UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y COMUNICACIONES

Tema:

SISTEMA DE CONTROL DE ACCESO POR MEDIO DE RECONOCIMIENTO FACIAL CON USO DE MASCARILLA Y MONITOREO DE TEMPERATURA

Trabajo de Titulación Modalidad: Proyecto de Investigación, presentado previo a la obtención de título de Ingeniera en Electrónica y Comunicaciones.

ÁREA: Electrónica

LÍNEA DE INVESTIGACIÓN: Tecnología de la información y sistemas de control.

AUTOR: Verónica de los Ángeles Untuña Toalombo

TUTOR: Ing. Santiago Altamirano Meléndez, Mg.

Ambato – Ecuador

marzo 2022

APROBACIÓN DEL TUTOR

En calidad de tutor del Trabajo de Titulación con el tema:" SISTEMA DEL CONTROL DE ACCESO POR MEDIO DEL RECONOCIMIENTO FACIAL CON USO DE MASCARILLA Y MONITOREO DE TEMPERATURA", desarrollado bajo la modalidad Proyecto de Investigación por la señorita Verónica de los Ángeles Untuña Toalombo, estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que la estudiante ha sido tutorada durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 15 del Reglamento para obtener el Título de Tercer Nivel, de Grado de la Universidad Técnica de Ambato, y el numeral 7.4 del respectivo instructivo.

Ambato, marzo 2022.

Firmado electrónicamente por:
SANTIAGO MAURICIO ALTAMIRANO MELENDEZ

……..

Ing. Santiago Altamirano Meléndez, Mg.

TUTOR

AUTORÍA

El presente Proyecto de Investigación titulado: SISTEMA DE CONTROL DE ACCESO POR MEDIO DE RECONOCIMIENTO FACIAL CON USO DE MASCARILLA Y MONITOREO DE TEMPERATURA es absolutamente original, auténtico y personal. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, marzo 2022.

Verónica de los Ángeles Untuña Toalombo
C.C 180540008-0
AUTORA

APROBACIÓN TRIBUNAL DE GRADO

En calidad de par calificador del Informe Final del Trabajo de Titulación presentado por la señorita Verónica de los Ángeles Untuña Toalombo , estudiante de la Carrera de Ingeniería en Electrónica y Comunicaciones, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado SISTEMA DEL CONTROL DE ACCESO POR MEDIO DEL RECONOCIMIENTO FACIAL CON USO DE MASCARILLA Y MONITOREO DE TEMPERATURA, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 17 del Reglamento para obtener el Título de Tercer Nivel, de Grado de la Universidad Técnica de Ambato, y al numeral 7.6 del respectivo instructivo. Para cuya constancia suscribimos, conjuntamente con la señora Presidenta del Tribunal.

Ambato, marzo 2022.

Firmado electrónicamente por:
ELSA PILAR URRUTIA

……..

Ing. Pilar Urrutia, Mg.

PRESIDENTA DEL TRIBUNAL

Firmado electrónicamente por:
MARIO GEOVANNY GARCIA CARRILLO

……

Ing. Mario García

PROFESOR CALIFICADOR

Firmado electrónicamente por:
ANDREA PATRICIA SANCHEZ ZUMBA

…….....

Ing. Andrea Sánchez

PROFESOR CALIFICADOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consultas y procesos de investigación.

Cedo los derechos de mi Trabajo de Titulación en favor de la Universidad Técnica de Ambato, con fines de difusión pública. Además, autorizo su reproducción total o parcial dentro de las regulaciones de la institución.

Ambato, marzo 2022.

Verónica Untuña

Verónica de los Ángeles Untuña Toalombo

C.C 180540008-0

AUTORA

DEDICATORIA

El presente trabajo dedico primeramente a Dios por darme la fortaleza y sabiduría necesaria para seguir adelante he ir cumpliendo con todas mis metas.

A mi madre Rosa Toalombo que ha estado conmigo en todo momento brindándome su apoyo incondicional y a mis hermanos Jairo, Moisés y Mayra que me impulsaron para seguir adelante.

A mis amigos y amigas por el apoyo que me han brindado y permitirme aprender más de la vida.

Verónica de los Ángeles Untuña Toalombo

AGRADECIMIENTO

Agradezco a Dios por cuidar de mi en todo momento, por brindarme su amor incondicional y guiarme en cada etapa de mi vida.

A mi madre por siempre estar al pendiente de mí, por enseñarme que cada situación que suceda siempre hay que seguir adelante.

Al Ingeniero Santiago Altamirano por orientarme en el desarrollo de mi proyecto de titulación.

A los docentes que me han impartido de sus conocimientos en toda la carrera universitaria.

A mis amigos quienes han formado parte de la carrera universitaria y han estado conmigo en los buenos y malos momentos.

Verónica de los Ángeles Untuña Toalombo

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN EJECUTIVO

El mundo atraviesa por una crisis sanitaria que ha afectado a multitud de personas, el número de contagios ha ido incrementando paulatinamente debido al incumplimiento de las normas de bioseguridad por la ciudadanía.

El presente trabajo de titulación describe el desarrollo de un sistema de control por medio de reconocimiento facial con uso de mascarilla y monitoreo de temperatura, donde se tiene un control de los datos adquiridos los cuales son almacenados para posteriormente ser analizados. El sistema de control está integrado por tres etapas siendo estas: adquisición de datos por medio del sensor de temperatura y una cámara web para posteriormente ser enviados a la tarjeta de desarrollo Raspberry Pi de modo que recibe la información y realiza el proceso de reconocimiento del uso de mascarillas por medio de redes neuronales convolucionales, las cuales detectan si una persona está utilizando o no la mascarilla, la información es enviada por medio del protocolo de red MQTT que transporta mensajes entre dispositivos, para la visualización de los datos se utiliza una app móvil, la cual es enlazada con ThingSpeak que es una plataforma de Internet de las cosas, en caso que la temperatura supere su valor nominal se enviará una señal de alerta a la aplicación de mensajería de Telegram.

Este proyecto se orienta al control del ingreso de las personas ya sean en lugares públicos o privados, evitando el contacto directo con el personal de control, y monitorizando una temperatura correcta para el respectivo ingreso.

Palabras Claves: Reconocimiento, Open CV, redes neuronales convolucionales, Raspberry Pi.

ABSTRACT

The world is going through a health crisis that has affected many people, the number of infections has been gradually increasing due to non-compliance with biosafety regulations by citizens.

The present degree work describes the development of a control system through facial recognition with the use of a mask and temperature monitoring, where there is control of the acquired data, which are stored for later analysis. The control system is made up of three stages, these being: data acquisition through the temperature sensor and a webcam to later be sent to the Raspberry Pi development card so that it receives the information and performs the process of recognition of use. of masks through convolutional neural networks, which detect whether or not a person is wearing the mask, the information is sent through the MQTT network protocol that transports messages between devices, a mobile app is used to view the data which is linked to ThingSpeak, which is an Internet of things platform, in case the temperature exceeds its nominal value, an alert signal will be sent to the Telegram messaging application.

This project is aimed at controlling the entry of people, whether in public or private places, avoiding direct contact with control personnel, and monitoring the correct temperature for the respective entry.

Keywords: Recognition, Open CV, convolutional neural networks, Raspberry Pi.

CAPÍTULO I

MARCO TEÓRICO

1.1. Tema de investigación

SISTEMA DE CONTROL DE ACCESO POR MEDIO DE RECONOCIMIENTO FACIAL CON USO DE MASCARILLA Y MONITOREO DE TEMPERATURA.

1.2. Antecedentes Investigativos

La investigación fue realizada en diferentes repositorios de Universidades, además, se recolectó información de artículos publicados en diferentes revistas científicas que abordan el monitoreo del uso de mascarilla, obtención de temperatura y almacenamiento de datos.

En el año 2019, Gutiérrez Segales Juan Pablo en su tesis "Implementación de un prototipo de una red inalámbrica de sensores biomédicos, para la adquisición y almacenamiento de datos, usando cloud computing, para pacientes en casa" de la Universidad Nacional de San Agustín de Arequipa-Perú describe un prototipo el cual manipula nodos, un microcontrolador y un sensor para la respectiva adquisición de datos, donde emplea programación del Node MCU mediante el entorno de desarrollo Arduino basada en un lenguaje de programación C++, el cual envía los datos de los sensores al servidor ThingSpeak, que es una API de código abierto a través de un módem para su enrutamiento, se utilizó el protocolo MQTT para la comunicación M2M entre sensores por el bajo consumo de ancho de banda, obteniendo como resultado el control preciso de las señales biométricas a monitorear, donde cada dato de los sensores se muestran en tiempo real con sus respectivas gráficas, considerado un prototipo de bajo costo para el monitoreo remoto [1].

En el año 2020, Rafizah Ab Rahman, Ummi Raba'ah Hashim y Sabrina Ahmad en su artículo de investigación denominado "IoT based temperature and humidity monitoring framework" de la Universiti Teknikal Malaysia Melaka en la revista Boletín de Ingeniería Eléctrica e Informática (BEEI), desarrollaron la monitorización de la temperatura y la humedad de un centro de datos en tiempo real, donde la lectura de los sensores se envía a la plataforma IoT de AT&T M2X a través de la API RESTful, para ser registradas y almacenadas donde; se indican los gráficos en tiempo real de los sensores de temperatura y humedad. La plataforma M2x otorga la integración al servicio IFTTT (If this, then that) el cual permite conectar diferentes servicios, cada sensor conectado al dispositivo se identifica por su ID y clave API, el servicio M2X admite al usuario crear soluciones de IoT sin administrar la infraestructura de almacenamiento obteniendo como resultado la detección de cambios de temperatura, humedad y el envío automático de una notificación por medio de correo electrónico y servicio de mensajes [2].

En el año 2020, Montesdeoca Erik en su tesis "Implementación de un Sistema de Reconocimiento del Uso de mascarillas como medida de precaución contra el covid19 usando Deep Learning" de la Universidad Técnica de Machala, describe un sistema que detecta a las personas que usan y no usan mascarilla, donde realizaron y entrenaron una red neuronal mediante OpenCV, Python y TensorFlow con varias imágenes que clasificaron en dos grupos de los cuales son: personas con mascarilla y sin mascarilla, donde aplicó la técnica MobileNetV2, también efectuó una página web obteniendo como resultado la examinación del uso de mascarilla de las personas por medio de una cámara IP en tiempo real que emite la información a la red neuronal mostrando los resultados en el navegador web [3].

En el año 2021, Pereira Alexandre, Donadon Thiago y Oliveira Fabio en su artículo científico titulado "Aplicación de inteligencia artificial para monitorear el uso de mascarillas de protección" publicado en la Revista Científica General José María Córdova aplicaron un algoritmo Haar Cascade disponible en un repositorio OpenCV para detectar patrones en imágenes el cual consta de tres capas 1) rastrea la imagen, 2) utiliza un clasificador boosting para la selección de las características y 3) conecta en cascada a los

clasificadores, utilizaron la conexión de la aplicación a la base de datos Atlas de MongoDB conocida como NoSQL, donde la biblioteca PyMongo ayuda a la influencia recíproca entre el lenguaje de programación Python y la base de datos MongoDB en la cual obtuvieron 194 imágenes que se dividieron en dos clases: la primera clase se refería a personas con mascarilla y la segunda clase a personas sin mascarilla, emplearon el algoritmo Haar Cascade mediante medidas de precisión, revocación y f_β-score, obteniendo como resultado la localización de las personas que no usaban mascarilla con una exactitud del 63%, registrando la información en una base de datos, y así efectuar medidas de protección para combatir y prevenir el contagio [4].

En el año 2021, Minh Long Hoang, Marco Carratú, Vicenzo Paciello y Antonio Pietrosanto en su artículo científico titulado "Body temperatura-Indoor Condition Monitor and Activity Recognition by MEMS Accelerometer Base don IoT-Alert System for People in Quarantine Due to Covid-19", de la Universidad of Salerno de Italia en la revista Sensors, exponen un dispositivo portátil colocado en la muñeca el cual monitorea datos de temperatura y condiciones interiores que son transferidos al teléfono inteligente a través de Bluetooth y Wifi, donde los usuarios pueden elegir cualquier red inalámbrica; se permite la conexión de sensores utilizando el protocolo MQTT por medio de Arduino de código abierto para la transferencia de mensajes entre la nube y dispositivos portátiles, donde se utilizó un M5StickC alimentado por ESP32, el cual está constituido por un termómetro infrarrojo MLX90614 que se utiliza para la temperatura corporal y un ENV HAT para la temperatura interior y la humedad, obteniendo como resultado que el sistema monitoreó exitosamente a la persona en cuarentena ya que el dispositivo proporcionó información apropiada para el usuario y a sus familiares, siendo un dispositivo de bajo costo y tamaño pequeño equipado para supervisar la salud del paciente [5].

1.2.1. Contextualización del problema

Hoy por hoy la población está enfrentando una pandemia causada por el coronavirus (Covid-19), aproximadamente 215 países y territorios en todo el mundo se han visto afectados por la enfermedad, los síntomas notificados con más frecuencia son fiebre y tos

seca. A nivel mundial, el número de muertes por (Covid-19) es de 5,521,807 personas, Estados Unidos uno de los países con mayor número de muertes en un total de 863,896 personas. En Ecuador el número de muertes que afecta al país por la pandemia es de 33,713 personas al 12 de enero de 2022 [6]. En cambio, el número de casos confirmados de coronavirus (Covid-19) hasta el 12 de enero del 2022 es de 559,950 casos, siendo Ecuador uno de los países más afectados por la pandemia en Latinoamérica, se encuentra en onceavo lugar; cada día las cifras siguen aumentando de manera alarmante, ocasionando que se tome las medidas de prevención [7].

La afluencia de personas fomenta el descontrol de las medidas de bioseguridad ya sea en espacios públicos o privados por consiguiente se han tomado normas de prevención que son necesarias por el incremento de transmisión del virus. Según la Organización Mundial de la Salud (OMS) la mejor forma de prevenir el aumento del virus Covid-19 son el distanciamiento social, vacuna contra el Covid-19, lavado de manos, uso de alcohol y mascarillas para disminuir el número de personas contagiadas [8]. Por esta razón la utilización de la tecnología es de gran ayuda debido a que en tiempo real se puede monitorear el uso correcto de la mascarilla y una temperatura estable evitando el contacto físico entre el usuario y la persona responsable del área de control.

Por lo cual, es necesario un sistema que controle la temperatura y el uso adecuado de la mascarilla por medio de fotografías capturadas por la cámara, enviando los datos a una aplicación de mensajería como telegram, además, se visualizará la información en una plataforma web y aplicación móvil, para ayudar a controlar las medidas de bioseguridad, en caso de que alguna persona tenga la temperatura alta o no esté usando correctamente la mascarilla, se emitirá un sonido en señal de alerta por lo tanto, no podrá ingresar a un lugar establecido para proteger la salud de los trabajadores, clientes y visitantes.

Esta investigación está orientada a solucionar la falta de control y monitorización de los ciudadanos que utilizan incorrectamente la mascarilla, de manera que, se debe verificar una temperatura estable, para que pueda ingresar a un cierto lugar, por consiguiente, el personal de control encargado no tenga que estar al pendiente haciendo cumplir el uso

adecuado de la mascarilla y control de la temperatura ya que se podrá realizar este proceso mediante el uso automático del sistema.

1.2.2. Fundamentación teórica

1.2.2.1. Sistemas de Control

Un sistema de control automático puede simbolizarse a través de bloques, de modo que se interconectan varios elementos dando como resultado a un sistema o proceso $g(t)$, $r(t)$ viene a ser la señal de entrada para conseguir una salida $y(t)$, cómo se pude observar en la figura 1 [9].

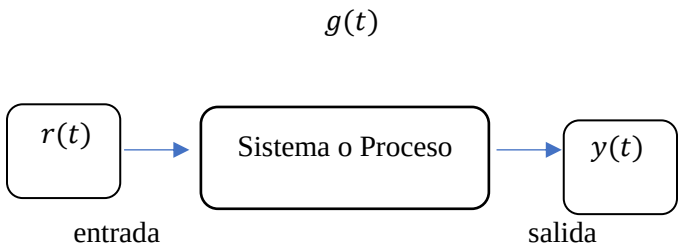


Figura 1.-Sistema de Control [9].

1.2.2.2. Neuronas

Las neuronas son los componentes básicos del sistema nervioso, las cuales están compuestas por un cuerpo celular, un axón y las dentritas, las neuronas reciben, procesan y transmiten la información a otras células del cuerpo humano [10].

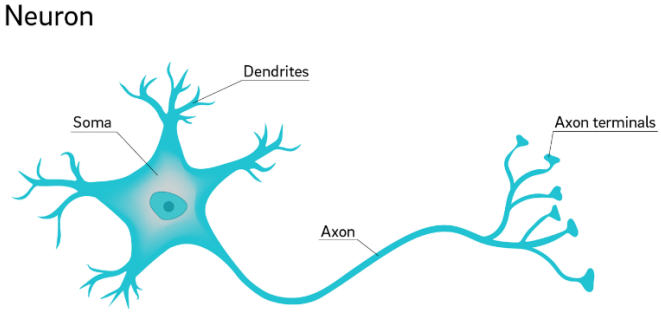


Figura 2.-Estructura de una neurona [10].

1.2.2.3. Redes Neuronales

Una red neuronal es un modelo de computación que se asimila a la forma en que el cerebro procesa la información, estas redes van interconectadas entre sí y se relacionan con objetos del exterior; una red puede ser simple o profunda dependiendo del número de neuronas por las que está compuesta la red [11].

Una neurona está conformada por tres componentes, los cuales son: peso(w) que establecen el vector de entrada, umbral interno (∅) magnitud de compensación que afecta al nodo de salida y la función de activación en la señal de salida efectúa una operación matemática. El funcionamiento de una neurona depende de la señal de entrada y la información que se desea procesar, los valores de entrada y peso son multiplicados obteniendo como resultado la función de ponderación que es la combinación lineal de las entradas y los pesos [11].

$$X * W^t = (x_1, x_2, \dots x_n) * \begin{pmatrix} w_1 \\ w_2 \\ . \\ . \\ w_n \end{pmatrix} = \sum_{i=1} x_i * w_i \qquad (1)$$

Donde:

X: vector entrada

W^t: vector peso

$x_1, x_2, \dots x_n$: valores de la señal de entrada

$w_1, w_2, \dots w_n$: valores del peso

Se aplica una función de activación:

$$Y = \varphi\left(\sum_{i=1} x_i * w_i\right) \qquad (2)$$

Donde:

Y: salida

φ: función de activación

x_i: valor de entrada

w_i:peso asociado

Este resultado, se obtiene a la salida, la cual puede ser la entrada nueva de una neurona, o también puede ser el resultado final, esto depende de cuántas capas ocultas esté conformada la red neuronal, en la figura 3, se observa el comportamiento de una neurona artificial [11].

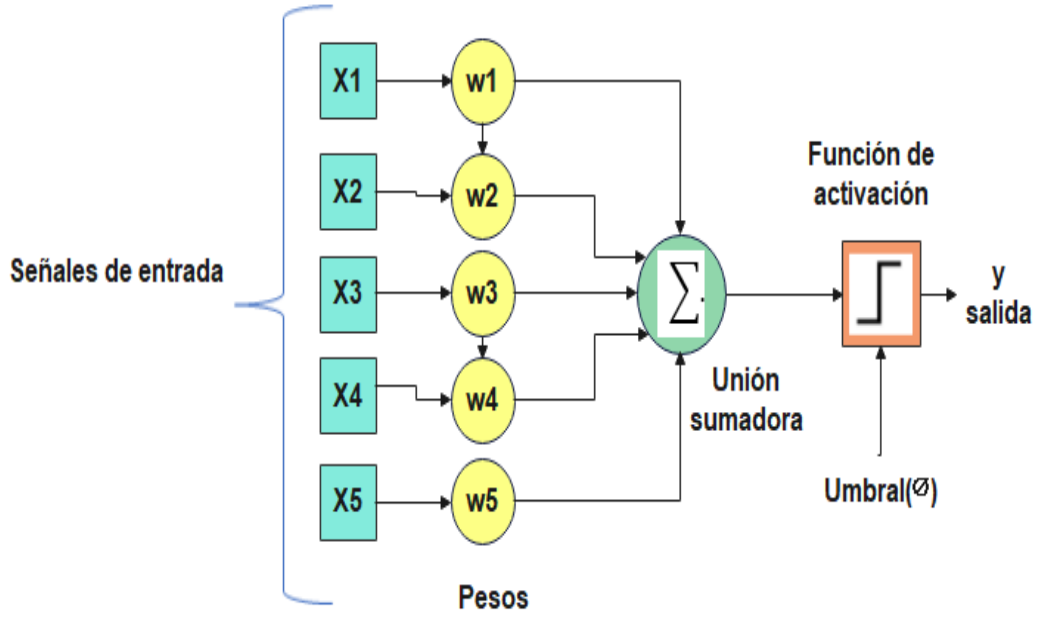


Figura 3.- Esquema de una neurona artificial [11].

La función de activación es el resultado de la información que transmite la combinación lineal de las entradas y los pesos. En la tabla 1, se observan las funciones de activación más utilizadas.

Tabla 1.-Funciones de activación [11].

Función de activación	Ecuación
Función escalón	$\emptyset(x) = \begin{cases} 0 \; si \; x < 0 \\ 1 \; si \; x \geq 0 \end{cases}$ (3)
Función Sigmoidal	$\emptyset(x) = \frac{1}{1 + e^{-x}}$ (4)
Función Rectificadora	$\emptyset(x) = max\{0, x\}, siendo \; x \geq 0$ (5)
Función Tangente Hiperbólica	$\emptyset(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$ (6)
Función de Base Radial	Gausianas, multicuadráticas, multicuadráticas inversas

Elaborado por: Investigadora

Las redes neuronales se clasifican en dos tipos:

-Según su arquitectura: Se trata de la disposición y conexión de las neuronas, una red se diferencia por el número y tipo de capas, estas redes se dividen en:

Redes monocapa. **–** está formada por una capa de neuronas, que conmutan señales hacia el exterior como la Red de Hopfield [12].

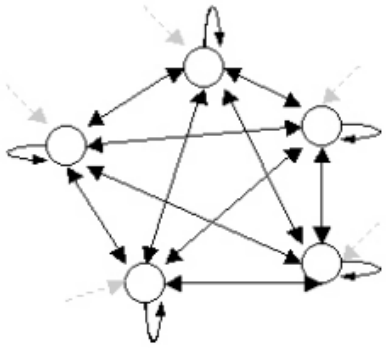


Figura 4.- Arquitectura de la Red de Hopfield [12].

-Redes multicapa. - compuesto por varias capas: capa de entrada, capa oculta y capa de salida, las mismas que están conectadas mediante neuronas o nodos [13].

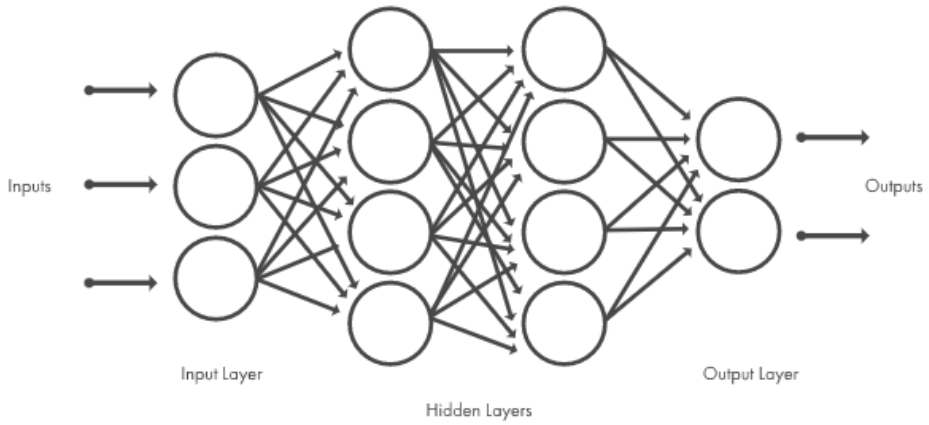


Figura 5.- Arquitectura de una Red Neuronal Multicapa [13].

