una camara web, de esta forma captura el rostro y esta es procesada mediante analisis de características del individuo en comparacion con los rostros almacenados dentro del sistema.

Tabla 8. Pruebas tecnicas del sistema

Pruebas	Descripcion
Prueba 1: usuario con mascarilla	Se realiza un prueba con una persona con mascarilla puesta, el sistema al detectar la mascarilla mostrara una etiqueta color verde.

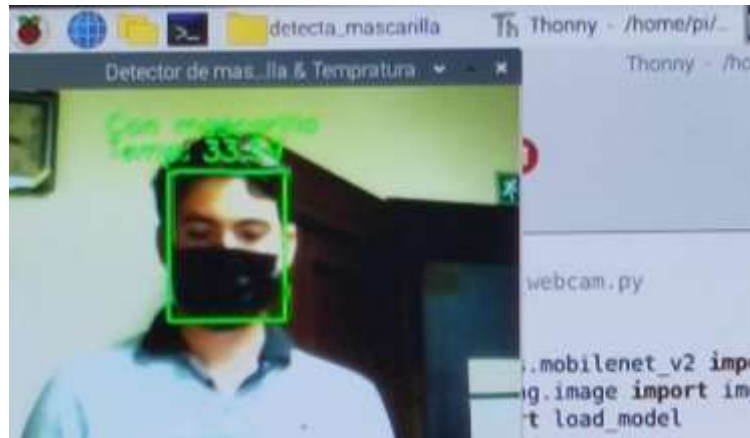


Figura 31. Prueba 1

Elaboracion: el autor

Así como la anterior prueba se la realiza con una persona pero que esta vez no lleve mascarilla, el sistema al detectar que no lleva mascarilla mostrara una etiqueta de color rojo y hara sonar una alerta.

Prueba 2: usuario sin mascarilla

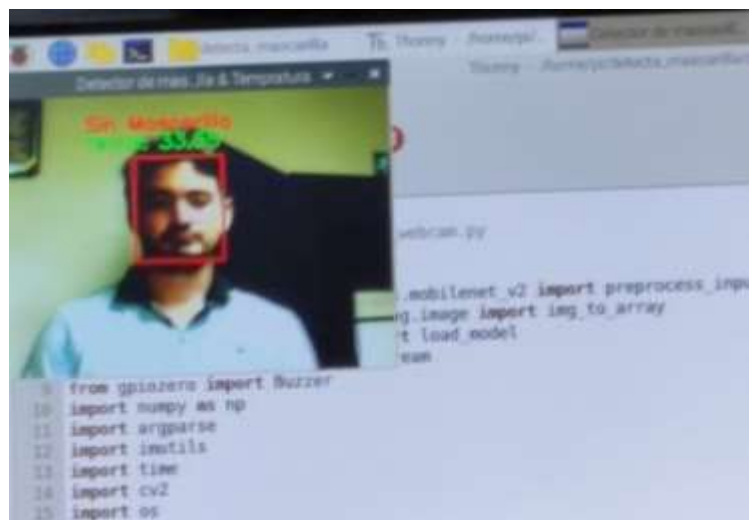


Figura 32. Prueba 2

Elaboracion: el autor

Prueba 3: test del sensor de temperatura

Aquí se le realiza una prueba al sensor de temperatura con un objeto caliente en este caso un cautín ya que no se encontró un sujeto con fiebre. A medida que el cautín va calentándose superando los 37.5° suena la alerta.



Figura 33. Prueba 3

Elaboración: el autor

Elaboración: el autor

3.8 Conclusiones

- Se utilizó una tarjeta Raspberry Pi 4 para el desarrollo del proyecto por sus características relacionadas a la visión artificial y al machine learning ya que cumple con la mayoría de las especificaciones tanto hardware como software para la implementación de algoritmos de procesamiento de imágenes.
- Se llegó a determinar que el número de iteraciones influye en el desempeño del modelo de la red neuronal, a medida que se aumenta el número de iteraciones se puede observar que el modelo incrementa en la precisión en la determinación de cada clase.
- El prototipo fue sometido a pruebas para verificar y monitorear su correcto funcionamiento.
- Por último, es importante mencionar que el tema de investigación queda abierto a una serie de mejoras, como la creación de una interfaz gráfica que sea amigable con el usuario o hacer que el sistema aparte de reconocer mascarillas reconozca rostros de personas y la etiqueten por su nombre como un sistema de control de ingreso de personal.

