

**SEP**

**TECNM**

**INSTITUTO TECNOLÓGICO DE TIJUANA**  
**DIVISIÓN DE ESTUDIOS DE POSGRADOS E INVESTIGACIÓN**



**REDES NEURONALES CONVOLUCIONALES PARA LA**  
**CLASIFICACIÓN MULTICLASE DEL CUBREBOCAS**

TRABAJO DE TESIS PRESENTADO POR:

**ALEXIS CAMPOS LÓPEZ**

PARA OBTENER EL GRADO DE:

**MAESTRO EN CIENCIAS DE LA COMPUTACIÓN**

DIRECTOR DE TESIS:

**DRA. ELBA PATRICIA MELIN OLMEDA**

CO-DIRECTOR DE TESIS:

**DRA. DANIELA ADRIANA SÁNCHEZ VIZCARRA**

TIJUANA, BAJA CALIFORNIA, MÉXICO. ENERO DE 2023



Tijuana, Baja California, 08/febrero/2023

OFICIO No. 007/DEPI/2023

Asunto: Autorización de Impresión de Tesis

**MARÍA MAGDALENA SERRANO ORTEGA**  
**JEFA DEL DEPARTAMENTO DE SERVICIOS ESCOLARES**  
**PRESENTE**

En lo referente al trabajo de tesis, "Redes neuronales convolucionales para la clasificación multiclase del cubrebocas" Presentado por C. **Alexis Campos López**, alumno de la Maestría en Ciencias de la Computación con numero de control **M21210006**; informo a usted que a solicitud del comité de tutorial, tengo a bien **Autorizar la impresión de Tesis**, atendiendo las disposiciones de los Lineamientos para la Operación de Estudios de Posgrado del Tecnológico Nacional de México.

Sin más por el momento le envió un cordial saludo.

**A T E N T A M E N T E**

*Excelencia en Educación Tecnológica.*  
*Por una Juventud Integrada al Desarrollo de México.*



**GUADALUPE HERNÁNDEZ ESCOBEDO**  
**JEFE DE DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN**

ccp. Archivo  
GHE/lap



Calzada del Tecnológico S/N esquina Castillo de Chapultepec y calle Cuauhtemotzin,  
Fracc. Tomás Aquino C.P.22414 Tijuana, Baja California. Tel. 01 (664) 6078400  
dir\_tijuana@tecnm.mx | tecnm.mx | tijuana.tecnm.mx



**2023**  
AÑO DE  
**Francisco**  
**VILLA**  
EL REVOLUCIONARIO DEL PUEBLO

Tijuana, B.C., 07 de Febrero de 2023.  
Asunto: Se autoriza impresión de Trabajo de Tesis

**C. Dr. Guadalupe Hernández Escobedo**  
Jefe de División de Estudios de Posgrado e Inv.  
Presente.

En lo referente al trabajo de tesis escrito, con título "**REDES NEURONALES CONVOLUCIONALES PARA LA CLASIFICACIÓN MULTICLASE DEL CUBREBOCAS**", presentado por el **C. ALEXIS CAMPOS LÓPEZ** alumno de la Maestría en Ciencias de la Computación con número de control **M21210006**, informamos a usted que se autoriza el escrito de tesis y se aprueba en todas sus partes, en virtud de reunir los requisitos de un trabajo de grado de maestría y a la vez se autoriza al interesado para que proceda de inmediato a la impresión del mismo y a presentar su examen de grado, ya que cumple con todos los requisitos.

ATENTAMENTE



DRA. ELBA PATRICIA MELIN OLMEDA  
PRESIDENTE



DRA. DANIELA ADRIANA SANCHEZ VIZCARRA  
SECRETARIO



DRA. MARTHA ELENA PULIDO  
VOCAL

c.c.p. Oficina de Titulación  
c.c.p. División de Estudios de Posgrado e Investigación  
c.c.p. Expediente  
c.c.p. Interesado

EPMO/\*inf


## DECLARACIÓN DE ORIGINALIDAD

Tijuana, BC., 2 de febrero de 2023,

Yo, **ALEXIS CAMPOS LOPEZ** estudiante de la Maestría en Ciencias de la Computación, en mi calidad de autor manifiesto que este documento de tesis es producto de mi trabajo original y que no infringe los derechos de terceros, tales como derechos de publicación, derechos de autor, patente y similitud. Por lo tanto, la obra realizada es de mi exclusiva autoría y no infringí en copiar el texto o imágenes, de fuentes de información por lo cual soy responsable del escrito que aquí se presenta.

Así mismo, declaro que en las citas textuales que he incluido (las cuales aparecen entre comillas) y en los resúmenes que he realizado de publicaciones ajenas, indico explícitamente los datos de los autores y las publicaciones.

En caso de presentarse cualquier reclamación o acción por parte de terceros en cuanto a los derechos de autor sobre la obra en cuestión, acepto toda la responsabilidad de tal infracción y relevo de esta a mi director de tesis, así como al Tecnológico Nacional de México, al Instituto Tecnológico de Tijuana y a sus respectivas autoridades.



Alexis Campos López

Nombre completo del estudiante y firma autógrafa  
Estudiante de la Maestría en Ciencias de la Computación




## CARTA DECLARACIÓN DE PROPIEDAD INTELECTUAL

Tijuana, BC a 1 de febrero del 2023

Yo CAMPOS LOPEZ ALEXIS reconozco que el Trabajo de Tesis de Maestría que realice durante mis estudios en la Maestría en Ciencias de la Computación del Instituto Tecnológico de Tijuana fue parte del Proyecto de Investigación Titulado: Nuevos modelos híbridos inteligentes para análisis, agrupación y clasificación utilizando sistemas difusos y redes neuronales No.13758.22-P que desarrolla mi directora de tesis el Dra. Elba Patricia Melin Olmeda-y del cual es responsable del proyecto de investigación. Por esta razón, los métodos, modelos, algoritmos, y software realizados, así como datos y resultados obtenidos durante el desarrollo de mi tesis de maestría son propiedad intelectual de mi Director de Tesis, del Tecnológico Nacional de México, Instituto Tecnológico de Tijuana y del Conacyt, y No podré utilizarlos por mi cuenta durante, Ni después de terminar mi beca o estudios, excepto a solicitud escrita para poder utilizarlos bajo una colaboración directa con mi directora de tesis la cual es responsable del proyecto de investigación. Por tanto, estoy de acuerdo en que No podre utilizar ni tomar modelos, ni datos utilizados en este proyecto de investigación y en el desarrollo de tesis para: presentaciones, publicaciones ni desarrollo de mi propia investigación que pudiera desarrollar una vez concluidos mis estudios.

Atentamente

Alexis Campos López   
NOMBRE COMPLETO Y FIRMA AUTOGRAFA DEL ESTUDIANTE

Estudiante de la Maestría en Ciencias de la Computación

## RESUMEN

En los últimos años, debido a la pandemia del COVID-19 se han producido gran cantidad de contagios entre los humanos, provocando que el virus se expanda alrededor del mundo. El uso de cubrebocas de acuerdo con estudios ha ayudado a evitar en gran parte su propagación, por eso es muy importante el uso correcto del mismo. Usar el cubrebocas en lugares públicos se ha vuelto una práctica común en estos días y si no es usado de manera correcta el virus se seguirá transmitiendo. Se propone el uso de redes neuronales convolucionales para detectar y clasificar el uso correcto de cubrebocas en combinación con un sistema tiempo real para el monitoreo del mismo. Los métodos de aprendizaje profundo son los más eficaces para detectar si una persona está utilizando correctamente el cubrebocas. El modelo fue entrenado utilizando el conjunto de datos de MaskedFace-Net y evaluado con diferentes imágenes del mismo. Se utiliza el modelo Caffe para la detección de rostro, posteriormente se hace un preprocesamiento de la imagen para extraer características. Estas imágenes son la entrada de la red neuronal convolucional, donde se clasifica entre mask, no mask e incorrect mask. De igual manera se propone un sistema tiempo real para el monitoreo e identificación de las personas que no utilicen el cubrebocas, implementando el uso de sistemas programables como la raspberry pi 4, clasificando según la clase en color verde, amarillo o rojo para encender un LED según sea el caso. El modelo propuesto alcanza un porcentaje de precisión de 99.69% en el porcentaje de prueba, el cual es mayor comparado con otros autores.

## **ABSTRACT**

In recent years, due to the COVID-19 pandemic, there have been a large number of infections among humans, causing the virus to spread around the world. According to studies, the use of face masks has significantly reduced its spread, which is why its proper application is critical. Using the mask in public places has become a common practice these days, and if it is not used correctly, the virus will continue to be transmitted. This research proposes the use of convolutional neural networks to detect and classify the correct use of face masks and a real-time monitoring system to monitor it. Deep learning methods are the most effective in detecting whether a person is wearing a mask correctly. The model was trained using the MaskedFaceNet dataset and evaluated with different images of it. The Caffe model is used for face detection, and then a pre-processing of the image is done to extract features. These images are the input of the convolutional neural network, where they are classified as mask, no mask, and incorrect mask. Additionally, a real-time system is proposed for monitoring and identifying people who do not wear a face mask, utilizing programmable systems such as the Raspberry Pi 4, classifying according to the class in green, yellow, or red to turn on an LED, depending on the case. The proposed model reaches an accuracy percentage of 99.69% in the test, which is higher compared to other authors.

## DEDICATORIA

Dedico esta tesis a Dios que es el principal motor para seguir adelante día tras día, motivándome a superar los obstáculos de la vida.

A mi madre Consuelo que me ha dado su apoyo incondicional durante todo mi proceso educativo, ayudándome a lograr mis objetivos académicos.

A mi esposa Paola que siempre estuvo a mi lado proporcionando aliento y motivación para que se pudieran alcanzar las metas.

A mis directoras, Dra. Melin y Dra. Daniela quienes me asesoraron y suministraron consejos para realizar el trabajo.

A los sinodales quienes evaluaron mi tesis y la aprobaron.

A todas las personas que me apoyaron en la redacción y realización de esta tesis.

A CONACYT por el apoyo otorgado a través de la beca con CVU 1106406 durante el periodo de Febrero 2021 a Enero 2023 con el cual pude concluir los estudios de Maestría en Ciencia de la Computación.

Esta tesis es para ellos porque se la debo a su apoyo incondicional.



## AGRADECIMIENTOS

Al término de esta etapa de mi vida, quiero expresar mi más sincero agradecimiento a aquellos que me brindaron su apoyo incondicional durante mi carrera académica. Sin su ayuda, no habría podido alcanzar este logro.

Dios, tu amor y tu bondad son eternos, me permites celebrar mis logros que son producto de tu ayuda, me has guiado en este camino y estarás siempre presente en futuros desafíos. Gracias por estar no solo en esta importante etapa de mi vida, sino siempre brindando lo mejor y buscando lo mejor para mí.

Gracias a mi esposa por su amor incondicional y por entenderme en todo momento. Su apoyo ha sido fundamental en esta etapa de mi vida, y su motivación me ha impulsado a seguir trabajando duro para alcanzar mis objetivos. Juntos, hemos superado obstáculos y desarrollado habilidades para alcanzar nuestras metas.

Expreso mi gratitud profunda por todo lo que has hecho por mí, madre. Tu dedicación y esfuerzos son impresionantes, y tu amor es invaluable para mí. Me has proporcionado todo lo que necesitaba para crecer y desarrollarme en la persona que soy hoy en día. Tus enseñanzas han sido fundamentales en mi formación y crecimiento personal. Te doy las gracias por todo lo que has hecho por mí. Eres mi roca y mi apoyo constante.

A pesar de que es imposible expresar mi gratitud completamente por todo el esfuerzo y sacrificio que hicieron, me gustaría hacerles saber que el logro alcanzado también es suyo y que su apoyo fue lo que me permitió lograrlo.

# ÍNDICE GENERAL

<b>RESUMEN .....</b>	<b>I</b>
<b>ABSTRACT.....</b>	<b>II</b>
<b>DEDICATORIA.....</b>	<b>III</b>
<b>AGRADECIMIENTOS.....</b>	<b>IV</b>
<b>CAPÍTULO I INTRODUCCIÓN.....</b>	<b>1</b>
<b>CAPÍTULO II ANTECEDENTES.....</b>	<b>4</b>
2.1.    Visión Artificial.....	4
2.1.1.    Detección facial con Visión Artificial.....	6
2.2.    Inteligencia Artificial.....	6
2.3.    Aprendizaje automático.....	7
2.3.1.    Aprendizaje supervisado.....	7
2.3.2.    Aprendizaje no supervisado.....	8
2.4.    Aprendizaje profundo.....	9
2.4.1.    Neurona.....	10
2.5.    Redes Neuronales.....	11
2.5.1.    Redes Neuronales Artificiales.....	12
2.5.2.    Redes Neuronales Convoluciones.....	12
2.5.3.    Modelos de Redes Neuronales convolucionales existentes.....	19
2.6.    Estado del Arte.....	24
<b>CAPÍTULO III METODOLOGÍA.....</b>	<b>29</b>
3.1.    Bases de Datos.....	29
3.1.1.    Base de datos MaskedFace-NeT.....	29
3.1.2.    Base de datos Face Mask Detection.....	31
3.1.3.    Base de datos MAFA.....	31
3.1.4.    Base de datos WIDER FACE.....	32

3.2.	Conceptos del preprocesamiento .....	33
3.2.1.	Caffe model .....	34
3.2.2.	Sustracción de RGB.....	36
3.3.	Especificaciones de Software .....	37
3.3.1.	Python.....	38
3.3.2.	Tensorflow .....	38
3.3.3.	Keras.....	39
3.4.	Raspberry pi.....	40
<b>CAPÍTULO IV DESARROLLO.....</b>		<b>41</b>
4.1.	Método General Propuesto .....	41
4.2.	Preprocesamiento.....	41
4.3.	Modelo General Propuesto .....	43
4.4.	Modelo del sistema tiempo real.....	44
<b>CAPÍTULO V Resultados.....</b>		<b>46</b>
5.1.	Experimentos realizados.....	46
5.1.1.	Caso de estudio 1 (4 clases).....	46
5.1.2.	Caso de estudio 2 (3 clases).....	53
5.2.	Sistema tiempo real .....	56
<b>CAPÍTULO VI Conclusiones .....</b>		<b>62</b>
<b>REFERENCIAS.....</b>		<b>64</b>

**ÍNDICE DE TABLAS**

<b>Tabla 2.1. Diferentes tipos de funciones de pérdida que se pueden utilizar y su respectiva ecuación. ....</b>	<b>19</b>
<b>Tabla 2.2. Artículos científicos relacionados Parte 1.....</b>	<b>25</b>
<b>Tabla 2.3. Artículos científicos relacionados Parte 2.....</b>	<b>26</b>
<b>Tabla 2.4. Artículos científicos relacionados Parte 3.....</b>	<b>27</b>
<b>Tabla 5.1. Experimentos para el caso de estudio 1. ....</b>	<b>49</b>
<b>Tabla 5.2. Resumen de comparación de modelos. ....</b>	<b>49</b>
<b>Tabla 5.3. Promedios de experimentos del caso de estudio 2.....</b>	<b>54</b>
<b>Tabla 5.4. Matriz de confusión del caso de estudio 2.....</b>	<b>55</b>

## ÍNDICE DE FIGURAS

<b>Figura 2.1.</b>	<b>Diagrama de bloques para el sistema de visión artificial .....</b>	<b>5</b>
<b>Figura 2.2.</b>	<b>Flujo de trabajo del aprendizaje supervisado .....</b>	<b>8</b>
<b>Figura 2.3.</b>	<b>Flujo de trabajo del aprendizaje no supervisado .....</b>	<b>9</b>
<b>Figura 2.4.</b>	<b>Función de propagación de una neurona.....</b>	<b>11</b>
<b>Figura 2.5.</b>	<b>Ejemplo de red neuronal convolucional.....</b>	<b>13</b>
<b>Figura 2.6.</b>	<b>Convolución de una imagen .....</b>	<b>14</b>
<b>Figura 2.7.</b>	<b>Función de activación ReLu .....</b>	<b>15</b>
<b>Figura 2.8.</b>	<b>Aplicación de Max-pooling.....</b>	<b>17</b>
<b>Figura 2.9.</b>	<b>Ejemplo flatten a una matriz.....</b>	<b>18</b>
<b>Figura 2.10.</b>	<b>Muestra la arquitectura de ResNet.....</b>	<b>22</b>
<b>Figura 3.1.</b>	<b>MaskedFace-Net base de datos ejemplo.....</b>	<b>29</b>
<b>Figura 3.2.</b>	<b>Clasificación de MaskedFace-Net .....</b>	<b>30</b>
<b>Figura 3.3.</b>	<b>Ejemplo de imágenes del conjunto de datos WIDERFACE .....</b>	<b>33</b>
<b>Figura 3.4.</b>	<b>Clasificación de MaskedFace-Net .....</b>	<b>34</b>
<b>Figura 3.5.</b>	<b>Resta de la media de una imagen.....</b>	<b>36</b>
<b>Figura 3.6.</b>	<b>Resta de la media de una imagen.....</b>	<b>40</b>
<b>Figura 4.1.</b>	<b>Flujo de trabajo básico .....</b>	<b>41</b>
<b>Figura 4.2.</b>	<b>Detección de la región de la cara.....</b>	<b>42</b>
<b>Figura 4.3.</b>	<b>Aplicando sustracción de RGB .....</b>	<b>42</b>
<b>Figura 4.4.</b>	<b>Modelo general de RNC para clasificación de uso de cubrebocas.....</b>	<b>43</b>
<b>Figura 4.5.</b>	<b>Método general propuesto para el sistema tiempo real.....</b>	<b>45</b>
<b>Figura 5.1.</b>	<b>Ejemplo de imágenes de MaskedFace-Net.....</b>	<b>47</b>
<b>Figura 5.2.</b>	<b>Estructura RNC del modelo A .....</b>	<b>47</b>
<b>Figura 5.3.</b>	<b>Estructura RNC del modelo B .....</b>	<b>48</b>
<b>Figura 5.4.</b>	<b>Estructura RNC del modelo C .....</b>	<b>48</b>
<b>Figura 5.5.</b>	<b>Gráficas de número de épocas contra la precisión.....</b>	<b>50</b>
<b>Figura 5.6.</b>	<b>Gráficas de número de épocas contra la pérdida .....</b>	<b>50</b>
<b>Figura 5.7.</b>	<b>Prueba de hipótesis 1 para el caso de estudio 1 .....</b>	<b>52</b>
<b>Figura 5.8.</b>	<b>Prueba de hipótesis 2 para el caso de estudio 1 .....</b>	<b>53</b>
<b>Figura 5.9.</b>	<b>Ejemplo de imágenes para el caso de estudio 2 .....</b>	<b>53</b>
<b>Figura 5.10.</b>	<b>Evaluando imagen real con cubrebocas.....</b>	<b>55</b>

## INDICE DE FIGURAS

<b>Figura 5.11.</b>	<b>Evaluando imagen real sin cubrebocas. ....</b>	<b>56</b>
<b>Figura 5.12.</b>	<b>Evaluando imagen real con cubrebocas colocado incorrectamente. ....</b>	<b>56</b>
<b>Figura 5.13.</b>	<b>Logitech C922 .....</b>	<b>57</b>
<b>Figura 5.14.</b>	<b>Circuito de la Raspberry pi4 .....</b>	<b>58</b>
<b>Figura 5.15.</b>	<b>Prueba de resultados para luz LED .....</b>	<b>59</b>
<b>Figura 5.16.</b>	<b>Ejemplo del sistema tiempo real con la clase Mask .....</b>	<b>59</b>
<b>Figura 5.17.</b>	<b>Ejemplo del sistema tiempo real con IncorrectMask y NoMask .....</b>	<b>60</b>



# CAPÍTULO I

## INTRODUCCIÓN

Las redes neuronales convolucionales han ido evolucionando a lo largo del tiempo esto debido a la gran cantidad de personas que se han dedicado a trabajar entorno a ellas, mejorando sus resultados o proponiendo nuevas ideas.

El uso de cubrebocas ha funcionado como una estrategia para disminuir la propagación del virus COVID-19, la cual ha infectado a más de 430,000,000 de personas alrededor del mundo según la Organización Mundial de la Salud (hasta febrero 2022) [1]. Una de las indicaciones básicas sobre la colocación correcta del cubrebocas es que debe colocarse cubriendo la nariz, boca y mentón [2]. El no acatar estas indicaciones podría ocasionar el esparcimiento del virus con las personas alrededor.