La capa de entrada recibe los datos que ingresan del exterior de la red, seguidamente, se dirige a la capa oculta la cual está conformada por varias capas internas donde, se procesa la información y, por último, consigue llegar a la capa de salida donde, se obtiene la respuesta final enviada al exterior de la red.

La red neuronal es conveniente para el reconocimiento de patrones, esta red emplea elementos que operan en paralelo y adopta varias capas de procesamiento, primordialmente las redes neuronales son usadas para funciones de predicción y clasificación [13].

1.2.2.3.1. Entrenamiento de las Redes Neuronales

Es el proceso que permite configurar para obtener las salidas requeridas de acuerdo a las entradas emitidas por medio de las conexiones, para conseguir este objetivo, se toma en consideración los pesos anteriormente conocidos; y otro método, se basa en las técnicas de retroalimentación y patrones de aprendizaje que varía los pesos hasta conseguir el apropiado. Los pesos convergen gradualmente hacia los valores que logran que cada entrada ocasione el vector de salida requerido, esto sucede durante el proceso de entrenamiento [14].

-Según el aprendizaje: Se trata del entrenamiento de red con patrones.
Aprendizaje supervisado. – se alteran los pesos, para ajustar la entrada a la salida.
Aprendizaje no supervisado. - suministrar a la red los patrones de entrada y no de salida.
Aprendizaje Híbrido. - no se provee los patrones, solo se informa la respuesta de falla o acierto [12].

En la tabla 2, se presentan las distintas aplicaciones que puede emplear las redes neuronales y en la tabla 3, se encuentran las ventajas y desventajas.

Tabla 2.-Aplicación de las Redes Neuronales [12].

Tipo de Aplicación	Característica
Biología	-Adquiere modelos de retina.
Empresa	-Reconocimiento de caracteres escritos. -Modelar sistemas para control y automatización.
Medio ambiente	-Previsión del tiempo.
Finanzas	-Interpreta firmas. -Identifica falsificaciones.
Manufacturación	-Sistemas de control. -Robots automatizados.
Medicina	-Monitoreo en cirugías. -Tratamiento y diagnóstico por medio de síntomas o datos analizados.
Militares	-Crea armas inteligentes. -Clasifica señales de radas.

Elaborado por: Investigadora

Tabla 3.-Ventajas y Desventajas de las redes neuronales [12].

Ventajas	**Desventajas**
Aprendizaje Adaptativo: Ejecuta tareas basadas en un entrenamiento.	Se precisa mayor procesamiento de los datos.
Auto-organización: Establece su propia información por medio de la fase de aprendizaje.	Para grandes tareas mayor será la complejidad de la red.
Tolerancia a fallos: Si tiene alguna imperfección la red, se puede conservar varias capacidades de la red si padece algún daño.	Es elevado el tiempo de aprendizaje: si se incrementa la cantidad de patrones a clasificar y si se requiere mayor capacidad de adaptación de la red neuronal.
Operación en tiempo real: Se realiza en paralelo los cómputos neuronales.	Cantidad elevada de datos para el entrenamiento.
Inserción dentro de la tecnología: Mediante chips para redes neuronales se puede perfeccionar diversas tareas.	Son más costosas computacionalmente que los algoritmos tradicionales.

Elaborado por: Investigadora

1.2.2.3.2. Red neuronal artificial (ANN)

Una red neuronal artificial tiene una estructura similar a la neurona humana, donde la capa de entrada recibe información de otras redes neuronales, la capa sumatoria funciona como el cuerpo celular de las neuronas, la capa de activación adquiere la información agregada y envía una señal, pero solo si la entrada agregada atraviesa un valor umbral caso contrario no envía la señal y la capa de salida puede estar conectado a otras neuronas o trabaja como capa de salida final para formar las predicciones como se puede visualizar en la figura 6 [10].

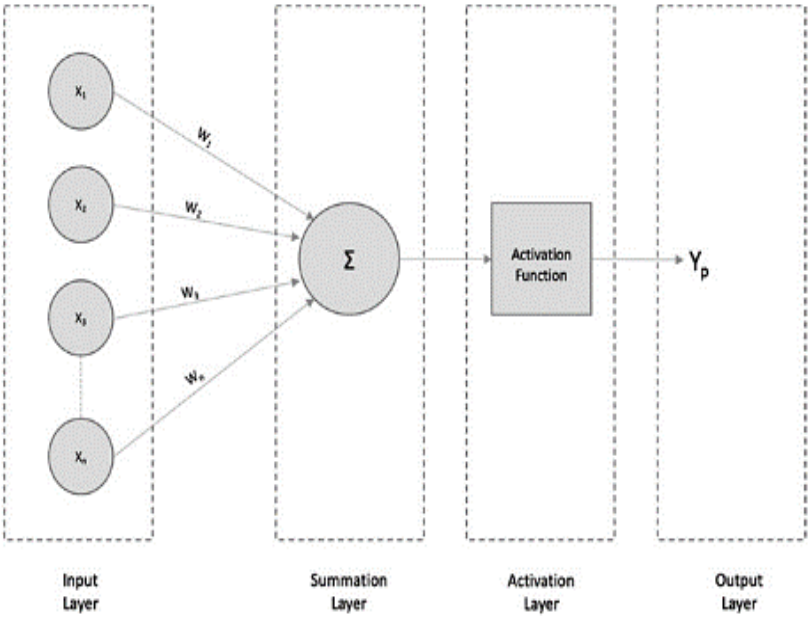


Figura 6.- Arquitectura de una Red Neuronal Artificial [10].

Cada una de las neuronas de la capa oculta está conectada a las neuronas de la anterior capa de entrada y por lo tanto, a la capa de salida, la función principal de la capa de entrada es actuar como una vía hacia las capas ocultas, donde transfiere información.

En cambio la capa oculta es necesaria debido a que sin ella, las redes neuronales solo estarían como un conjunto de combinaciones lineales ponderadas, la capa de salida es la responsable de la entrega de datos hacia el mundo externo [15].

1.2.2.3.3. Red Neuronal Simple

Una Red Ned Neuronal Simple consta de una sola capa oculta, está conformada por una capa de entrada, la cual va conectada a cada una de las neuronas de la capa oculta y finalmente a la capa de salida [10].

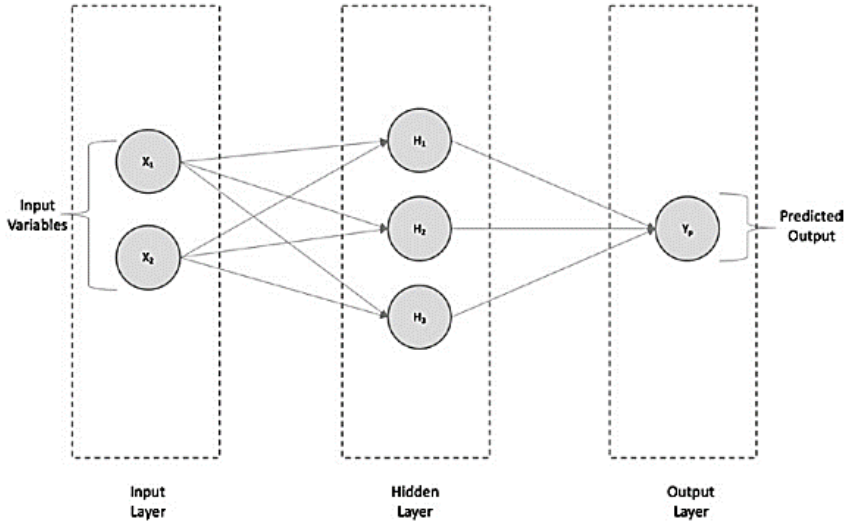


Figura 7.-Arquitectura de la Red Neuronal Simple [10].

1.2.2.3.4. Red Neuronal Profunda (DNN)

Una red neuronal profunda contiene entre las capas de entrada y salida un elevado número de capas ocultas, mientras más neuronas y capas ocultas tenga la red neuronal profunda se puede decir que existe más precisión y los datos se clasifican de mejor manera, sin embargo, mientras más capas ocultas tenga la red el costo de cómputo aumenta [10].

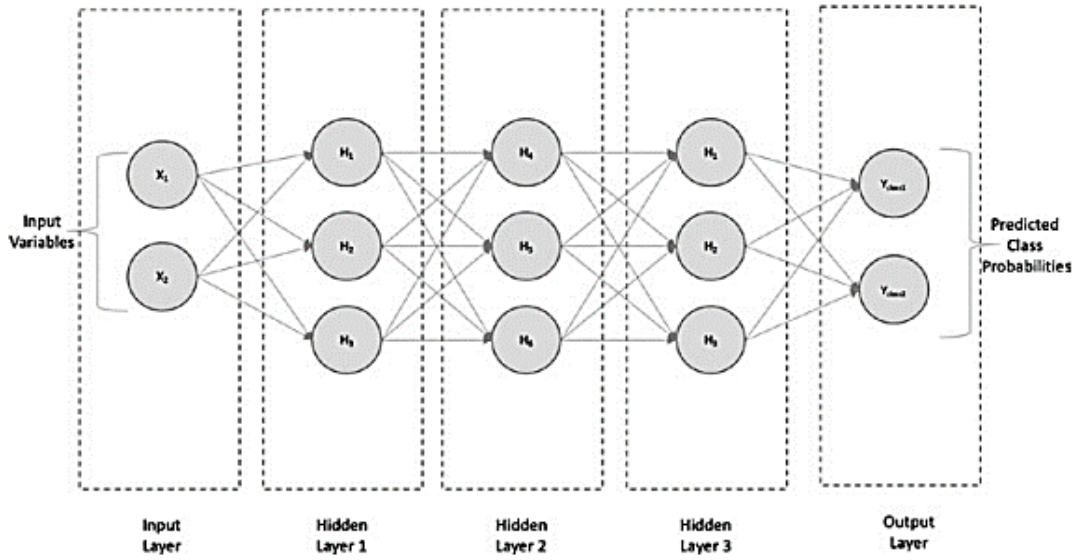


Figura 8.-Arquitectura de una red neuronal profunda [10].

1.2.2.3.5. Red Neuronal Convolucional (CNN)

Una red convolucional es una arquitectura de red para Deep Learning la cual puede contener cientos de capas que extrae las características más notables de los rostros alcanzando una precisión alta en el reconocimiento facial. En la figura 9, se puede observar una red que contiene diversas capas convolucionales donde se emplean filtros a cada una de las imágenes de entrenamiento con diferentes resoluciones, de manera que la salida de la imagen convolucionada es la entrada de la siguiente capa, y en la tabla 4, se encuentran las más utilizadas en las redes neuronales convolucionales [16].

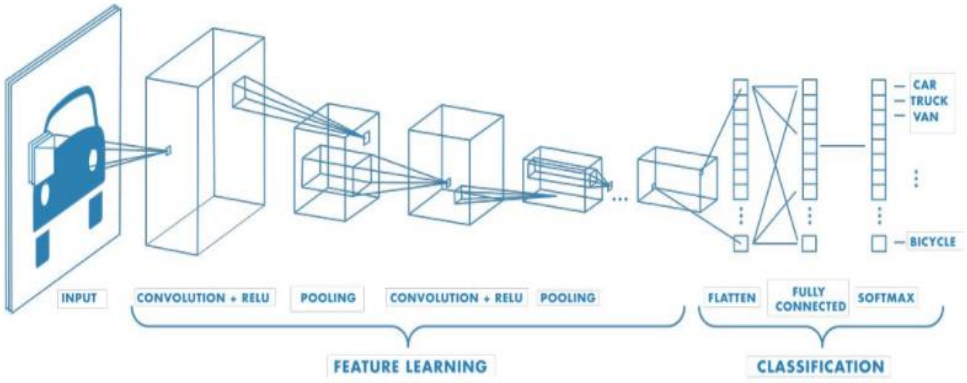


Figura 9.- Red con varias capas convolucionales [16].

Tabla 4.-Capas de CNN [16].

Capas	Función
Convolución	Las imágenes de entrada van a un conjunto de filtros convolucionales.
Unidad Lineal Rectificada (ReLU)	Determina los valores negativos a cero manteniendo los valores positivos, logrando un entrenamiento más eficaz y rápido.
Pooling	Simplifica la salida disminuyendo la tasa de muestreo no lineal, mitigando el número de parámetros que la red requiere aprender.

Tabla 5.-Funciones y Aplicaciones de la Red Neuronal Convolucional [16].

Funciones	Aplicaciones
Aprende directamente de los datos, sin necesidad de extraer manualmente.	Imágenes médicas
Generar resultados de reconocimiento altamente precisos.	Procesamiento de audios
	Generación de datos sintéticos
Permite aprovechar las redes preexistentes	Detección de señales de stop

Elaborado por: Investigadora

En la figura 10, se puede observar que existe un grupo de imágenes que se envían a la red neuronal convolucional de modo que asimila las características y clasifica cada uno de los objetos de manera automática [16].

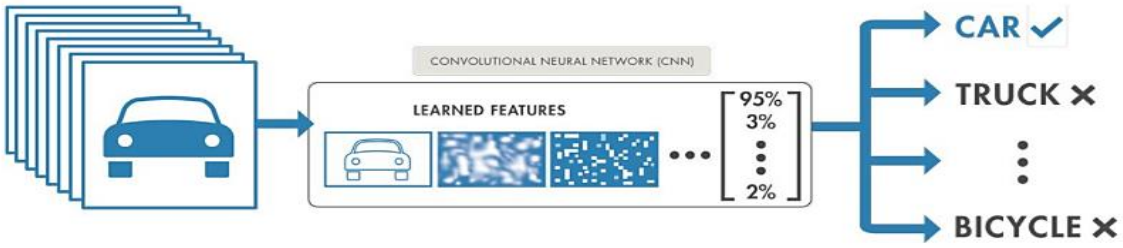


Figura 10.-Ejemplo de una Red Neuronal Convolucional [16].

1.1.2.4. Reconocimiento Facial

El reconocimiento facial es un método utilizado para identificar a una persona mediante una imagen o un video por medio de un análisis de las particularidades del rostro, por esto, se emplean dos asignaciones básicas para el proceso de identificación las cuales son detección y reconocimiento, hay que tomar en consideración la posición y la iluminación para el entrenamiento de la red neuronal, ya que son factores muy importantes al momento en que la cámara detecte el rostro, debido a que pueden existir errores si se omiten estos dos parámetros al momento de identificar a la persona, en la tabla 6, se indican las ventajas, desventajas y en la tabla 7, las aplicaciones del Reconocimiento Facial [17].

Tabla 6.- Ventajas y Desventajas del Reconocimiento Facial [18].

Ventajas	**Desventajas**
Control del sistema desde cualquier PC.	Los sistemas de identificación biométrica son más costosos que las tradicionales.
Eficiente sistema de vigilancia para mantener la seguridad.	No siempre son precisas debido a que si la persona cambia la apariencia o hay un cambio en el ángulo de la cámara ocasionará errores en el reconocimiento.
Solamente el operador puede efectuar el sistema, no hay manipulación del mismo por parte del usuario.	Intrusión a la privacidad por lo que puede rastrear a las personas en cualquier momento.
Difícil de vulnerar debido a que el acceso no depende de métodos físicos.	Si no existe una correcta iluminación existirá errores en el reconocimiento.

Elaborado por: Investigadora

Tabla 7.-Aplicaciones del Reconocimiento Facial [19].

Aplicación	**Función**
Control de Acceso	Acceso a Establecimientos, Acceso a Vehículos.
Biometría	Pasaportes, Fraude, Teléfonos Inteligentes, Accesos a lugares restringidos.
Seguridad de la información	Seguridad en Base de datos, Seguridad en internet, Registros Médicos, Cajeros Automáticos.
Cumplimiento de la ley y vigilancia.	Control CCTV, Análisis Post-event, Seguimiento de Sospechosos.
Tarjetas Inteligentes.	Autenticación de usuarios.

Elaborado por: Investigadora

Para el análisis de reconocimiento facial, se toma en cuenta diversas etapas que debe cumplir el sistema de forma general.

En la figura 11, se visualiza las etapas del reconocimiento facial; primeramente, es la detección del rostro, el cual, permite ingresar una imagen que es procesada, después, se extraen las características de la imagen como puede ser el color de los ojos, forma de la nariz, entre otros aspectos por medio de algoritmos, finalmente, ingresa a la etapa de reconocimiento que se refiere a la clasificación de patrones extraídos de la imagen para la toma de decisiones. [20].

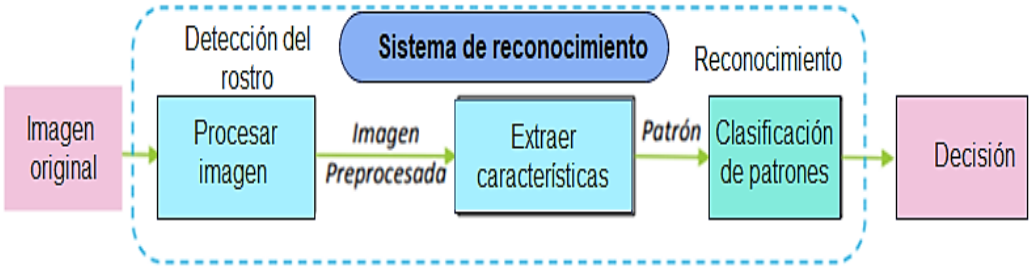


Figura 11.-Etapas del reconocimiento facial [20].

1.2.2.5. Modelo de aprendizaje Profundo

1.1.2.5.1. FaceNet

FaceNet es un modelo de aprendizaje profundo de uso libre, siendo así uno de los sistemas más precisos en relación a otros modelos de identificación facial, por consiguiente, admite extraer las características más importantes de una cara, asimila un mapeo de imágenes de rostros a un espacio euclidiano donde las distancias corresponden claramente a una medida de semejanza de rostros.

Las tareas como reconocimiento facial, la verificación y la agrupación se implementan usando técnicas con incrustaciones de FaceNet como vectores de funciones [21].

1.1.2.5.1.1. Estructura del modelo

En la figura 12, se puede observar la estructura del modelo que está conformada por una capa de entrada por lotes, una red neuronal convolucional profunda, consecutivamente L2 es la normalización, obteniendo el embedding como el mapeo de los atributos de entrada a un vector y seguidamente, viene el Triplet Loss durante el entrenamiento [22].

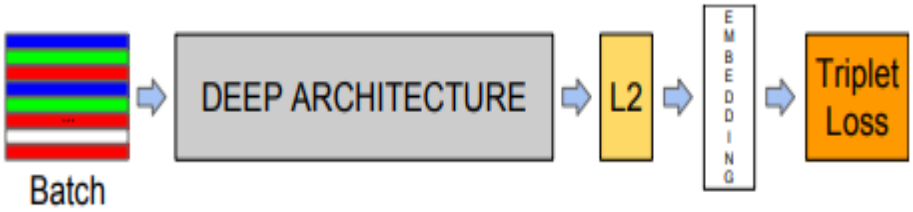


Figura 12.-Estructura del Modelo [22].

El modelo emplea como función de pérdida "Triplet Loss", la cual minimiza la distancia de los ejemplos positivos y al mismo tiempo maximiza la distancia de los ejemplos negativos, debido a que son rostros de una misma entidad surgirán más cerca que los rostros de distintas identidades como se puede observar en la figura 13 [22].

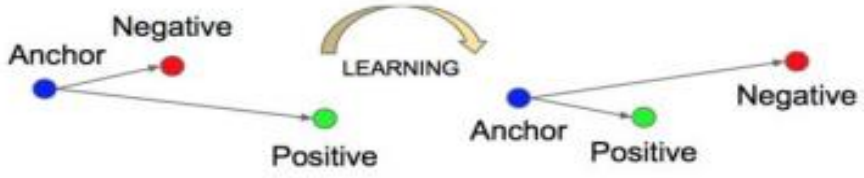


Figura 13.- Triplet Loss Learing [22].

1.2.2.6. MobileNetV2

MobileNetV2 es una red neuronal convolucional la cual está basada en una arquitectura que usa convoluciones separadas para obtener redes neuronales profundas, la cual impulsa el estado del arte para el reconocimiento visual móvil, incluyendo la clasificación, detección de objetos y la segmentación semántica [23].

1.2.2.6.1. Arquitectura MobileNet V2

La arquitectura de MobileNetV2 utiliza bloques residuales invertidos con funciones de cuello de botella, además emplea circunvoluciones ligeras en profundidad para filtrar entidades en la capa de expansión intermedia. Los cuellos de botella codifican las entradas y salidas intermedias del modelo, la capa interna encapsula la capacidad del modelo para convertirse, de conceptos de nivel inferior como los pixeles, a descriptores de nivel superior como categorías de imágenes; los bloques azules figuran bloques de construcción convolucionales compuestos como se puede ver en la figura 14 [23].

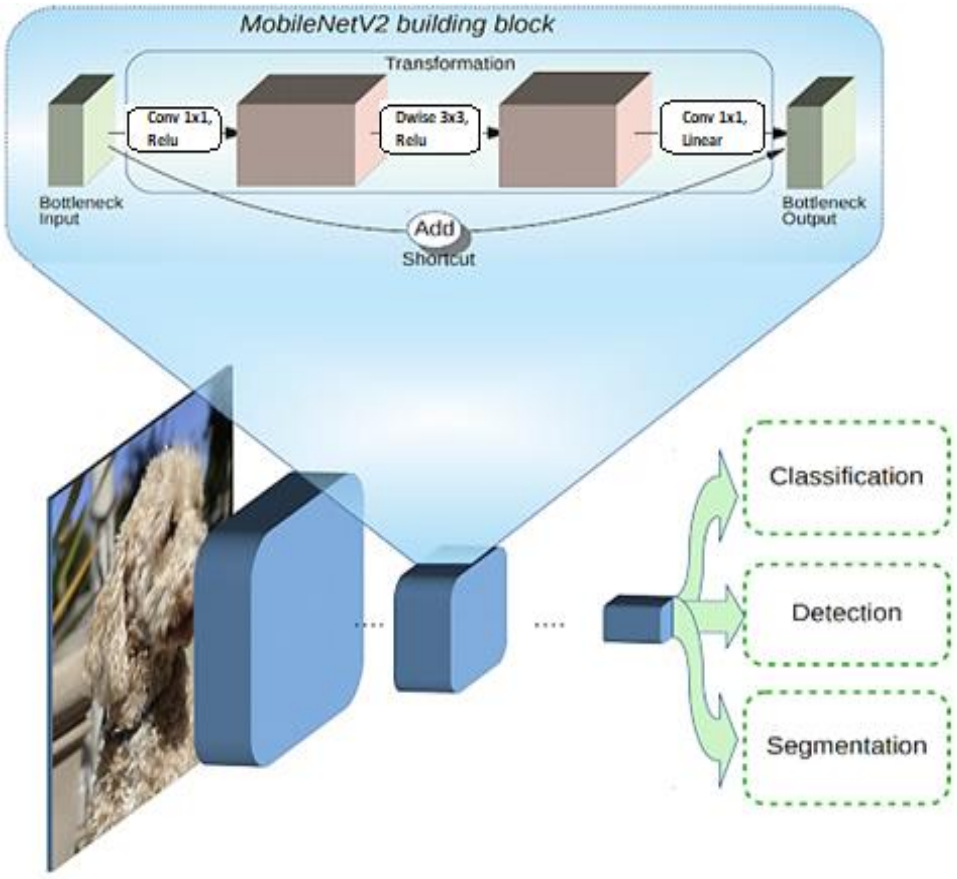


Figura 14.-Descripción General de la arquitectura MobileNetV2 [23].