3.9 Recomendaciones

- Es recomendable trabajar con tipos de cámara que posean una buena resolución o un módulo de cámara original PiNoIR de la misma fundación Raspberry que está diseñada para la misma Raspberry y que esta está diseñada también para capturar imágenes en alta resolución.
- Para mejor precisión del sistema aumentar la base de datos de las personas con mascarilla y sin mascarilla.
- Es recomendable el uso de ventiladores y disipadores para la Raspberry Pi para que no se sobrecaliente ya que forzosamente el sistema tiene que estar activa por mucho tiempo.
- También es recomendable para estos tipos de trabajos usar una Raspberry Pi con una RAM de 4GB o mayor a que el procesamiento de imágenes y videos lo requieren.
- Es recomendable una adecuada instalación del sistema, tomando en cuenta como factor importante la ubicación de la cámara para evitar variaciones de luminosidad o factores externos que afecten su correcto funcionamiento.
- Comprobar si las conexiones y soldaduras de los componentes están efectuadas de manera correcta.

ANEXOS

Anexo 1

Instalación del sistema operativo

- Dirigirse a la pagina de Raspberrry www.raspberrypi.org en el apartado de ordenadores posteriormente encontraremos la opcion de Software encontraremos la ubicación del programa Raspberry Pi imager que nos servira descargar e instalar el software de manera oficial.

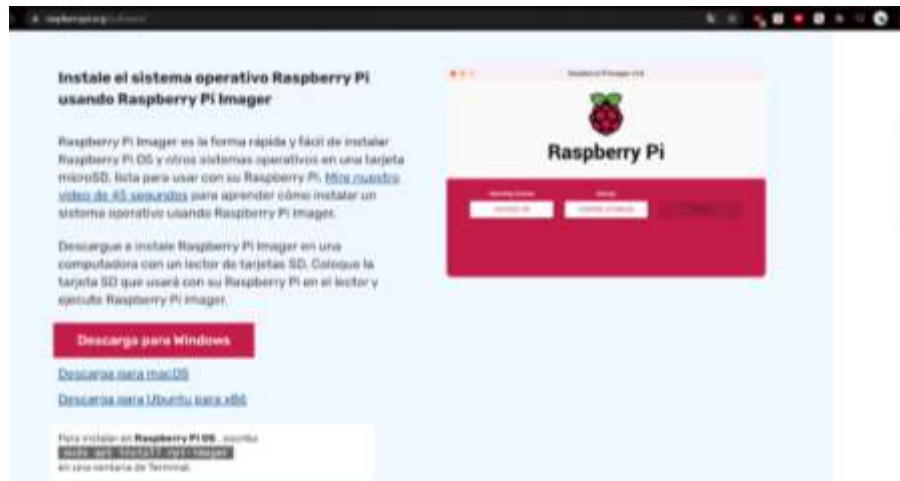
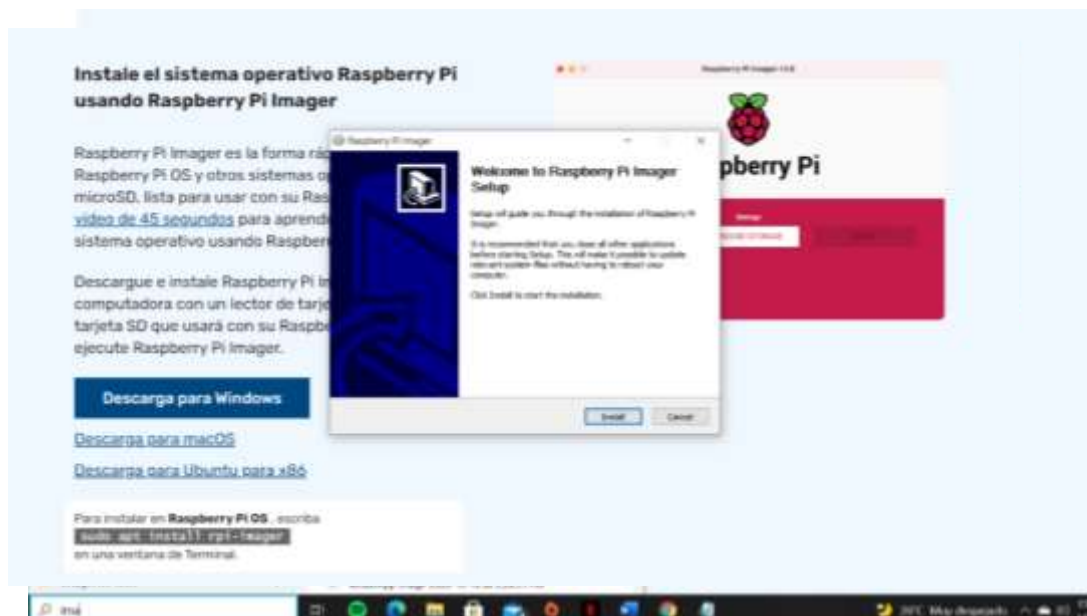