Como justificación se tiene que durante la pandemia del COVID-19 en la mayoría de los países se volvió una obligación el uso de cubrebocas, a la fecha en el mes de febrero del 2022 se estimaron 50, 595,554 de casos nuevos confirmados en el mundo [1], por lo cual es necesario identificar a las personas que utilizan correctamente el cubrebocas.

Estas cifras siguen en aumento puesto que en muchos países no se ha controlado el esparcimiento del virus, las medidas contra la propagación del virus COVID-19 en ocasiones no se realizan de manera correcta.

Algunas limitaciones que se encontraron en esta investigación son determinadas por la base de datos conocida como MaskedFace-Net [3], esta cuenta con imágenes de cubrebocas simulada por lo que podría limitar los resultados.

El objetivo principal del proyecto es diseñar un nuevo modelo de red neuronal convolucional para la detección acerca del uso de cubrebocas, específicamente nos enfocamos en:

- Desarrollar diferentes modelos de redes neuronales convolucionales para la clasificación sobre el uso correcto del cubrebocas.

- Realizar pruebas estadísticas para comparar los resultados con otros autores que utilicen las mismas clases.
- Desarrollar experimentos con el modelo propuesto.

De acuerdo a la Organización Mundial de la Salud (OMS), utilizar cubrebocas disminuye la propagación de cualquier síndrome gripal e infecciones por coronavirus humanos, es por ello que la OMS aconseja que se fomente el uso de cubrebocas al público general, con el objetivo de evitar transmitir el virus, exponerse a propagarlo o infectar el entorno que habitué [16].

El sistema propuesto utiliza técnicas de visión por computadora para detectar la región de la cara, así como técnicas de aprendizaje profundo. Se indicará automáticamente usando una raspberry pi4 y una cámara a las personas que usan cubrebocas, que no usan cubrebocas, o las usan incorrectamente.

Las aportaciones generadas al realizar este presente trabajo tesis fueron:

1.- A. Campos, P. Melin, D. Sánchez., Artículo. 22th International Conference on Hybrid Intelligent Systems, 2022. (Aceptado).

2.- A. Campos, P. Melin, D. Sánchez., Capítulo de libro, Hybrid Intelligent Models: Theory and Applications, 2022. (Aceptado)

3. - A. Campos, P. Melin y D. Sánchez , «Multiclass Mask Classification with a New Convolutional Neural Model and Its Real-Time Implementation,» *Life*, vol. 13, nº 2, pp. 1-15, 2023 (Publicado) [4].

El trabajo de tesis está organizado en capítulos que se describen brevemente a continuación:

En el capítulo 2 se definen los conceptos básicos relacionados al tema de investigación. Los temas que se profundizan son: redes neuronales, cubrebocas, aprendizaje profundo, entre otros conceptos básicos.

La metodología utilizada para realizar el modelo es presentada el capítulo 3, donde se abordan temas como las bases de datos utilizadas, tipo de preprocesamiento utilizado y tecnologías usadas.

En el capítulo 4 se explica el desarrollo del método general propuesto, así como el modelo de red neuronal convolucional y la presentación del sistema tiempo real.

Los resultados son presentados en el capítulo 5, se muestran los experimentos realizados y los casos de estudios hechos, así como sus respectivas pruebas estadísticas. Finalizando el capítulo 6 donde se plasman las conclusiones finales y el posible trabajo futuro.

## **CAPÍTULO II**

### **ANTECEDENTES**

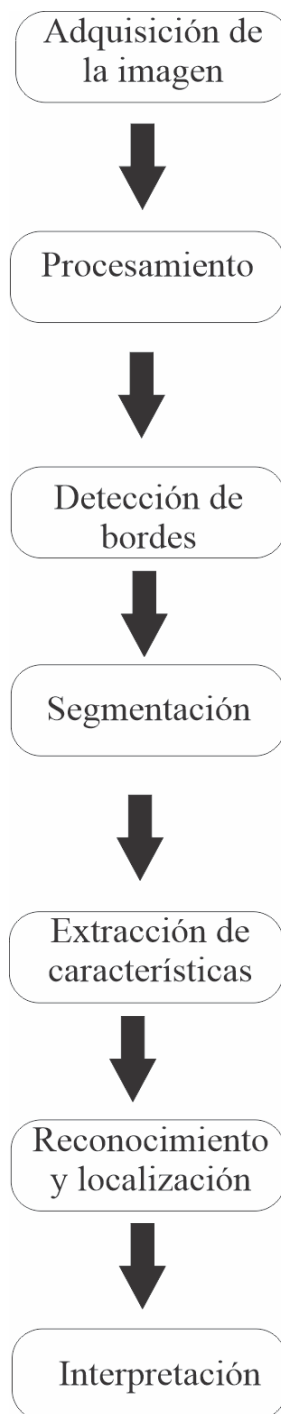
En este capítulo se explica en qué consisten las redes neuronales convolucionales explicando un poco acerca de su funcionamiento, así como los elementos que las componen, además de describir conceptos básicos de aprendizaje automático, aprendizaje profundo y visión por computadora, se explicarán las diferentes arquitecturas de red neuronal convolucional ya establecidas y utilizadas por otros autores.

#### **2.1. VISIÓN ARTIFICIAL**

La visión artificial, o lo que otros autores denominan visión por computadora, se define como el estudio de la programación informática para procesar imágenes o películas, o incluso para comprenderlas. Se explica que es el proceso de transformar datos de un encuadre o cámara en una nueva representación o decisión. La Figura 2.1 muestra el proceso paso a paso realizado en visión artificial, mostrando la secuencia de cada paso desde la adquisición y procesamiento de imágenes hasta las interpretaciones específicas utilizando los datos adquiridos [5].

Cada etapa es independiente entre sí ya que no todas las etapas son obligatorias. La primera fase de adquisición de imágenes se basa en la digitalización y digitalización física, seguida de un procesamiento adecuado, donde la imagen estará en el área de interés, eliminando partes de la imagen que no se tienen en cuenta. [5]. Un paso relevante en este proceso es detectar bordes que no sean de fondo que separen objetos de interés. Esto reduce los datos mostrados. Luego se realiza la segmentación de la imagen para mostrar que los píxeles se pueden seleccionar en función de su valor de color para poder resaltar objetos a través de la información de color. Finalmente, se muestra la detección y la localización se clasifica en objetos con propiedades 3D comunes y limitados espacialmente por triangulación para que se pueda seleccionar e interpretar una sola imagen. [5]

## ANTECEDENTES



*Figura 2.1. Diagrama de bloques para el sistema de visión artificial*

## ANTECEDENTES

### **2.1.1. DETECCIÓN FACIAL CON VISIÓN ARTIFICIAL**

Los sistemas de reconocimiento facial han cobrado impulso en los últimos años debido a su uso generalizado en sistemas biométricos que tienen como objetivo identificar automáticamente a las personas en las fotos de las cámaras digitales. El reconocimiento facial automático, que debutó en la década de 1960, fue el primer sistema semiautomático que requería que un administrador encontrara características como ojos, cejas, narices y bocas en una imagen para realizar cálculos. A continuación, se analiza el sistema de cálculo de distancia para varias paradas utilizando los datos almacenados en la base de datos. Los siguientes sistemas de reconocimiento utilizan diferentes marcadores y técnicas para mejorar el proceso de reconocimiento. Actualmente, hay diversas aplicaciones de tecnología madura en diferentes ámbitos de la sociedad, como el entretenimiento, los documentos de identificación como pasaportes y licencias de conducir, la seguridad de la información, la aplicación de la ley y la vigilancia.

La implementación de tecnología de reconocimiento facial fue un proceso desafiante que requirió una investigación exhaustiva para optimizar el sistema. El desarrollo se dividió en varias áreas, incluyendo la identificación facial, seguimiento, coincidencia, extracción de características, entrenamiento, clasificación, análisis de expresiones faciales y análisis en 2D y 3D [5].

### **2.2. INTELIGENCIA ARTIFICIAL**

La Inteligencia Artificial (IA) es un campo de estudio que busca desarrollar algoritmos, sistemas y técnicas que permitan a las computadoras realizar tareas que requieren inteligencia humana. Según [6], la IA es la capacidad de las máquinas de aprender de los datos, tomar decisiones y realizar tareas de manera similar a como lo hacen los seres humanos. Con la IA, se busca crear sistemas que puedan procesar grandes cantidades de información, cometer menos errores y no requerir descanso.



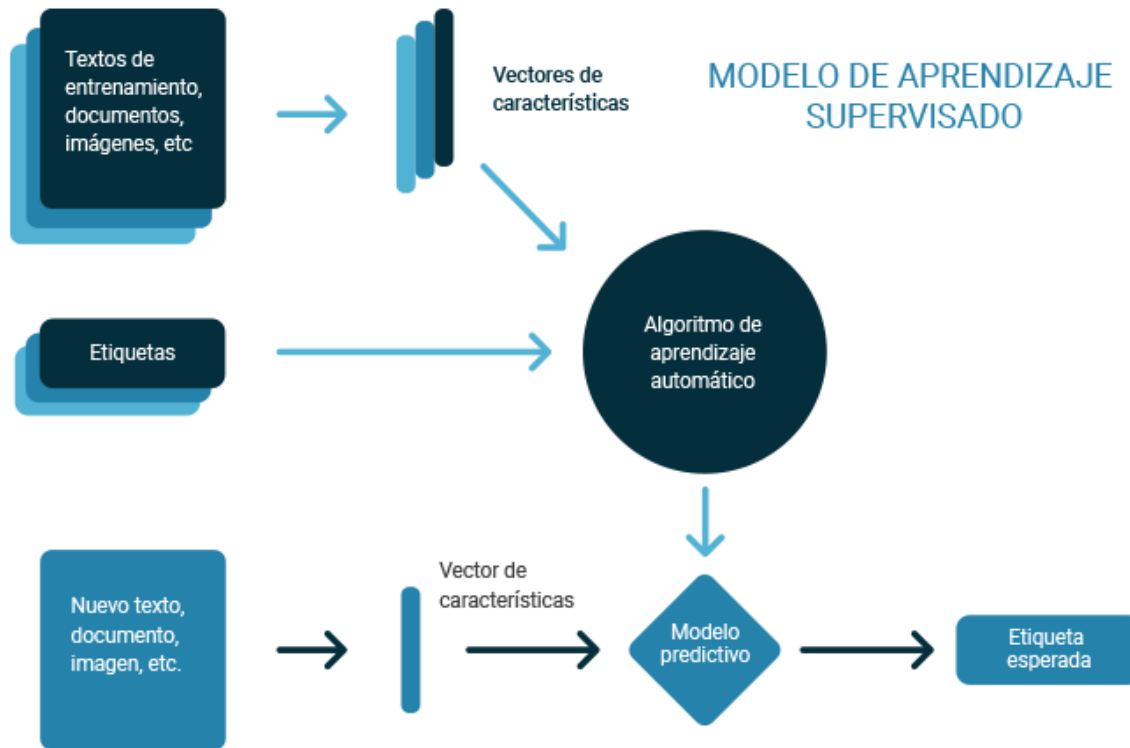
## **2.3. APRENDIZAJE AUTOMÁTICO**

El aprendizaje automático, también conocido como machine learning (ML), es una disciplina que se dedica a crear algoritmos y modelos estadísticos que permiten a las computadoras realizar tareas sin la necesidad de programación específica. Estos algoritmos se utilizan en varios campos, tales como la minería de datos, el procesamiento de imágenes y el análisis predictivo. La principal ventaja de utilizar el aprendizaje automático es que, una vez entrenado el algoritmo con los datos, puede llevar a cabo la tarea de manera autónoma [7].

### **2.3.1. APRENDIZAJE SUPERVISADO**

El aprendizaje supervisado es un tipo de aprendizaje automático en el que un algoritmo es entrenado para aprender una función que asigna entradas a salidas basadas en ejemplos de entradas y salidas previamente etiquetadas. La función se construye a partir de un conjunto de datos de entrenamiento etiquetado. Los algoritmos de aprendizaje automático supervisado requieren una fuente externa de datos etiquetados para poder funcionar. El conjunto de datos de entrada suele dividirse en un conjunto de datos de entrenamiento y un conjunto de datos de prueba para evaluar el rendimiento del algoritmo [7]. El conjunto de datos de entrenamiento contiene las variables de salida que se van a predecir o clasificar. Todos los algoritmos derivan algún tipo de modelo del conjunto de datos de entrenamiento y lo aplican al conjunto de datos de prueba para predicción o clasificación. El flujo de trabajo del algoritmo de aprendizaje automático supervisado se muestra en la Figura 2.2.

## ANTECEDENTES

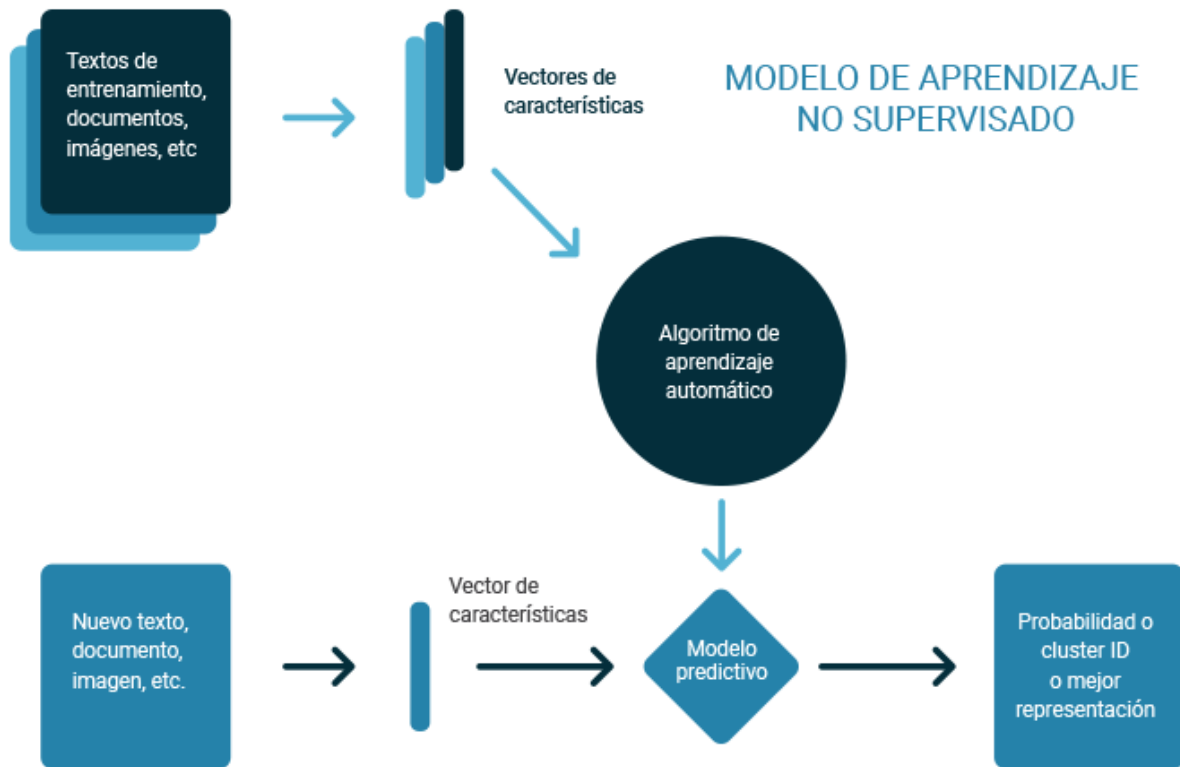


*Figura 2.2. Flujo de trabajo del aprendizaje supervisado*

### 2.3.2. APRENDIZAJE NO SUPERVISADO

Esto se llama aprendizaje no supervisado porque, en comparación con aprendizaje supervisado anterior, no hay una respuesta correcta ni un maestro. Los algoritmos están reservados para descubrir y presentar estructuras interesantes en los datos. Los algoritmos de aprendizaje no supervisados aprenden ciertas características de los datos [7]. Cuando se ingresan nuevos datos, utiliza una función previamente aprendida para identificar la categoría de los datos. Se utiliza principalmente para el agrupamiento y la reducción de funciones. De igual manera la Figura 2.3 muestra el modelo del aprendizaje no supervisado.

## ANTECEDENTES



*Figura 2.3. Flujo de trabajo del aprendizaje no supervisado*

## 2.4. APRENDIZAJE PROFUNDO

El aprendizaje profundo o deep learning (DL) es un subcampo del aprendizaje automático. Lo "profundo" en aprendizaje profundo es tener clases representativas consecutivas; por lo tanto, la profundidad del modelo se refiere a la cantidad de capas en un modelo de red neuronal artificial (RNA) y se denomina básicamente aprendizaje profundo [8].

El aprendizaje profundo es excelente cuando se trata de tareas complejas, al usar capas de acumulación apiladas, podemos formar redes profundas [8]. Pero esto viene con algunos desafíos:

- Al entrenar la red se observan problemas de fugas y gradientes explosivos.

## ANTECEDENTES

- Muchas veces, al considerar la precisión del entrenamiento y las pruebas, las redes se superponen y, por lo tanto, no son útiles para conjuntos de datos ocultos.
- Además, elegir el mejor tamaño de núcleo es una decisión difícil.

Para resolver estos desafíos, los investigadores pensaron, en porque no se puede ir más allá en lugar de solo profundizar, es por ello que pensaron diferentes modelos para resolver estos problemas.

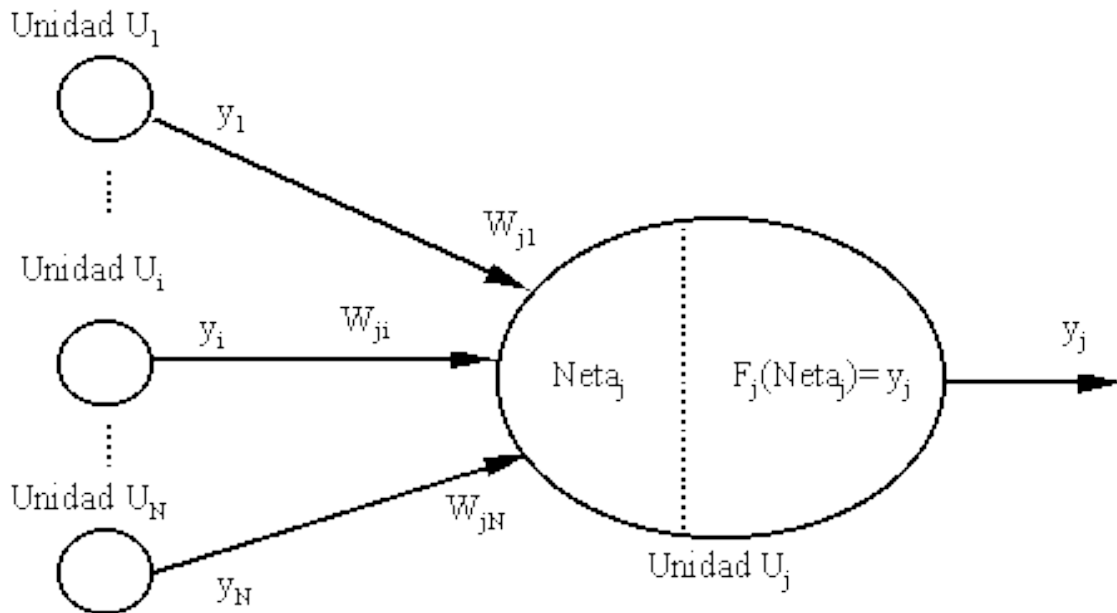
### 2.4.1. NEURONA

Las neuronas o neuronas artificiales son la base de las redes neuronales, son una técnica de aprendizaje automático que se basa en el funcionamiento de las neuronas biológicas. Estas redes están compuestas por una serie de neuronas artificiales que procesan y transmiten información a través de los datos de entrada. Estos cálculos se realizan en una sola neurona, y una capa de red puede contener varias neuronas. La información se procesa utilizando funciones de activación y luego se genera una salida mediante una función de salida [8].

La entrada de una neurona se recibe de la salida de sus neuronas predecesoras y sus respectivas conexiones. La entrada recibida se calcula como una suma ponderada y, a menudo, también se agrega un término de sesgo. Es la función de una función de propagación. Como se muestra en la Figura 2.4,  $f$  es la función de activación,  $w$  es el término de ponderación  $y_n$  es el término de sesgo. Cuando los cálculos están hechos, obtenemos la salida. La salida es la tarea a realizar, por ejemplo, identificar un objeto o si queremos clasificar una imagen, etc [8].