1.2.2.7. Raspberry Pi

Raspberry Pi, se trata de un sistema SoC (Sistema en un chip) que emplea un microprocesador con arquitectura ARM, memoria RAM y tarjeta gráfica o GPU (Unidad de Procesamiento Gráfico) siendo así un ordenador de placa reducida de bajo coste, debido a que no incluye un disco duro, la memoria varía según el modelo, entre 256 MB y 8 GB, para el almacenamiento se usa una tarjeta SD, SDHC o MicroDS para la instalación del Sistema Operativo GNU/Linux ARM (Debian, Raspberry Pi OS versión adaptada de Debian, Fedora, RISC OS) donde se localiza los archivos que se ejecuta en la placa, se debe emplear una fuente de alimentación externa 5.1 Volt / 3 Amperios para su funcionamiento.[24]

Figura 15.- Placa Raspberry Pi 3 [24].

1.2.2.8. Sistema Operativo Raspbian

Raspbian llamado también Raspberry Pi OS es una distribución del sistema operativo de Linux basado en Debian favorecido para Raspberry Pi, el cual viene preinstalado con software para la programación, y cuenta también con navegador web, office entre otros, para su uso educativo, los comandos se efectúan en la terminal que esta instalada en el Sistema Operativo [24].

1.2.2.9. Lenguaje de programación

1.2.2.9.1. Python

Python tiene eficientes estructuras de datos de alto nivel y un procedimiento de programación enfocado a objetos por ende es un lenguaje de programación viable con una sintaxis entendible que está compuesto por elementos de diferentes tipos como palabras reservadas, identificadores, operadores, delimitadores, literales y funciones integradas [24].

1.2.2.2.2. Open CV

Open CV es una biblioteca de código abierto que puede ejecutarse en varios sistemas operativos como Windows, iOS, Android, Linux y Mac OS; tiene más de 2500 algoritmos diseñado para las aplicaciones en tiempo real, además, tiene diversas funciones que se pueden aplicar al sistema de visión artificial y aprendizaje automático. En la figura 16, se visualiza la estructura de la librería Open CV [25].

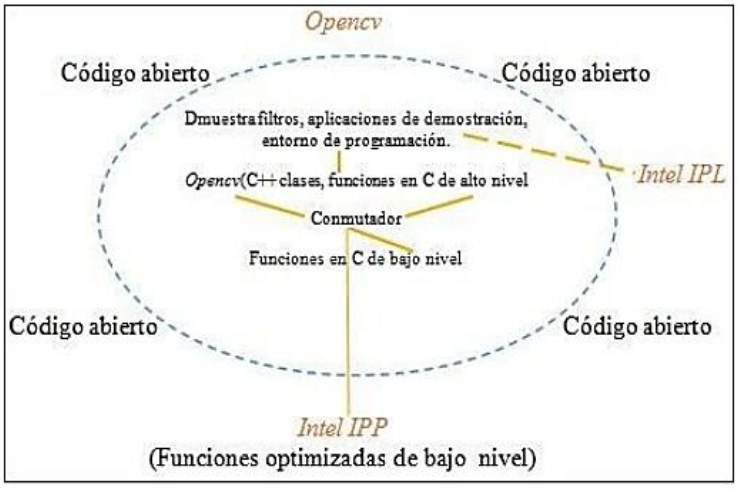


Figura 16.- Estructura de la librería Open CV [25].

En la tabla 8, se observan las ventajas, desventajas y funciones que posee Open CV para la captura, análisis y manipulación de información.

Tabla 8.- Funciones, Ventajas y Desventajas de Open CV [26].

Open CV		
Funciones	**Ventajas**	**Desventajas**
-Identifica, detecta, clasifica objetos y rostros.	-Es Multiplataforma.	-No se recupera la información perdida.
-Utilizado en áreas de robótica y realidad aumentada.	-Se puede usar otros lenguajes de programación.	-Tiene compresiones/ descompresiones complejas y costosas.
-Examina escenarios.	-Es gratuita. - Es portable.	-Documentación débil, no siempre explica lo que significa cada parámetro.
-Extrae modelo 3D.	-Bajo uso de RAM.	-Pequeña biblioteca de aprendizaje automático.

Elaborado por: Investigadora

1.2.2.10. Biblioteca de Redes Neuronales

1.2.2.10.1. TensorFlow

TensorFlow es una biblioteca de código abierto para aprendizaje automático y profundo, el cual construye y entrena redes neuronales, la palabra Tensorflow viene de tensor, que representa tensores como arreglos n-dimensionales, siendo un tensor la generalización de matrices y vectores, y flow, que se refiere al flujo el cual es un marco de cálculo gráfico subyacente que hace uso de tensores para que puedan ejecutarse [10].

1.2.2.10.2. Keras

Keras es una biblioteca desarrollada en python, empleado para aligerar la creación de redes neuronales, la arquitectura de cada red neuronal está construida por bloques también envuelve redes convolucionales y recurrentes; utiliza librerías como TensorFlow, Theano y Microsoft Cognitive Toolkit para la creación de modelos de Deep Learning [27].

1.2.2.11. Parámetros

1.2.2.11.1. Iluminación

La iluminación es un parámetro muy importante, afecta al reconocimiento facial, el rostro debe ser reconocido bajo ciertas variaciones, ya sea fuente de luz o nivel de iluminación, estas medidas ocasionan cambios en el color de la piel denominados sobreexposición y subexposición, por lo tanto, la iluminación debe ser estable para evitar errores de datos al momento de capturar una imagen.

La sobreexposición, se refiere al exceso de luz en la imagen y la subexposición, se trata de la carencia de luz en la imagen causando que la fotografía se torna obscura, por lo tanto, afectaría al reconocimiento facial como se puede observar la figura 17 [28].

Figura 17.-Subexposición y Sobreexposición de una imagen.

1.2.2.12. Temperatura corporal

La temperatura corporal es la consecuencia del equilibrio entre el calor generado por los procesos orgánicos y la eliminación de calor a través del sudor de manera que, se ve impactada por diversos elementos como el sexo, edad, deporte, comida, ambiente entre otros [29].

La tabla 9, indica los valores de temperatura según la edad de la persona.

Tabla 9.-Valores de Temperatura según la edad [30].

Valores de Temperatura	
Edad	**Grados Centígrados**
0-10 años	35.5 °C - 37.5°C
11-65 años	36.4 °C - 37.6°C
Mas de 65 años	35.8 °C – 36.9°C

Elaborado por: Investigadora

1.2.2.13. Asistente de Mensajería Telegram

Telegram es una aplicación de mensajería, la cual permite estar en contacto con amigos, familiares, puede ser utilizada en dispositivos como tablets, computadores y celulares inteligentes.

La aplicación incluye varios servicios como bot normales, el cual interactúa mediante comandos en mensajes privados, se utilizará a través de python ya que es un lenguaje de programación viable.

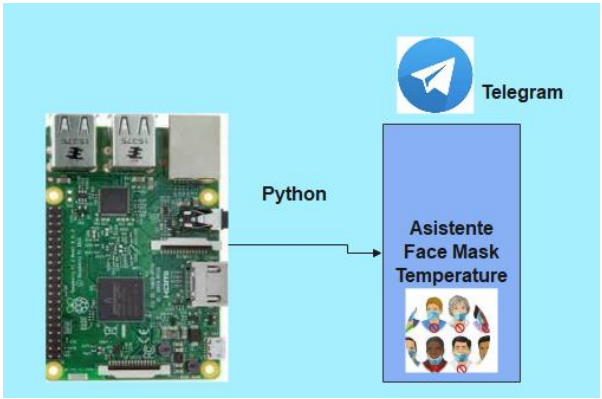


Figura 18.- Asistente en Telegram.

Elaborado por: Investigadora

1.2.2.14. ThingSpeak

ThingSpeak es un servicio de plataforma de análisis de IoT que permite añadir, visualizar y analizar los datos en tiempo real en la nube, además, se puede almacenar datos sin la necesidad de configurar servidores web. Se caracteriza por su sencillez al momento de programar aplicaciones del mundo real, debido a que se usa poco código y una configuración sencilla en relación a otros entornos de trabajo [31].

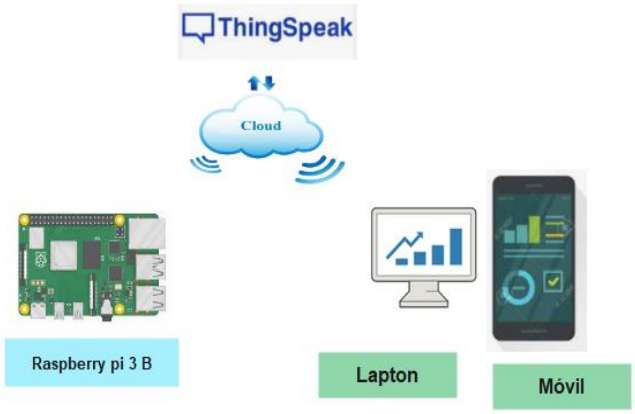


Figura 19.-Servicio de Plataforma ThingSpeak.

Elaborado por: Investigadora

Tabla 10.-Características, con que Trabaja y Aplicaciones de ThingSpeak [31].

Características	Funciona con	Aplicaciones
Recolecta datos en canales privados	Matlab-Simulink	Monitoreo Ambiental
Comparte datos con canales públicos	Arduino	Monitoreo de Energía
Programación de eventos	Módulos ESP8266 y ESP32	Agricultura Inteligente
Alertas	Raspberry pi	
Integraciones de aplicaciones	LoRaWAN	
API RESTful y MQTT	Libelium, Senet, Beckhoff	

Elaborado por: Investigadora

1.3. Objetivos

1.3.1. Objetivo General

- Implementar un sistema de control de acceso por medio de reconocimiento facial con uso de mascarilla y monitoreo de temperatura.

1.3.2. Objetivos Específicos

- Analizar los componentes y sensores apropiados que se van a utilizar en la construcción del proyecto.
- Determinar los elementos técnicos de reconocimiento facial con uso de mascarilla y monitoreo de temperatura.
- Construir un prototipo de sistema de control de acceso de detección de reconocimiento facial con uso de mascarilla y monitoreo de temperatura.

CAPÍTULO II

METODOLOGÍA

2.1. Materiales

Para el desarrollo del proyecto de investigación, se utilizaron materiales como libros, artículos, tesis, revistas, internet, hoja de datos o datasheet de los componentes electrónicos, además se utilizó un software adecuado para la programación.

2.2. Métodos

2.2.1. Modalidad de la Investigación

El presente proyecto se orienta a un problema dentro de la sociedad, por esta razón, se realizó un sistema de control de mascarillas y temperatura siendo así, se mencionan las diferentes modalidades de investigación:

Investigación Aplicada

Para realizar este proyecto de investigación se toma en cuenta la técnica de investigación aplicada, puesto que se pone en práctica los conocimientos obtenidos para implementar el sistema de control de acceso por medio de reconocimiento facial con uso de mascarilla y monitoreo de temperatura.

Investigación Bibliográfica

En la recolección de información se utiliza la investigación bibliográfica, ya que se investigó de libros, artículos científicos, tesis, revistas los cuales son de mucha utilidad para el desarrollo del proyecto, se debe saber información relacionada con el sistema de

control de acceso por medio de reconocimiento facial con uso de mascarilla y monitoreo de temperatura.

Investigación experimental

Se realiza la investigación experimental debido a que por medio de pruebas se examinó los beneficios que tiene el sistema de control y monitoreo.

Investigación de campo

Se efectúa la investigación de campo ya que se obtiene datos reales acerca de los parámetros de la temperatura por medio del sensor y el reconocimiento facial con uso de mascarilla, lo que permitió la implementación del sistema de control de acceso.

2.2.2. Recolección de Información

Para la recolección de información, se analizó artículos científicos, libros y tesis efectuados en los últimos años, los cuales, tienen relación con dispositivos de monitoreo de temperatura corporal, sistema de reconocimiento facial, IoT basado en la temperatura que son utilizados como base para la implementación del proyecto.

2.2.3. Procesamiento y Análisis de Datos

Para el procesamiento y análisis de datos se realizaron las siguientes actividades:

- Organización de la información recopilada.
- Revisión de la información seleccionada.
- Análisis de la información obtenida.
- Interpretación de los resultados.
- Determinación de la propuesta de solución.
- Los resultados obtenidos se añaden al prototipo para observar si las condiciones

son adecuadas caso contrario se ajustará a parámetros apropiados.

2.2.4. Desarrollo del Proyecto

Para cumplir con los objetivos planteados en el proyecto de investigación se llevó a cabo los siguientes pasos:

- Análisis, descripción y evaluación de los componentes para el sistema de control de acceso y monitoreo.
- Investigación de los elementos técnicos de reconocimiento facial con uso de mascarilla para el sistema de control y monitoreo.
- Determinación de materiales a manejar en el sistema de control y monitoreo.
- Comparación de características, precios de los materiales para el proyecto de investigación.
- Elección del hardware y software para el prototipo
- Realización de una placa del circuito donde se coloca los elementos para la implementación del proyecto de investigación.
- Instalación del sensor y los elementos que se emplea para el funcionamiento del sistema de control y monitoreo.
- Ensamblaje del prototipo.
- Investigación de una herramienta que permite mostrar los resultados en tiempo real de los datos que genera el sistema.
- Ejecución de pruebas de funcionamiento del sistema de control de acceso por medio de reconocimiento facial con uso de mascarilla y monitoreo de temperatura.
- Análisis de posibles fallas del dispositivo.
- Elaboración del informe final.

CAPÍTULO III

RESULTADOS Y DISCUSIÓN

3.1. Análisis y discusión de los resultados

La implementación del sistema de control de acceso por medio de reconocimiento facial con uso de mascarilla y monitoreo de temperatura, permitió detectar el correcto uso de mascarillas y la variación de temperatura, facilitando la obtención de datos de forma estable y eficaz; puesto que la información se envió mediante mensajería para el análisis de datos, si la temperatura sobrepasa su valor normal o no se use adecuadamente la mascarilla, se emite un sonido en señal de alerta, para que la persona no pueda ingresar, y así ayudar al control seguro y permanente de los usuarios.

3.2 Desarrollo de la propuesta

3.2.1. Etapas del Sistema

El sistema está compuesto por 3 etapas las cuales son: adquisición de datos, procesamiento de datos, y visualización de los datos en tiempo real. La figura 20, indica cada una de las etapas del sistema de control del uso mascarilla y temperatura. La primera etapa, se refiere a la adquisición de datos donde los sensores obtienen una magnitud física, pero para su adecuado procesamiento son convertidas a señales eléctricas, además, consta de una cámara web, la cual toma fotos con una adecuada resolución y son remitidas a la tarjeta de desarrollo para su posterior análisis. La segunda etapa, es aquella que recibe los datos para ser procesados por la tarjeta de desarrollo a través de un algoritmo, para la detección de mascarillas en tiempo real. La tercera etapa, se trata de la visualización de la información por medio de mensajería como telegram y el servicio de plataforma ThingSpeak, donde se observarán los datos estadísticos en la computadora y en la app móvil.

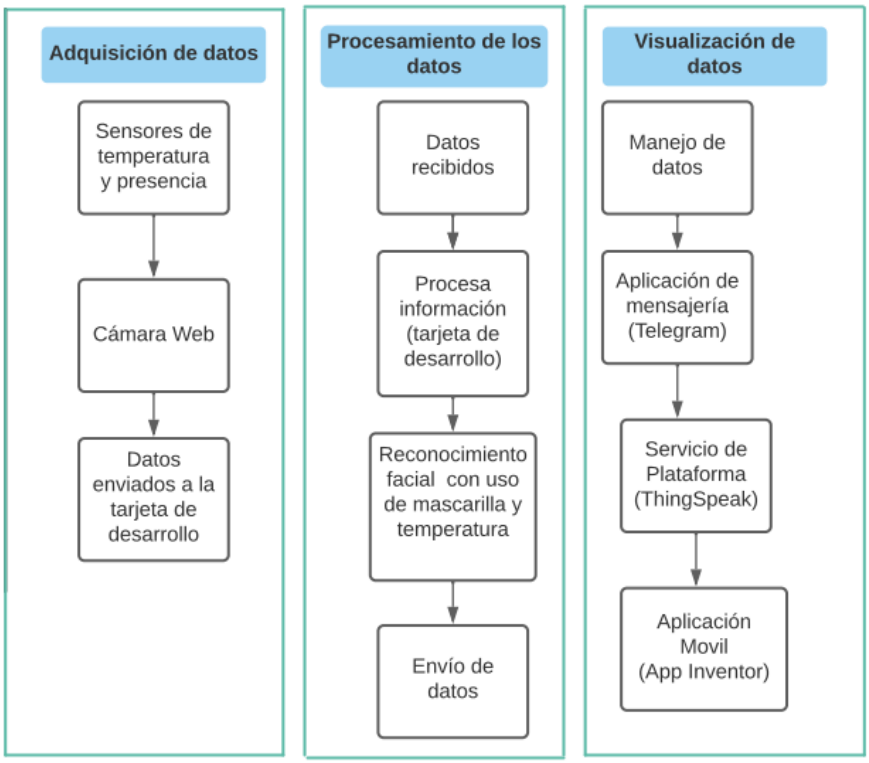


Figura 20.-Etapas del sistema.

Elaborado por: Investigadora

3.2.2. Selección de los elementos para la implementación del sistema

Se ejecutó un análisis técnico de cada uno de los elementos tomando en cuenta las características, el precio de cada uno de ellos y sobre todo si se encuentran disponibles en el país. La selección de los diferentes componentes para el sistema de control de acceso y monitoreo se indica a continuación:

Tarjeta de desarrollo

Son circuitos que incluyen un microcontrolador principal que realiza varias instrucciones de un programa en específico siendo ventajoso para los diferentes procesos de diseño ya sea sistemas digitales o analógicos [32].

Tabla 11.-Tabla comparativa de tarjetas de desarrollo [33], [34].

Características	**Tarjetas de Desarrollo**		
	Raspberry pi 3 B 	**Banana PI BPI-M64** 	**Arduino Uno** 
Tipo	Microordenador	Microcontrolador	Microcontrolador
RAM	1GB	2GB	2 KB
GPU	VideoCore IV 400 MHz	Dual core Mail 400 MP2	ATmega 328
Procesador	Broadcom BCM2837, Cortex-A53(ARMv8) 64 bits SoC	ARM A53 Quad-Core	ATmega 328
Almacenamiento	Micro SD	Micro SD	EEPROM 1KB
Wifi	2.4 GHz 802.11n	802.11 b/g/n	No
Bluetooth	BT 4.1	BT 4.0	No
Frecuencia de reloj	1,2 GHz	1,2 GHz	16 MHz
Conectividad de red	Fast Ethernet 10/100 Gbps	Ethernet 100 Mbps	No
Cámara	Puerto CSI	MIPI-CSI	No
Precio	$60	$75	$15
Disponibilidad en el país	Si	No	Si

Elaborado por: Investigadora

Una vez realizado el análisis en la tabla 11, la tarjeta de desarrollo seleccionada es la Raspberry pi 3B (Anexo A) debido a que es un ordenador de tamaño reducido, el cual contiene una conectividad de red, se puede conectar también a través de wifi, y es de fácil acceso en el mercado.

Sensor de temperatura infrarrojo

Dispositivo que mide la temperatura ambiente y de objetos, no tiene la necesidad de estar en contacto directo con el sensor ya que opera con la radiación emitida por el objeto.

Tabla 12.-Tabla comparativa de sensores de temperatura infrarrojo [35],[36].

Características	**Sensores de temperatura**		
	MLX90614 -ESFBAA 	**AMG8833** 	**MLX90614-ESFDCI** 
Voltaje	3.6-5 voltios	3-5 voltios	5 voltios
Interfaz	SMBUS I2C	I2C	SMBUS I2C
Temperatura del objeto	-70 °C a 380°C	0°C a 80°C	-40°C a 170°C
Precisión de temperatura	+/- 0.5°C +/- 0.3 °C (16-40) °C	+/-2.5 °C	+/-0.5°C
Distancia	2 cm-5cm	7 m	50 cm
Ángulo de visión	90°	60°	5°
Precio	$15	$89.00	$70
Disponibilidad en el país	Si	No	Si

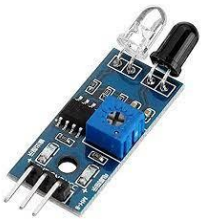
Elaborado por: Investigadora

De acuerdo al análisis realizado en la tabla 12, se seleccionó el sensor MLX90614-ESFBAA (Anexo B), porque tiene un rango de temperatura suficiente para aplicar en el proyecto a un precio económico, no necesita un circuito externo para su funcionamiento y está disponible en el mercado.

Sensor de presencia

Dispositivo que permite detectar la presencia o movimiento de un objeto o persona, se activa cuando detecta calor por medio de los infrarrojos, caso contrario permanece desactivado, estos sensores son empleados para sistemas de seguridad.

Tabla 13.-Tabla comparativa de los sensores de distancia [37] [38].

Características	**Sensores de presencia**		
	Sensor de movimiento PIR HC-SR505 	**Sensor Infrarrojo IR FC-51** 	**Sensor de movimiento PIR HC-SR501** 
Voltaje	4,5-20 voltios	3.3-5 voltios	5-20 Voltios
Corriente	-	18-20 mA	1 mA
Ángulo de inducción	<100°	35°	110°
Distancia	3m	2 cm -30 cm	3m-7 m
Precio	$3,50	$2,00	$2,70
Disponibilidad en el país	Si	Si	Si

Elaborado por: Investigadora

En la tabla 13, se realizó la comparación de los sensores disponibles en el mercado, se decidió utilizar el sensor infrarrojo IR FC-51 (Anexo C), el ángulo de inducción y la distancia es adecuada para la ejecución del proyecto, además el costo es accesible y está disponible en el mercado.

Cámara Web

Dispositivo para capturar imágenes, va conectado a la Raspberry Pi por medio del puerto USB donde las fotografías se transmiten a través de internet.

Tabla 14.-Tabla comparativa de las cámaras [39], [40], [41].

Características	**Cámaras**		
	Conexis 	Logitech C270 HD 	Adesso CyberTrack H4 
Resolución	720P 1280×720 pixeles	-720P 1280×720 pixeles	1080 pixeles
Velocidad de Fotogramas	30 FPS	30 FPS	30 FPS
Tipo de sensor	CMOS	CMOS	CMOS
Interfaz	USB2.0 + Cable de Audio 3.5mm	USB2.0	USB 2.0
Precio	$15	$55	$36
Disponibilidad en el país	Si	Si	Si

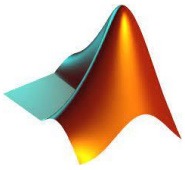
Elaborado por: Investigadora

En la tabla 14, se realizó la comparación de las posibles cámaras a utilizar para un funcionamiento adecuado del prototipo, se eligió la Cámara Web HD 720P Conexis, debido a que tiene una resolución adecuada para aplicar al sistema y el costo es económico, además como la Raspberry pi tiene puertos USB es factible utilizar este tipo de cámara.

Lenguaje de programación

Conjunto de instrucciones que permiten tener comunicación entre el ser humano y una máquina mediante algoritmos que la computadora pueda entender, también, posibilita que procese la información de manera segura y eficiente [42].

Tabla 15.-Tabla comparativa de los distintos lenguajes de programación [42], [43],[44].