Figura 34. Página de inicio del sitio web www.raspberrypi.org

Elaboración: el autor

- Una vez descargado, instalamos el programa.



Instalación del programa descargado posteriormente

Elaboración: el autor

- Seguimos el proceso de instalacion y le damos finish.



Figura 35. Finalización de la instalación

Elaboración: el autor

- Una vez ya el software este descargado e instalado, lo ejecutamos y este nos aparecerá de la siguiente manera.



Figura 36. Raspberry Pi imager instalado correctamente

Elaboración: el autor

- una vez en esta pantalla, elegimos el software con el que trabajaremos en la opción CHOOSE OS como se mencionó anteriormente utilizaremos el Raspberry Pi OS, el instalador nos ofrece una versión estándar que requiere 1.2 GB de espacio más otras 2 opciones, una versión lite y una versión full en nuestro caso usaremos la versión full este ocupara un espacio de 2.4 GB.

Figura 37. Selección de tipo y arquitectura del SO.

Elaboración: el autor

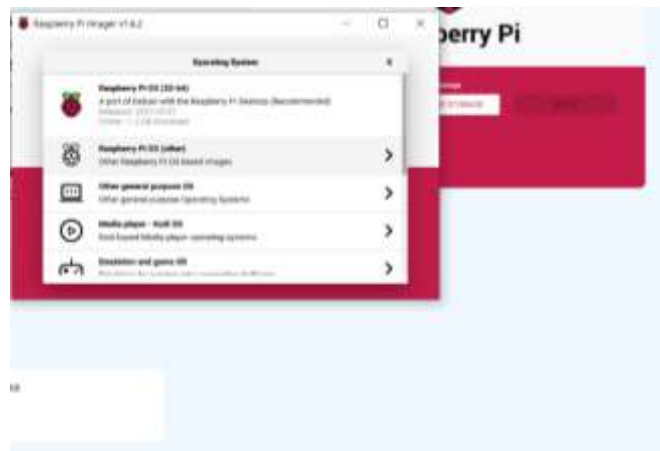
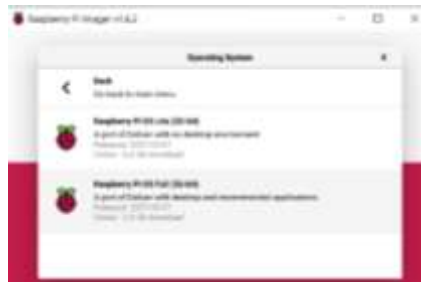


Figura 38. Selección de la versión del SO

Elaboración: el autor

- Una vez elegida la versión optima del SO seleccionamos el almacenamiento donde se guardara el S.O, en este caso en una Micro SD.