Por ejemplo, los datos de entrenamiento de entrada tendrán imágenes sin procesar o procesadas. Estas imágenes se pasarán a la capa de entrada. Los datos ahora se mueven a las capas ocultas donde se realizan todos los cálculos. Estos cálculos son realizados por las neuronas de cada capa.

## ANTECEDENTES



*Figura 2.4. Función de propagación de una neurona*

## 2.5. REDES NEURONALES

Una red neuronal está compuesta por varias capas de neuronas artificiales, cada una de las cuales procesa la información de entrada y la transmite a las capas siguientes. A través de un proceso de aprendizaje automático, las redes neuronales se ajustan para optimizar su rendimiento en una tarea específica, como el reconocimiento de patrones, la clasificación de imágenes o el análisis predictivo. Estas redes son utilizadas en una amplia variedad de campos, desde la medicina y la robótica hasta el comercio y el marketing. Las redes neuronales pueden adaptarse a los cambios en la capa de entrada. Permita que la red brinde los mejores resultados posibles sin rediseñar los criterios de terminación [7]. Derivado de la inteligencia artificial, el concepto de redes neuronales está ganando rápidamente popularidad en el desarrollo de sistemas comerciales. Una red neuronal puede extraer información por sí misma, pero se deben inicializar algunos parámetros para entrenar la red.

## ANTECEDENTES

### 2.5.1. REDES NEURONALES ARTIFICIALES

Una red neuronal artificial (RNA) o artificial neural network (ANN) por sus siglas en inglés, funciona en tres capas. La capa de entrada recibe la información de entrada. La capa oculta manipula la entrada. Finalmente, la capa de salida envía el resultado calculado [7].

De manera similar, las redes neuronales artificiales se entrenan con un conjunto de datos etiquetados para aprender a asociar entradas con salidas específicas. A través de un proceso de aprendizaje automático, las redes neuronales aprenden a reconocer patrones y a generar respuestas precisas a partir de los datos de entrada. Esto las convierte en una herramienta poderosa para tareas como el análisis de imágenes, el procesamiento del lenguaje natural, y el aprendizaje automático en general.

Las RNA aprenden a realizar tareas similares a través del aprendizaje o entrenamiento. Lo hace al observar varios ejemplos de puntos de datos históricos, como datos transaccionales e imágenes, la mayoría de los cuales no están programados para seguir ninguna regla específica. Por ejemplo, para distinguir entre automóviles y humanos, el ARN comienza sin conocimiento previo o comprensión de las propiedades de cada clase. A continuación, genere atributos y rasgos discriminatorios a partir de los datos de entrenamiento. Luego aprenda estas propiedades y utilícelas para hacer predicciones. [8].

En resumen, el objetivo principal del entrenamiento de una red neuronal es optimizar los pesos y sesgos de la red para minimizar el error entre los valores predichos y los valores reales. Se utilizan algoritmos de aprendizaje automático supervisado como el descenso del gradiente para ajustar los pesos y sesgos en función de la función de costo y lograr una mayor precisión en la tarea específica para la cual fue diseñada la red neuronal.

### 2.5.2. REDES NEURONALES CONVOLUCIONES

Las redes neuronales convolucionales (RNC) o convolutional neural networks (CNN) por sus siglas en inglés) son un tipo específico de redes neuronales profundas que se basan en los mecanismos de percepción visual de los organismos vivos. Estas redes están compuestas por



## ANTECEDENTES

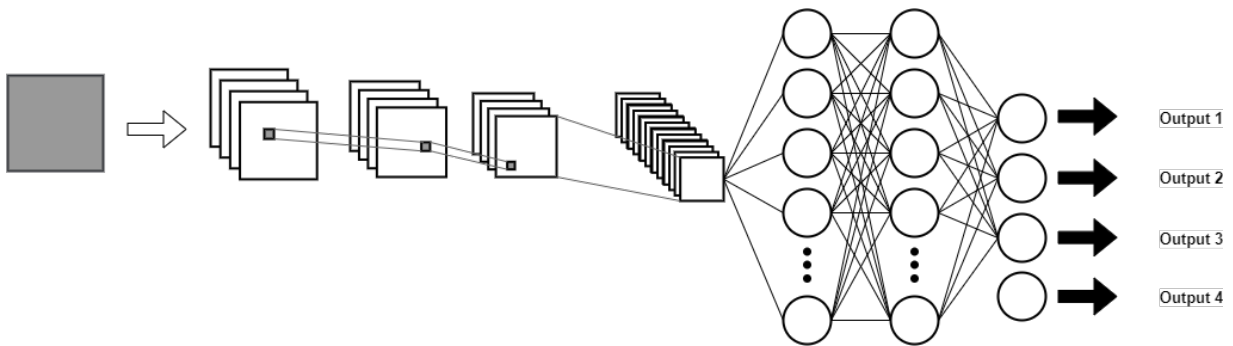
varias capas apiladas, entre las cuales se encuentran la capa convolucional, la capa de grupo y la capa totalmente conectada.

La primera capa de cualquier modelo de CNN es la capa de entrada, donde el ancho, la altura y la profundidad de la imagen de entrada se especifican como parámetros de entrada. Inmediatamente después de la capa de entrada, las capas convolucionales se definen con el número de filtros, el tamaño de la ventana del filtro, el relleno y la activación como parámetros [9].

La capa de salida se aplana para formar un único vector de características de red, que se envía a una capa completamente conectada. Finalmente, se define un clasificador con funciones de activación como sigmoid, softmax y tanh. El número de neuronas de salida se especifica en esta capa y las características extraídas se agregan en una puntuación de capa [9].

Las clases de normalización por lotes se aplican después de la capa de entrada o después de las clases de activación para estandarizar el proceso de aprendizaje y reducir el tiempo de aprendizaje. Otro parámetro importante es la función de pérdida, que resume el error en las predicciones durante el entrenamiento y la validación. La pérdida se devuelve al modelo CNN después de cada época para mejorar el proceso de aprendizaje [9].

Las redes neuronales convolucionales, observe Figura 2.5, se utilizan para resolver diferentes problemas prácticos en el cual se necesita realizar un procesamiento de imágenes. Lo que nos permite aprovechar la estructura que este tipo de red para solucionar de una manera práctica los problemas donde se trabaje con señales particulares [10].



*Figura 2.5. Ejemplo de red neuronal convolucional*

### 2.5.2.1.Convolución

La convolución es una operación matemática que cambia una función a otra y mide la integral de su punto de multiplicación, observe Figura 2.6. Tiene conexiones profundas con la transformada de Fourier y la transformada de Laplace, y se usa mucho en el procesamiento de señales. Las capas convolucionales en realidad usan correlaciones cruzadas, que son muy similares a las convoluciones [10].

Las capas convolucionales se utilizan para extraer mapas de características significativos para la ubicación de entrada mediante el cálculo de la suma ponderada. Luego, cada mapa de características se pasa a través de una función de activación y se agrega sesgo para formar la salida [9].

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

 $*$ 

-1	0	1
-1	0	1
-1	0	1

 $=$ 

5	4	0	-8
10	2	-2	-3
0	2	4	7
3	2	3	16

*Figura 2.6. Convolución de una imagen*

### 2.5.2.2.Funciones de Activación

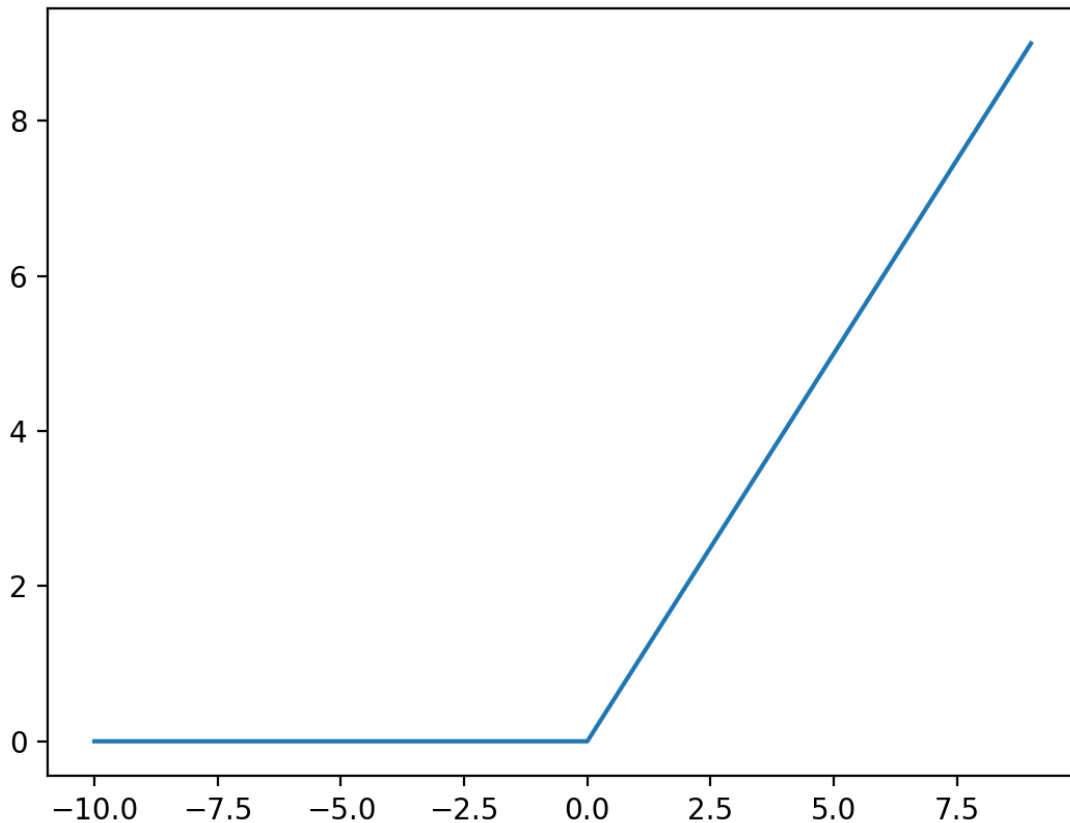
El papel principal de la función de activación es decidir si estimular la neurona o perceptrón. Desempeñan un papel central en la corrección del gradiente después de la formación tardía de la red. A veces se les llama funciones de transferencia. El comportamiento no lineal de las funciones de activación permite que las redes de aprendizaje profundo aprendan comportamientos complejos [8]. La función de activación genera unidades de salida. Se puede

## ANTECEDENTES

considerar como una función de decisión, es decir, determina si una neurona artificial tiene alguna característica en particular o no.

### 2.5.2.2.1. Rectificador Lineal Unitario o ReLu

La Unidad Lineal Rectificada o ReLU es una función de activación que elimina los lados negativos de un argumento, tal y como se muestra en la Figura 2.7.



*Figura 2.7. Función de activación ReLu*

La función de activación ReLU, vea Ecu. 2.1, es una de la más utilizadas dentro del aprendizaje profundo, debido a que es computacionalmente muy eficiente provocando un fuerte gradiente en la región calculada. Una característica de la activación de ReLU es que puede producir neuronas muertas durante el entrenamiento. Una neurona muerta siempre devuelve 0 para cada muestra en el conjunto de datos. La ventaja de esta propiedad es que la salida de una capa puede tener entradas que siempre son cero [11].

## ANTECEDENTES

ReLU es una función simple, la que consume menos poder computacional y se puede entrenar mucho más rápido. Es ilimitado y no tiene su centro en cero. Es distinguible en todos los lugares excepto cero. Dado que una función ReLU se puede entrenar mucho más rápido, encontrará que se usará con más frecuencia [8].

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \quad (2.1)$$

### 2.5.2.2.2. Softmax

La función softmax se usa en la capa final de la red neuronal para generar la salida de la red. El resultado puede ser una clasificación final de una imagen para distintas categorías.

La función softmax, vea Ecu. 2.2 calcula las probabilidades para cada una de las clases objetivo sobre todas las posibilidades. Es una función de activación que es útil para problemas de clasificación multiclase y obliga a la red neuronal a generar la suma de 1 [8].

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}} \quad (2.2)$$

Donde  $\mathbf{Z}_i$  son todos elementos de entrada vectoriales y pueden tomar cualquier valor. El último término de la expresión es el término de normalización, que representa una distribución de probabilidad válida porque todos los valores de salida de la función suman 1.

Como ejemplo, si la entrada es [2.33, -1.46, 0.56] y tomamos un softmax, entonces la salida correspondiente será [0.8383, 0.0189, 0.1428]. Esta salida asigna el peso más alto al valor más alto de en este caso. Y así se puede utilizar para resaltar el valor más alto.

### 2.5.2.3.Pooling

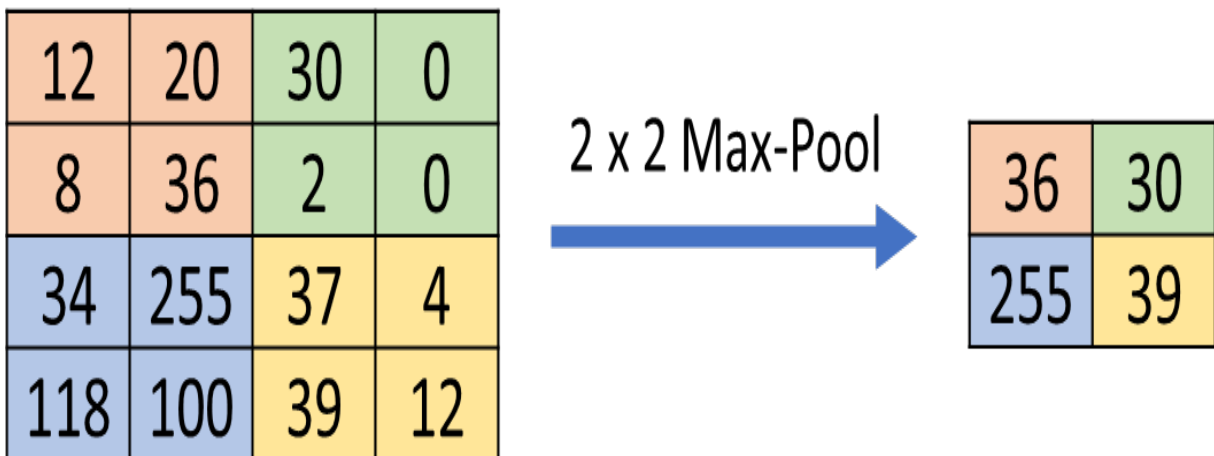
Las capas de agrupación se utilizan para reducir el tamaño de salida de las capas convolucionales. A medida que el tamaño del modelo aumenta con más y más filtros en la capa de acumulación, el tamaño de salida también aumenta exponencialmente, lo que dificulta la gestión de la computadora. Se agregan grupos de capas para reducir el tamaño para facilitar el

## ANTECEDENTES

cálculo y, a veces, para eliminar el ruido. La clase de agrupamiento puede ser la clase de agrupamiento máximo [9]. La clase de grupo más utilizada es la clase de grupo máxima o también conocida como Max Pooling.

### 2.5.2.3.1. Max Pooling

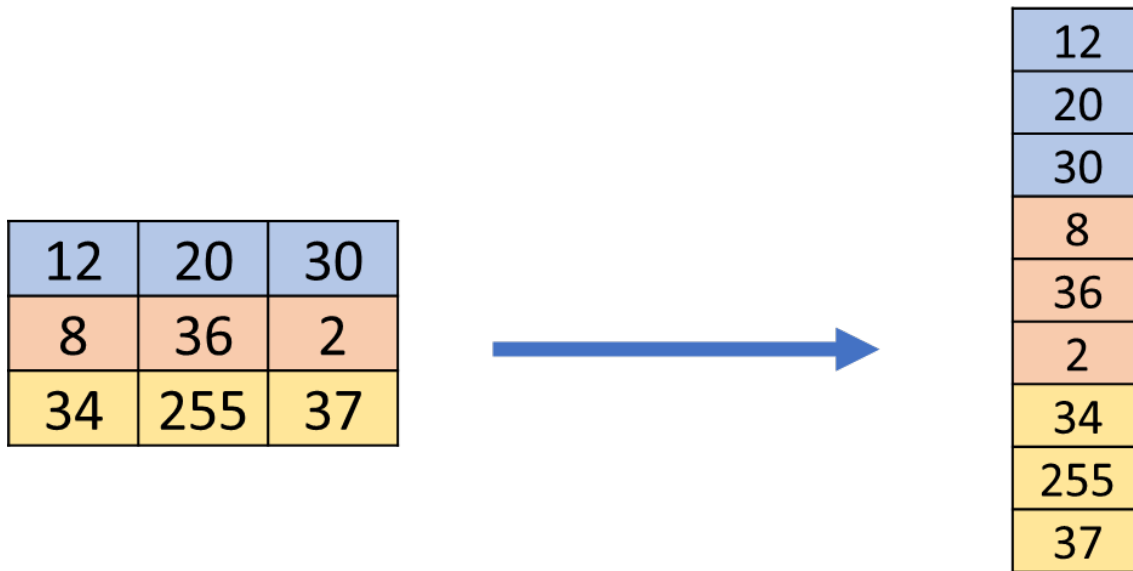
La salida de la capa de max pooling es dada por la activación máxima sobre regiones rectangulares de tamaño, creando invarianzas de posición sobre las regiones locales más grandes con el objetivo de reducir la muestra de la imagen de entrada por cierto factor en cada dirección, mejorando el rendimiento de generalización para seleccionar las características de la imagen rápido [12]. Tal y como se muestra en la Figura 2.8, donde se le está aplicando una función de max-pooling a una imagen.



*Figura 2.8. Aplicación de Max-pooling*

### 2.5.2.4.Flatten

Es una capa cuya función es convertir cada una de las imágenes de entrada en una matriz de una dimensión [13], observe Figura 2.9, para posteriormente ser utilizado como una futura entrada de una red neuronal artificial.



*Figura 2.9. Ejemplo flatten a una matriz*

#### 2.5.2.5. Funciones de Pérdida

La precisión de nuestro modelo se mide utilizando la función de pérdida. Esta función calcula la diferencia entre el valor real y el valor predicho. Cada tipo de pérdida tiene una función de pérdida diferente. Debido a que la pérdida es diferente, el rendimiento del modelo también será diferente [8].

Existen diferentes funciones de pérdida para regresión matemática y clasificación. Crossentropy se puede utilizar para eliminar una función para optimizar la pérdida. Para medir el error entre la salida real y la salida deseada, a menudo se usa la raíz cuadrada del error cuadrado. Algunos investigadores sugieren usar el error de entropía cruzada en lugar del error cuadrático medio del original. La Tabla 2.1 ofrece un breve resumen de las diferentes funciones de pérdida.



## ANTECEDENTES

**Tabla 2.1.** *Diferentes tipos de funciones de pérdida que se pueden utilizar y su respectiva ecuación.*

Función de Pérdida	Ecuación para la pérdida	Usada para
Cross-entropy	$-y(\log(p) + (1 - y)\log(1 - p))$	Clasificación
Pérdida de bisagra	$\max(0, 1 - y * f(x))$	Clasificación
Error absoluto	$ y - f(x) $	Regresión
Error cuadrático	$(y - f(x))^2$	Regresión
Pérdida de Huber	$L_{\delta} = \frac{1}{2}(y - f(x))^2, \text{ si }  y - f(x)  < \delta$ $= \delta \text{ sino } \delta y - f(x)  - \frac{1}{2}\delta^2$	Regresión

### 2.5.3. MODELOS DE REDES NEURONALES CONVOLUCIONALES EXISTENTES

Uno de los desafíos que enfrentan los analistas en el campo de la imagen es que los datos de entrenamiento pueden no ser adecuados para una aplicación específica. Es posible que desee realizar una clasificación en un conjunto de datos con un conjunto específico de etiquetas que pueden tener una disponibilidad limitada [14]. Si se desea construir un clasificador de imágenes, pero no se cuenta con la suficiente información para el entrenamiento, entonces es una buena idea utilizar modelos pre entrenado.

Estas configuraciones causan problemas porque las redes neuronales requieren una gran cantidad de datos de entrenamiento para construir desde cero. Sin embargo, un punto clave sobre los datos de imágenes es que las características extraídas de un conjunto de datos en particular son altamente reutilizables en todas las fuentes de datos. Por ejemplo, si se usa la misma cantidad de píxeles y canales de color en diferentes fuentes de datos, la forma en que se muestra el dato no cambiará mucho. En este caso, es útil una fuente de datos común que represente varios tipos de imágenes. Por ejemplo, el conjunto de datos de ImageNet [15], contiene más de un millón de imágenes extraídas de 1000 categorías que se encuentran en la vida cotidiana. Las 1000 categorías seleccionadas y la gran cantidad de imágenes en el conjunto de datos son lo suficientemente representativas y completas como para usarlas para extraer características de imágenes para configuraciones comunes.