Características	**Lenguajes de programación**		
	Python 	LabVIEW 	Matlab 
Sistema Operativo	Linux, Mac y Windows	Multiplataforma	Multiplataforma
Licencia	Libre	Propietaria	Software privativo
Aplicaciones	*Desarrollo Web *Procesamiento de imágenes *Interfaz gráfica *Juegos de desarrollo	*Diseño de sistemas industriales *Procesamiento de imágenes *Sistemas de pruebas automatizadas	*Manipulación de matrices *Interfaz de usuario *Procesamiento de imágenes

Ventajas	*Lenguaje de código abierto orientado a objetos *Facilita la programación concurrente *Portabilidad	*Simplificación de tareas *Gran flexibilidad al Sistema *Dotado de un compilador gráfico	*Extenso soporte de funciones ya desarrolladas *Visualización de gráficos de alta calidad *Proporciona IDE más rápido para el cálculo matemático
Desventajas	*Consumo de memoria *Lentitud	*Poca capacidad de procesamiento *Lentitud	*Problemas de velocidad *La computadora necesitan MCR (Matlab Component Runtime) para que funcionen adecuadamente.

Elaborado por: Investigadora

Se compararon los distintos lenguajes de programación mencionados en la tabla 15, por lo cual, el más óptimo es Python debido a que es un lenguaje de código abierto orientado a objetos que soporta procesamiento de imágenes y es de licencia libre, además, es compatible con Linux lo que facilita su implementación en la Raspberry Pi porque se va a instalar Raspberry Pi OS llamado también Raspbian para la realización del proyecto de investigación.

Detección de rostros usando clasificadores

La detección de rostros es una técnica, la cual localiza en una imagen el rostro de la persona, omitiendo el fondo de la imagen, en la tabla 16, se observa los clasificadores preentrenados para el reconocimiento de rostros de cada una de las personas.

Tabla 16.-Tabla comparativa de los clasificadores

Clasificador	Concepto	Características	Ventajas
Haar-Cascade	-Aprendizaje Automático, entrenada a partir de imágenes positivas y negativas.	-Detecta objetos en otras imágenes.	-Detecta la estructura de los objetos, aunque no sea uniforme.
FaceNet	-Red convolucional la cual extrae características más relevantes de un rostro, usa redes neuronales pre-entrenadas.	-Para la verificación de rostros tiene una precisión del 99.63% -La salida debe ser un vector de 128 elementos.	-Sistema de reconocimiento facial más robusto. -Escalabilidad más simple.
OpenFace	-Software de Código abierto basado en FaceNet, es un framework para el desarrollo de redes neuronales.	-Tiene un 87% de precisión. -Detección y reconocimiento de rostros.	-Verifica si dos imágenes corresponden a la misma persona.
FaceAPI	-Módulo de Java Scripts desarrollado por Microsoft.	-Herramienta potente -Implementa tres arquitecturas de red neuronal convolucional.	-Realiza pruebas a partir de la página web.

Elaborado por: Investigadora

En la tabla 16, se comparó los clasificadores para la detección de rostro siendo el más óptimo FaceNet, tiene un porcentaje de precisión sumamente alto con un 99,63% siendo apto para un reconocimiento robusto del sistema.

3.2.3. Diagrama de bloques del dispositivo

En la figura 21 se indica el diagrama general del sistema, el cual está compuesto por una Raspberry Pi 3, a la que va conectada la cámara USB, un sensor de temperatura y presencia para la adquisición de datos, una fuente de alimentación de 5V, una pantalla para su respectiva visualización, un ventilador para que la Raspberry Pi no se recaliente y el sistema pueda funcionar adecuadamente.

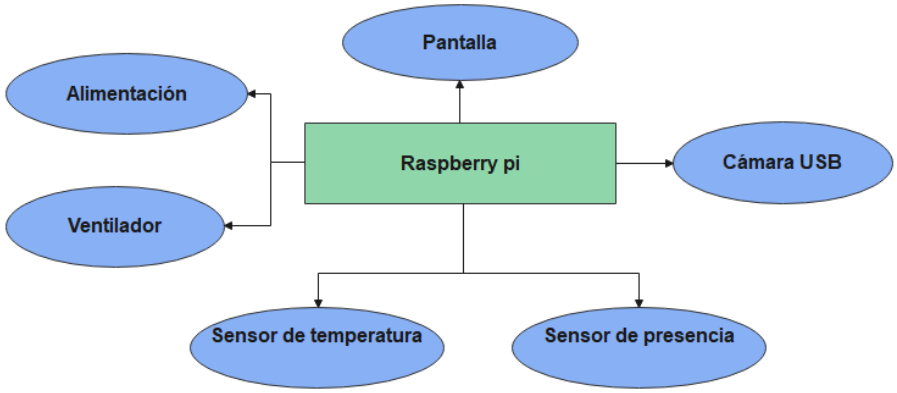


Figura 21.- Diagrama general del proyecto.

Elaborado por: Investigadora

3.2.4. Diseño esquemático del proyecto

Mediante el análisis realizado anteriormente, se seleccionaron los distintos elementos para la implementación del sistema como se puede ver en la figura 22, el esquema del prototipo indica que los datos que adquiere el sensor de temperatura por medio de la Raspberry Pi serán enviados en tiempo real a la base de datos, donde será almacenada la información, por otro lado, la cámara adquiere imágenes, las cuales son procesadas y enviadas a la aplicación de mensajería Telegram, donde se visualiza la fotografía y temperatura de la persona.

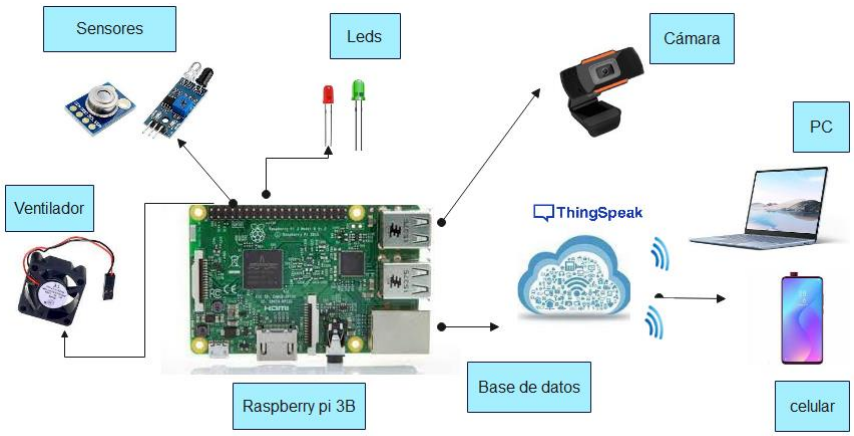


Figura 22.-Esquema General del prototipo.

Elaborado por: Investigadora

3.2.5. Diseño de hardware del proyecto

En la figura 23, se muestra el esquema con sus respectivas conexiones, se lo realizó en el software Fritzing, donde se observa cada uno de los elementos que se utilizó en el prototipo.

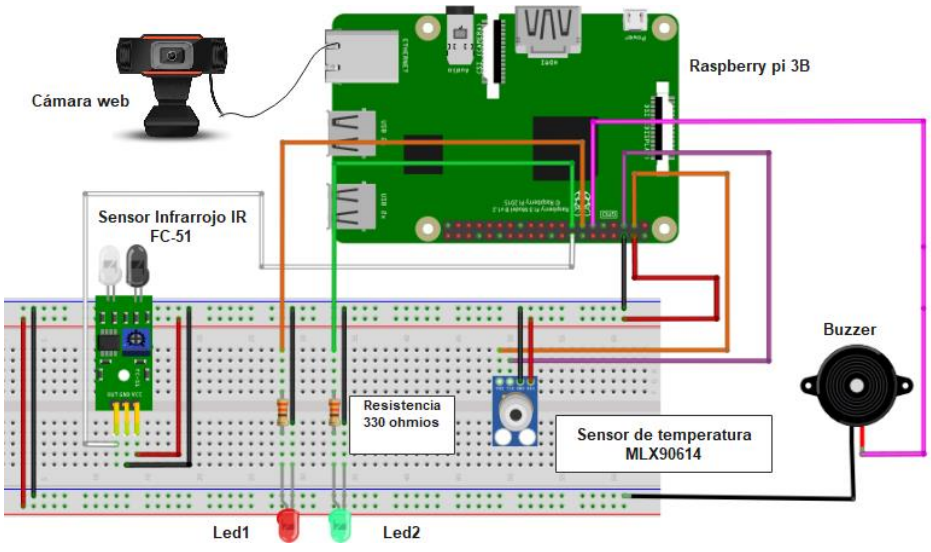


Figura 23.- Esquema del circuito físico.

Elaborado por: Investigadora

3.2.6. Montaje del circuito

En la figura 24, se observan las conexiones de los elementos del circuito implementado en una protoboard.

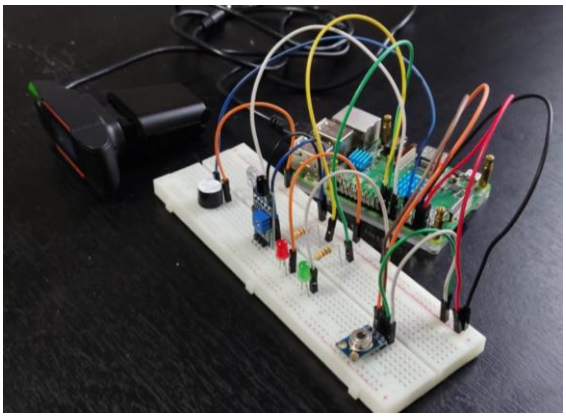


Figura 24.-Circuito en protoboard.
Elaborado por: Investigadora

En la figura 25, se indica el diagrama del circuito en la baquelita, para lo cual se usó el software proteus, que permite realizar simulaciones y pistas de circuitos.

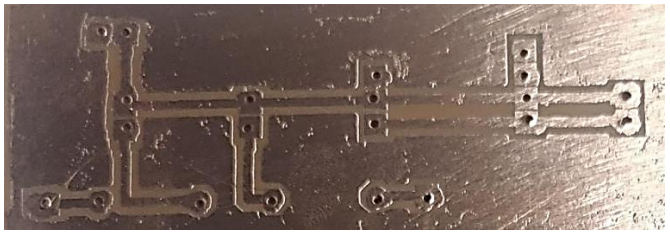


Figura 25.- Diagrama del circuito en baquelita.
Elaborado por: Investigadora

3.2.7. Diseño de software para el sistema de reconocimiento facial con uso mascarilla y temperatura

En la figura 26, se muestra el diagrama de flujo del funcionamiento del sistema, en donde se utiliza una cámara web para la captura de fotografías en tiempo real, la imagen es procesada para optimizar la calidad de los datos adquiridos, el sistema detecta si es o no

un rostro, mediante redes convolucionales en este caso FaceNet, consecutivamente se realiza, el reconocimiento de mascarilla en tiempo real mediante maskNet una red convolucional , se utiliza un modelo pre entrenado en MobileNetv2, para adquirir redes neuronales profundas debido a que, se puede implementar en la Raspberry Pi por sus bajos recursos computacionales, si el clasificador detectó que la persona tiene mascarilla, se forma un cuadro de color verde caso contrario, se forma un cuadro color rojo y sonará un buzzer en señal de alerta, en cada uno de los casos se mide la temperatura; los datos se visualizan en una pantalla de un celular que se conectó mediante VNC viewer, el cual permite acceder remotamente ya sea desde la computadora o el celular.

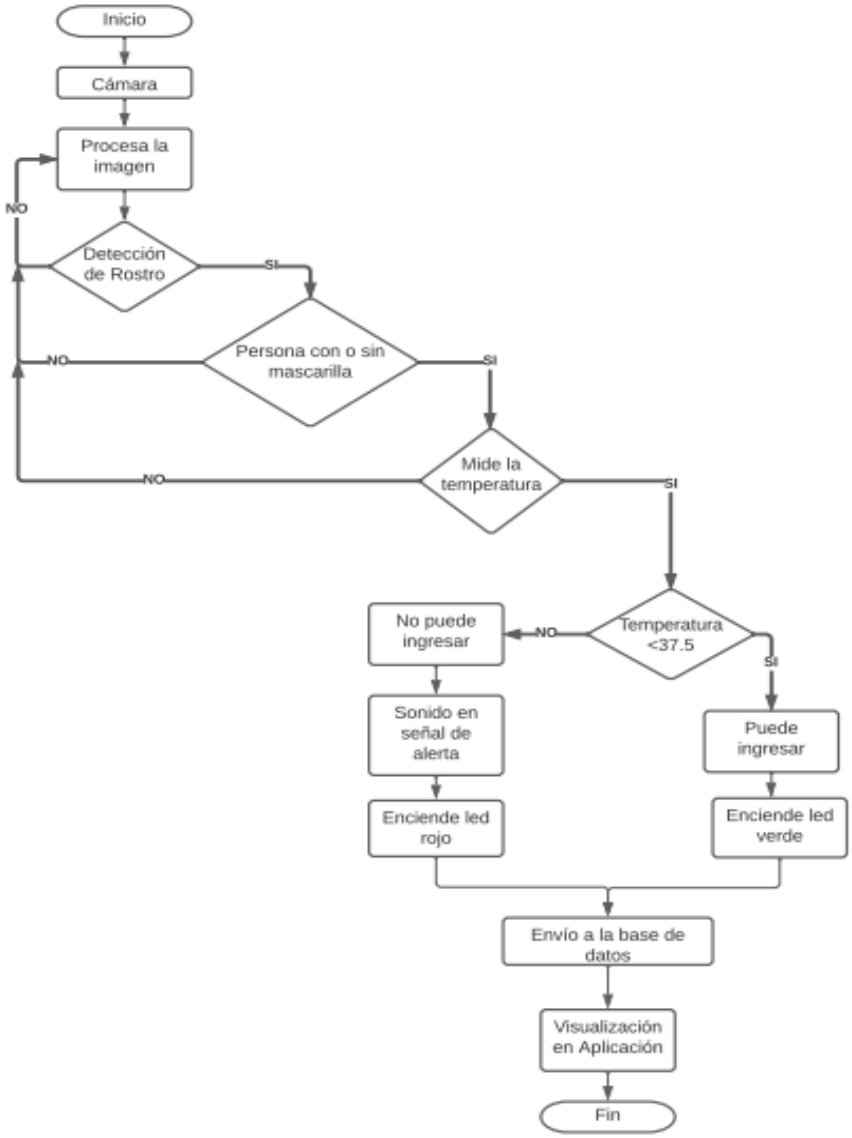


Figura 26.-Diagrama de flujo del sistema.

A continuación, se describe cada uno de los procesos a realizarse para el reconocimiento facial con uso de mascarilla y temperatura.

3.2.8.1. Instalación del Sistema Operativo en la Raspberry pi

Para la instalación del sistema operativo en la Raspberry Pi hay se usa una tarjeta microSD, revisar Anexo D.

3.2.8.2. Instalación de dependencias para el sistema

Para el reconocimiento facial con uso de mascarilla se necesita instalar TensorFlow, Keras entre otras librerías, para lo cual, se debe crear un ambiente virtual para instalar los paquetes específicos que se va a emplear en el desarrollo del proyecto y así no tener conflictos con la actualización de las librerías, revisar Anexo E.

3.2.8.3. Análisis del modelo y Validación de la Red Neuronal

Se necesita recolectar variedad de imágenes para un entrenamiento correcto, mientras más imágenes se tiene más confiable es el sistema. Se utilizó el modelo MobileNetv2 debido a la optimización de recursos para implementar en este proyecto. Para la validación de algoritmo, se obtiene una base de datos de 3830 imágenes que son distribuidas igualitariamente en dos clases: la primera clase consta de imágenes de personas con mascarilla y la segunda clase con imágenes de personas sin mascarilla.

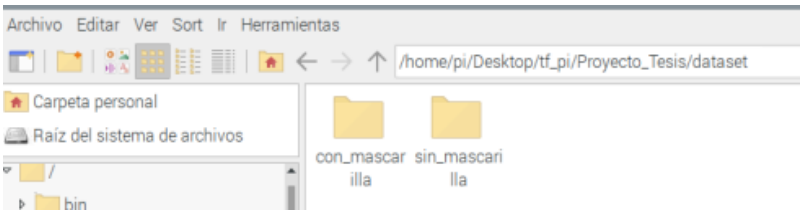


Figura 27.-Dataset para el entrenamiento.

Elaborado por: Investigadora

3.2.8.3.1. Importar paquetes necesarios para la ejecución del código

Es necesario importar las librerías que se van a utilizar, empleando el software Thonny Python IDE.

Las líneas de programación 3, 4 y 5 indican el procesamiento de una matriz que codifica las imágenes, al usar MobileNetv2 se debe llamar a las entradas antes de pasar al modelo, seguidamente convierte una instancia de una imagen en una matriz y por último al momento de instanciar un modelo los pesos se descargan automáticamente en ". Keras. Models".

```
3 from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
4 from tensorflow.keras.preprocessing.image import img_to_array
5 from tensorflow.keras.models import load_model
```

Figura 28.-Importar paquetes de TensorFlow.

-Librerías para la captura de imágenes y video.

```
7  from imutils.video import VideoStream
8  import numpy as np
9  import imutils
10 import time
11 import cv2
```

Figura 29.- Importar librerías para captura de imágenes y video.

Tabla 17.- Librerías para capturar Imágenes

Comandos	Función
from imutils. video import VideoStream	Facilidad de la interacción con la cámara web
numpy:	Crea, modifica vectores y matrices
imutils:	Realiza las funciones de procesamiento de imágenes
time:	Función relacionada con el tiempo
OpenCV	Biblioteca de código abierto, que realiza operaciones en imágenes y videos

Elaborado por: Investigadora

En la tabla 17, se especifica la función de cada uno de los comandos a utilizarse para que capture imágenes y videos en tiempo real.

Librerías para subir datos a ThinkSpeak

import paho.mqtt.publish as publish

Librería para manejo de sensores

Se utilizó la librería mlx90614 para el sensor de temperatura, controlador SMBus, Buzzer y puerto GPIO para el control de los pines de la Raspberry Pi como observa en la figura 30.

```
17 from smbus2 import SMBus
18 from mlx90614 import MLX90614
19 from gpiozero import Buzzer
20 import RPi.GPIO as GPIO
```

Figura 30.-Librerías para el uso de sensores

Elaborado por: Investigadora

3.2.8.4. Configuración de los sensores, leds, Buzzer

Los sensores utilizados para la ejecución del proyecto son MLX90614 sensor de temperatura infrarrojo utilizado para la adquisición de los datos de temperatura de la persona, sensor infrarrojo IR FC-51 que detecta la presencia de una persona, los leds y buzzer los cuales usan pines GPIO, observar Anexo D.

3.2.8.5. Entrenamiento de la red neuronal y generación del modelo

Para el entrenamiento de la red neuronal, se necesitan una cantidad considerable de recursos computaciones, por lo que se emplea TensorFlow y una red pre-entrenada en keras para reducir recursos.

El modelo, que se utiliza es MobileNetv2 se basa en los pesos preentrenados y el dataset; el código, se efectuó en Thonny Python que es un entorno de desarrollo que viene por defecto instalado en el sistema operativo Raspbian utilizado en la Raspberry Pi.

3.2.8.5.1. Función para la de Detección de mascarillas

En la línea 61 de programación se observa tres parámetros:

- Frame: cuando, se visualiza la imagen, se forma un cuadro alrededor de la misma.
- faceNet: modelo para la identificación facial.
- maskNet: modelo para la identificación de la mascarilla.

```
61 def detect_and_predict_mask(pantalla, faceNet, maskNet):
```

Figura 31.-Parámetros para la Detección de Mascarillas

En la figura 32, se muestra los bloques del proceso de detección de rostro que refiere a que dentro de una imagen o fotograma encuentre un rostro.

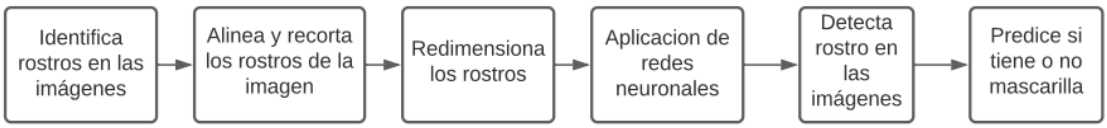


Figura 32.-Diagrama de bloques del proceso de imágenes

La línea 63, adquiere las dimensiones del rectángulo, la línea 64 y 65 adecua las imágenes de entrada para su clasificación, la cual, consiste en realizar la resta media, que es una técnica que ayuda a los cambios de iluminación en las imágenes, tiene como parámetros la imagen de entrada para ser procesada, el factor de escala que viene a ser 1.0 un valor predeterminado, que se lo efectúa después de aplicar la resta media y por último el tamaño de la imagen.

```
63     (h, w) = pantalla.shape[:2]
64     blob = cv2.dnn.blobFromImage(pantalla, 1.0, (224, 224),
65         (104.0, 177.0, 123.0))
```

Figura 33.-Parámetros de la imagen de entrada

Las líneas 67,68,69 pasa el conjunto de imágenes (blob) por medio de la red neuronal y se puede detectar cada uno de los rostros.

```
67    faceNet.setInput(blob)
68    detections = faceNet.forward()
69    print(detections.shape)
```

Figura 34.-Detección de Rostros

Las líneas 70,71,72 almacena en listas los rostros, ubicación y las predicciones de las mascarillas.

```
70    rostros = []
71    direcciones = []
72    prediccion = []
```

Figura 35.-Almacena en listas los datos

Las líneas 73,74 y 75 extrae la probabilidad que es el valor de confianza con la respectiva predicción.

```
73    for i in range(0, detections.shape[2]):
74        valor_confianza = detections[0, 0, i, 2]
75        if valor_confianza > 0.5:
```

Figura 36.-Extrae valor de confianza

Las líneas 76 hasta la 81, se calcula el cuadro delimitador para la imagen y se extrae las coordenadas del cuadro.

```
76            box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
77            (startX, startY, endX, endY) = box.astype("int")
78
79
80            (startX, startY) = (max(0, startX), max(0, startY))
81            (endX, endY) = (min(w - 1, endX), min(h - 1, endY))
```

Figura 37.-Coordenadas del cuadro delimitador

Las líneas 84 hasta la 88, extraen la región de interés (ROI) de la imagen, por lo que, solo se obtiene una sección de la imagen, por consiguiente, aumenta la precisión y disminuye

el tiempo de procesamiento, convierte la imagen BGR a RGB y cambia el tamaño de la imagen a 224x 224 para que la imagen sea procesada.

```
84              face = pantalla[startY:endY, startX:endX]
85              face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
86              face = cv2.resize(face, (224, 224))
87              face = img_to_array(face)
88              face = preprocess_input(face)
```

Figura 38.-Proceso de conversión y cambio de tamaño de la imagen

Las líneas 91 hasta la 97, añaden los rostros y los cuadros delimitadores, realizan las predicciones si se ha detectado una cara, y en la línea 99 retorno una tupla (asocia en un solo conjunto diversas variables).

```
91              rostros.append(face)
92              direcciones.append((startX, startY, endX, endY))
93
94
95      if len(rostros) > 0:
96          rostros = np.array(rostros, dtype="float32")
97          prediccion = maskNet.predict(rostros, batch_size=32)

99      return (direcciones, prediccion)
```

Figura 39.-Predicción si ha detectado una cara

3.2.8.6. Programación para el control de la temperatura y mascarilla

Se envía un mensaje a la aplicación de telegram si está o no con mascarilla, se emite la temperatura, se activa el led verde cuando está colocada adecuadamente la mascarilla en cambio, se activa el buzzer y se enciende el led rojo cuando sobrepase la temperatura y no este colocada la mascarilla, además, sube los datos a ThingSpeak para que se visualice en tiempo real; se utiliza def para definir una función, la cual, permite segmentar un programa complejo observar anexo G.