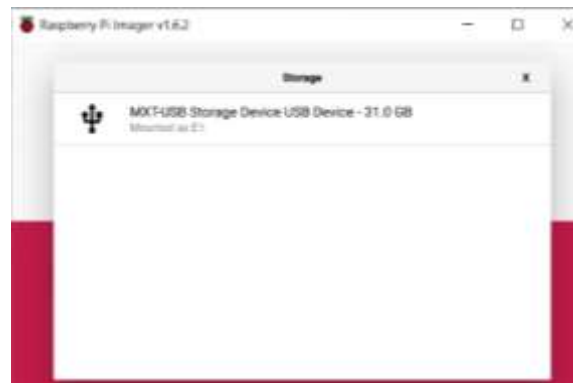


Figura 39. Selección del almacenamiento del SO. Elaborado por el autor

Elaboración: el autor

- Ya solo nos queda seleccionar a la opcion write, para que se instale el SO en la tarjeta Micro SD, una vez iniciamos el proceso nos saldra un aviso que se borrara todos los datos que esten actualmente en el almacenamiento asi que es recomendable que anteriormente la Micro SD este en blanco.



Figura 40. Escritura del SO en la SD. Elaborado por el autor

Elaboración: el autor



Figura 41. Iniciado la instalación de Raspberry Pi OS. Elaborado por el autor

Elaboración: el autor

- Una vez terminada la instalacion del sistema operativo en la MicroSD procedemos en insertarla en la Raspberry Pi y la encendemos para iniciar el equipo y si todo salio bien deberia salir igual que en la figura.



Figura 42. Pestaña de inicio Raspberry Pi Os.

Elaboración: el autor

Anexo 2

Instalar Python, Open CV

```

sudo raspi-config
df -h
sudo apt-get update && sudo apt-get upgrade
sudo apt-get install build-essential cmake pkg-config
sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
sudo apt-get install libxvidcore-dev libx264-dev
sudo apt-get install libgtk2.0-dev libgtk-3-dev
sudo apt-get install libatlas-base-dev gfortran
sudo apt-get install python2.7-dev python3-dev
cd ~
wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.3.0.zip
unzip opencv.zip
wget -O opencv_contrib.zip https://github.com/Itseez/opencv_contrib/archive/3.3.0.zip
unzip opencv_contrib.zip
install numpy
install tensorflow
install keras
cd ~/opencv-3.3.0/
mkdir build
cd build
cmake -D CMAKE_BUILD_TYPE=RELEASE \ -D CMAKE_INSTALL_PREFIX=/usr/local \
-D INSTALL_PYTHON_EXAMPLES=ON \
-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.3.0/modules \
-D BUILD_EXAMPLES=ON ..
Make
sudo make install
sudo ldconfig
ls -l /usr/local/lib/python2.7/site-packages/
cd ~/.virtualenvs/cv/lib/python2.7/site-packages/
ls -l /usr/local/lib/python3.5/site-packages/
cd /usr/local/lib/python3.5/site-packages/
sudo mv cv2.cpython-35m-arm-linux-gnueabi.so cv2.so
cd ~/.virtualenvs/cv/lib/python3.5/site-packages/
ln -s /usr/local/lib/python3.5/site-packages/cv2.so cv2.so