## ANTECEDENTES

Esta nueva representación se puede utilizar para aplicaciones completamente diferentes, como la agrupación en clústeres. Este enfoque es tan común que las redes neuronales acumulativas rara vez se entrenan desde cero. Este enfoque de extracción de características independiente puede considerarse un tipo de aprendizaje de transferencia, ya que utilizamos fuentes públicas como ImageNet para extraer características para abordar varios problemas con los datos disponibles.

Este enfoque se ha convertido en una práctica común para muchas tareas de reconocimiento de imágenes, y muchos marcos de software como Caffe brindan acceso rápido a estas funciones. De hecho, Caffe ofrece una amplia gama de estos modelos pre entrenados que se pueden descargar y utilizar. Se puede usar para ajustar solo las capas más profundas (es decir, las capas más cercanas a la capa de salida) si hay datos de entrenamiento adicionales disponibles. Los pesos de la primera capa (más cercana a la entrada) son fijos.

La razón para formar solo capas más profundas y mantener la primera capa fija es que la primera capa solo captura características primitivas como bordes, mientras que las capas más profundas capturan características más complejas. La funcionalidad inicial no cambia mucho con la aplicación en cuestión y la funcionalidad más profunda puede ser sensible a la aplicación en cuestión. Por ejemplo, todos los tipos de imágenes requieren bordes de diferentes orientaciones para representarlos (obtenidos en la primera capa), pero las características correspondientes a las ruedas de un camión estarán asociadas al conjunto. Los datos contienen imágenes de camiones. En otras palabras, las primeras capas tienden a capturar características muy generalizadas (en diferentes conjuntos de datos de visión por computadora), mientras que las capas posteriores tienden a capturar características específicas de los datos.

### **2.5.3.1. ResNet**

ResNet, llamada así por Residual Neural Network, fue propuesto por Microsoft Research dirigido por He Kaiming en su artículo de 2016 Deep Residual Learning for Image Recognition. Los investigadores tomaron en cuenta los siguientes hechos: Sí, para la capa de red neuronal de 200 neuronas, colocamos 36 capas que calculan funciones de identidad simples, y la red resultante

## ANTECEDENTES

(56 capas) debería tener exactamente la misma (y no menos) eficiencia que la primera red. Recuerda que la función identidad devuelve exactamente el mismo valor que su argumento.

Esta arquitectura ganó la competencia ILSVRC en 2015 y logró un error entre los 5 primeros del 3,6 %, lo que resultó en el primer clasificador con un rendimiento a nivel humano [14], en la Figura 2.10, se observa una muestra parcial de lo que es la arquitectura de ResNet.

Los problemas como las pendientes de escape y las explosiones son causados por una mayor profundidad. Sin embargo, el trabajo muestra que el principal problema de entrenamiento en redes tan profundas no es necesariamente causado por estos problemas, especialmente si se usa la normalización por lotes. El principal problema es que es difícil hacer converger adecuadamente el proceso de aprendizaje en un tiempo razonable. Tales problemas de convergencia son comunes en redes con superficies de pérdida complejas. Aunque algunas redes profundas tienen una gran brecha entre el error de entrenamiento y el error de prueba, el error en los datos de entrenamiento y reevaluación es alto en muchas redes profundas. Esto implica que el proceso de optimización no ha progresado lo suficiente.

ResNet utiliza saltos entre capas para permitir la copia entre capas e introduce una vista iterativa de la ingeniería de características (a diferencia de una vista jerárquica). Las redes de memoria a largo plazo y los bloques repetitivos cerrados explotan principios similares en los datos secuenciales al permitir que partes del estado se copien de una capa a la siguiente mediante compuertas de regulación.

### 34-layer residual

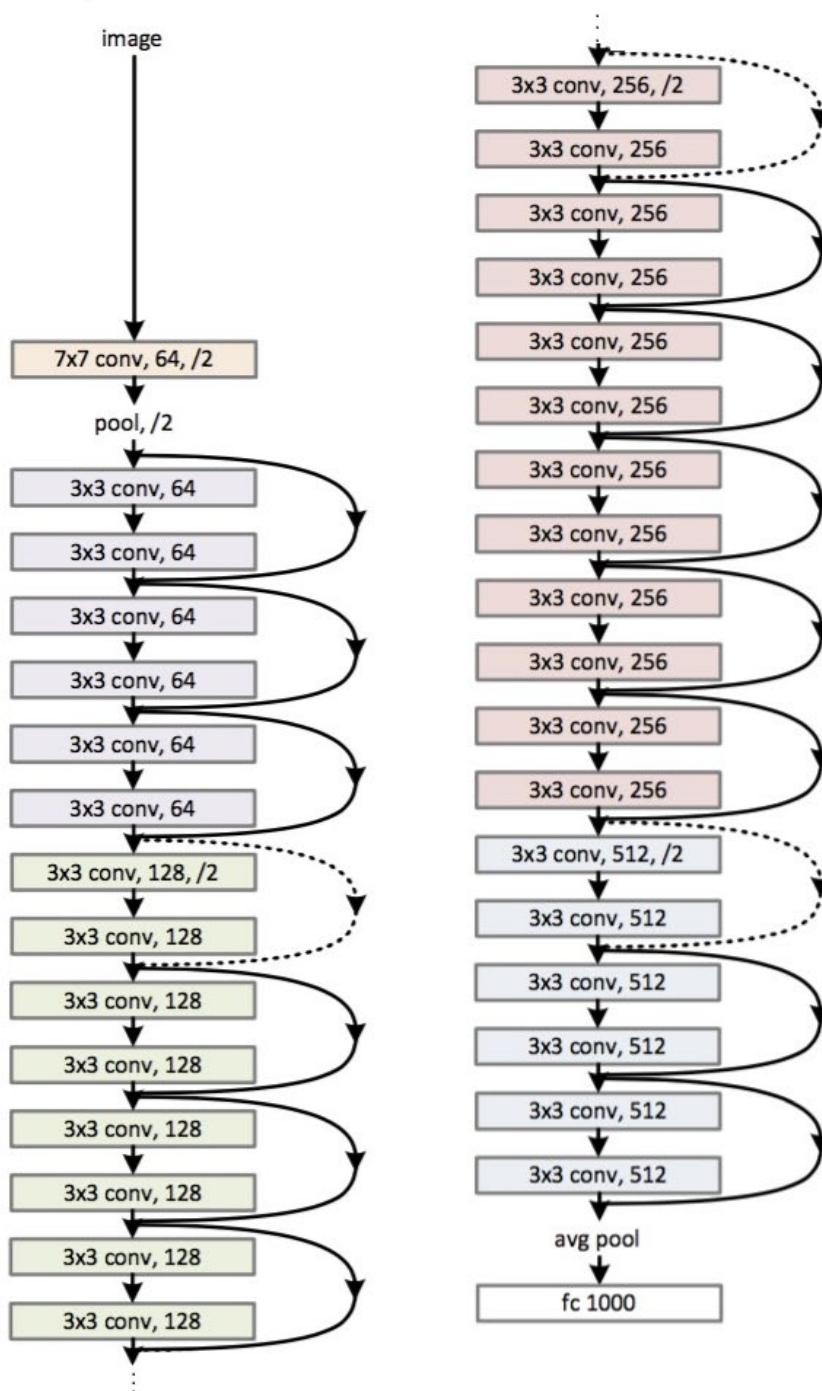


Figura 2.10. Muestra la arquitectura de ResNet

## ANTECEDENTES

### 2.5.3.2.MobileNet

Creado por Google, la característica principal de MobileNet es que utiliza una forma diferente de bloque convolucional "sándwich". Divide las circunvoluciones de 3x3 en una convolución de profundidad de 3x3, seguida de una CONV de 1x1 puntos [16].

MobileNet tiene algunas de las mejores proporciones de precisión, velocidad y parámetros para cualquier diseño de red neuronal, también es más liviana y rápida en dispositivos móviles.

Sin embargo, actualmente no existe una implementación buena (rápida) de convoluciones profundas para ejecutar en GPU; por lo tanto, el entrenamiento será probablemente más lento que usar una operación de convolución normal. Sin embargo, donde esta red realmente brilla en este momento es en diseños de CPU más pequeños, donde la mayor eficiencia es más visible.

### 2.5.3.3.InceptionNet

En la primera versión, una versión ingenua de Inception, se realizaron tres tamaños diferentes de convolución: 1x1, 3x3 y 5x5. Además, también se propuso una agrupación máxima de 3x3. Todas las salidas respectivas se apilan y alimentan al siguiente módulo Inception.

Pero a medida que aumenta el costo de computación, los investigadores agregaron una capa convolucional adicional de 1x1 para reducir la dimensionalidad. Esto limita la cantidad de canales de entrada y 1x1 es menos costoso computacionalmente que 3x3 o 5x5. Una característica notable es que la convolución 1x1 está detrás de la capa compuesta máxima [8].

Usando esta segunda versión reducida, se creó una red completa, llamada GoogLeNet. Los investigadores eligieron este nombre para rendir homenaje a Yann LeCuns, el pionero de la arquitectura LeNet-5 [8].

### 2.5.3.4.VGGNet

Creado por Visual Geometry Group (VGG) en la Universidad de Oxford, VGGNet fue una de las primeras arquitecturas en introducir la idea de apilar varias capas juntas. VGGNet usa

## ANTECEDENTES

solo filtros muy pequeños con dimensiones espaciales de  $3 \times 3$ , en comparación con AlexNet, que es de hasta  $11 \times 11$ . Estos filtros convolucionales de  $3 \times 3$  generalmente se intercalan con capas de agrupación de hasta  $2 \times 2$  [16].

El uso de filtros tan pequeños significa que la vecindad de los píxeles vistos también es muy pequeña. Inicialmente, esto puede dar la impresión de que el modelo solo tiene en cuenta la información local. Sin embargo, es interesante apilar los filtros pequeños uno tras otro para el mismo "campo de recepción" que un filtro grande. Por ejemplo, apilar tres filtros de  $3 \times 3$  tendrá el mismo campo de recepción que un solo filtro de  $7 \times 7$ .

Esta vista previa de filtros apilados ofrece la ventaja de poder tener una estructura más profunda (que generalmente encontramos que siempre es mejor) manteniendo el mismo tamaño del campo receptor, mientras reduce la cantidad de parámetros. Esta idea se explora más adelante en el capítulo.

El subcampeón del desafío ILSVRC 2014 fue VGGNet, desarrollado por K. Simonyan y A. Zisserman. Es una arquitectura muy simple y clásica con 2 o 3 capas de convoluciones, capas de agrupación, luego 2 o 3 capas de convoluciones, capas de agrupación, etc. (solo 16 capas de convoluciones en total) y densas 2 capas ocultas y la red final. capa, incluida la capa inicial con solo filtros  $3 \times 3$ , pero muchos filtros [13].

## 2.6. ESTADO DEL ARTE

En esta sección se agrupa la investigación realizada del estado del arte para la detección y clasificación del uso de cubrebocas, existen diversos artículos científicos que se concentran en diferentes métodos para la detección del uso de cubrebocas.

A causa del virus del COVID-19, se han adaptado diferentes técnicas a través de inteligencia artificial para la detección y clasificación de personas que utilicen cubrebocas. En esta sección, se abordarán algunos de los trabajos más relevantes en torno a la clasificación de cubrebocas multiclase.

## ANTECEDENTES

La mayoría de los artículos publicados se enfocan en una clasificación binaria, donde, el objetivo es identificar a aquellas personas están utilizando cubrebocas, así como, las personas que no estén utilizando cubrebocas. Se incluyeron también las publicaciones, que utilizan diferentes modelos pres entrenados.

La Inteligencia Artificial (IA) y el aprendizaje profundo son tecnologías que han ido en constante crecimiento y en torno a ellas se han desarrollado e implementado muchas aplicaciones en la industria, se ha trabajado en diversas áreas como reconocimiento de patrones, procesamiento de imágenes, entre otras. Las CNN han demostrado ser bastante eficientes para resolver problemas de patrones, algunas arquitecturas estándar como ResNet [17], YOLO [18] y MobileNet [19] ya tienen un modelo de red neuronal convolucional cargado.

En muchos casos se trabajan [20] con las mismas bases de datos que se utilizaron en este trabajo, sin embargo, existen algunos casos en que utilizaron bases de datos propias, o algunos otros publicados sitios web. En la Tabla 2.2, Tabla 2.3 y Tabla 2.4, se resumen los resultados de los trabajos relacionados.

**Tabla 2.2.** *Artículos científicos relacionados Parte 1*

<b>1er Autor</b>	<b>Tipo de Detección</b>	<b>Modelo de Clasificación</b>	<b>Conjunto de datos</b>	<b>Software</b>	<b>Precisión</b>
Singh [21]	Binario	YOLOv3	MAFA, WIDER FACE, Manual	Tensorflow, Keras	55%
Syed [22]	Binario	RNC	MFDD, RMFRD, SMFRD	Librerías Python	98%
Sethi [23]	Binario	RNC	MAFA	PyTorch	98.2%

ANTECEDENTES

**Tabla 2.3.** *Artículos científicos relacionados Parte 2*

<b>1er Autor</b>	<b>Tipo de Detección</b>	<b>Modelo de Clasificación</b>	<b>Conjunto de datos</b>	<b>Software</b>	<b>Precisión</b>
Deshmukh [24]	Triple	MobileNetV2	RFMD, MaskedFace-Net	-	99%
Bhattarai [25]	Triple	ResNet50	Kaggle [26, 27], MaskedFaceNet	OpenCV, Tensorflow, Keras	91%
Pham-Hoang-Nam [28]	Triple	ResNet50	Kaggle [29, 30], MaskedFaceNet, MAFA,	Tensorflow, Keras	94.59%
Yu [31]	Triple	YOLO-v4 Mejorado	RFMD, MaskedFace-Net	-	98.3%
Aydemir [32]	Triple	RNC	Manual, MaskedFace-Net	MATLAB	99.75
Soto-Paredes [33]	Triple	ResNet-18	MaskedFace-Net, Kaggle [34]	PyTorch	99.05



## ANTECEDENTES

**Tabla 2.4.** *Artículos científicos relacionados Parte 3*

<b>1er Autor</b>	<b>Tipo de Detección</b>	<b>Modelo de Clasificación</b>	<b>Conjunto de datos</b>	<b>Software</b>	<b>Precisión</b>
Wang [35]	Triple	InceptionV2	RMFRD, MAFA, WIDER FACE, MaskedFace- Net	OpenCV, MATLAB	91.1%
Rudraraju [36]	Triple	MobileNet	RMFRD	OpenCV, Keras	90%
Jones [37]	Triple	RNC	MaskedFace- Net	Tensorflow, Keras	98.5%
Jiang [38]	Triple	YOLOv3	WIDER FACE, MAFA, RFMD, Internet	Tensorflow	-

En [37] proponen una red neuronal convolucional (CNN) para detectar personas con cubrebocas que se utilice correctamente, incorrectamente o que no tenga cubrebocas, utilizando MaskedFaceNet y Flickr-Faces-HQ dataset [39] alcanzando un porcentaje de precisión del 98.5%. En [24], los autores presentan un sistema para identificar violaciones de protocolos de cubrebocas, utilizando el Haar Cascades para obtener la región de interés, así como la arquitectura de MobileNetV2 como modelo, logrando un 99% de precisión con un conjunto de datos de MaskedFaceNet y Real Facemask Dataset. De igual manera en [25], el autor presenta una interfaz gráfica de usuario (GUI) para detectar el uso de cubrebocas clasificándolo en las tres clases de los autores anteriores, utilizando la arquitectura ResNet50 con una precisión del 91%, de la misma manera [40] utilizó ResNet50 con 4 diferentes conjuntos de datos entre ellos MAFA, MaskedFace-Net y dos de Kaggle. El autor [31] propone el reconocimiento de cubrebocas y algoritmo de detección de uso estándar basado en el YOLO-v4 mejorado, alcanzando un 98.3% de precisión. Otros estudios [41] [42] [23] presentan modelos de redes

## ANTECEDENTES

neuronales convolucionales para la detección del uso de cubrebocas utilizando paquetes de aprendizaje automático, tales como Tensorflow, Keras, OpenCV y Scikit-Learn.

Proponen en [43] el uso de la arquitectura de red neuronal MobileNetV2 [44] en combinación con Single Shot Detector (SSD) [45] realizando predicción en tiempo real usando bibliotecas como OpenCV con un sistema de alerta para detectar personas que usan o no hacen uso del cubrebocas a través de una raspberry pi 4 logrando entre un 85% y un 95% de porcentaje de precisión.

En otros trabajos [46], utilizan tecnologías para la identificación de la máscara, analizando en tiempo real la categoría a la que pertenece, clasificando en dos clases Mask y NoMask, agregando métodos para mejorar el conjunto de datos y eliminando imágenes con poca luz.

Las principales diferencias que encontramos en los artículos publicados son la arquitectura de sus modelos, el conjunto de datos, pre procesamiento y librerías de software utilizadas para entrenar sus modelos.

## CAPÍTULO III

### METODOLOGÍA

La metodología aplicada para elaborar el modelo propuesto es presentada en este capítulo, donde se muestran las diversas bases de datos de rostros utilizando cubrebocas que existen, el pre procesamiento utilizado es presentado, así como las tecnologías utilizadas para desarrollar el modelo.

#### 3.1. BASES DE DATOS

Existen diversas bases de datos relacionados al uso de cubrebocas, esto debido a que la pandemia del COVID-19 han surgido diversos trabajos relacionando a la posición del cubrebocas. En esta sección se presentan algunos conjuntos de datos que muestran el uso de cubrebocas.

##### 3.1.1. BASE DE DATOS MASKEDFACE-NET

MaskedFace-Net [3] [47] es un conjunto de datos de caras humanas con una correcta o incorrecta manera de usar el cubrebocas, cuenta con 133783 imágenes y se encuentra basada en el conjunto de datos de Flickr-Faces-HQ (FFHQ) [48].

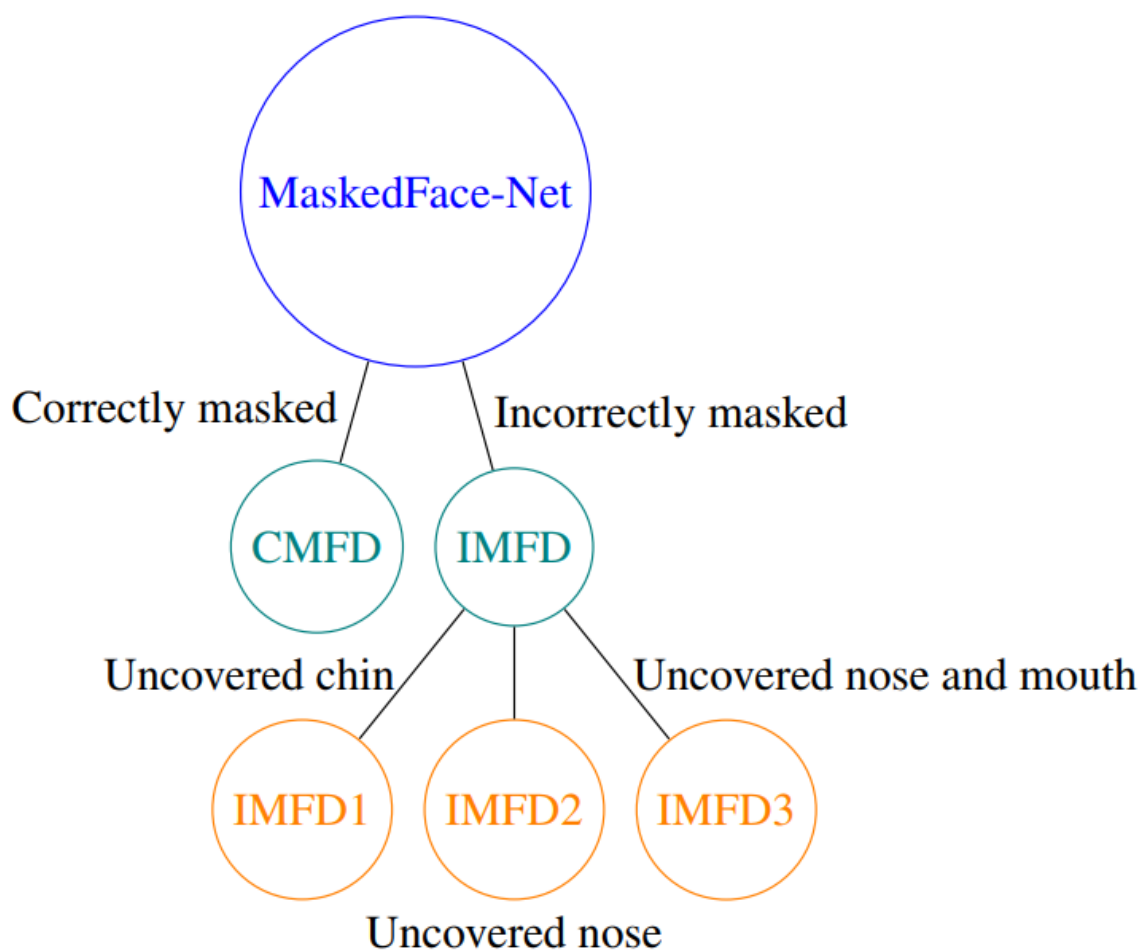


*Figura 3.1. MaskedFace-Net base de datos ejemplo*

Se encuentra dividida en 4 clases diferentes, observe Figura 3.1 y Figura 3.2:

- Correctly masked
- Incorrectly masked
  - Undercovered chin
  - Undercovered nose
  - Undercovered nose and mouth

### View of the MaskedFace-Net data Tree



*Figura 3.2. Clasificación de MaskedFace-Net*

El uso de cubrebocas aparece como una solución para limitar la propagación de la COVID-19. En este contexto, se esperan sistemas de reconocimiento eficientes para comprobar que los

rostros de las personas están enmascarados en áreas reguladas. Para realizar esta tarea, se necesita un gran conjunto de datos de rostros con cubrebocas para entrenar modelos de aprendizaje profundo para detectar a las personas que usan cubrebocas y a las que no las usan. Algunos grandes conjuntos de datos de rostros con cubrebocas están disponibles en la literatura. Sin embargo, por el momento, no hay un gran conjunto de datos disponible de imágenes de rostros con cubrebocas que permita verificar si los rostros con cubrebocas detectados se usan correctamente o no. De hecho, muchas personas no usan correctamente sus cubrebocas debido a malas prácticas, malos comportamientos o vulnerabilidad de las personas (por ejemplo, niños, ancianos).