Guardar los datos en .csv

Cada uno de los datos adquiridos fecha, hora, uso de mascarilla y temperatura, se van guardando en la base de datos en un archivo plano para luego enviar mediante el protocolo mqtt, ver anexo H.

```
151 def save_log():
152     with open("/home/pi/Desktop/tf_pi/Proyecto_Tesis/datos_mascarilla_temperatura.csv", "a") as log:
153         log.write("{},{},{}\n".format(timeString, label, str(round(temp,1))))
154     log.close()
```

Figura 40.-Guardar datos en un archivo .csv

Ejecución del prototipo

Para poner en funcionamiento el prototipo, se ejecuta el archivo mask_temp_detection.py desde el IDE Thonny Python.

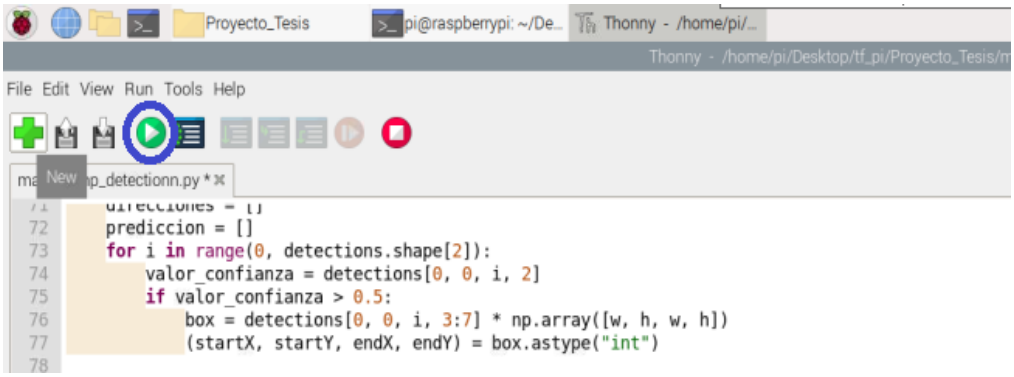


Figura 41.- Ejecución del programa

Elaborado por: Investigadora

3.2.9. Desarrollo de la programación para el sistema

3.2.9.1. Aplicación de mensajería Telegram

Se necesita diversas librerías para adquirir los datos e integración de la aplicación Telegram mostradas en la figura 42.

csv: se emplea para acceder a los archivos guardados en la base de datos

datetime: obtiene el tiempo cuando tome la fotografía y se graba en la Raspberry.

```
23 import csv
24 import telegram as tg
25 import datetime
26 from time import sleep
```

Figura 42.- Librerías para el uso de Telegram

Elaborado por: Investigadora

Telegram Bot

Para el desarrollo del asistente de Telegram, se realiza con la ayuda de BotFather, el cual, permite crear y controlar nuestro propio bot personalizado. Para comenzar a crear un bot, ingrese a la conversación de BotFather, coloque /start como se observa en la figura 43.

Figura 43.-Pantalla de BotFather

Elaborado por: Investigadora

Seguidamente, obtiene como resultado una lista de comandos, los cuales, se utiliza para personalizar el bot, teclear "/newbot" comando utilizado para crear un nuevo bot, luego colocar el nombre para el bot en este caso es Sistema de Monitoreo de Mascarilla y Temperatura Bot, después colocar el usuario para el bot que es facemask_temp_bot y finalmente el bot, se creará, observe la figura 44.

Figura 44.-Creación del bot

Elaborado por: Investigadora

También, se coloca una foto de perfil usando el comando "/setuserpic", se escoge el bot creado anteriormente @facemask_temp_bot y se elige la nueva foto del perfil para el bot de Telegram como se observa en la figura 45.

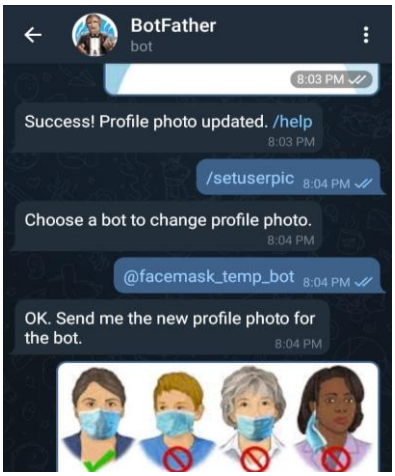


Figura 45.- Foto de perfil para el bot de Telegram

Elaborado por: Investigadora

Despúes de haber creado el bot Sistema de Monitoreo de Mascarilla y Temperatura, se crea un token, como se observa en la figura 46 el mismo que se emplea en la programación en python para él envió de datos a la aplicación.

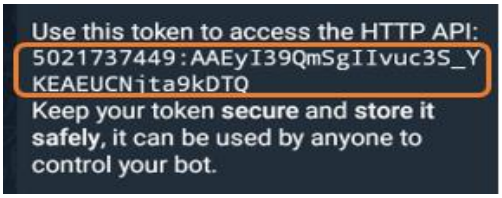


Figura 46.- Token para acceso de HTTP API

Elaborado por: Investigadora

Se utiliza el siguiente link https://api.telegram.org/bot<**YourBOTToken**>/getUpdates, para generar el id que se utiliza en la programación para chat en telegram.

Figura 47.- Link para generar Id en Telegram

Teniendo como resultado la siguiente línea de código donde está el id del chat '1930098683' que es un número positivo lo que quiere decir que corresponde al chat de un usuario como, se puede observar en la figura 48, en cambio si el número es negativo significa que pertenece a un grupo de chats.

{"ok":true,"result":[{"update_id":378710761,
"message":{"message_id":2,"from":{"id":1930098683,"is_bot":false,"first_name":"Verito","username":"VeronicadelosAngeles","language_code":"es"},"chat":
{"id":1930098683,"first_name":"Verito","username":"VeronicadelosAngeles","type":"private"},"date":1642615505,"text":"/start","entities":
[{"offset":0,"length":6,"type":"bot_command"}]}}]}

Figura 48.-Chat_ID generado por telegram

Elaborado por: Investigadora

Ahora, en el software Thonny de Python en la Raspberry Pi se coloca el bot_token, chat_ID que se adquirió anteriormente; y para activar el bot se utiliza la línea 31, que se observa en la figura 49.

```
29 bot_token = '5021737449:AAEyI39QmSgIIvuc3S_YKEAEUCNjta9kDTQ'
30 chat_ID = '1930098683'
31 mask_temp_bot = tg.Bot(token = bot_token)
```

Figura 49.- Token e ID en Python

3.2.9.2. Configuración de la Plataforma ThingSpeak

ThingSpeak es una plataforma IoT totalmente gratuito, para, lo cual, se crea una cuenta en MathWorks para acceder a ThingSpeak, ver anexo I.

Despúes de haber creado los canales en ThingSpeak, se crea automáticamente un Channel ID y un Api Key, los cuales, se debe colocar en la programación como, se puede ver figura 50 línea 34.

```
32 #----------ThingSpeak----------
33 channel_ID = "1635946"
34 apiKey = "D5LVRC05DFE7EPCG"
35 topic = "channels/" + channel_ID + "/publish/" + apiKey
36 mqttHost = "mqtt.thingspeak.com"
37 tTransport = "tcp"
38 tPort = 1883
39 tTLS = None
```

Figura 50.- Configuración de ThingSpeak

Elaborado por: Investigadora

3.2.9.3. Aplicación Móvil en App Inventor

La aplicación móvil, se realizó en el software App Inventor, es un sistema gratuito y, se puede crear aplicaciones de software para dispositivos móviles por medio de bloques, en este caso App Inventor la aplicación se conecta mediante Wifi, tiene la ventaja de ser enlazado con ThingSpeak por medio de la Api que son generadas en la plataforma de código abierto ver anexo J.

3.2.10. Implementación del prototipo

La implementación del prototipo permite verificar que los usuarios acaten una de las normas de bioseguridad como es el uso de mascarilla, además, se tiene el monitoreo de la temperatura que es visualizada por medio de la aplicación móvil de la persona a ingresar y del personal de control.

El personal encargado debe estar al pendiente de las personas que ingresen a cierto lugar para la medición de temperatura, ahora con el sistema automático, se tomará la medición de la temperatura acercando la muñeca de la mano al prototipo donde, se encuentra

conectado el sensor de temperatura infrarrojo, el prototipo adquiere los datos cada 4 segundos y son enviados a un archivo plano.

En la figura 51, se indica el circuito implementado, la persona se coloca frente a la cámara, la cual, captura la imagen, predice si usa o no mascarilla y se visualiza en la pantalla, se coloca la muñeca frente al sensor de temperatura para obtener la medición, y mediante el sensor infrarrojo se envía los datos en tiempo real.

<table>
<tr><td colspan="2">Con mascarilla y monitoreo de temperatura</td></tr>
<tr><td>  </td><td>  </td></tr>
<tr><td colspan="2">Sin mascarilla y monitoreo de temperatura</td></tr>
<tr><td>  </td><td>  </td></tr>
</table>

Figura 51.-Implementación del Sistema de Control

Elaborado por: Investigadora

3.3. Verificación de la hipótesis

3.3.1. Pruebas de Funcionamiento

Para las pruebas de funcionamiento, se utiliza dataset conjunto de imágenes para el análisis de los datos adquiridos y el uso de mascarilla.

Se utilizó un dataset de 300 imágenes y un valor de confianza mayor a 0.3 para la primera prueba, donde se obtuvo muchas inconsistencias en los resultados, la cámara capturó la imagen de una persona sin mascarilla que utiliza lentes dando un resultado erróneo en la pantalla porque reconoce los lentes como mascarilla. En la segunda prueba, se utilizó un dataset de 700 imágenes con un valor de confianza mayor a 0.8, también se obtuvo errores, proporcionando un reconocimiento incorrecto por lo que detectó a la persona sin mascarilla como persona con mascarilla, finalmente, se empleó un dataset de 1915 imágenes y un valor de confiabilidad mayor a 0.5, el cual, reconoce adecuadamente el uso de mascarilla, asimismo, se realizó pruebas con una iluminación alta donde el sistema no pudo distinguir correctamente el rostro, por el brillo que se genera en el área, en cambio, con una iluminación media se detectó el uso de mascarilla adecuadamente, y si la iluminación es completamente baja no se distingue el rostro frente a la cámara.

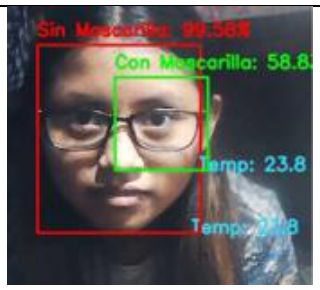
Prueba 1-Dataset de 300 imágenes	Prueba 2-Dataset de 700 imágenes	Prueba 3.- Dataset de 1915 imágenes
		
Valor de confiabilidad >0.3 Reconocimiento incorrecto	Valor de confiabilidad >0.8 Reconocimiento incorrecto	Valor de confiabilidad >0.5 Reconocimiento correcto

Figura 52.- Pruebas para la detección de mascarillas

Elaborado por: Investigadora

Para la adquisición de temperatura, se realizó varias pruebas, la distancia de medición del sensor es de 2cm-5cm según los datos técnicos del sensor, entonces el sensor mlx90614 se colocó a la distancia máxima que es 5 cm pero se obtuvo errores de medición, no se tenía una lectura adecuada del sensor de temperatura, por lo cual, se comprobó con una distancia de 2 cm, sin embargo la respuesta de temperatura variaba y no coincidía con el

termómetro digital, para lo tanto, se decidió colocar a un 1 cm de distancia, para que la medición de temperatura concuerde con la información obtenida por el termómetro digital, además, el sensor de temperatura daba como resultado un valor estable, lo que quiere decir que el dato se leía una sola vez, por lo tanto, se colocó la variable de temperatura en una función para que los datos se actualicen constantemente.

3.3.2. Resultados

Confiabilidad del dispositivo

Se puede determinar la confiabilidad de un dispositivo mediante el coeficiente de Cronbach, el cual, oscila entre valores de 0 y 1, el valor aceptable para el coeficiente es de 0.7, en general los valores, que se admiten son de 0,80 y 0,90 en caso, que se obtenga un valor negativo existe inconsistencia de la escala o error en el cálculo [45].

Fórmula del coeficiente de Cronbach

$$\alpha = \frac{K}{K-1}\left[1 - \frac{\sum V_i}{V_t}\right] \tag{7}$$

Donde:

α: Coeficiente de Cronbach

K: Número de muestras

V_i: Varianza Individual

V_t: Varianza Total Intentos

En la tabla 18, se puede observar el análisis de temperatura obtenido por el termómetro digital y por el prototipo. Se determinó el coeficiente de confiabilidad en cada uno de los dispositivos donde el termómetro digital tiene una confiablidad de 0,63 que pertenece a un rango de confiabilidad buena, en cambio, el prototipo adquirió un valor de 0,42, el cual, es un rango de confiabilidad moderado, ver anexo K.

Tabla 18.-Temperatura del termómetro digital y del prototipo

Intentos	Temperatura					
	Día 1		Día 2		Día 3	
	Termómetro Digital	Prototipo	Termómetro Digital	Prototipo	Termómetro Digital	Prototipo
1	36,6	36,7	36,4	36,5	36,4	36,6
2	36,5	35,6	36,4	36,4	36,5	36,2
3	36,7	35,4	36,5	36,7	36,4	36,4
4	36,6	36,4	36,3	36,6	36,3	36,4
5	36,4	36,6	36,4	36,5	36,3	36,5
6	36,5	37,3	36,3	36,6	36,4	36,4
7	36,4	36,5	36,4	36,4	36,3	36,1
8	36,5	36,6	36,3	36,5	36,3	36,3
9	37,2	37,2	36,4	36,6	36,4	36,7
10	37,1	37,4	36,4	36,7	36,4	36,6
11	36,9	37,2	36,5	36,6	36,5	36,6
12	36,7	37,1	36,4	36,6	36,8	36,9
13	36,2	36,5	36,2	36,3	36,3	36,4
14	36,6	36,4	36,7	37,1	36,9	37,1
15	36,6	36,3	37,2	37,4	37,2	37,1
Varianza	0,06	0,32	0,05	0,07	0,07	0,08
Promedio	36,63	36,61	36,45	36,63	36,49	36,55

Elaborado por: Investigadora

En la tabla 19, se muestran los datos de precisión adquiridos por el prototipo al detectar el uso correcto de mascarilla, donde el porcentaje de precisión de usuarios con mascarilla es de 95.89 %, en cambio el porcentaje de precisión de usuarios sin mascarilla es de 99.58, información relativamente alta y factible para su ejecución.

Tabla 19.-Precisión del Uso de mascarilla

Usuario	Precisión % Con Mascarilla	Precisión % Sin Mascarilla
1	96.22	99.90
2	96.48	99.96
3	98.71	97.36
4	99.90	99.67
5	91.50	99.97
6	95.49	99.62
7	94.42	99.96
8	91.73	99.51
9	98.87	99.87
10	95.63	100
Promedio	95.89	99.58

Elaborado por: Investigadora

Matriz de confusión

La matriz de confusión permite observar cómo se desempeña un algoritmo de aprendizaje supervisado, esta matriz es utilizada para localizar la precisión y exactitud del modelo; cada una de las filas simboliza a las instancias en la clase real, en cambio las columnas representan el número de predicciones de cada clase [46].

Tabla 20.-Términos asociados con la Matriz de confusión [46].

VP	Verdadero Positivo	Persona con mascarilla y el modelo detectó con mascarilla.
FP	Falso Positivo	Persona que no tiene mascarilla y el modelo detectó con mascarilla.
FN	Falso Negativo	Persona con mascarilla y el modelo detectó con mascarilla.
VN	Verdadero Negativo	Persona que no tiene mascarilla y el modelo detectó con mascarilla.

Elaborado por: Investigadora

Tabla 21.-Matriz de confusión [46].

	Predicción		
Actual		**Positivo**	**Negativo**
	Positivo	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativo	Falsos Positivos (FP)	Verdaderos Negativos (VN)

Elaborado por: Investigadora

Precisión

$$Precisión = \frac{VP}{VP + FP} \tag{8}$$

Sensibilidad

$$Sensibilidad = \frac{VP}{VP + FN} \tag{9}$$

Valor de Referencia

$$Valor\ deReferencia = 2x\frac{Precision\ x\ Sensibilidad}{Precision + Sensibilidad} \tag{10}$$

Resultados de la matriz de confusión

Con un dataset de 300 imágenes, se obtuvieron los siguientes resultados.

VP=189	**FP=111**	**FN=99**	**VN=201**
Precisión=0,63	Sensibilidad=0,65	Valor de Referencia=0,64	

Con un dataset de 700 imágenes, se obtuvieron los siguientes resultados.

VP=595	**FP=105**	**FN=100**	**VN=600**
Precisión=0,83	Sensibilidad=0,85	Valor de Referencia=0,83	

Con un dataset de 1915 imágenes, se obtuvieron los siguientes resultados.

VP=1890	**FP=25**	**FN=90**	**VN=1825**
Precisión=0,98	Sensibilidad=0,95	Valor de Referencia=0,96	

De acuerdo a los datos obtenidos en la matriz de precisión con diferentes cantidades de dataset se considera que con un dataset de 1915 imágenes por clase es un sistema factible, con una precisión de 98%.

Los resultados obtenidos en tiempo real para la detección del uso de mascarillas, se realizaron con diferentes tipos de mascarilla como se indica en la figura 53.

Mascarilla negra
Con mascarilla: 99,70%

Mascarilla KN95
Con mascarilla: 99,96%

Mascarilla Quirúrgica
Con mascarilla: 99.89%

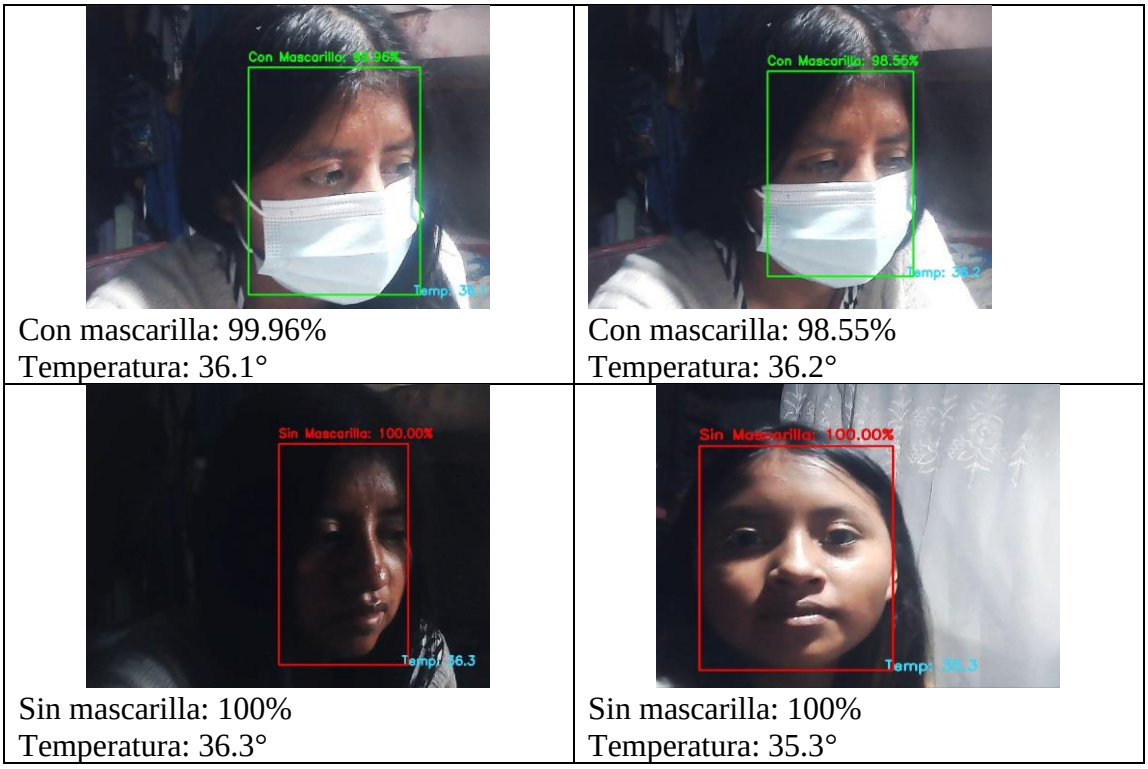
Mascarilla con figuras
Con mascarilla:99.98%

Sin mascarilla: 96.52%

Sin mascarilla: 76.98%

Figura 53.-Resultados de detección del uso de mascarillas
Elaborado por: Investigadora

En la figura 54, se muestra los resultados de la mascarilla conjuntamente con la medición de temperatura y poca luminosidad teniendo un porcentaje alto de precisión cuando la persona no está con mascarilla.



Con mascarilla: 99.96%
Temperatura: 36.1°

Con mascarilla: 98.55%
Temperatura: 36.2°

Sin mascarilla: 100%
Temperatura: 36.3°

Sin mascarilla: 100%
Temperatura: 35.3°

Figura 54.-Resultados con poca luminosidad
Elaborado por: Investigadora

Aplicación telegram

Los datos son adquiridos por el sensor y la cámara web, los cuales, son enviados en tiempo real a la aplicación de mensajería telegram, como se puede observar en la figura 55 emitiendo un mensaje “no puede pasar” en caso de que la persona este sin mascarilla o haya sobrepasado el valor nominal de temperatura 37.5 ºC, en cambio se emite un mensaje de “puede pasar” cuando tiene colocada correctamente la mascarilla y una temperatura normal.