```

Anexo 3

Líneas de código del entrenamiento del detector de mascarilla

```
# import the necessary packages
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import AveragePooling2D
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.utils import to_categorical
from sklearn.preprocessing import LabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from imutils import paths
import matplotlib.pyplot as plt
import numpy as np
import argparse
import os

# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-d", "--dataset", required=True,
    help="path to input dataset")
ap.add_argument("-p", "--plot", type=str, default="plot.png",
    help="path to output loss/accuracy plot")
ap.add_argument("-m", "--model", type=str,
    default="mask_detector.model",
    help="path to output face mask detector model")
args = vars(ap.parse_args())

# Inicializamos con una tasa inicial de aprendizaje y el conteo de epocas para el entrenamiento
INIT_LR = 1e-4
EPOCHS = 20
BS = 32

#se toma la lista de imagenes grabadas en el dataset del directorio
# the list of data (i.e., images) and class images
print("[INFO] loading images...")
imagePaths = list(paths.list_images(args["dataset"]))
data = []
labels = []

# loop over the image paths
for imagePath in imagePaths:
```

```

# extract the class label from the filename
label = imagePath.split(os.path.sep)[-2]

# load the input image (224x224) and preprocess it
image = load_img(imagePath, target_size=(224, 224))
image = img_to_array(image)
image = preprocess_input(image)

# update the data and labels lists, respectively
data.append(image)
labels.append(label)

# convert the data and labels to NumPy arrays
data = np.array(data, dtype="float32")
labels = np.array(labels)

# perform one-hot encoding on the labels
lb = LabelBinarizer()
labels = lb.fit_transform(labels)
labels = to_categorical(labels)

(trainX, testX, trainY, testY) = train_test_split(data, labels,
                                                  test_size=0.20, stratify=labels, random_state=42)

# construct the training image generator for data augmentation
aug = ImageDataGenerator(
    rotation_range=20,
    zoom_range=0.15,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.15,
    horizontal_flip=True,
    fill_mode="nearest")

# load the MobileNetV2 network, ensuring the head FC layer sets are
# left off
baseModel = MobileNetV2(weights="imagenet", include_top=False,
                        input_tensor=Input(shape=(224, 224, 3)))

# construct the head of the model that will be placed on top of the
# the base model
headModel = baseModel.output
headModel = AveragePooling2D(pool_size=(7, 7))(headModel)
headModel = Flatten(name="flatten")(headModel)
headModel = Dense(128, activation="relu")(headModel)
headModel = Dropout(0.5)(headModel)
headModel = Dense(2, activation="softmax")(headModel)

# place the head FC model on top of the base model (this will become
# the actual model we will train)
model = Model(inputs=baseModel.input, outputs=headModel)

```

```

for layer in baseModel.layers:
    layer.trainable = False

# compile our model
print("[INFO] compiling model...")
opt = Adam(lr=INIT_LR, decay=INIT_LR / EPOCHS)
model.compile(loss="binary_crossentropy", optimizer=opt,
              metrics=["accuracy"])

# train the head of the network
print("[INFO] training head...")
H = model.fit(
    aug.flow(trainX, trainY, batch_size=BS),
    steps_per_epoch=len(trainX) // BS,
    validation_data=(testX, testY),
    validation_steps=len(testX) // BS,
    epochs=EPOCHS)

# make predictions on the testing set
print("[INFO] evaluating network...")
predIdxs = model.predict(testX, batch_size=BS)

# for each image in the testing set we need to find the index of the
# label with corresponding largest predicted probability
predIdxs = np.argmax(predIdxs, axis=1)

# show a nicely formatted classification report
print(classification_report(testY.argmax(axis=1), predIdxs,
                          target_names=lb.classes_))

# serialize the model to disk
print("[INFO] saving mask detector model...")
model.save(args["model"], save_format="h5")

# plot the training loss and accuracy
N = EPOCHS
plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, N), H.history["loss"], label="train_loss")
plt.plot(np.arange(0, N), H.history["val_loss"], label="val_loss")
plt.plot(np.arange(0, N), H.history["accuracy"], label="train_accuracy")
plt.plot(np.arange(0, N), H.history["val_accuracy"], label="val_accuracy")
plt.title("Training Loss and Accuracy")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="lower left")
plt.savefig(args["plot"])

```

Ejecutamos el código del entrenamiento para comprobar el comportamiento de la red neuronal en este caso la convolución de las imágenes se ejecutaron sin ningún inconveniente

1) En la imagen se muestra que el entrenamiento fue exitoso aquí en este paso solo hay que esperar a que el sistema termine su entrenamiento según a las épocas que uno le de al entrenamiento entre más épocas le de al entrenamiento más preciso será pero una desventaja es que va a tener más procesamiento y sería más tardío en este caso se le dio 20 épocas a este entrenamiento