### **3.1.2. BASE DE DATOS FACE MASK DETECTION**

Existen diversos conjuntos de datos publicados en diferentes sitios web, Face Mask Detection se destaca por estar clasificado en 3 diferentes clases, entre ellas, con cubrebocas, sin cubrebocas y uso incorrecto del cubrebocas.

El conjunto de datos es publicado en Kaggle, según el autor [29], el conjunto de datos estaba muy desequilibrado y sin limpiar. Entonces, para mejorar este conjunto de datos, las imágenes debían aumentarse de tal manera que cada clase tuviera una distribución equitativa de imágenes y eliminar las imágenes ruidosas que podrían considerarse atípicas. Por lo tanto, este conjunto de datos que he creado es una combinación de un conjunto de datos existente que se limpió y se distribuyó por igual en cada clase.

Contiene 3 carpetas etiquetadas según la clase a la que pertenecen. Las 3 clases son "with\_mask", "without\_mask" y "mask\_wearred\_incorrect". Cada carpeta contiene 2994 imágenes de personas que pertenecen a dicha clase etiquetada con un total de 8982 imágenes.

### **3.1.3. BASE DE DATOS MAFA**

El conjunto de datos MAFA consta de 30.811 imágenes de Internet, de las cuales se anotaron manualmente 35.806 rostros humanos con cubrebocas. Durante el proceso de anotación, se

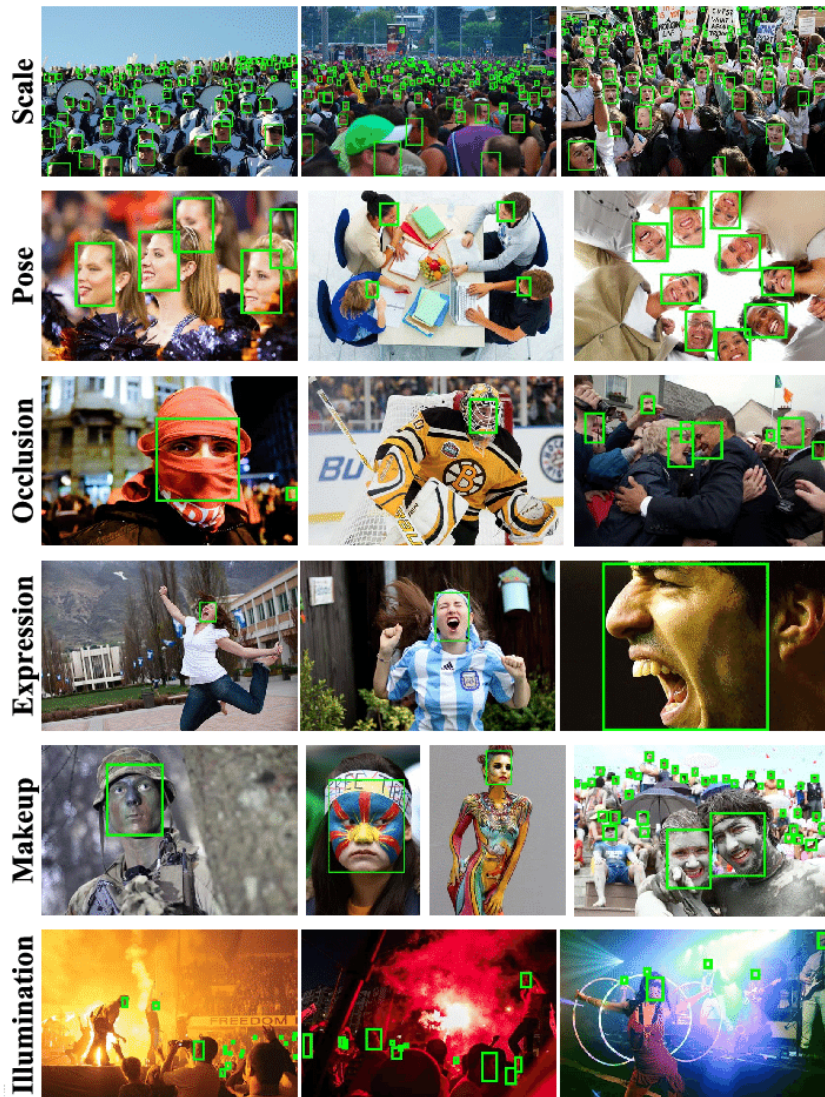
aseguraron de que cada imagen contenga al menos un rostro cubierto por diferentes tipos de máscara, mientras que seis atributos clave de cada rostro, incluyendo la posición del rostro, los ojos y el cubrebocas, la orientación del rostro, el grado de oclusión y el tipo de cubrebocas, los cuales fueron anotados manualmente y cotejados por nueve personas. El conjunto de datos se publicará próximamente en Internet, puede facilitar el desarrollo de nuevos dispositivos de detección facial en el futuro [49].

Al examinar las características clave de los rostros con cubrebocas en MAFA, se encontró con que la mayoría de las propiedades faciales pueden perderse en los rostros muy ocluidos (por ejemplo, los rostros sin ojos solamente), aunque son muy diversos pueden producir diferentes tipos de ruido.

#### **3.1.4. BASE DE DATOS WIDER FACE**

El conjunto de datos de WIDER FACE contiene muchas anotaciones, como oclusiones, posturas, categorías de eventos y cajas de delimitación de rostros. Los rostros del conjunto de datos propuesto son extremadamente difíciles debido a las grandes diferencias de escala, postura y oclusión, como se muestra en la Figura 3.3. Además, demostramos que el conjunto de datos WIDER FACE es una fuente de entrenamiento eficaz para la detección de rostros. Comparamos varios sistemas de detección representativos, proporcionamos información sobre el mejor rendimiento y recomendamos soluciones para hacer frente a las variaciones a gran escala. Por último, se discuten casos de error comunes que merecen una investigación más detallada [50].

El conjunto de datos WIDER FACE incluye 393.703 cuadros delimitadores de rostros etiquetados en 32.203 imágenes.



*Figura 3.3. Ejemplo de imágenes del conjunto de datos WIDERFACE*

### 3.2. CONCEPTOS DEL PREPROCESAMIENTO

El procesamiento de imágenes digitales es un conjunto de técnicas utilizadas para mejorar la calidad de las imágenes digitales y extraer información relevante de ellas. Es una disciplina que combina teoría y algoritmos para automatizar el proceso de recuperación de información del mundo real a partir de imágenes individuales o secuencias. Uno de los métodos utilizados en



este proceso es la correlación, que permite el reconocimiento de imágenes mediante el uso de filtros específicos.

### 3.2.1. CAFFE MODEL

Caffe model es un algoritmo de detección de rostros que utiliza técnicas de visión por computadora y aprendizaje profundo, este modelo pre-entrenado es muy eficiente con caras en diferentes ángulos, escrito en C++ y proporciona enlaces para Python y MATLAB, se encuentra enfocada en aplicaciones de visión artificial y rápida de implementar en CNN [51].

Caffe (Convolutional Architecture for Fast Feature Embedding) es un marco de aprendizaje profundo de código abierto que admite una variedad de arquitecturas de aprendizaje profundo como CNN, Region with CNN (RCNN), Long short-term memory (LSTM) y redes totalmente conectadas.

Caffe proporciona a los científicos y profesionales multimedia un marco limpio y editable para algoritmos modernos de aprendizaje profundo y una colección de modelos de referencia. Un ejemplo de la detección del rostro se observa en la Figura 3.4



*Figura 3.4. Clasificación de MaskedFace-Net*

Proporciona un conjunto de herramientas para entrenar, probar, ajustar e implementar modelos, con licencias para uso académico [52]. El marco Caffe es una herramienta comúnmente utilizada en el desarrollo de aplicaciones de aprendizaje profundo, especialmente en el campo de la visión artificial. Se enfoca en la facilidad de uso y la portabilidad, permitiendo



a los desarrolladores probar, entrenar e implementar modelos de aprendizaje profundo de manera eficiente en diferentes plataformas. Además, Caffe sigue las mejores prácticas de ingeniería de software, incluyendo pruebas unitarias para garantizar la precisión y la velocidad de implementación. También es adecuado para la investigación debido a su estructura modular y la claridad en la separación de la definición de la red y su implementación.

El modelo para la detección del rostro que fue utilizado es pre entrenado con imágenes de tamaño de 300x300 pixeles con 140000 iteraciones, el framework fue creado y mantenido por Berkeley AI Research (BAIR) y otros compañeros de la comunidad [53], los archivos necesarios pueden ser descargados del repositorio en Github de OpenCV.

La función `cv2.dnn.readNet`, la cual es parte de Caffe, toma dos parámetros ("ruta/a/prototxtfile", "ruta/a/caffemodelweights"). Se recibe el número de rostros identificados después de usar el modelo de detección de rostros, que luego se proporciona como entrada al modelo de detección de mascarillas [53]. Los rostros se pueden detectar utilizando este modelo de detección de rostros tanto en imágenes estáticas como en transmisiones de video en tiempo real. En general, este modelo es rápido y preciso y tiene un uso mínimo de recursos.

Es un modelo de Caffe que se basa en Single Shot-Multibox Detector (SSD) y utiliza la arquitectura ResNet-10 como su columna vertebral. Se introdujo después de OpenCV 3.3 en su módulo de red neuronal profunda.

El modelo Caffe de OpenCV del módulo DNN es el mejor, a comparación de otros diferentes modelos pre entrenados como Haar Cascades, Dlib o MTCNN según el autor [54], dado que trabaja bien con la oclusión, los movimientos rápidos de la cabeza y también puede identificar las caras laterales. Además, también dio los fps más rápidos entre todos.

#### Ventajas [51]

- Fácil implementación.
- Modelos pre entrenados disponibles.
- Velocidad de entrenamiento rápida.
- Se utiliza para redes hacia adelante.

Desventajas [51]

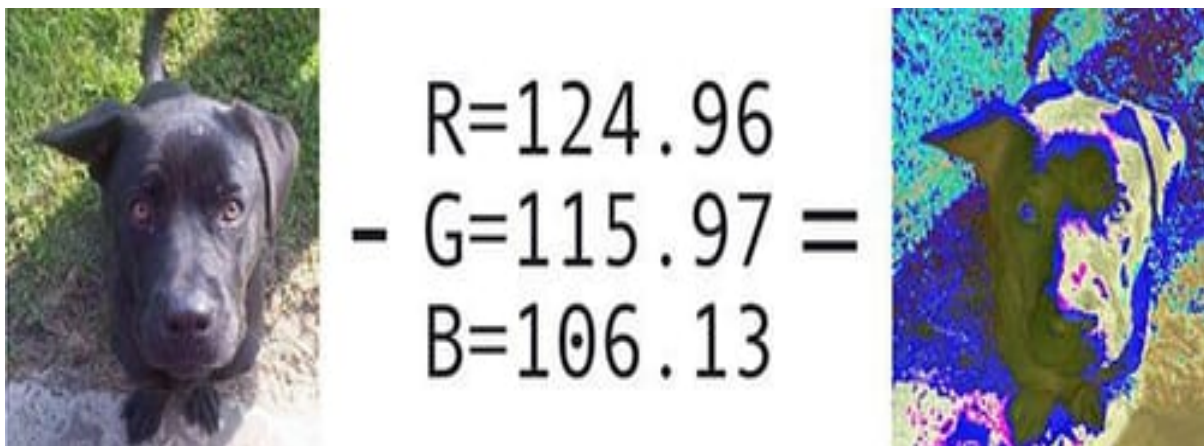
- Requiere escribir código para crear nuevas capas.
- Poco soporte para redes recurrentes.
- No hay soporte para entrenamiento distribuido.

### 3.2.2. SUSTRACCIÓN DE RGB

En el contexto del aprendizaje profundo y la clasificación de imágenes, estas tareas de preprocesamiento generalmente implican restar el promedio y escalar según varios factores. El nuevo módulo de red neuronal profunda de OpenCV contiene dos funciones que se pueden usar para preprocesar imágenes y prepararlas para la clasificación a través de modelos de aprendizaje profundo pre-entrenados.

Casi todos los modelos de aprendizaje profundo de última generación realizan restas y escalas promedio; la ventaja aquí es que OpenCV hace que estas tareas de preprocesamiento sean muy simples.

Antes de comenzar a entrenar la red neuronal profunda, primero se calcula las intensidades de píxeles promedio en todas las imágenes en el conjunto de entrenamiento para cada canal rojo, verde y azul.



*Figura 3.5. Resta de la media de una imagen*

La Figura 3.5 muestra una representación visual de la resta media donde la media RGB se calculó a partir de un conjunto de datos de imagen y se restó de la imagen original dando como resultado la imagen de salida.

Cuando se esté lista para transmitir una imagen a través de nuestra red (para entrenamiento o prueba), se resta la media, según la Ecuación 3.1, Ecuación 3.2 y Ecuación 3.3 de cada canal de entrada de la imagen:

$$R_i = R_a - \mu_R \quad (3.1)$$

$$B_i = B_a - \mu_B \quad (3.2)$$

$$G_i = G_a - \mu_G \quad (3.3)$$

Donde  $R_a$  es el pixel anterior,  $R_i$  es el pixel actual y  $\mu_R$  es la media de los pixeles del canal.

Donde  $B_a$  es el pixel anterior y  $B_i$  es el pixel actual y  $\mu_B$  es la media de los pixeles del canal.

Donde  $G_a$  es el pixel anterior y  $G_i$  es el pixel actual y  $\mu_G$  es la media de los pixeles del canal.

No todas las arquitecturas de aprendizaje profundo realizan restas y escalas intermedias. Antes de preprocesar una imagen, es importante asegurarse de que este sea un proceso que se pueda realizar para la neurona profunda de la red que está utilizando.

De manera informal, un blob es solo un (conjunto potencial) de imágenes con las mismas dimensiones espaciales (es decir, ancho y alto), la misma profundidad (número de canales), procesadas antes de la misma manera. Estos métodos se utilizan para preparar imágenes de entrada para la clasificación utilizando modelos de aprendizaje profundo previamente entrenados [55].

### 3.3. ESPECIFICACIONES DE SOFTWARE

Existen diversos programas y lenguajes de programación que facilitan el desarrollo de sistemas con inteligencia artificial, entre ellos, Python, C++, Java, cada uno de ellos cuentan

con librerías especializadas para el desarrollo de código para realizar trabajos de aprendizaje profundo.

### 3.3.1. PYTHON

Python es un lenguaje de programación interpretado y orientado a objetos, con una sintaxis clara y una gran cantidad de bibliotecas estándar para realizar tareas comunes de manera fácil. Fue creado por Guido van Rossum, un programador holandés, en 1989.

Python es un lenguaje de programación ampliamente utilizado en ciencia de datos debido a su gran comunidad de desarrolladores y una variedad de bibliotecas útiles desarrolladas en código abierto. Aunque el rendimiento de Python puede ser inferior para tareas computacionalmente intensivas en comparación con lenguajes de bajo nivel, bibliotecas extendidas como NumPy y SciPy, que están basadas en C y Fortran, ofrecen un rendimiento mejorado para operaciones vectoriales en arreglos multidimensionales. Para tareas de aprendizaje automático, la biblioteca scikit-learn es ampliamente utilizada y reconocida como una de las bibliotecas de aprendizaje automático de código abierto más populares y accesibles disponibles actualmente [56].

El uso de Python está muy extendido. Según las estadísticas de la comunidad de código abierto GitHub, es uno de los lenguajes de programación más populares de los últimos 10 años. Su estructura gramatical es muy concisa, lo que anima a la gente a escribir la mayor cantidad de código comprensible y a escribir la menor cantidad de código posible. Desde el punto de vista funcional, Python cuenta con un gran número de bibliotecas estándar y de terceros.

### 3.3.2. TENSORFLOW

TensorFlow es una interfaz de programación escalable y multiplataforma para implementar y ejecutar algoritmos de aprendizaje automático, incluidos prácticos paquetes para el aprendizaje profundo. TensorFlow está desarrollado por investigadores e ingenieros del equipo de Google Brain; y aunque el desarrollo principal está liderado por un equipo de investigadores

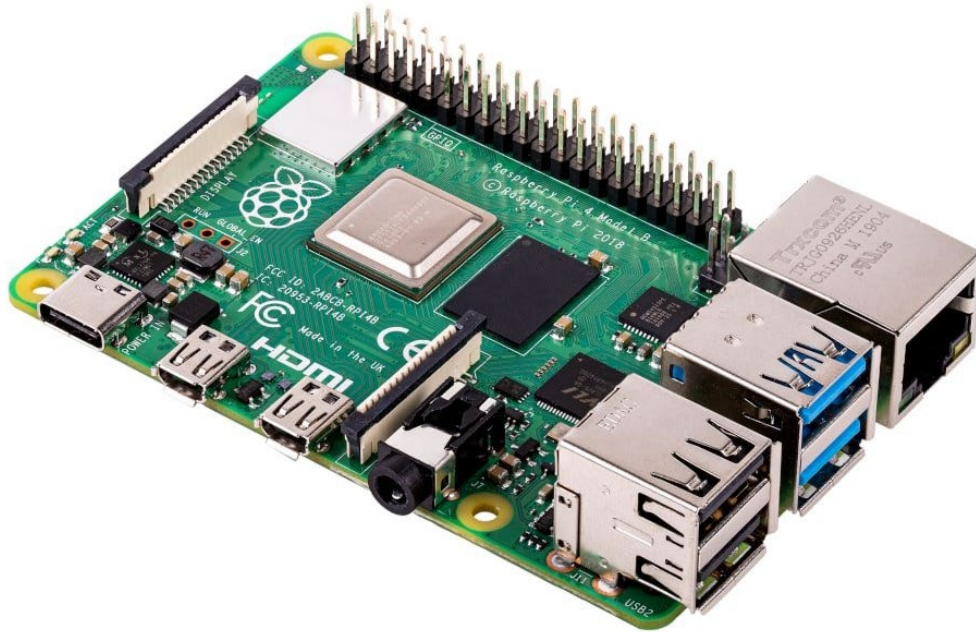
e ingenieros de software de Google, su desarrollo también cuenta con muchas contribuciones de la comunidad de código abierto. TensorFlow se diseñó originalmente para uso interno solo en Google, pero luego se lanzó bajo una licencia de código abierto autorizada en noviembre de 2015. Para mejorar el rendimiento de los modelos de aprendizaje automático de entrenamiento, TensorFlow permite la ejecución tanto en CPU como en GPU. Sin embargo, sus excelentes posibilidades de rendimiento se pueden descubrir al usar la GPU. TensorFlow admite oficialmente las GPU CUDA. El soporte para dispositivos compatibles con OpenCL aún está en versión beta. Sin embargo, es probable que OpenCL sea compatible oficialmente en un futuro próximo. TensorFlow actualmente admite interfaces de usuario para varios lenguajes de programación. Afortunadamente los usuarios de Python, la API de Python de TensorFlow es actualmente la más completa y atrae a muchos profesionales del aprendizaje automático y el aprendizaje profundo [56]. Se conforma de matrices multidimensionales, también para ejecutar operaciones en el conjunto de datos, construye un gráfico computacional similar a un diagrama de flujo que determina cómo fluyen los datos de una operación a la siguiente, entonces se llama TensorFlow porque está definiendo cómo fluirán los datos o los tensores a través del sistema.