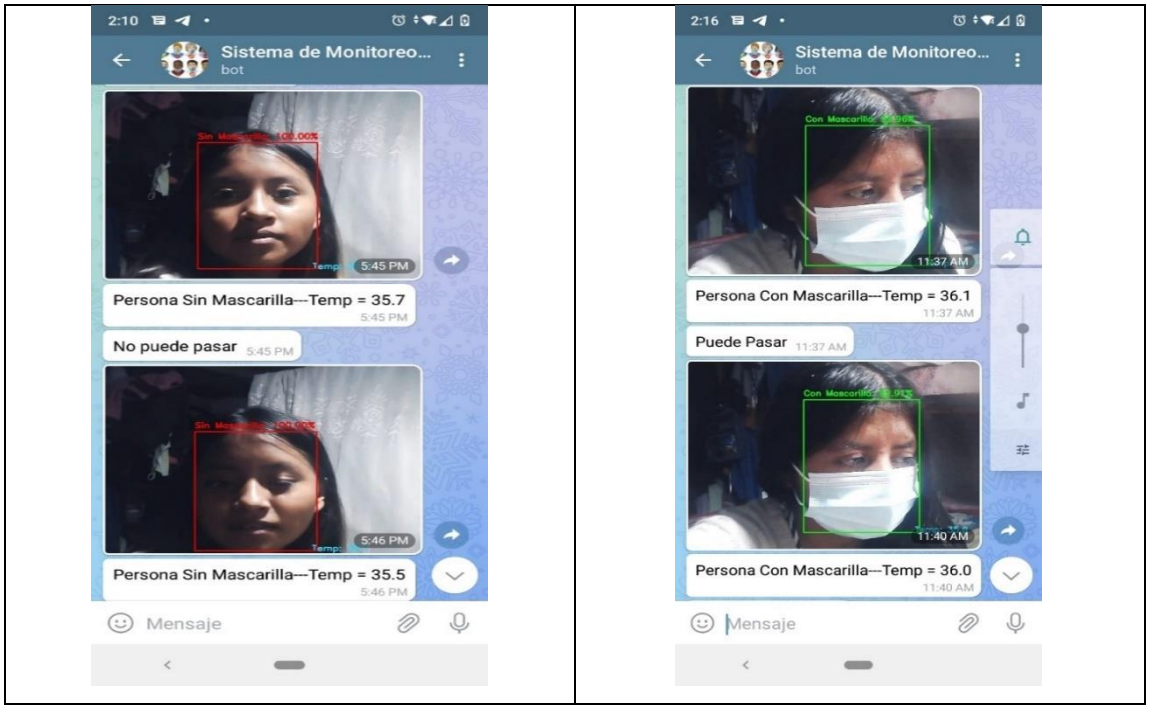


Figura 55.-Datos en Aplicación de Mensajería

Elaborado por: Investigadora

Herramienta ThingSpeak

En la figura 56, se observa los datos estadísticos y gráficos de la temperatura y uso de mascarilla con hora, día y mes, en la detección de mascarilla el valor 1 representa que está con mascarilla y 0 si esta sin mascarilla, el gráfico del reloj, se activa cuando detecte que la temperatura ha superado su valor nominal, el gráfico del led, se enciende cuando esté colocado bien la mascarilla caso contrario se apaga, además, está colocado la ubicación de donde, se encuentra instalado el prototipo, ver datos solamente por el administrador, con el siguiente link: https://thingspeak.com/channels/1635946/private_show

Si el usuario desea visualizar solo los datos estadísticos de temperatura y mascarilla ingresar al siguiente link: https://thingspeak.com/channels/1635946

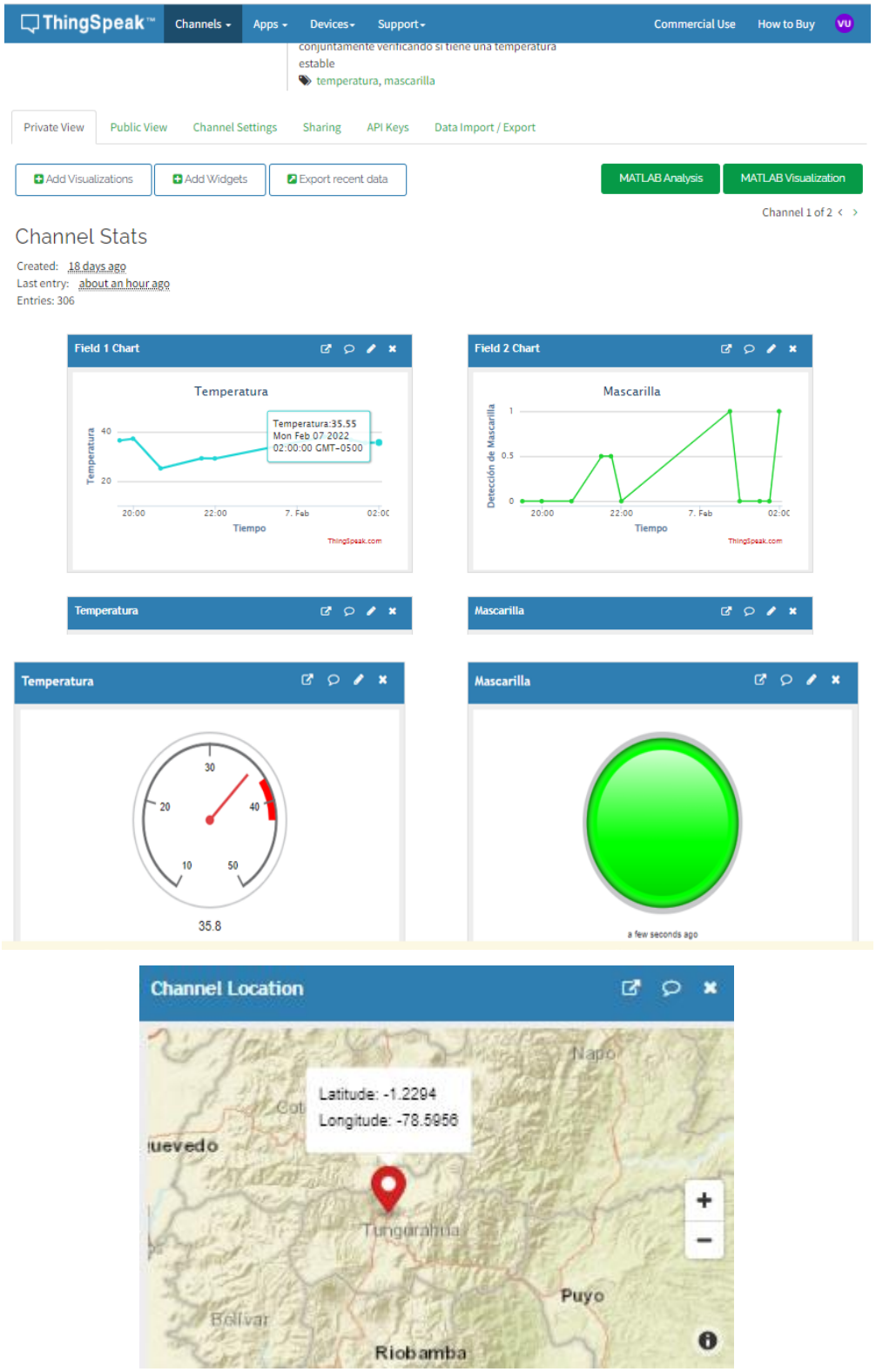


Figura 56.- Visualización de los datos en ThingSpeak

Elaborado por: Investigadora

App Inventor Visualización

Una vez realizada la App, al iniciarla se obtiene la pantalla con los datos de temperatura y detección de mascarilla como se observa en la figura 57.

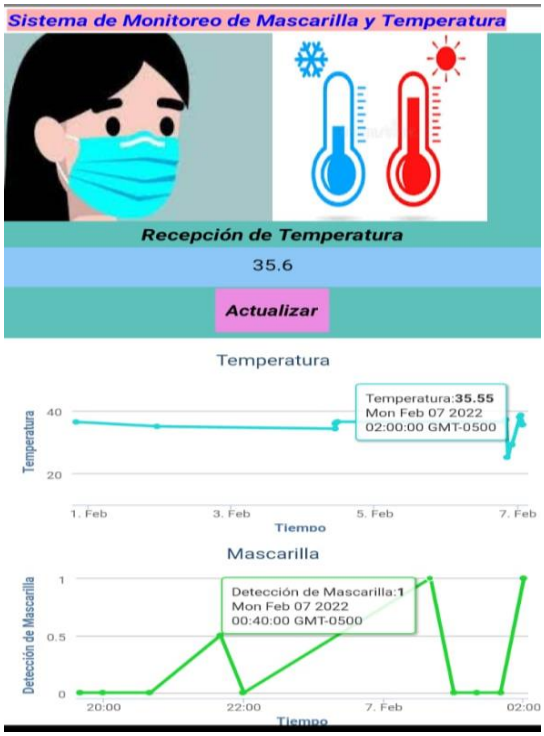


Figura 57.-Pantalla de la App

Para comprobar la App debe tener instalado en el dispositivo móvil la aplicación MIT AI2 Companion, abrir el archivo.APK Sistema_de_monitoreo_de_mascarilla_y_temperatura, habilitar la opción para admitir la instalación de aplicaciones de fuentes desconocidas y se inicia la aplicación del proyecto en el móvil.

3.3.3. Costo del prototipo

Precios de Hardware

Se obtuvo diferentes componentes que fueron utilizados para el desarrollo del prototipo los cuales son descritos en la tabla 22.

Tabla 22.-Precios del Hardware del prototipo

Detalle	Cantidad	Precio Unitario	Total
Raspberry pi 3	1	$60,00	$60,00
Cámara USB	1	$15,00	$15,00
Sensor MLX90614	1	$15,00	$15,00
Sensor Infrarrojo IR FC-51	1	$2,00	$2,00
Led	2	$0.10	$0,20
Resistencia 220 ohmios	2	$0,10	$0,20
Ventilador	1	$3.00	$3,00
Pantalla	1	$30,00	$30,00
Cables	Varios	$0,50	$0,50
Baquelita	1	$1,00	$1,00
Estaño	1	$0,75	$0,75
Cloruro Férrico	1	$0,75	$0,75
Espadines	2	$0,25	$0,50
Case	1	$8,00	$8,00
		Subtotal	$136,90
		I.V. A	$16,43
		Total	$153,33

Elaborado por: Investigadora

Precio del Software

Los programas empleados en el desarrollo del prototipo son gratuitos los cuales son:

*Thonny Python IDE

*BotFather de Telegram

*ThingSpeak

*App Inventor

Precio de mano de obra

Para determinar la mano de obra se toma en cuenta el sueldo de un Ingeniero en Electrónica y Comunicaciones que según el Ministerio de Trabajo es de $457,52 y puede aumentar de acuerdo al rango en que se desempeñe y el tiempo de ejecución del proyecto lo cual fue 6 meses teniendo un total de $2745.12.

CAPÍTULO IV

CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

Al finalizar con el proyecto de titulación se alcanzó las siguientes conclusiones:

- Se analizó los componentes para el desarrollo del proyecto, por lo cual, se optó por la Raspberry Pi ordenador de tamaño reducido de bajo costo, la conexión se realizó mediante Wifi IEEE 802.11n que se utilizó para conectar otro dispositivo a la Raspberry Pi como el teléfono móvil, para la visualización de los datos en la pantalla, y tiene la suficiente capacidad de memoria RAM-1GB aceptable para la ejecución del proyecto, el sensor de temperatura MLX90614-ESFBAA fue la mejor opción pues mide la temperatura sin estar en contacto con la persona, es económico y tiene un precisión de +/-0.3 ℃ a una temperatura de 16 °C a 40 °C, aceptable para la ejecución del sistema y se utilizó una cámara web con una resolución de 1280x720 pixeles apta la captura de imágenes en la implementación del proyecto.

- Se creó un bot Sistema de Monitoreo de Mascarilla y Temperatura en la aplicación de mensajería de telegram para enviar las fotografías y la temperatura, se importa la librería de telegram y csv para guardar los datos que se genera en tiempo real, y se tiene como respuesta un mensaje de alerta “puede o no ingresar”, sistema controlado por el operario.

- Se utilizó el software ThingSpeak plataforma IoT de código abierto, que almacena, analiza, y visualiza los datos en tiempo real de la temperatura y mascarilla con la hora y fecha que ingresan las personas, se empleó la dirección

mqtt.thingspeak.com para la conexión de mensajes desde el servidor local a ThingSpeak a través del puerto TCP 1883, Protocolo de Control de Transmisión por lo que garantiza el envío de datos en el orden que fueron enviados , además, se realizó la aplicación móvil en app inventor para que el usuario pueda monitorear su temperatura en todo momento, la misma que se enlazó por medio de la Api que se genera en la plataforma IoT.

- Se determinó que las redes neuronales profundas tienen un alto valor de precisión para el reconocimiento de mascarilla en comparación con los algoritmos Eigenfaces, Fisherfaces, LBPH (Local Binary Pattern Histogram) no son tan precisas como las redes neuronales ya que detectan rostros donde no existe, el clasificador FaceNet tiene una precisión de 99.63% para la verificación de rostros, y la arquitectura MobileNetV2 tiene un bajo consumo de recursos, y de acuerdo a los datos obtenidos en la matriz de precisión, con un dataset de 1915 imágenes por clase de obtuvó una precisión de 98% lo que resulta factible la implementación del prototipo.

4.2 Recomendaciones

- Es necesario crear un propio interpretador de python, al instalar el sistema operativo, viene preinstalado por defecto varias librerías lo que ocasiona errores al momento de utilizarlas, para el desarrollo del reconocimiento facial del uso de mascarilla, se emplean versiones específicas para su ejecución como Python versión 3.7.3 con TensorFlow 2.0.0, si se instalaba una versión superior de TensorFlow ocasionaba errores ya que no eran compatibles.

- La base de datos debe contener la mayor cantidad de fotografías disponibles y un valor de confiabilidad mayor a 0.5 para que el reconocimiento del uso de mascarilla se realice sin errores, mientras más fotos existan en la base de datos se tendrá mayor precisión del reconocimiento.

- Se debe extraer la región de interés (ROI) de la imagen que se desea procesar, en Open CV se requiere un formato de lectura de color, la imagen se debe convertir de BGR a RGB con dimensiones de 224 x 224 para que la imagen sea procesada.

- Se debe importar todas las librerías en el entorno de desarrollo integrado para Python (Thonny) para que no exista errores al momento de ejecutar el programa, también hay que tomar en cuenta los pines de la Raspberry Pi al momento de realizar las conexiones porque el módulo RPI.GPIO usa dos formas de aplicarse, numeración física del pin o numeración Broadcom GPIO.

- Se debe ubicar de forma correcta de la cámara con un ángulo de 90° y tener una adecuada iluminación ya que son factores importantes al momento de realizar el reconocimiento, para que no se produzca errores en la detección del rostro con la mascarilla, también, para el funcionamiento adecuado del sensor de temperatura se coloca la muñeca hasta un 1 cm de distancia para tener resultados precisos.

- Debido a que la pandemia COVID-19, está cada vez más controlada, el presente proyecto de investigación se puede emplear para el control del uso de mascarilla del personal médico como los podólogos en el servicio de quiropodia, y el control de la temperatura se puede emplear para el monitoreo de zonas pecuarias debido a que el sensor mlx90614, también, mide la temperatura ambiente y se puede modificar en la programación para aplicar en ese ámbito.

BIBLIOGRAFÍA

[1] Gutiérrez Segales Juan Pablo, «Implementación de un prototipo de una red inalámbrica de sensores biomédicos, para la adquisición y almacenamiento de datos, usando cloud computing, para pacientes en casa», Perú-Arequipa, 2019.

[2] R. A. Rahman, U. R. Hashim, y S. Ahmad, «IoT based temperature and humidity monitoring framework», *Bulletin of Electrical Engineering and Informatics*, vol. 9, n.º 1, Art. n.º 1, feb. 2020, doi: 10.11591/eei.v9i1.1557.

[3] Montesdeoca Ordoñez Erik Daniel, «Implementación de un Sistema de Reconocimiento del Uso de mascarillas como medida de precaución contra el covid19 usando Deep Learning», Machala, 2020.

[4] A. J. Pereira, T. P. Donadon Homem, y F. Oliveira Teixeira, «Aplicación de inteligencia artificial para monitorear el uso de mascarillas de protección», *Revista Científica General José María Córdova*, vol. 19, n.º 33, Art. n.º 33, ene. 2021, doi: 10.21830/19006586.725.

[5] M. L. Hoang, M. Carratù, V. Paciello, y A. Pietrosanto, «Body Temperature—Indoor Condition Monitor and Activity Recognition by MEMS Accelerometer Based on IoT-Alert System for People in Quarantine Due to COVID-19», *Sensors*, vol. 21, n.º 7, Art. n.º 7, ene. 2021, doi: 10.3390/s21072313.

[6] Elflein Juan, «Coronavirus deaths worldwide by country», *Statista*. https://www.statista.com/statistics/1093256/novel-coronavirus-2019ncov-deaths-worldwide-by-country/ (accedido 14 de enero de 2022).

[7] Statista Research Department, «Coronavirus en Latinoamérica: países con más casos», *Statista*. https://es.statista.com/estadisticas/1105121/numero-casos-covid-19-america-latina-caribe-pais/ (accedido 14 de enero de 2022).

[8] Organización Mundial de la Salud, «Coronavirus». https://www.who.int/westernpacific/health-topics/coronavirus (accedido 21 de diciembre de 2021).

[9] R. Hernández Gaviño, *Introducción a los Sistemas de control: conceptos, aplicaciones y simulación con MATLAB*. 2010.

[10] P. Singh y A. Manure, «Neural Networks and Deep Learning with TensorFlow», en *Learn TensorFlow 2.0-Implement Machine Learning and Deep Learning Models with Python*, India, 2020.

[11] F. Ramírez, «Las matemáticas del Machine Learning: Redes Neuronales (Parte I) - Think Big Empresas», *Think Big*, 26 de noviembre de 2019. https://empresas.blogthinkbig.com/las-matematicas-del-machine-learning-redes-neuronales-parte-i/ (accedido 26 de enero de 2022).

[12] H. Galán y A. Martínez, «Inteligencia artificial. Redes neuronales y aplicaciones», Madrid. [En línea]. Disponible en: https://www.it.uc3m.es/jvillena/irc/practicas/10-11/06mem.pdf

[13] MathWords, «¿Qué es una red neuronal?» https://la.mathworks.com/discovery/neural-network.html (accedido 25 de enero de 2022).

[14] P. Rodríguez y J. Carlos, «Plataforma cloud de monitoreo, adquisición, visualización y predicción de la contaminación del aire, basado en modelos de redes neuronales artificiales», Universidad Técnica de Ambato, 2018. Accedido: 1 de febrero de 2022. [En línea]. Disponible en: https://repositorio.uta.edu.ec/jspui/handle/123456789/28482

[15] D. A. Gavilanes Proaño, «Sistema de monitoreo apícola mediante el uso de redes neuronales artificiales para identificar la variación de población», Universidad Técnica de Ambato, 2020. Accedido: 1 de febrero de 2022. [En línea]. Disponible en: https://repositorio.uta.edu.ec/jspui/handle/123456789/31507

[16] MathWords, «Redes neuronales convolucionales». https://la.mathworks.com/discovery/convolutional-neural-network-matlab.html (accedido 2 de febrero de 2022).

[17] F. A. V. Naranjo, A. P. Arcos, F. P. Baño, y H. W. Baño, «APP móvil para el reconocimiento facial», *Ciencias de la Ingeniería y Aplicadas*, vol. 1, n.º 2, Art. n.º 2, mar. 2018.

[18] N. Quispe y R. Jimena, «Reconocimiento facial aplicado a la autentificación de usuarios en cursos online de la carrera de docencia en informática de la Facultad de Ciencias Humanas y de la Educación de la Universidad Técnica de Ambato»,

Universidad Técnica de Ambato, 2016. Accedido: 1 de febrero de 2022. [En línea]. Disponible en: https://repositorio.uta.edu.ec/jspui/handle/123456789/24037

[19] R. D. Castro Arias, «Sistema de control de acceso al personal de la Lavadora de Jeans Fashion mediante reconocimiento facial», Universidad Técnica de Ambato, 2016. Accedido: 1 de febrero de 2022. [En línea]. Disponible en: https://repositorio.uta.edu.ec/jspui/handle/123456789/20347

[20] A. B. Amaya Arcos, «Sistema alternativo de seguridad vehicular basado en reconocimiento facial», Universidad Técnica de Ambato, 2015. Accedido: 27 de enero de 2022. [En línea]. Disponible en: https://repositorio.uta.edu.ec/jspui/handle/123456789/10586

[21] J. F. Diaz Diaz y R. H. Di Pasquale, «Reconocimiento facial: FaceNet y AWS Rekognition», Argentina, mar. 2020. Accedido: 2 de febrero de 2022. [En línea]. Disponible en: https://repositorio.uca.edu.ar/bitstream/123456789/11212/1/reconocimiento-facial-facenet-aws.pdf

[22] F. Schroff, D. Kalenichenko, y J. Philbin, «FaceNet: A Unified Embedding for Face Recognition and Clustering», *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815-823, jun. 2015, doi: 10.1109/CVPR.2015.7298682.

[23] M. Sandler y A. Howard, «MobileNetV2: The Next Generation of On-Device Computer Vision Networks», *Google AI Blog*, 3 de abril de 2018. http://ai.googleblog.com/2018/04/mobilenetv2-next-generation-of-on.html (accedido 2 de febrero de 2022).

[24] Eugenio López Aldea, «Hardware de la Placa Raspberry Pi», en *Raspberry Pi Fundamentos y aplicaciones*, España: RA-MA, 2017, pp. 63-177.

[25] O. González y J. Guadalupe, «Reconocimiento de objetos utilizando Open CV y Python en una Raspberry Pi 2 en una tlapalería», feb. 2017, Accedido: 27 de enero de 2022. [En línea]. Disponible en: http://ri.uaemex.mx/handle/20.500.11799/68150

[26] H. Rodríguez, «¿Qué es OpenCV y para qué sirve?», *https://www.crehana.com*. https://www.crehana.com/ec/blog/desarrollo-web/que-es-opencv/ (accedido 27 de enero de 2022).

[27] J. M. Silva Caamaño, «Desarrollo de redes de neuronas profundas para procesado de imagen», 2019, Accedido: 28 de enero de 2022. [En línea]. Disponible en: https://ruc.udc.es/dspace/handle/2183/24116

[28] J. D. Alonso Sierra y D. L. Castaño Saavedra, «Sistema de reconocimiento facial para control de acceso a viviendas», 2019, Accedido: 27 de enero de 2022. [En línea]. Disponible en: https://repository.ucatolica.edu.co/handle/10983/24032

[29] R. Aragonés y D. Rincón, «La higiene en cuidados intensivos. Control de temperatura», en *Manual de Cuidados Intensivos para Enfermería*, España: Medica Panamericana, 2020, pp. 61-62.

[30] R. Nall, «Temperatura corporal: Rangos normales en adultos y niños», 9 de junio de 2020. https://www.medicalnewstoday.com/articles/es/temperatura-normal-del-cuerpo (accedido 8 de diciembre de 2021).

[31] «IoT Analytics - ThingSpeak Internet of Things». https://thingspeak.com/ (accedido 2 de febrero de 2022).

[32] R. Concepción, «Tarjetas de Desarrollo (Episodio #8)», *rjconcepcion*, 19 de diciembre de 2019. https://www.rjconcepcion.com/podcast/tarjetas-de-desarrollo-episodio-8/ (accedido 18 de enero de 2022).

[33] Sinovoip, «Banana pi BPI-M64 vs Raspberry pi 3 vs odroid vs Pine64», *banana pi single board computer open source project official forum SinoVoip BPI team*, 1 de agosto de 2016. https://forum.banana-pi.org/t/banana-pi-bpi-m64-vs-raspberry-pi-3-vs-odroid-vs-pine64/2070 (accedido 17 de enero de 2022).

[34] Rodrigo Alonso, «Raspberry Pi vs Arduino: características técnicas y diferencias», *HardZone*, 3 de abril de 2020. https://hardzone.es/reportajes/comparativas/raspberry-pi-vs-arduino/ (accedido 17 de enero de 2022).

[35] Dynamo Electronics, «Cámara térmica AMG8833 IR», *DynamoElectronics*. https://dynamoelectronics.com/tienda/camara-thermica-amg8833-ir/ (accedido 18 de enero de 2022).

[36] Mouser Electronics, «MLX90614ESF-DCI-000-SP Melexis | Mouser», *Mouser Electronics*. https://www.mouser.ec/ProductDetail/482-90614ESFDCI000SP (accedido 18 de enero de 2022).

[37] «Sensores de Movimiento PIR HC-SR501 / HC-SR505», *UNIT Electronics*. https://uelectronics.com/producto/sensores-de-movimiento-pir-hc-sr501-hc-sr505/ (accedido 26 de enero de 2022).

[38] Tecnopura, «Módulo sensor infrarrojo evasor de obstáculos / seguidor de línea», *Tecnopura*. https://www.tecnopura.com/producto/modulo-sensor-infrarrojo-evasor-de-obstaculos-seguidor-de-linea/ (accedido 18 de enero de 2022).

[39] MediaPrice, «Camara Web Pc HD 720P Webcam Windows MAC 720px reales por solo $13.99 · MediaPrice Ecuador», *MediaPrice*. https://www.mediaprice.com.ec/producto/camara-web-pc-hd-720p-1280x720-webcam/ (accedido 18 de enero de 2022).