Anexo 4

Líneas de código del detector de mascarilla

```
# importar paquetes
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model
from imutils.video import VideoStream
from gpiozero import Buzzer
import numpy as np
import argparse
import imutils
import time
import cv2
import os

import math
import board
import adafruit_mlx90614

def detect_and_predict_mask(frame, faceNet, maskNet):

    (h, w) = frame.shape[:2]
    blob = cv2.dnn.blobFromImage(frame, 1.0, (300, 300),
                                  (104.0, 177.0, 123.0))

    faceNet.setInput(blob)
    detections = faceNet.forward()

    # 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 > args["confidence"]:
            # compute the (x, y)-coordinates of the bounding box for
            # the object
            box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
            (startX, startY, endX, endY) = box.astype("int")

            # ensure the bounding boxes fall within the dimensions of
```



```

        # the frame
        (startX, startY) = (max(0, startX), max(0, startY))
        (endX, endY) = (min(w - 1, endX), min(h - 1, endY))

        # extract the face ROI, convert it from BGR to RGB channel
        # ordering, resize it to 224x224, and preprocess it
        face = frame[startY:endY, startX:endX]
        face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
        face = cv2.resize(face, (224, 224))
        face = img_to_array(face)
        face = preprocess_input(face)

        # add the face and bounding boxes to their respective
        # lists
        faces.append(face)
        locs.append((startX, startY, endX, endY))

    if len(faces) > 0:
        # for faster inference we'll make batch predictions on all
        # faces at the same time rather than one-by-one predictions
        # in the above `for` loop
        faces = np.array(faces, dtype="float32")
        preds = maskNet.predict(faces, batch_size=32)

    return (locs, preds)

# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-f", "--face", type=str,
                default="face_detector",
                help="path to face detector model directory")
ap.add_argument("-m", "--model", type=str,
                default="mask_detector.model",
                help="path to trained face mask detector model")
ap.add_argument("-c", "--confidence", type=float, default=0.5,
                help="minimum probability to filter weak detections")
args = vars(ap.parse_args())

# load our serialized face detector model from disk
print("[INFO] Cargando el modelo de deteccion facial...")
prototxtPath = os.path.sep.join([args["face"], "deploy.prototxt"])
weightsPath = os.path.sep.join([args["face"],
                                "res10_300x300_ssd_iter_140000.caffemodel"])
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

# load the face mask detector model from disk
print("[INFO] Cargando el modelo de deteccion de mascarilla...")
maskNet = load_model(args["model"])

# Inicializando sensor de temperatura
i2c = board.I2C()

```

```

# inicializa el buzzer
buzzer= Buzzer(17)

# Temperatura maxima permitida
tmpmax=37.50

print("[INFO] Iniciando el video stream...")
vs = VideoStream(src=0).start()
time.sleep(2.0)

# loop over the frames from the video stream
while True:
    "# grab the frame from the threaded video stream and resize it"
    # to have a maximum width of 400 pixels
    frame = vs.read()
    frame = imutils.resize(frame, width=500)

    # face mask or not
    (locs, preds) = detect_and_predict_mask(frame, faceNet, maskNet)

# capturar temperatura
mlx = adafruit_mlx90614.MLX90614(i2c)
to=round(mlx.object_temperature + 4, 2)

buzzer.off()

# loop over the detected face locations and their corresponding
# locations
for (box, pred) in zip(locs, preds):
    # unpack the bounding box and predictions
    (startX, startY, endX, endY) = box
    (mask, withoutMask) = pred

    "# determine the class label and color we'll use to draw"
    # the bounding box and text

    if mask > withoutMask:
        label_1 = "Con mascarilla"
        label_2 = "Temp: "+ str(to)
        color_1 = (0, 255, 0)
        color_3 = (0, 255, 0)
        if to > tmpmax:
            color_2 = (0, 0, 255)
            color_3 = (0, 0, 255)
            buzzer.on()
        else:
            color_2 = (0, 255, 0)
    else:
        label_1 = "Sin Mascarilla"