### 3.3.3. KERAS

El desarrollo de Keras comenzó en los primeros meses de 2015. Hasta la fecha, se ha convertido en una de las bibliotecas más populares y utilizadas, construida sobre Theano y TensorFlow. Al igual que TensorFlow, Keras nos permite utilizar nuestra GPU para acelerar la formación de redes neuronales. Una de sus principales características es que tiene una API muy intuitiva y fácil de usar que nos permite implementar redes neuronales en pocas líneas de código. Keras se lanzó por primera vez como una API independiente que podía aprovechar Theano como backend, y posteriormente se añadió el soporte de TensorFlow. Keras también se ha integrado con TensorFlow desde la versión 1.1.0. Por lo tanto, si usted tiene la versión 1.1.0 de TensorFlow, no hay necesidad de instalar Keras adicional [56].

### 3.4. RASPBERRY PI

La Raspberry Pi, vea Figura 3.6 es un dispositivo pequeño y de bajo costo que puede ser utilizado como una computadora y ejecutar programas en Python [57], La unidad de procesamiento gráfico (GPU), la entrada y la salida funciona en un solo circuito. Los pines GPIO son un elemento importante que permite a la RPi acceder a la programación del hardware para controlar los circuitos electrónicos del dispositivo de E/S y el procesamiento de datos. Añade una fuente de alimentación, un teclado, un ratón y un monitor que se ejecutan en la Raspberry Pi a través del conector HDMI. Hay nuevos modelos disponibles que pueden comunicarse con Internet a través de Wi-Fi y puerto Ethernet. La RPi puede utilizarse con el sistema operativo Raspbian [58].



*Figura 3.6. Resta de la media de una imagen*

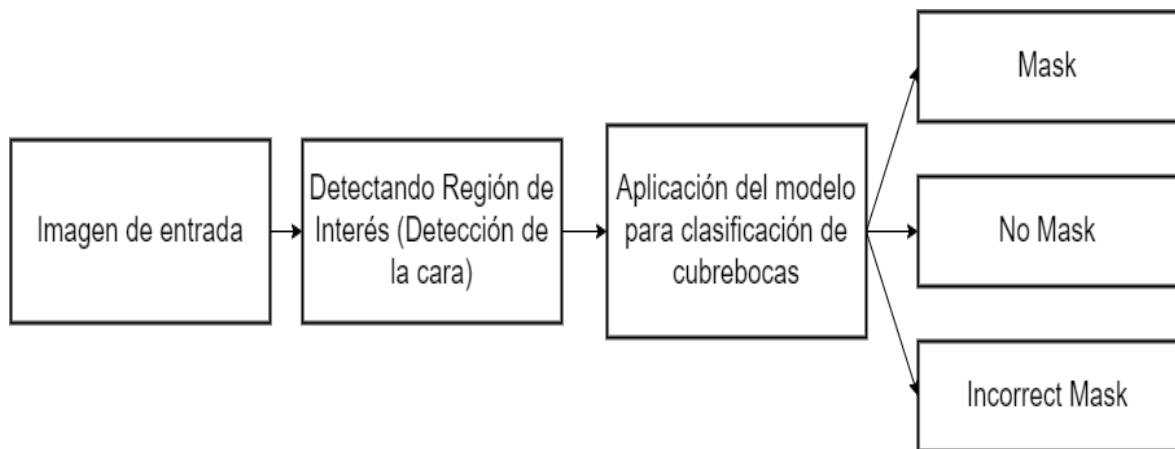
## CAPÍTULO IV

### DESARROLLO

En este capítulo, se presenta una descripción detallada de los métodos y técnicas utilizadas para lograr los objetivos de la investigación, incluyendo el proceso de creación del modelo de red neuronal convolucional y las estrategias de preprocesamiento empleadas.

#### 4.1. MÉTODO GENERAL PROPUESTO

Este trabajo propone un modelo combinando aprendizaje automático, aprendizaje profundo junto con librerías de Python. La propuesta incluye una RNC que permita la clasificación del uso de cubrebocas en tres diferentes clases (Mask, Incorrect Mask y No Mask). El flujo de trabajo básico del modelo se muestra en la Figura 4.1



*Figura 4.1. Flujo de trabajo básico*

#### 4.2. PREPROCESAMIENTO

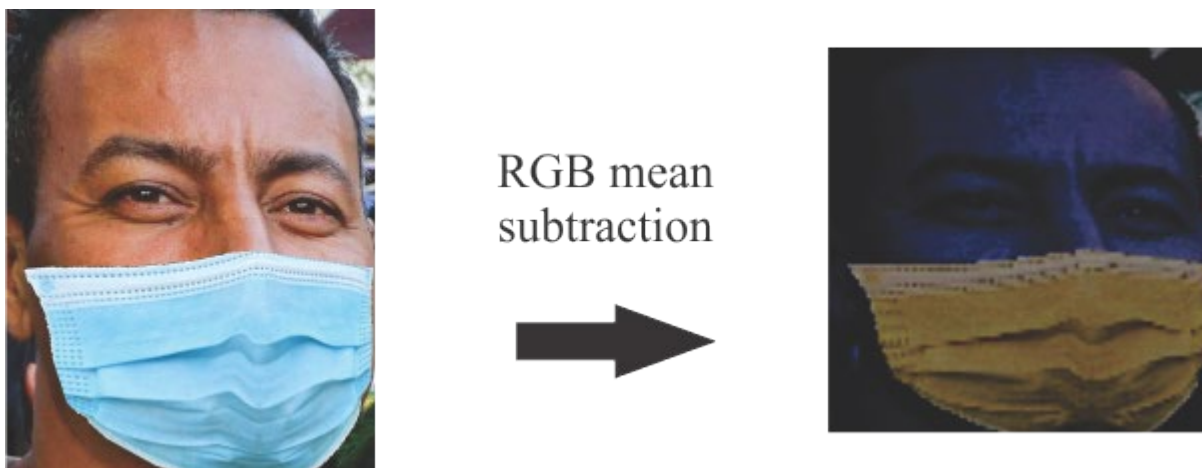
Para la etapa de preprocesamiento se utilizó el modelo de Caffe para identificar la región de interés (ROI) en este caso la cara, este modelo pre entrenado nos facilita esta tarea, para solamente trabajar con la región de la cara, como se observa en la Figura 4.2.

## DESARROLLO



*Figura 4.2. Detección de la región de la cara*

Parte del preprocesamiento incluye el redimensionamiento de la imagen de entrada, con el objetivo de trabajar con las características de la arquitectura de la RNC. Se usó la sustracción de la media para ayudar a contrarrestar las variaciones de luz en la imagen de entrada de nuestro conjunto de datos. Por lo tanto, se puede considerar la resta media como una técnica utilizada para ayudar a nuestra red neuronal convolucional, esta técnica utiliza el método `cv2.dnn.blobFromImage` de OpenCV. En la Figura 4.3 se observa una imagen del conjunto de datos aplicando el mencionado preprocesamiento.

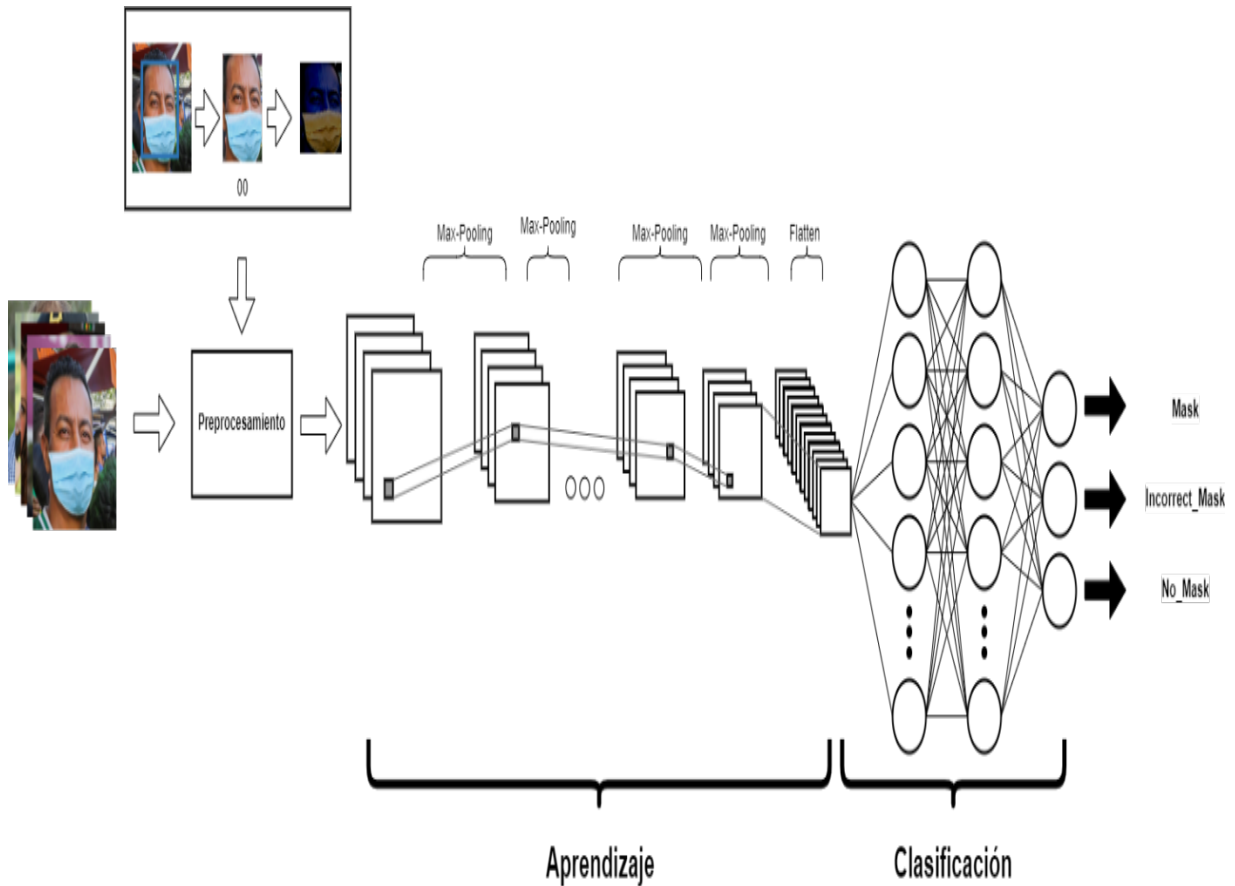


*Figura 4.3. Aplicando sustracción de RGB*



### 4.3. MODELO GENERAL PROPUESTO

Para el entrenamiento del modelo para la clasificación del uso de cubrebocas, se propone utilizar una red neuronal convolucional, observe Figura 4.4.



**Figura 4.4.** Modelo general de RNC para clasificación de uso de cubrebocas

Donde a partir la base de datos entrada se aplicará el preprocesamiento mencionado en el capítulo 4.2, seguido de la arquitectura de la red neuronal convolucional iniciando con la etapa de aprendizaje donde se selecciona el número de capas convolucionales. En la etapa de clasificación se utilizan los datos de entrada para clasificarla por medio de una red neuronal artificial donde se identificará si utiliza cubrebocas, utiliza incorrectamente el cubrebocas o no tiene cubrebocas.

#### 4.4. MODELO DEL SISTEMA TIEMPO REAL

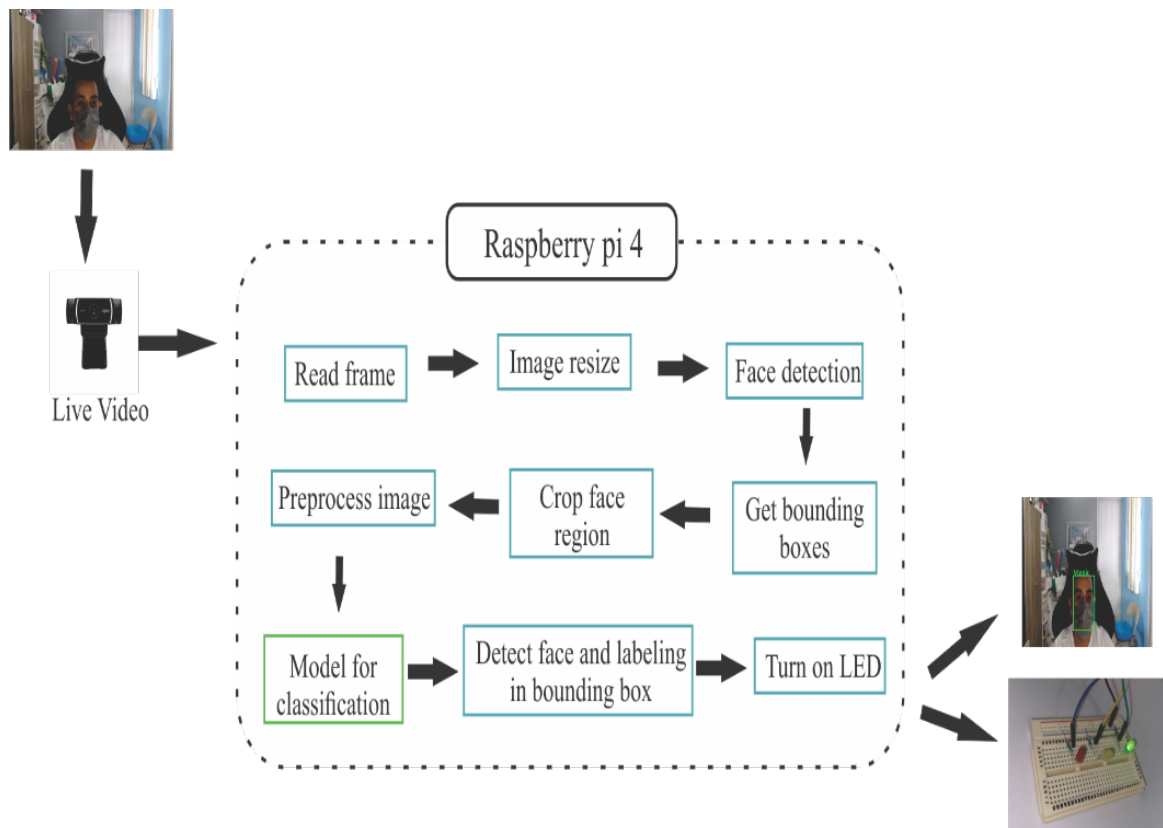
En este estudio se propone un nuevo modelo de RNC donde se combinan técnicas de visión por computadora, aprendizaje automático y aprendizaje profundo, para lograr la clasificación en las tres clases (NoMask, Mask, IncorrectMask). En la Figura 4.1. Se observa el plan para detectar el uso de cubrebocas, identificando la ROI de la imagen de entrada, aplicando el modelo de RNC para identificar en una de las tres clases.

El sistema tiempo real propuesto ayuda principalmente a disminuir la propagación de algún virus que se transmita por vías aéreas, identificando a las personas según el uso de cubrebocas. Esto se puede llevar a cabo a través de un monitoreo y alarma de acuerdo las normas que cada locación tenga. En esta sección se abarcará la solución propuesta para solucionar esta problemática, así como, la arquitectura que se propone para detectar a las personas de acuerdo al uso de cubrebocas y encerrar en un rectángulo la región de la cara.

Básicamente el sistema monitorea e identifica el uso de cubrebocas, donde a partir de nuestro modelo de RNC clasifica con imágenes en tiempo real a las personas en tres clases, Mask que es cuando la persona tiene cubrebocas y marca de color verde, IncorrectMask cuando la persona tiene colocado el cubrebocas incorrectamente identificado por el color amarillo, y NoMask que es cuando la persona no está utilizando ningún cubrebocas este de color rojo, como se observa en la Figura 4.5.

El sistema propuesto usa técnicas de visión por computadora para detectar la ROI, al igual que técnicas de aprendizaje profundo. Se indicará automáticamente por medio de una raspberry pi4 y una cámara las personas que utilizan cubrebocas, la que no utiliza cubrebocas o utilizan incorrectamente.

## DESARROLLO



**Figura 4.5.** Método general propuesto para el sistema tiempo real

## **CAPÍTULO V**

### **RESULTADOS**

En este capítulo se describen los experimentos llevados a cabo, incluyendo las pruebas estadísticas para evaluar el rendimiento del modelo de red neuronal convolucional, las tablas de resultados obtenidos y las pruebas de clasificación con imágenes reales. Además, se detalla la implementación del modelo en un sistema tiempo real.

#### **5.1. EXPERIMENTOS REALIZADOS**

Se tomaron en cuenta dos casos de estudio, donde el primero se basó exclusivamente utilizando la el conjunto de datos de MaskedFace-Net utilizando las cuatro clases que proporciona, mientras que el segundo caso de estudio utiliza dos conjuntos de datos de MaskedFace-Net en combinación con Flickr-Faces-HQ (FFHQ) para clasificarlo en tres clases durante en el entrenamiento.

En ambos casos se realizaron diversos experimentos donde cada uno de ellos fue ejecutado en 30 ocasiones, con el objetivo de conseguir la precisión media y la desviación estándar, a su vez se detectó el mejor caso en cada experimento.

##### **5.1.1. CASO DE ESTUDIO 1 (4 CLASES)**

El conjunto de datos de MaskedFace-Net está dividido en 4 clases totales, donde las dos categorías principales son Correctly masked e Incorrectly masked, esta última subclasificada en 3 clases Undercovered chin, Undercovered nose y Undercovered nose and mouth observe la Figura 5.1. Por lo cual a partir de una RNC es posible identificar a que clase pertenece una imagen.



Figura 5.1. Ejemplo de imágenes de MaskedFace-Net

Se propusieron 3 de modelos diferentes para identificar la arquitectura de la RNC que proporcione los mejores resultados.

La primera estructura de RNC propuesta nombrada Modelo A, consiste en una imagen de entra de tamaño 100x100x3 con 4 capas convolucionales, tal y como se muestra en la Figura 5.2. En la cual al final se clasifica en alguna de las 4 clases del conjunto de datos.