[40] LDLC experience, «Logitech HD Webcam C270 - Webcam Logitech en LDLC». https://www.ldlc.com/es-es/ficha/PB00213751.html (accedido 18 de enero de 2022).

[41] ADESSO, «Adesso CyberTrack H4 1080p USB Webcam with Built-in Microphone». https://www.bhphotovideo.com/c/product/1567510-REG/adesso_cybertrackh4_cybertrack_h4_1080p_desktop.html (accedido 18 de enero de 2022).

[42] C. Villalba Martín, A. Urquía Moraleda, y M. Á. Rubio Gonzáles, «Fundamentos de programación», en *Lenguaje de programación*, Madrid, 2021, pp. 1-50. Accedido: 22 de enero de 2022. [En línea]. Disponible en: https://elibro.net/es/ereader/uta/184827

[43] J. R. Lajara Vizcaíno y J. Pelegri Sebastiá, «Introducción a Labview. Entorno», en *Labview: Entorno gráfico de programación*, México: AlfaOmega, 2007, pp. 1-6. Accedido: 22 de enero de 2022. [En línea]. Disponible en: https://elibro.net/es/ereader/uta/35715

[44] J. Rodríguez García, «El programa Matlab», en *Matlab: guía de aprendizaje*, Primera., Argentina: Científica Universitaria, 2020, pp. 1-7. Accedido: 22 de enero de 2022. [En línea]. Disponible en: https://elibro.net/es/ereader/uta/181970

[45] J. A. Bojórquez Molina, L. López Aranda, M. E. Hernández Flores, y E. Jiménez López, «Utilización del alfa de Cronbach para validar la confiabilidad de un instrumento de medición de satisfacción del estudiante en el uso del software

Minitab», *Eleventh LACCEI Latin American and Caribbean Conference for Engineering and Technology*, pp. 1-9, 2013.

[46] A. G. Vargas Machuca Del Salto y V. S. Manzano Villafuerte, «Sistema de detección de intrusos basado en machine learning», Universidad Técnica de Ambato, 2021. Accedido: 1 de febrero de 2022. [En línea]. Disponible en: https://repositorio.uta.edu.ec/jspui/handle/123456789/34028

ANEXOS

Anexo A: Raspberry pi 3B

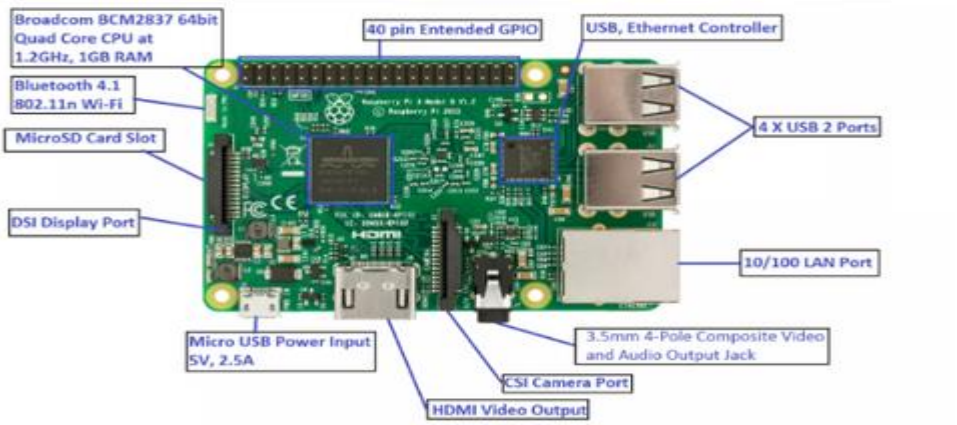


Figura 58.-Partes de la Raspberry Pi 3B

Especificaciones Generales:

	RASPBERRY PI 3 MODEL B
PROCESADOR	Broadcom BCM2837, Cortex-A53 (ARMv8) 64-bit SoC
FRECUENCIA DE RELOJ	1,2 GHz
GPU	VideoCore IV 400 MHz
MEMORIA	1GB LPDDR2 SDRAM
CONECTIVIDAD INALÁMBRICA	2.4GHz IEEE 802.11.b/g/n Bluetooth 4.1
CONECTIVIDAD DE RED	Fast Ethernet 10/100 Gbps
PUERTOS	GPIO 40 pines HDMI 4 x USB 2.0 CSI (cámara Raspberry Pi) DSI (pantalla táctil) Toma auriculares / vídeo compuesto Micro SD Micro USB (alimentación)
FECHA DE LANZAMIENTO	29/2/2016

Figura 59.-Especificaciones de la Raspberry Pi 3B

Diagrama de Pines

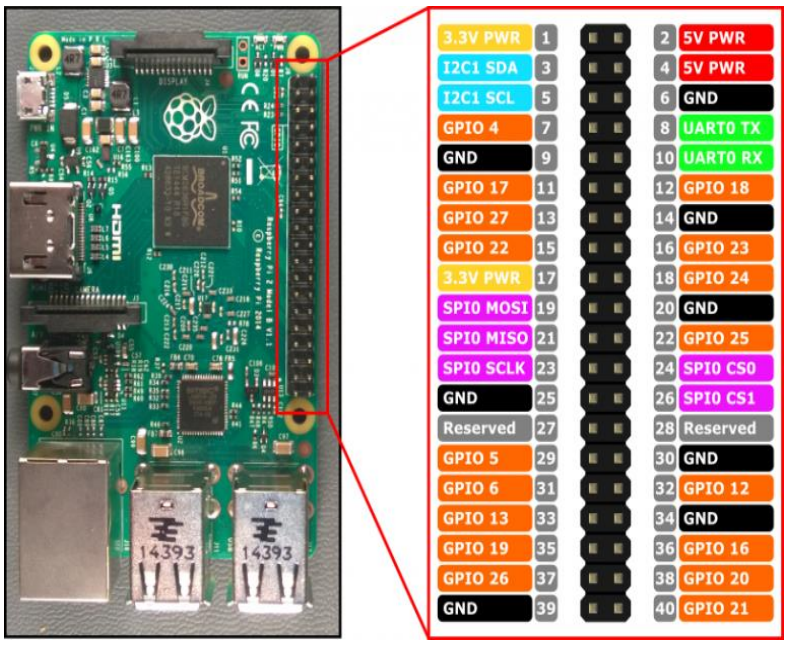


Figura 60.-Diagrama de Pines de la Raspberry Pi 3B

Anexo B: Sensor MLX90614-ESFBAA

Figura 61.- Sensor MLX90614-ESFBAA

ESPECIFICACIONES Y CARACTERÍSTICAS

- Tipo: Sensor de temperatura sin contacto
- Modulo: GY-906
- Sensor: MLX90614ESF-BAA
- Voltaje de Operación: 3.3V – 5V DC
- Rango de temperatura ambiente de trabajo: -40°C hasta +170°C
- Rango de temperatura de objeto: -70°C hasta +380°C
- Precisión: ±0.5°C
- Angulo de visión: 90° (FOV)
- Distancia para temperatura de objeto: Min. 2cm y Max. 3cm
- ADC incorporado de 17 bits
- Salida de PWM: 10 bits
- Protocolo de comunicación SMBUS (I2C)
- No necesita componentes adicionales

Figura 62.- Especificaciones del Sensor MLX90614-ESFBAA

MLX90614Bxx

All parameters are preliminary for T_A = 25 °C, **V_{DD} =3V** (unless otherwise specified)

Parameter	Symbol	Test Conditions	Min	Typ	Max	Units
Supplies						
External supply	V_{DD}		2.4	3	3.6	V
Supply current	I_{DD}	No load		1	2	mA
Supply current (programming)	I_{DDpr}	No load, erase/write EEPROM operations		1.5	2.5	mA
Power-down supply current	Isleep	no load	1	2.5	5	uA
Power-down supply current	Isleep	Full temperature range	1	2.5	6	uA
Power On Reset						
POR level	V_{POR}	Power-up, power-down and brown-out	1.6	1.85	2.1	V
V_{DD} rise time	T_{POR}	Ensure POR signal			1	ms
Output valid	Tvalid	After POR		0.15		s

Pulse width modulation[1]						
PWM resolution	PWMres	Data band		10		bit
PWM output period	$PWM_{T,def}$	Factory default, internal oscillator factory calibrated		1.024		ms
PWM period stability	$dPWM_T$	Internal oscillator factory calibrated, over the entire operation range and supply voltage	-4		+4	%
Output high Level	PWM_{HI}	I_{source} = 2 mA	V_{DD}-0.25			V
Output low Level	PWM_{LO}	I_{sink} = 2 mA			V_{SS}+0.25	V
Output drive current	$Idrive_{PWM}$	Vout,H = V_{DD} - 0.8V		4.5		mA
Output sink current	$Isink_{PWM}$	Vout,L = 0.8V		11		mA
Output settling time	Tset	100 pF capacitive load, full operating Ta range			150	ns
Output settling time	$Tset_{RC}$	220 Ohm in series with 47nF load on the wire, full Ta operating range		500		ns

Figura 63.- Parámetros del Sensor MLX90614-ESFBAA

Diagrama de bloques

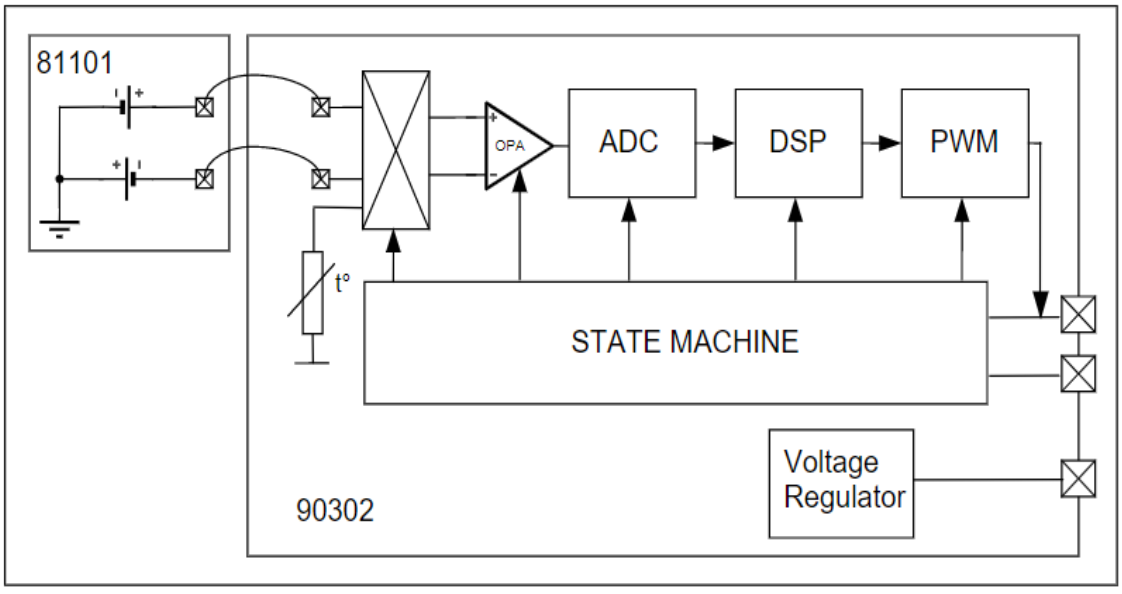


Figura 64.- Diagrama de Bloques del Sensor MLX90614-ESFBAA

Diagrama de Pines

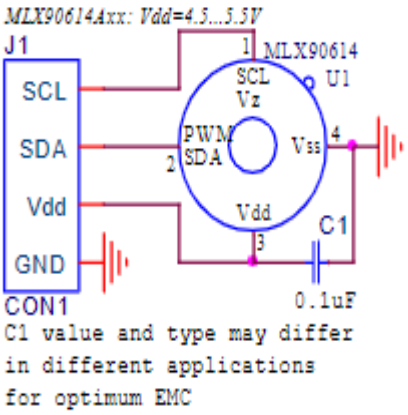


Figura 65.- Diagrama de Pines del Sensor MLX90614-ESFBAA

Pin Name	Function
VSS	Ground. The metal can is also connected to this pin.
SCL / Vz	Serial clock input for 2 wire communications protocol. 5.7V zener is available at this pin for connection of external bipolar transistor to MLX90614A to supply the device from external 8 -16V source.
PWM / SDA	Digital input / output. In normal mode the measured object temperature is available at this pin Pulse Width Modulated. In SMBus compatible mode automatically configured as open drain NMOS.
VDD	External supply voltage.

Figura 66.- Descripción de Pines del Sensor MLX90614-ESFBAA

Anexo C: Sensor Infrarrojo IR FC-51

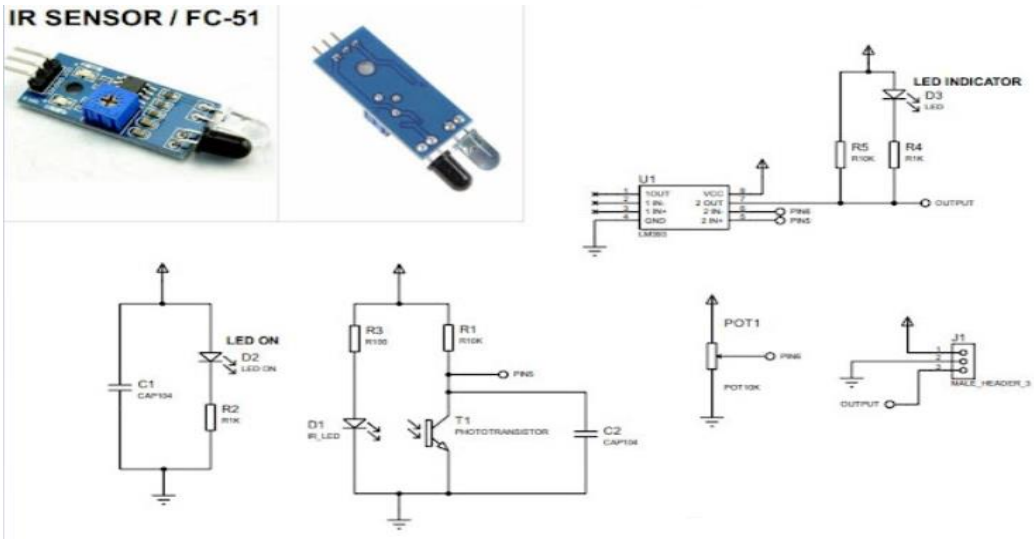


Figura 67.- Sensor Infrarrojo IR FC-51

Especificaciones Generales:

Especificaciones

- Función: Detector de obstáculos
- Chip principal: LM393
- Voltaje de alimentación mínimo: 3.3 V
- Voltaje de alimentación máximo: 5 V
- Distancia de detección mínima: 2 cm (Ajustable con el potenciómetro)
- Distancia detección máxima: 30 cm (Ajustable con el potenciómetro)
- Angulo de detección: 35°
- Distribución de pines:
  - Pin de alimentación VCC: 3.3 V - 5 V
  - Pin de alimentación 0 V: GND
  - Pin de OUT: Salida digital Pin
- Indicador de alimentación: LED rojo
- Indicador de salida digital: LED verde
- Dimensiones: 31 mm X 15 mm
- Número de pines: 3
- Modelo: FC-51
- Modelo: E14
- Modelo: OKY3127

Figura 68.- Especificaciones Generales del Sensor Infrarrojo IR FC-51

Anexo D: Instalación del Sistema Operativo Raspbian

Ingresa la tarjeta SD con la microSD a la PC selecciona el sistema operativo en este caso Raspberry Pi OS y comienza a instalar.

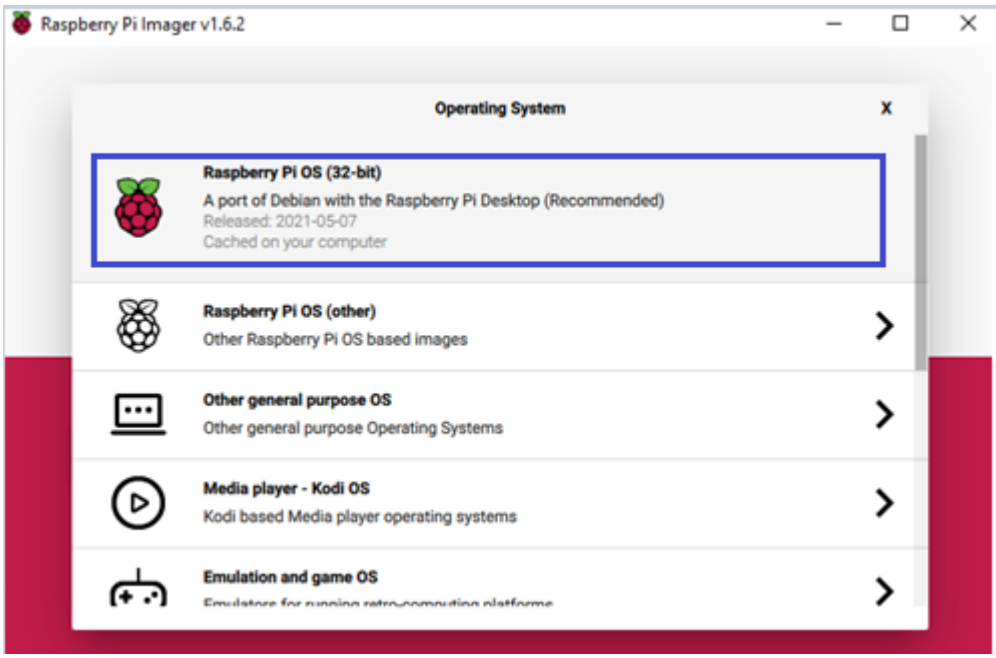


Figura 69.- Selección del Sistema Operativo

Activa SSH en la micro SD para configurar de forma remota.

Figura 70.- Activar SSH

Procede a extraer la micro SD e ingresa a la ranura de la Raspberry Pi, habilita VNC server con sudo raspi-config.

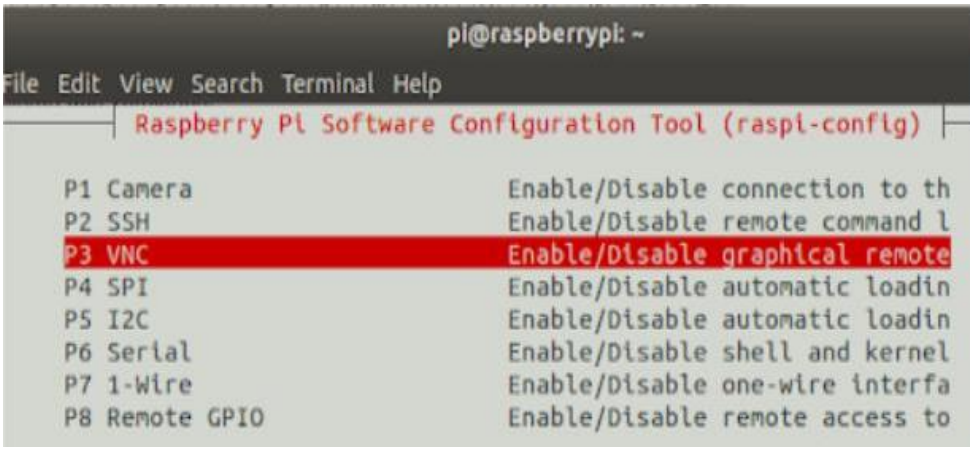


Figura 71.- Habilitar VNC Server

Con el software Advanced Ip Scanner observa la Ip que tiene la Raspberry Pi.

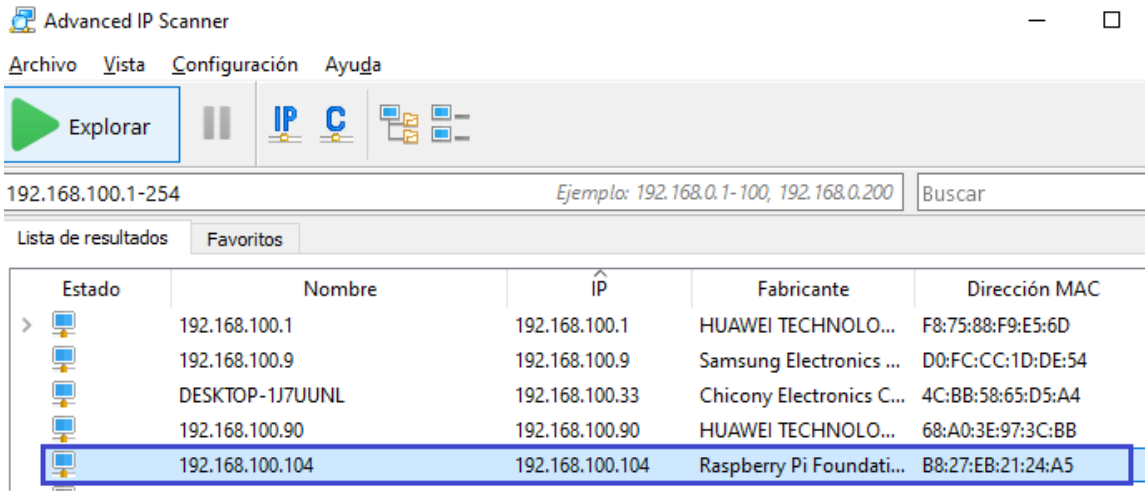


Figura 72.- Selección de la Ip de la Raspberry Pi

Coloca el nombre de usuario y la contraseña de la Raspberry Pi.

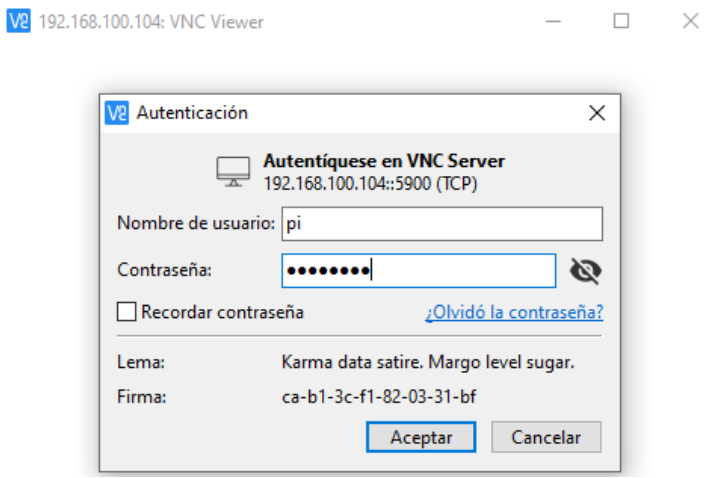


Figura 73.- Ingresar a la Raspberry Pi

Se conecta graficamente al sistema operativo Raspbian en la Raspberry Pi.

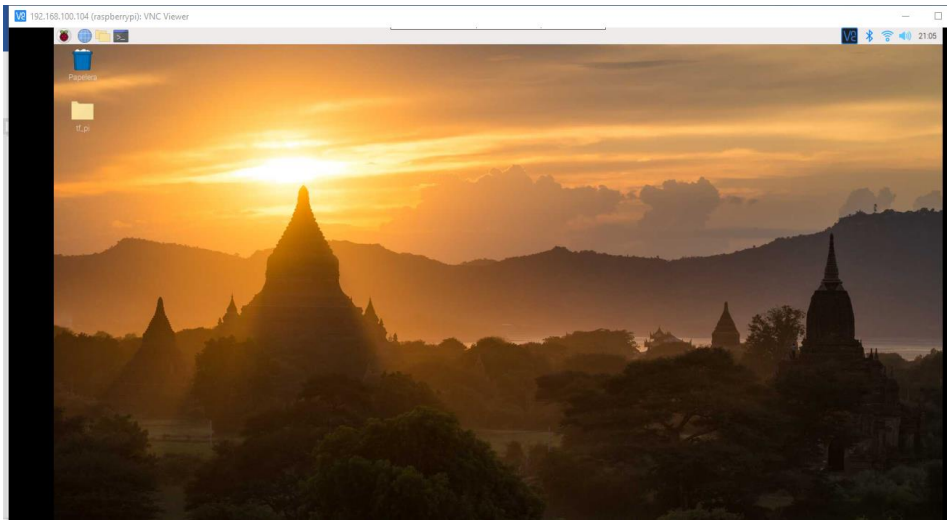


Figura 74.- Pantalla Principal de la Raspberry Pi.