```

```

        label_2 = "Temp: "+ str(to)
        color_1 = (0, 0, 255)
        color_3 = (0, 0, 255)
        buzzer.on()
        if to > tmpmax:
            color_2 = (0, 0, 255)
            buzzer.on()
        else:
            color_2 = (0, 255, 0)
    # display the label and bounding box rectangle on the output
    # frame
    cv2.putText(frame, label_1, (startX-50, startY - 30),
                cv2.FONT_HERSHEY_SIMPLEX, 0.7, color_1, 2)
    cv2.putText(frame, label_2, (startX-50, startY - 10),
                cv2.FONT_HERSHEY_SIMPLEX, 0.7, color_2, 2)
    cv2.rectangle(frame, (startX, startY), (endX, endY), color_3, 2)

# show the output frame
cv2.imshow("Detector de mascarilla & Temperatura", frame)
key = cv2.waitKey(1) & 0xFF

"# if the `q` key was pressed, break from the loop"
if key == ord("q"):
    break

# do a bit of cleanup
cv2.destroyAllWindows()
buzzer.off()
vs.stop()

```

Bibliografía

- El Universo . (29 de Febrero de 2020). Ecuador confirma primer caso de coronavirus.
- APD. (04 de Marzo de 2019). *¿Qué es Machine Learning y cómo funciona?* Obtenido de APD:
<https://www.apd.es/que-es-machine-learning/>
- Capacitarte.org. (14 de Abril de 2020). *¿Qué es y para qué sirve Python?* Obtenido de
capacitarte.org : <https://www.capacitarte.org/blog/nota/que-es-y-para-que-sirve-python>
- CCQ. (2020). *¿Como reducir la exposicion y riesgo de contagio de COVID-19 en trabajadores y clientes? PROTOCOLO DE BIOSEGURIDAD PARA LOCALES COMERCIALES*, 30.
- COE NACIONAL. (8 de Junio de 2021). Obtenido de CoronaVirusEcuador.com:
<https://www.coronavirusecuador.com/datos-provinciales/>
- Corona, L., Abarca, G., & Mares, J. (2019). Sensores Y Actuadores Aplicaciones Con Arduino. *Sensores Y Actuadores*, 17,18.
- cristhian, j. (22 de Enero de 2019). *Cómo funciona una cámara web*. Obtenido de
247tecno.com: <https://247tecno.com/como-funciona-una-camara-web/>
- Delgado, A. (21 de Noviembre de 2020). *¿Qué es Raspberry Pi y para qué sirve?* Obtenido de
GEEKNETIC: <https://www.geeknetic.es/Raspberry-Pi/que-es-y-para-que-sirve>
- Díaz, D. N. (2016). Obtenido de Tecnologia Biométrica: <http://www.alimenco.net/>
- El expreso. (23 de Marzo de 2020). Las pandemias más letales de la historia antes de la llegada del coronavirus.
- Europapress. (15 de Marzo de 2020). *"¿A qué temperatura se considera que una persona tiene fiebre?* Obtenido de Europapress: <https://www.europapress.es/sociedad/noticia-temperatura-considera-persona-tiene-fiebre-20200315170834.html>
- GEPAR. (Marzo de 2007). *Diseño de un sistema biométrico*. Obtenido de Revista Facultad de Ingeniería : <http://www.scielo.org.co/pdf/rfiua/n39/n39a02.pdf>
- Inca, G. P., & Inca, A. C. (2020). Evolución de la enfermedad por coronavirus (COVID-19) en Ecuador. *La Ciencia al Servicio de la Salud y la Nutrición*, 6.
- Julian, G. (21 de Enero de 2016). *Las redes neuronales: qué son y por qué están volviendo*. Obtenido de xataka: <https://www.xataka.com/robotica-e-ia/las-redes-neuronales-que-son-y-por-que-estan-volviendo>
- kimaldi. (31 de Octubre de 2017). *Reconocimiento facial*. Obtenido de kimaldi:
https://www.kimaldi.com/blog/biometria/reconocimiento_facial/