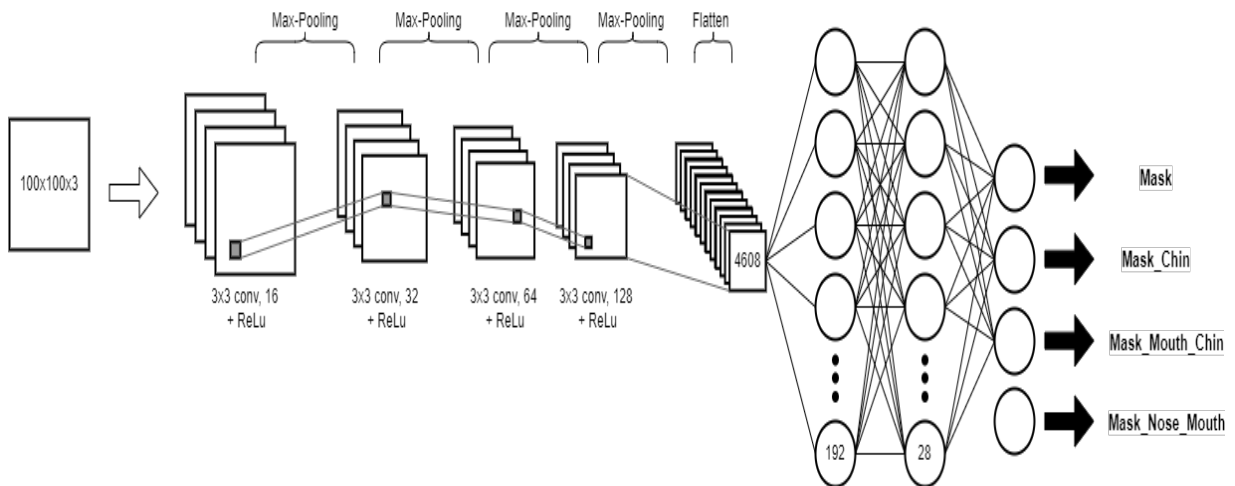
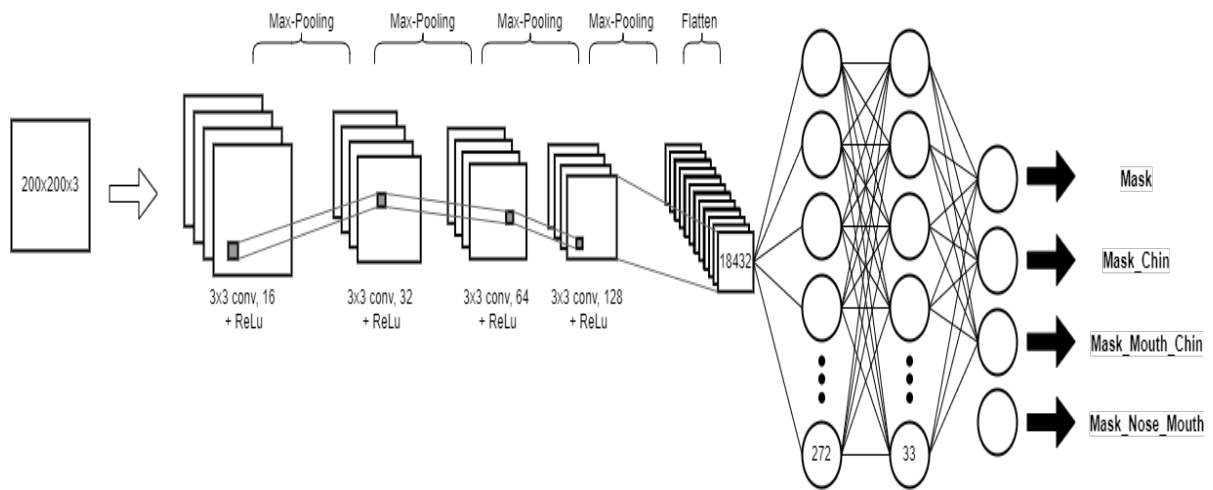


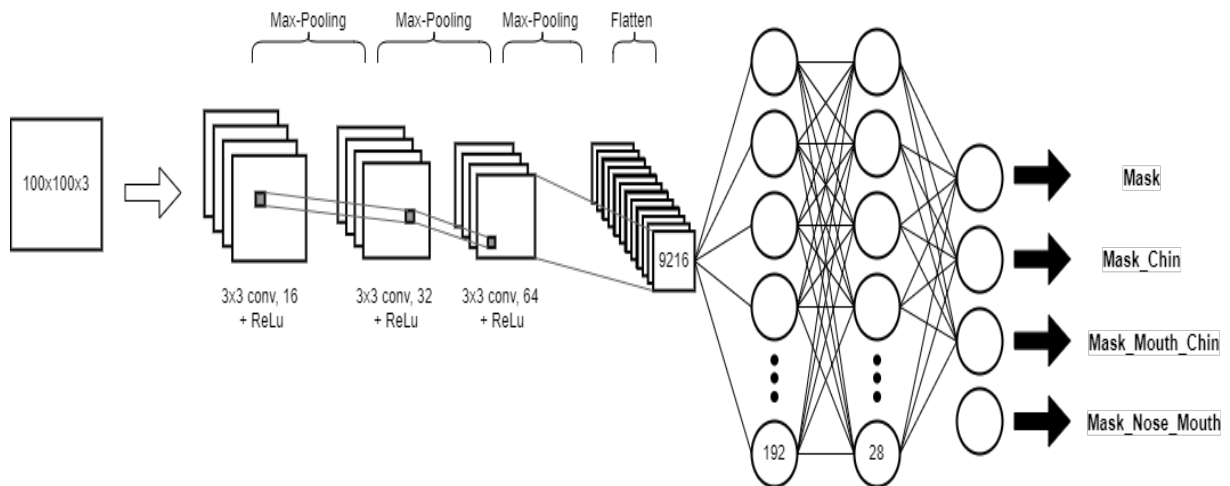
Figura 5.2. Estructura RNC del modelo A

El modelo B, observe Figura 5.3, cuenta con una imagen de entra más grande la cual es de tamaño 200x200x3, cuenta con 4 capas convolucionales de igual manera, pero debido a que es de mayor dimensión es más pesada computacionalmente hablando en la etapa de la clasificación.



**Figura 5.3.** Estructura RNC del modelo B

La última arquitectura de RNC, llamada modelo C utiliza una imagen de entrada de 100x100x3 pixeles, pero a comparación de las otras 2 esta cuenta con 3 capas convolucionales, como se muestra en la Figura 5.4., de igual manera con 4 neuronas de salida.



**Figura 5.4.** Estructura RNC del modelo C

Se realizaron 30 experimentos con cada uno de los tres modelos propuestos con el objetivo de identificar al que mejores resultados proporcione, tal como se observa en el Anexo 1. En la Tabla 5.1., se observan los promedios y desviación estándar de dichos modelos. Los datos completos de cada modelo se muestran en el Anexo 1.

**Tabla 5.1.** Experimentos para el caso de estudio 1.

<b>Modelo</b>	<b>A</b>		<b>B</b>		<b>C</b>	
	Promedio	0.9875	Promedio	0.9862	Promedio	0.9848
	Desviación Estándar	0.0042	Desviación Estándar	0.0021	Desviación Estándar	0.0031

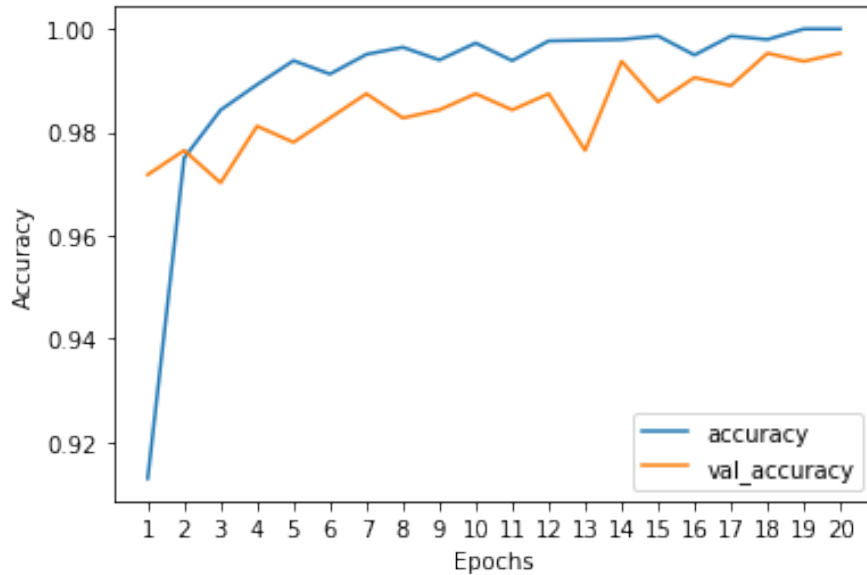
Cada modelo entreno durante 30 ocasiones con diferentes parámetros, entre ellos, el número de épocas, porcentaje de entrenamiento, pruebas y validación, tamaño de lote, coincidiendo el optimizador en este caso Adam (Adaptive Moment Estimation).

En la Tabla 5.2, se muestra un resumen donde a partir de los datos obtenidos de los experimentos, en el cual se comparan los 3 modelos identificando datos relevantes, en promedio el que presenta mejores resultados es el modelo A.

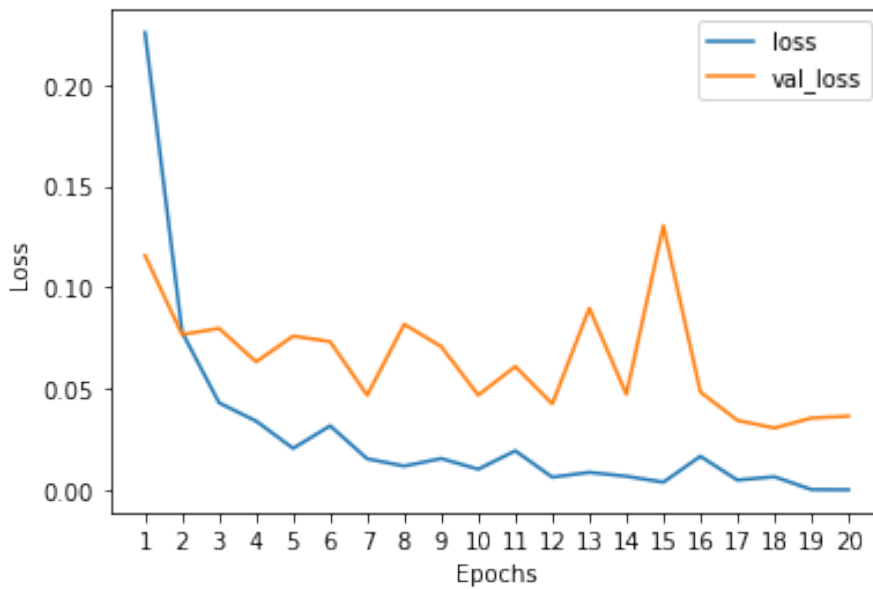
**Tabla 5.2.** Resumen de comparación de modelos.

<b>Parámetros</b>	<b>A</b>	<b>B</b>	<b>C</b>
<b>Épocas</b>	20	15	25
<b>Mejor caso</b>	0.9953	0.9878	0.9888
<b>Promedio</b>	0.9875	0.9862	0.9848
<b>Desviación Estándar</b>	0.0042	0.0021	0.0031
<b>% Entrenamiento</b>	70%	72%	72%
<b>% Pruebas</b>	20%	18%	18%
<b>% Validación</b>	10%	10%	10%
<b>Experimentos</b>	30	30	30
<b>Tamaño del lote</b>	30	50	20
<b>Optimizador</b>	Adam	Adam	Adam

El mejor caso se obtuvo en el modelo A; experimento número 1, donde el mejor caso fue 99.53% con una pérdida de 3.63%. Superando en promedio a los experimentos y sus respectivos modelos, utilizando la base de datos de MaskedFace-Net. En la Figura 5.5 y Figura 5.6 se muestran las gráficas del número de épocas contra la precisión de y la pérdida respectivamente, donde se puede observar cómo va convergiendo los la validación de cada uno de ellos.



**Figura 5.5.** Gráficas de número de épocas contra la precisión



**Figura 5.6.** Gráficas de número de épocas contra la pérdida



De acuerdo a los resultados obtenidos de la Tabla 5.2, se realizaron pruebas de hipótesis para comprobar la afirmación de que el modelo A es el que obtuvo mejores resultados.

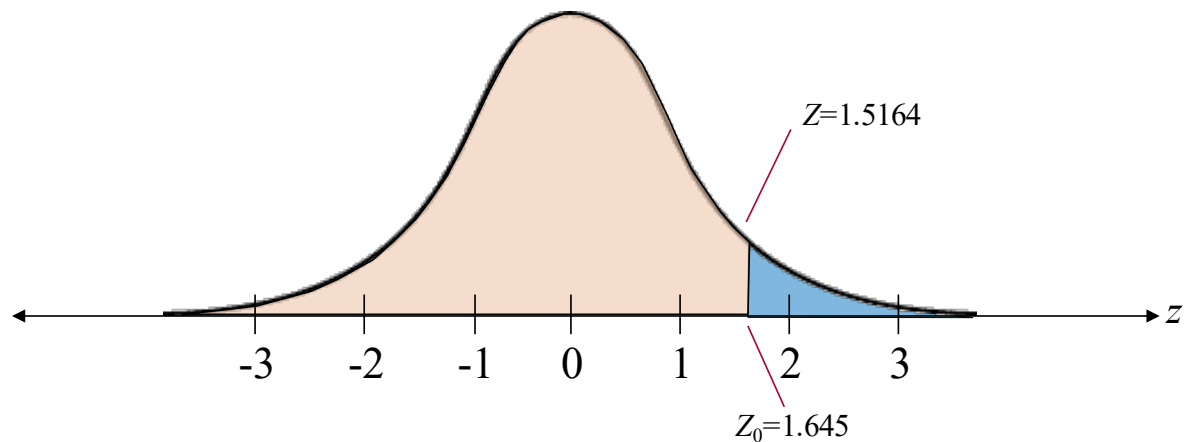
#### 5.1.1.1. Prueba de hipótesis 1

La prueba de hipótesis 1 tiene como afirmación de que “La precisión media del modelo A es mayor a la precisión media del modelo B”. Por lo tanto, hipótesis nula sería que la precisión media del modelo B es mayor o igual a la precisión media del modelo A, para realizar los cálculos de la prueba estadística es por medio de la Ecuación 5.1.

$$z = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}} \quad (5.1)$$

La media de los 30 experimentos realizados para la precisión media del modelo A es de 0.9875, mientras que la desviación estándar es de 0.0042. La precisión media del modelo B es de 0.9862 y la desviación estándar de 0.0021.

Para rechazar la hipótesis nula, el valor crítico debe ser superior a 1.645 porque el valor de alfa es 0.05. El resultado de la prueba estadística es 1.5164 obtenido, por lo que se falla al rechazar la hipótesis nula. Por lo tanto, No hay suficiente evidencia con un 5% de nivel de significancia para apoyar la afirmación de que la precisión de media del modelo A es mayor que la precisión media del modelo B. En la Figura 5.7 se observa la prueba estadística del caso de estudio 1.



*Figura 5.7. Prueba de hipótesis 1 para el caso de estudio 1*

### 5.1.1.2. Prueba de hipótesis 2

La prueba de hipótesis 1 tiene como afirmación de que “La precisión media del modelo A es mayor a la precisión media del modelo C”. Por lo tanto, hipótesis nula sería que la precisión media del modelo C es mayor o igual a la precisión media del modelo A.

La media de los 30 experimentos realizados para la precisión media del modelo A es de 0.9875, mientras que la desviación estándar es de 0.0042. La precisión media del modelo C es de 0.9848 y la desviación estándar de 0.0031.

Para rechazar la hipótesis nula, el valor crítico debe ser superior a 1.645 porque el valor de alfa es 0.05. El resultado de la prueba estadística es 2.833 obtenido, por lo que se falla al rechazar la hipótesis nula. Por lo tanto, hay suficiente evidencia con un 5% de nivel de significancia para apoyar la afirmación de que la precisión de media del modelo A es mayor que la precisión media del modelo C. En la Figura 5.8 se observa la prueba estadística del caso de estudio 1.

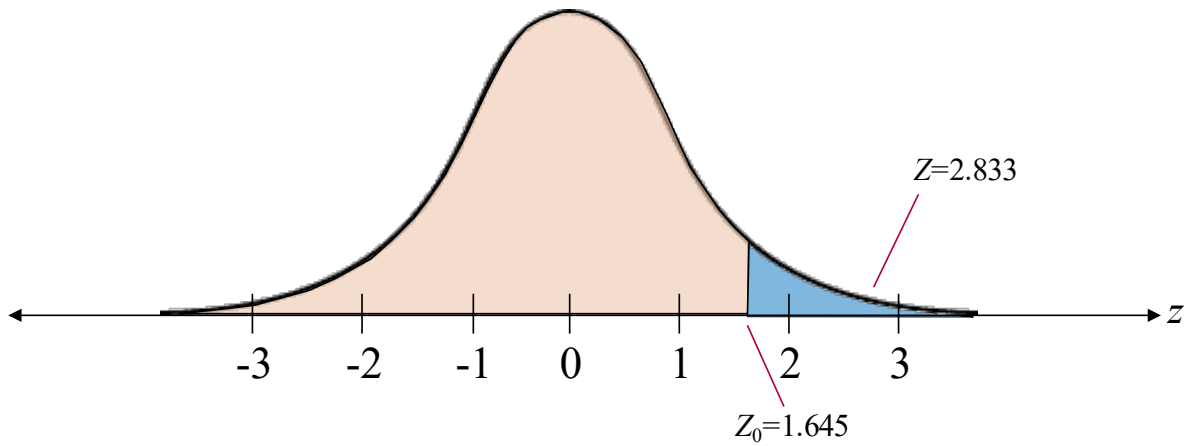


Figura 5.8. Prueba de hipótesis 2 para el caso de estudio 1

### 5.1.2. CASO DE ESTUDIO 2 (3 CLASES)

Para el caso de estudio 2 se utilizó la base de datos de MaskedFaceNet en combinación con imágenes de Flickr-Faces-HQ Dataset (FFHQ), para elaborar tres clases para el entrenamiento de la red neuronal convolucional; Mask, NoMask e IncorrectMask, observe Figura 5.9.



Figura 5.9. Ejemplo de imágenes para el caso de estudio 2

De acuerdo a los datos obtenidos del caso de estudio 1, donde se evaluaron tres modelos diferentes y se realizaron pruebas estadísticas, el modelo A de la Figura 5.2 es el modelo que mejores resultados presenta, por lo tanto, dicho modelo será el utilizado para realizar el entrenamiento de nuestro modelo propuesto, con la única diferencia que en este caso solamente se contará con tres salidas.

### 5.1.2.1.Experimentos

Se realizaron diversos experimentos con ciertas partes del conjunto de datos, para el primer experimento se tomaron en cuenta las primeras 15,000 imágenes de la base de datos, donde el porcentaje de estas imágenes utilizado para el entrenamiento fue del 70%, mientras que para las pruebas se utilizó el 20% y el 10% restante funcionó para la validación.

**Tabla 5.3.** *Promedios de experimentos del caso de estudio 2.*

	<b>Parte 1</b>	<b>Parte 2</b>	<b>Parte 3</b>
<b>Promedio</b>	0.9960	0.9964	0.9974
<b>Desviación Estándar</b>	0.0004	0.0019	0.0015

Los experimentos individuales para cada parte se observan en el Anexo 2, Anexo 3 y Anexo 4, en la Tabla 5.3 en la Parte 1 se observan los resultados obtenidos a partir de haber entrenado en 30 donde el mejor porcentaje de precisión se obtuvo en el experimento 7 con 99.69% y 2.15% de porcentaje pérdida. Un promedio general entre los 30 experimentos de 99.60% y una desviación estándar de 0.4%.

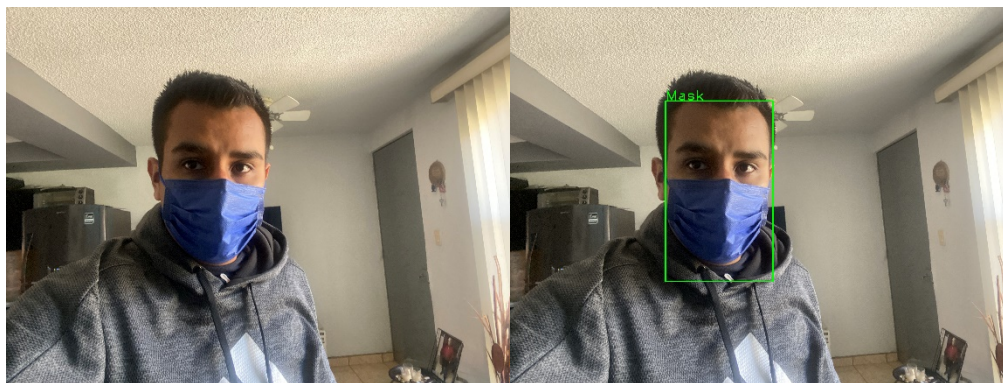
De acuerdo a los resultados obtenidos por Jones [37], con 98.5% de precisión, se obtuvo un mejor porcentaje de precisión en promedio con el modelo de la red neuronal propuesto donde se logró alcanzar como mejor caso un 99.69% utilizando la base de datos de MaskedFaceNet.

Se realizó una matriz de confusión, observe Tabla 5.4, evaluándola con el porcentaje de prueba por lo cual, de las 2991 imágenes de ese porcentaje, 2979 fueron evaluados correctamente del cual se obtuvo un 99.60% de precisión.