Anexo E: Instalación de dependencias para el sistema

Ingrese al directorio:

-cd Desktop

Cree una carpeta

-mkdir tf_pi

Ingrese al directorio

-cd tf_pi

Primero se crea un propio interpretador de python debido a que se necesita aplicar librerías específicas para la ejecución del proyecto y no ocasionar errores al momento de actualizar el sistema.

Instalar virtualenv

-python3 -m pip install virtualenv

Creación del entorno virtual con el comando **virtualenv** seguidamente del nombre que se desea poner en este caso env

-virtualenv env

Activar el entorno virtual

-source env/bin/activate

Desactivar el entorno virtual

-deactivate

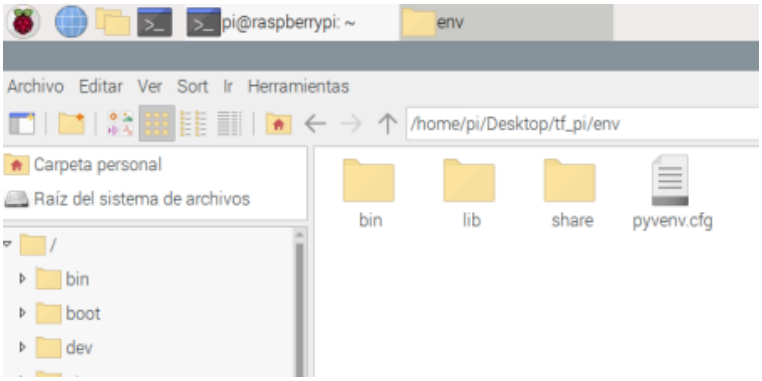


Figura 75.- Entorno Virtual Creado

Ejecutar los comandos para instalar las dependencias

-sudo apt-get install -y libhdf5-dev libc-ares-dev libeigen3-dev

-python3 -m pip install keras_applications==1.0.8 --no-deps

-python3 -m pip install keras_preprocessing==1.1.0 --no-deps

-python3 -m pip install h5py==2.9.0

-sudo apt-get install -y openmpi-bin libopenmpi-dev

-sudo apt-get install -y libatlas-base-dev

-python3 -m pip install -U six wheel mock

Instalar TensorFlow

-wget https://github.com/lhelontra/tensorflow

-python3 -m pip install tensorflow-2.0.0-cp37-none-linux_armv7l.whl

Reiniciar el terminal

-reboot

Activar nuevamente el entorno virtual

-cd Desktop

-cd tf_pi

-source env/bin/activate

Verificar que se haya instalado TensorFlow

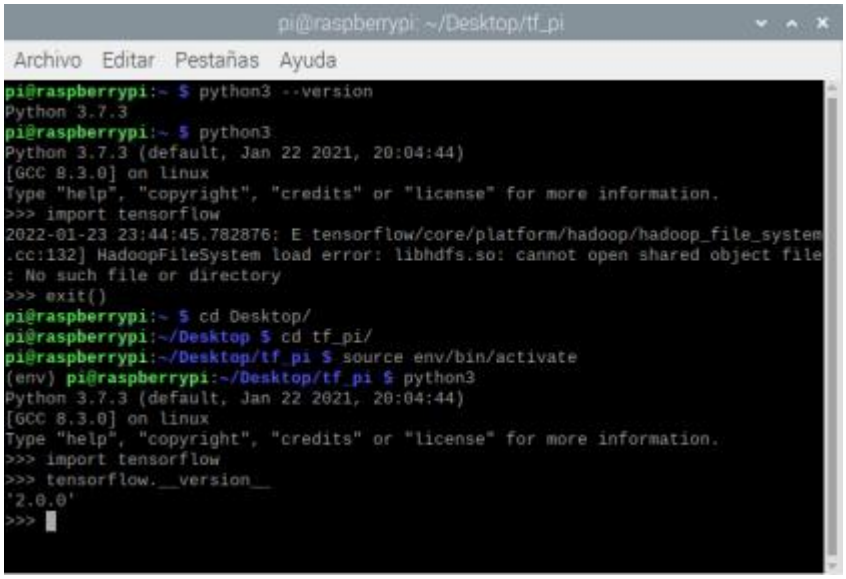


Figura 76.- Versiones utilizadas de Python y TensorFLow

Instalar librerías necesarias para la ejecución del proyecto

-pip3 install python-opencv-contrib

-pip3 install PyMLX90614

-pip3 install python-telegram-bot

-pip3 install imutils

Anexo F: Configuración de sensores, leds y buzzer

```
#----Inicialización del sensor de Temperatura
bus = SMBus(1)
sensor = MLX90614(bus, address=0x5a)
#------Configuración de Leds
greenLed = 27
redLed = 22
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(greenLed, GPIO.OUT, initial=GPIO.LOW)
GPIO.setup(redLed, GPIO.OUT, initial=GPIO.LOW)

#------------Configuración del buzzer
buzzer = Buzzer(17)
#-------------Configuración sensor de objeto infrarrojo
ir = 23
GPIO.setup(ir,GPIO.IN)
```

Anexo G: Configuración del sistema para que pueda o no ingresar el usuario

```
def applyLogic(label,chat_ID,bot):
  temp = getTempData()
  if temp >= 37.5:
    buzzer.on()
```

```
    GPIO.output(redLed, GPIO.HIGH)
    GPIO.output(greenLed, GPIO.LOW)
    sleep(4)
    buzzer.off()
    GPIO.output(redLed, GPIO.LOW)
    bot.send_message(chat_id = chat_ID, text='Persona Con Mascarilla---Temp = {} mayor a 37.5'.format(round(temp,1)))
    bot.send_message(chat_id = chat_ID, text='No puede pasar')
  elif (label=="Sin Mascarilla"):
    buzzer.on()
    GPIO.output(redLed, GPIO.HIGH)
    GPIO.output(greenLed, GPIO.LOW)
    sleep(4)
    buzzer.off()
    GPIO.output(redLed, GPIO.LOW)
#--Envía mensaje a la aplicación de mensajería de telegram
    bot.send_message(chat_id = chat_ID, text='Persona Sin Mascarilla---Temp = {}'.format(round(temp,1)))
    bot.send_message(chat_id = chat_ID, text='No puede pasar')
  elif (label == "Sin Mascarilla" and temp >= 37.5):
    buzzer.on()
    GPIO.output(redLed, GPIO.HIGH)
    GPIO.output(greenLed, GPIO.LOW)
    sleep(4)
    buzzer.off()
    GPIO.output(redLed, GPIO.LOW)
    bot.send_message(chat_id = chat_ID, text='Persona Sin Mascarilla---Temp = {} mayor a 37.5'.format(round(temp,1)))
    bot.send_message(chat_id = chat_ID, text='No puede pasar')
  else:
    buzzer.off()
```

```
    GPIO.output(greenLed, GPIO.HIGH)
    GPIO.output(redLed, GPIO.LOW)
    bot.send_message(chat_id = chat_ID, text='Persona Con Mascarilla---Temp = {}'.format(round(temp,1)))
    bot.send_message(chat_id = chat_ID, text='Puede Pasar')
    sleep(4)
    GPIO.output(greenLed, GPIO.HIGH)

#---Función para extraer temperatura de objeto para que lee la temperatura constantemente
def getTempData():
temp = sensor.get_object_1()
return temp

#--------Función para detección de objeto
def irData():
ir_value = GPIO.input(ir)
return ir_value

#-----------Aplicación de detección de mascarillas
def detect_mask(direcciones, prediccion, pantalla):
for (box, pred) in zip(direcciones, prediccion):
(startX, startY, endX, endY) = box
(mask, withoutMask) = pred

label = "Con Mascarilla" if mask > withoutMask else "Sin Mascarilla"
#Para tener datos gráficos en thingspeak
label_binary = 1 if mask > withoutMask else 0
#Código BGR
color = (0, 255, 0) if label == "Con Mascarilla" else (0, 0, 255)
print(label)
#Poner texto para la imagen
```

```
label_out = "{}: {:.2f}%".format(label, max(mask, withoutMask) * 100)
#Temperatura en función para que los datos se vayan actualizando
temp = getTempData()
#Texto en pantalla
tempe_usuario = "Temp: {:.1f}".format(temp)
#Se crea rectángulo alrededor del rostro e indica texto si tiene o no la mascarilla,
cv2.putText(pantalla, label_out, (startX, startY - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.65, color, 2)
cv2.putText(pantalla, tempe_usuario, (endX-10, endY),
cv2.FONT_HERSHEY_SIMPLEX, 0.65, (253, 216, 53), 2)
cv2.rectangle(pantalla, (startX, startY), (endX, endY), color, 2)
#Llamar a la función
dist = irData()
#si detecta un objeto toma el tiempo actual
if dist == 0:
act_date = datetime.datetime.now()
timeString = act_date.strftime("%d-%m-%Y %H:%M:%S")
#Guarda el frame del rostro
cv2.imwrite('/home/pi/Desktop/tf_pi/Proyecto_Tesis/fotos_resultado/img_{}.jpg'.format
(timeString),pantalla)
#Toma la imagen guardada y envía al chat ID para ser enviada a telegram
mask_temp_bot.send_photo(chat_id = chat_ID,
photo=open('/home/pi/Desktop/tf_pi/Proyecto_Tesis/fotos_resultado/img_{}.jpg'.format
(timeString),'rb'))
#Llama a función, y muestra los resultados
applyLogic(label, chat_ID, mask_temp_bot)
# Escribir al archivo cvs
with open("/home/pi/Desktop/tf_pi/Proyecto_Tesis/datos_mascarilla_temperatura.csv",
"a") as log:
log.write("{},{},{}\n".format(timeString, label, str(round(temp,1))))
log.close()
```

```
#Subir el dato a ThingSpeak
tPayload = "field1=" + str(round(temp,1))+ "&field2=" + str(label_binary)
#Publicar los datos en la página
try:
publish.single(topic, payload=tPayload, hostname=mqttHost, port=tPort, tls=tTLS, transport=tTransport)
#Guardar datos
save_log()
except (KeyboardInterrupt):
break
#Si no sube los datos espera un segundo e intenta subir nuevamente
except:
time.sleep(1)

#---------Aplicación de la detección de mascarillas en video a tiempo real
#Toma el video de la cámara
def run_video(detect_and_predict_mask):
while True:
pantalla = vs.read()
# Tamaño de la pantalla
pantalla = imutils.resize(pantalla, width=400)
#Valores de la detección de mascarillas y rostros
(direcciones, prediccion) = detect_and_predict_mask(pantalla, faceNet, maskNet)
detect_mask(direcciones, prediccion, pantalla)
#Muestra la imagen
cv2.imshow("pantalla", pantalla)
#Cerrar programa
key = cv2.waitKey(1) & 0xFF
if key == ord("q"):
break
```

```
cv2.destroyAllWindows()
GPIO.cleanup()
vs.stop()

#-----Función principal (Ejecuta todo)
if __name__=="__main__":
prototxtPath = "/home/pi/Desktop/tf_pi/Proyecto_Tesis/face_detector/deploy.prototxt"
weightsPath =
"/home/pi/Desktop/tf_pi/Proyecto_Tesis/face_detector/res10_300x300_ssd_iter_140000
.caffemodel"
faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)

maskNet = load_model("detector_mascarilla.model")

print("[INFO] starting video stream...")
vs = VideoStream(src=0, pantallarate=30).start()

run_video(detect_and_predict_mask)
```

Anexo H: Archivo generado en .csv

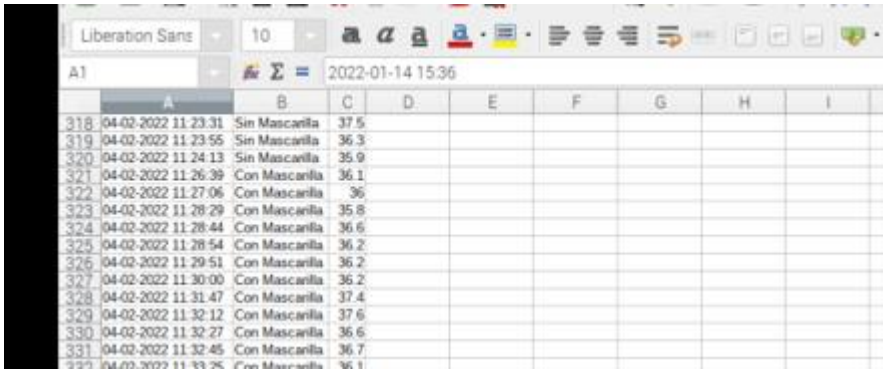


Figura 77.- Datos en .csv

Anexo I: Configuración de la Plataforma ThingSpeak

Creación de una cuenta en MathWords

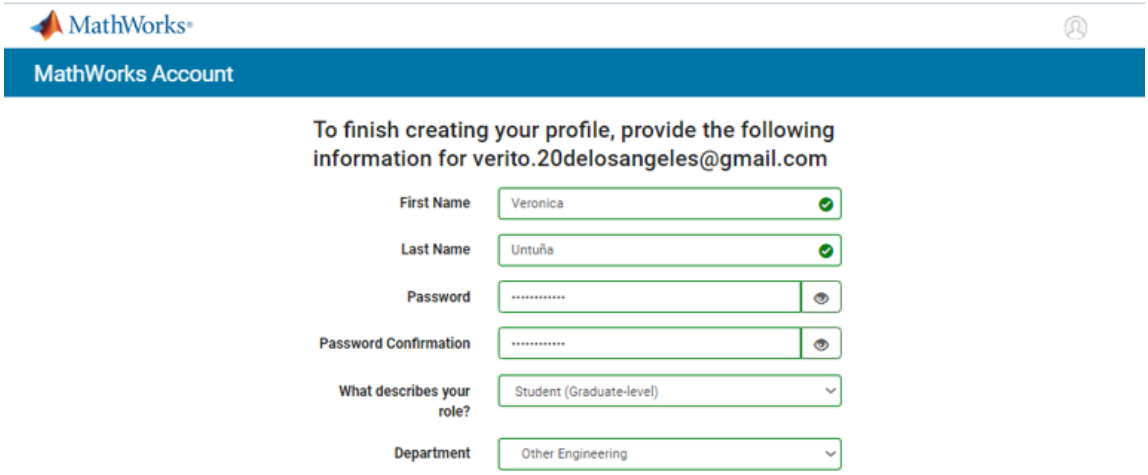


Figura 78.- Cuenta creada en ThingSpeak

Configurar los canales que se va a visualizar

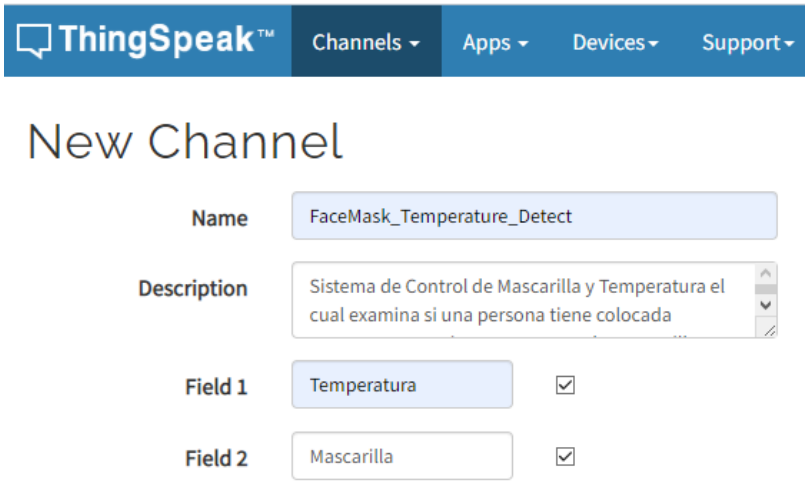


Figura 79.- Canales de ThingSpeak

ApiKeys para el envío de los datos con python

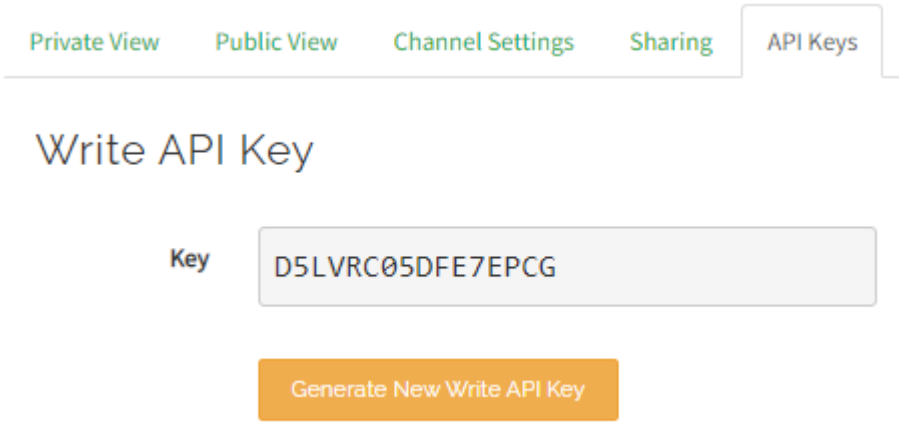


Figura 80.- Api Keys de ThingSpeak-Envío de datos

ApiKeys para la lectura de los datos con app inventor

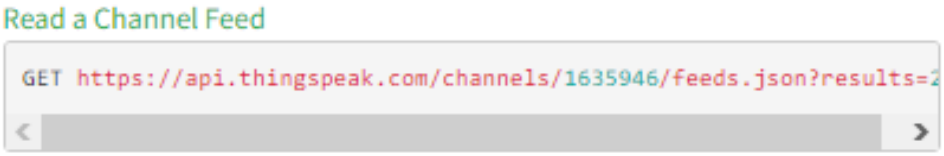


Figura 81.- Api Keys de ThingSpeak-Lectura de Datos

Anexo J: Configuración de la Aplicación Móvil en App Inventor

Creación de un proyecto en App inventor

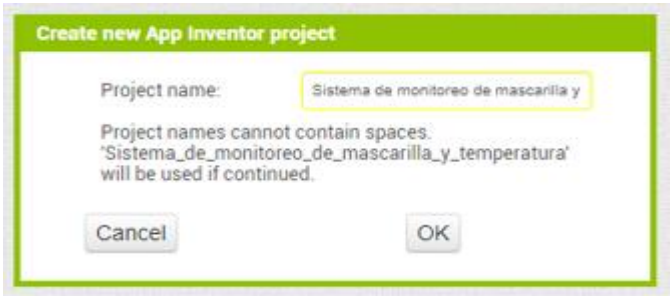


Figura 82.- Nuevo Proyecto App Inventor

Programación en Diagrama de Bloques

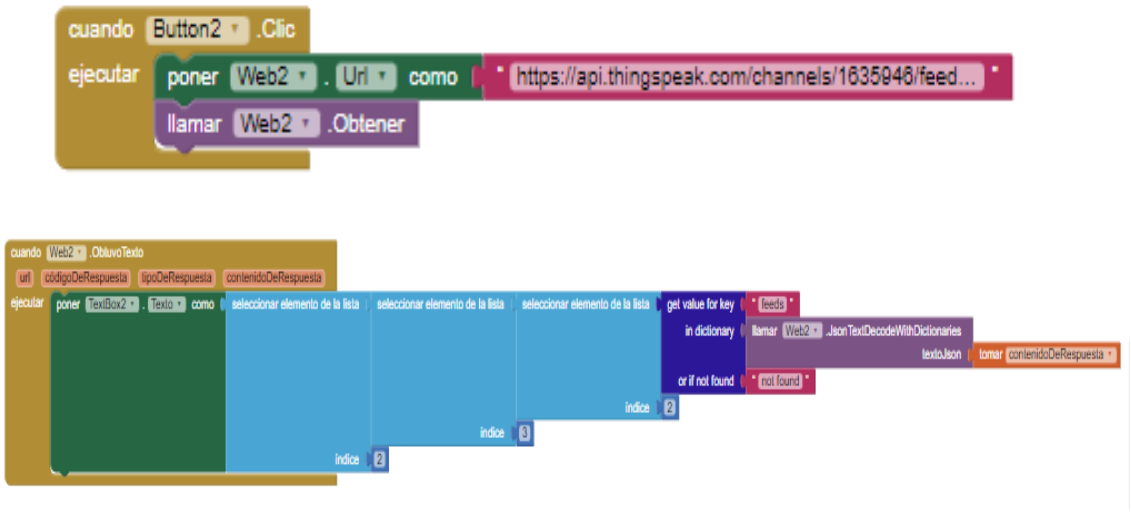


Figura 83.- Diagrama de Bloques en App Inventor

Anexo K: Calculo del valor de confiabilidad de temperatura

Intentos	Temperatura							
	Día 1		Día 2		Día 3			
	Termómetro Digital	Prototipo	Termómetro Digital	Prototipo	Termómetro Digital	Prototipo	Suma-Termómetro Digital	Suma-Prototipo
1	36,6	36,7	36,4	36,5	36,4	36,6	109,4	109,8
2	36,5	35,6	36,4	36,4	36,5	36,2	109,4	108,2
3	36,7	35,4	36,5	36,7	36,4	36,4	109,6	108,5
4	36,6	36,4	36,3	36,6	36,3	36,4	109,2	109,4
5	36,4	36,6	36,4	36,5	36,3	36,5	109,1	109,6
6	36,5	37,3	36,3	36,6	36,4	36,4	109,2	110,3
7	36,4	36,5	36,4	36,4	36,3	36,1	109,1	109
8	36,5	36,6	36,3	36,5	36,3	36,3	109,1	109,4
9	37,2	37,2	36,4	36,6	36,4	36,7	110	110,5
10	37,1	37,4	36,4	36,7	36,4	36,6	109,9	110,7
11	36,9	37,2	36,5	36,6	36,5	36,6	109,9	110,4
12	36,7	37,1	36,4	36,6	36,8	36,9	109,9	110,6
13	36,2	36,5	36,2	36,3	36,3	36,4	108,7	109,2
14	36,6	36,4	36,7	37,1	36,9	37,1	110,2	110,6
15	36,6	36,3	37,2	37,4	37,2	37,1	111	110,8
Varianza	0,06	0,32	0,05	0,07	0,07	0,08		
Promedio	36,63	36,61	36,45	36,63	36,49	36,55		

Coeficiente de Cronbach-Termómetro Digital	
$\propto (Alfa)$=	0,63
K(número de items)=	3,00
Vi(Varianza de cada item)=	0,18
Vt(Varianza total)=	0,31

$$\alpha = \frac{K}{K-1}\left[1 - \frac{\sum V_i}{V_t}\right]$$

Coeficiente de Cronbach-Prototipo	
$\propto (Alfa)$=	0,42
K(número de items)=	3,00
Vi(Varianza de cada item)=	0,47
Vt(Varianza total)=	0,65

Análisis de la confiabilidad de un instrumento	
0-0.2	Muy baja
0.2-0.4	Baja
0.4-0.6	Moderada
0.6-0.8	Buena
0.8-1.0	Alta

Figura 84.- Coeficiente de Cronbach