- knepublishing. (2020). *PROCESAMIENTO DE IMÁGENES PARA LA IDENTIFICACIÓN DE PERSONAS COMO SISTEMA DE SEGURIDAD EN ZONAS DOMICILIARIAS*. *Procesamiento de imágenes para identificación de personas como sistema de seguridad en zonas domiciliarias*. Obtenido de knepublishing: <https://knepublishing.com/index.php/KnE-Engineering/article/view/6233/11605>
- Llamas, L. (16 de Julio de 2016). *REPRODUCIR SONIDOS CON ARDUINO Y UN BUZZER PASIVO O ALTAVOZ*. Obtenido de Luis llamas.com: <https://www.luisllamas.es/reproducir-sonidos-arduino-buzzer-pasivo-altavoz/>
- Lopez, C. R. (Mayo de 2017). *Reconocimiento facial mediante vision artificial*. Obtenido de bibing.: <http://bibing.us.es/proyectos/abreproy/11025/fichero/Reconocimiento+Facial+Median+Visi%C3%B3n+Artificial+%252FPFC+Completo%252FPFC+Reconocimiento+Facial+Mediante+Visi%C3%B3n+Artificial.pdf>
- OMS. (27 de Abril de 2020). *Preguntas y respuestas sobre la enfermedad por coronavirus (COVID-19)*. Obtenido de Organizacion Mundial de la Salud : <https://www.int/es/emergencias/diseases/novel-coronavirus-2019/advice-for-public/q-a-coronaviruses>
- ORGANIZACION MUNDIAL DE LA SALUD. (01 de DICIEMBRE de 2020). *Mask use in the context of COVID-19*. Obtenido de Organizacion Mundial de la Salud: [file:///D:/WHO-2019-nCov-IPC_Masks-2020.5-eng%20\(1\).pdf](file:///D:/WHO-2019-nCov-IPC_Masks-2020.5-eng%20(1).pdf)
- ORGANIZACION MUNDIAL DE LA SALUD. (2020). *Panel de control de coronavirus (COVID-19) de la OMS*. Obtenido de Who Coronavirus: <https://covid19.who.int/>
- ORGANIZACION MUNDIAL DE LA SAULD. (1 de DICIEMBRE de 2020). *Coronavirus disease (COVID-19) advice for the public: When and how to use masks*. Obtenido de <https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public/when-and-how-to-use-masks>
- Pérez, J. P., & Merino, M. (2015). *Definicion.de*. Obtenido de Definicion de prototipo: <https://definicion.de/prototipo/>
- RAE. (16 de Diciembre de 2019). *Contagio*. Obtenido de Real academia española: <https://dle.rae.es/contagio>
- SEGURILATAM. (05 de Agosto de 2020). *Los cambios en los sistemas de control de accesos ante la COVID-19*. Obtenido de SEGURILATAM: <https://www.segurilatam.com/tecnologias-y-servicios/control-de-accesos-y->

antintrusion/los-cambios-en-los-sistemas-de-control-de-accesos-ante-la-covid-19_20200805.html

Shudhanshu, C., & Ankit, V. (2020). Front Line Protection against Covid-19. *International Research Journal of Engineering and Technology*.

SISCA. (15 de Febrero de 2015). *¿Qué es un control de acceso?* Obtenido de SISCA CCTV: <http://sisca.co/que-es-un-control-de-acceso/>

TOALA, A. R., & OMAR, M. S. (2020). *Desarrollo del prototipo de una Aplicación móvil que permita autoevaluar y reportar posibles casos de COVID-19 en la Unidad Educativa Instituto Británico (tesis de grado de sistemas computacionales)*. Guayaquil .

TORRES, A. K., & TRIANA, C. D. (2020). *IMPLEMENTACIÓN DE UN SISTEMA DE CONTROL DE ACCESO BASADO EN ARDUINO CON MONITOREO DE TEMPERATURA CORPORAL PARA PREVENIR CONTAGIOS DE COVID-19 Y AUMENTAR LA SEGURIDAD EN BLOQUES DE VIVIENDAS CERRADOS DEL GUASMO SUR DE GUAYAQUIL (tesis de grado)*. Guayaquil.