**Tabla 5.4.** *Matriz de confusión del caso de estudio 2*

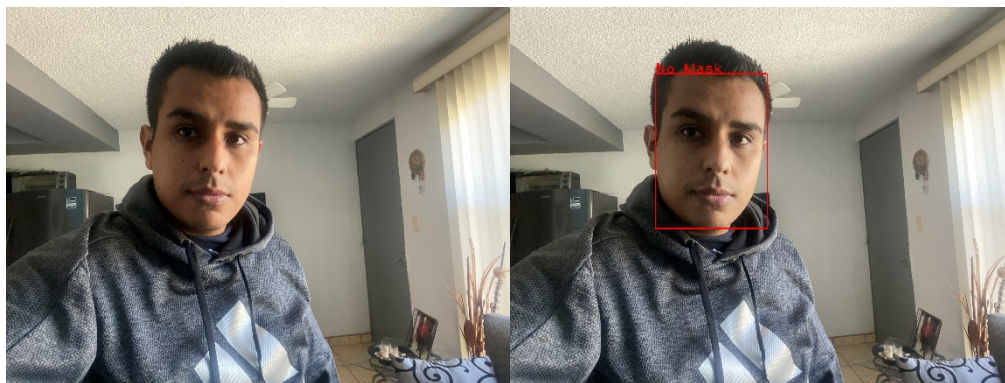
	<b>IncorrectMask</b>	<b>Mask</b>	<b>NoMask</b>
<b>IncorrectMask</b>	1033	2	2
<b>Mask</b>	6	1005	2
<b>NoMask</b>	0	0	941

De igual manera se tomaron imágenes de la vida real para comprobar los resultados del modelo entrenado con las imágenes de la base de datos de MaskedFaceNet, los resultados en la mayoría de los casos resulto positiva, en la Figura 5.10 evalúa una imagen utilizando el cubrebocas.



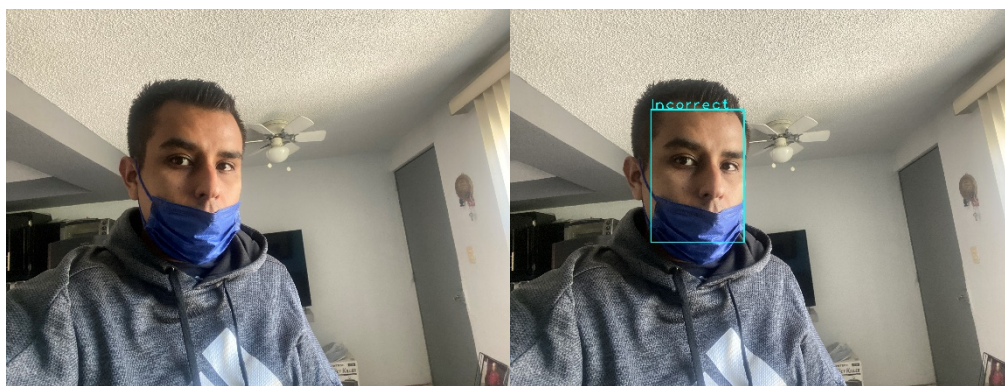
**Figura 5.10.** *Evaluando imagen real con cubrebocas.*

Para corroborar los resultados con todas las clases que el modelo propone se evaluó de igual manera con una imagen real no utilizando cubrebocas como se observa en la Figura 5.11.



*Figura 5.11. Evaluando imagen real sin cubrebocas.*

En la Figura 5.12 se evalúa una imagen utilizando el cubrebocas de manera incorrecta de tal manera que en la mayoría que se predice por medio del modelo de red neuronal convolucional que se encuentra mal colocado.



*Figura 5.12. Evaluando imagen real con cubrebocas colocado incorrectamente.*

Las imágenes de la vida real tal y como se mostraron, logran predecir satisfactoriamente los resultados esperados, en la mayoría de los casos se alcanza el objetivo principal, por lo que se comprueba que la arquitectura propuesta es eficiente.

## 5.2. SISTEMA TIEMPO REAL

Se desarrolló un sistema para el monitoreo e identificación del uso de cubrebocas dentro de un aula, donde a partir de modelo de red neuronal convolucional propuesto se realizaron validaciones para identificar el estado en el que el uso de cubrebocas se encuentra.



Una muestra de la solución que se propuso para el sistema tiempo real, podrá ser utilizado en diversas áreas donde el uso de cubrebocas es obligatorio ya sea por sanidad o reglas locales, o en lugares para monitoreo y alerta de actividades, o cualquier otra actividad en la que se requiera identificar el estado en el que se encuentra el cubrebocas.

El clasificador fue puesto a prueba e implementado en imágenes y videos en tiempo real. Las caras fueron reconocidas utilizando el mismo preprocesamiento propuesto, se extrajo la información de las caras y se detectó en color verde, rojo o amarillo la detección del uso de cubrebocas.

De acuerdo al estándar de los colores establecido se realizó un sistema de tiempo de real, en donde, a través de una Raspberry 4y una cámara Logitech C922, vea Figura 5.14 se identifica el estado en el que se encuentra el cubrebocas.

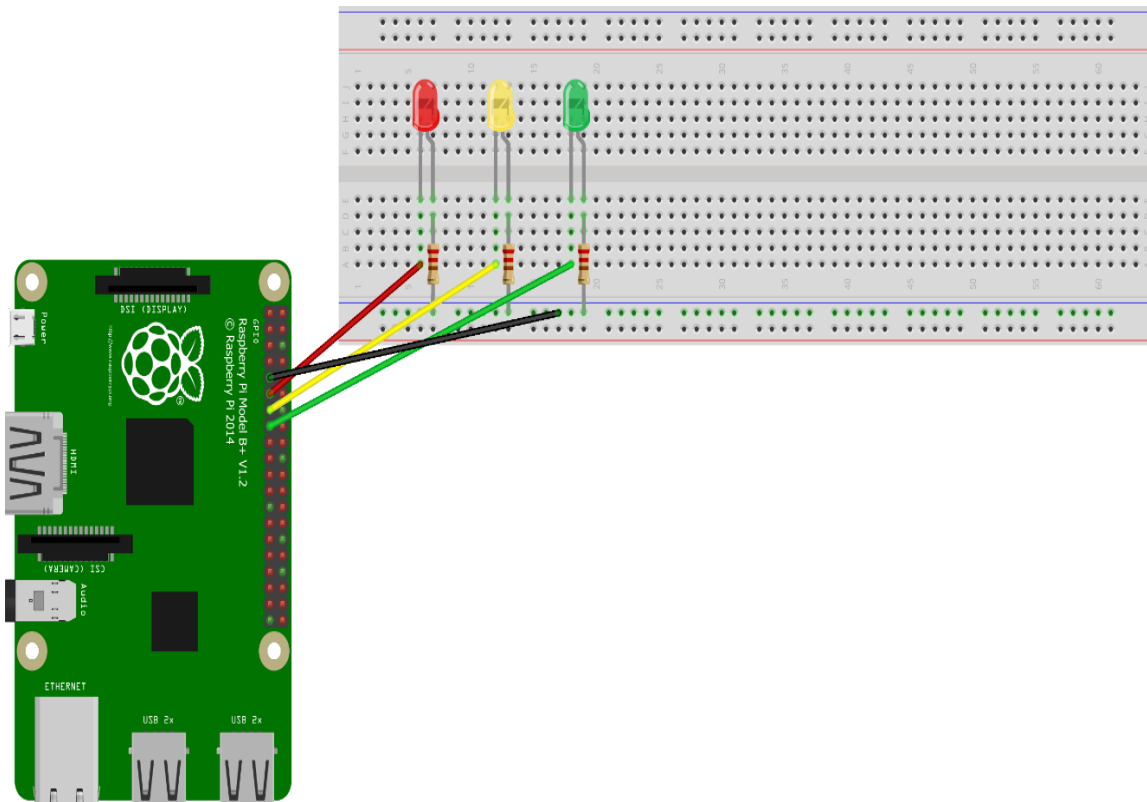
A través de la adquisición de la imagen por medio de la cámara, se realiza la predicción para clasificar el uso de cubrebocas. Evaluando cada frame que se recibe, aplicando el preprocesamiento y el modelo de red neuronal convolucional entrenado para identificar el estado en que se encuentra el cubrebocas en las personas.



**Figura 5.13.** Logitech C922

De acuerdo a la clase predicha por el modelo se encenderá un LED, conforme al color asignado, para la clase Mask se designó el color verde, mientras que para la clase IncorrectMask se asignó el color amarillo, para la clase restante NoMask se fijó el color rojo, estos colores con el fin de vincularlos al parecido del funcionamiento de un semáforo.

Una vez hecha la detección por medio del modelo, se creó un circuito para vincularla con la raspberry pi, observe Figura 5.15. Posteriormente la raspberry pi4 envía una señal para un LED de acuerdo a la clase predicha.

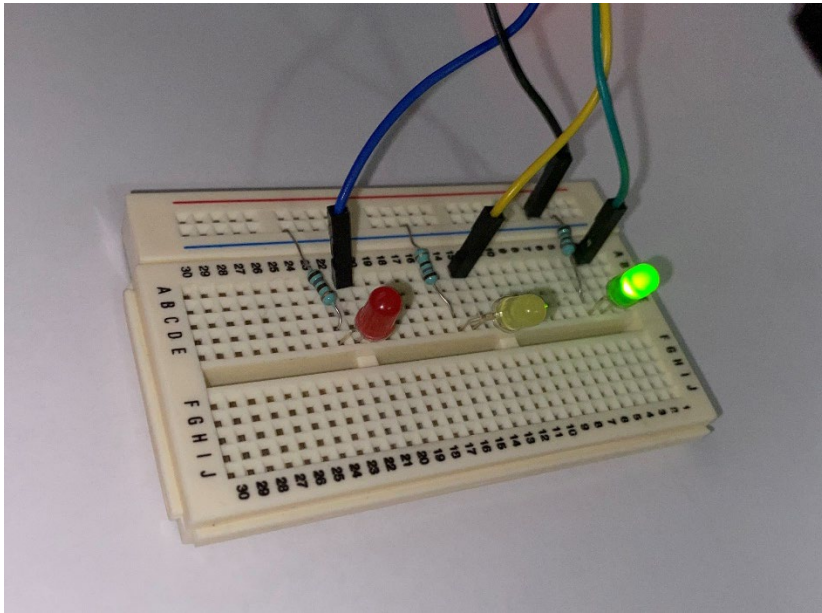


*Figura 5.14. Circuito de la Raspberry pi4*

El circuito de la raspberry pi4, tiene la función de enviar y recibir la información acerca del estado en el que se encuentra el cubrebocas, esto se logra por medio de la raspberry, utilizando el lenguaje de programación Python para la generación de código.

Una vez hecha la identificación del estado del cubrebocas en el que se encuentra se enviará una señal para activar la luz LED correspondiente a la clase señalada, como se muestra en la Figura. 5.16.

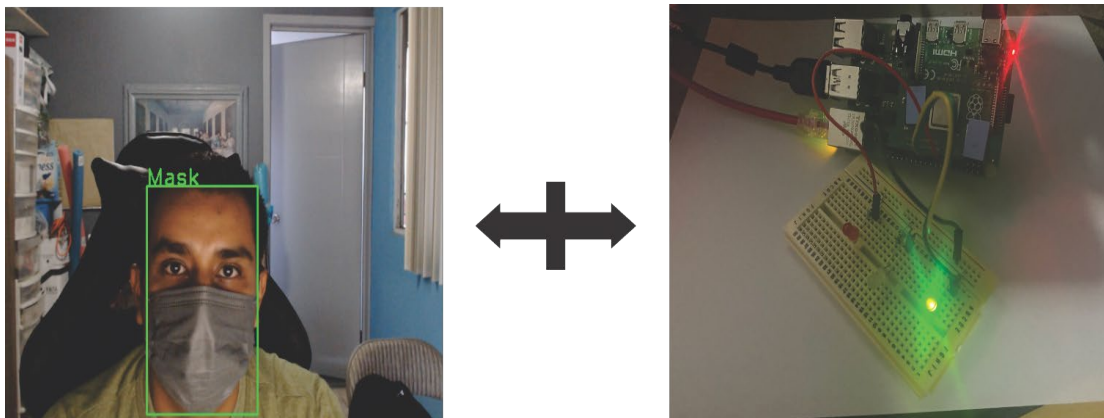




*Figura 5.15. Prueba de resultados para luz LED*

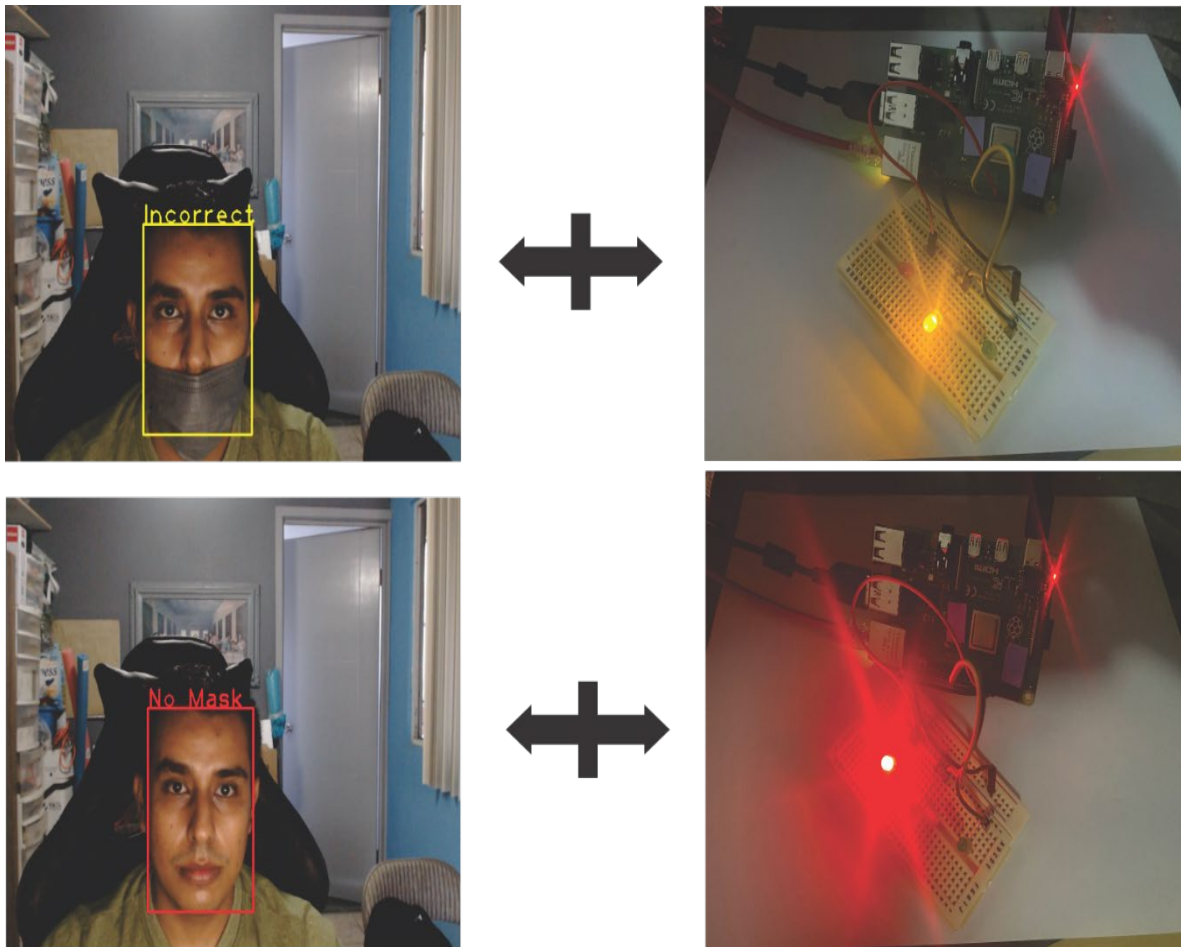
De acuerdo a los experimentos realizados para corroborar la efectividad del sistema se puede observar resultados favorables y prometedores, logrando identificar y enfatizar el estado en el que se encuentra el cubrebocas, tanto en video en vivo como utilizando señales para mostrar el LED correcto.

Se realizaron múltiples pruebas utilizando una cámara de video para obtener las imágenes de entrada del modelo para poder evaluarlo en tiempo real, en la Figura 5.17 se muestra el resultado de la clasificación de una imagen cuando un humano lleva puesta la máscara.



*Figura 5.16. Ejemplo del sistema tiempo real con la clase Mask*

Además de clasificar la clase a la que pertenece de igual forma enciende un led según la clase, en esta clase color verde, porque el cubrebocas está colocado correctamente. Mediante el etiquetado es más fácil detectar a los humanos que no están utilizando correctamente el cubrebocas, consiguiendo detectar si pertenecen a alguna de las tres clases, como Mask, NoMask o IncorrectMask. En la mayoría de los casos, los resultados son bastante satisfactorios logrando clasificarlos correctamente.



*Figura 5.17. Ejemplo del sistema tiempo real con IncorrectMask y NoMask*

La Figura 5.18 muestra la clasificación de las dos clases restantes, así como también, el LED muestra el color correspondiente a la clase, el color amarillo para la clase IncorrectMask y el color rojo para la clase NoMask.

La raspberry pi4 envía las señales para encender el LED correspondiente gracias a la placa GPIO y la programación realizada en Python para activar el pin adecuado y cargar el modelo para detección de rostros y el modelo entrenado para la clasificación multiclase del correcto uso de cubrebocas.

# CAPÍTULO VI

## CONCLUSIONES

En el presente capítulo quedan plasmadas las conclusiones obtenidas, después de haber realizado el proyecto de tesis, de igual manera se comentan los posibles trabajos futuros a implementar.

Nuestro trabajo es capaz de clasificar el uso de cubrebocas multiclase por medio de un modelo de red neuronal convolucional, en combinación de visión por computadora, con el objetivo de evitar transmitir el virus de COVID-19 o cualquier otro virus que se pueda transmitirse por vías aéreas, esto se logra por medio de un monitoreo tiempo real en zonas estratégicas, por medio de una raspberry pi 4, es posible realizar esta identificación para realizar acciones específicas.

El modelo logra clasificar el uso de cubrebocas en tres clases Mask, NoMask y IncorrectMask y a través del GPIO Board de la raspberry pi enviar señales para encender un LED verde, amarillo o rojo respectivamente obteniendo un porcentaje de precisión de 99.69%, evaluando con el conjunto de datos de MaskedFaceNet.

El sistema propuesto, por lo tanto, permitirá que el esparcimiento del virus disminuya, ayudando al sistema de salud de las personas. Esta solución podría evitar que se incumplan las restricciones en tiempo real, mejorando la seguridad de las personas a nuestro alrededor, pudiendo ser utilizado en diversas áreas como escuelas, plazas, lugares públicos o privados, entre otros.

Como trabajo futuro se plantean realizar más pruebas con otros modelos identificando los que mejores resultados y los posibles cambios que se puedan proporcionar. De igual manera, se utilizarían diferentes bases de datos para probar los modelos y entrenarlos con otros parámetros y diferentes técnicas de visión por computadora.

## CONCLUSIONES

Otro trabajo futuro consiste en implementar el sistema tiempo real en la industria mejorando la base de datos y alimentándola con imágenes reales para ser utilizada en la alarma de alguna infracción cometida según los estándares del lugar colocado.

**REFERENCIAS**

- [1] World Health Organization, «WHO Coronavirus (COVID-19) Dashboard,» World Health Organization, 25 02 2022. [En línea]. Available: <https://covid19.who.int/>. [Último acceso: 25 02 2022].
- [2] World Health Organization, «Consejos para la población sobre el nuevo coronavirus (2019-nCoV): cuándo y cómo usar mascarilla,» World Health Organization, 1 12 2020. [En línea]. Available: <https://www.who.int/es/emergencies/diseases/novel-coronavirus-2019/advice-for-public/when-and-how-to-use-masks>. [Último acceso: 03 11 2022].
- [3] A. Cabani , K. Hammoudi, H. Benhabiles y M. Melkemi, «MaskedFace-Net – A dataset of correctly/incorrectly masked face images in the context of COVID-19,» *Smart Health*, vol. 19, pp. 1-6, 2020.
- [4] A. Campos, P. Melin y D. Sánchez , «Multiclass Mask Classification with a New Convolutional Neural Model and Its Real-Time Implementation,» *Life*, vol. 13, nº 2, pp. 1-15, 2023.
- [5] V. Alvear-Puertas, P. Rosero-Montalvo, D. Peluffo-Ordóñez y J. Pijal-Rojas, «Internet de las Cosas y Visión Artificial, Funcionamiento y Aplicaciones: Revisión de Literatura,» *Enfoque UTE*, vol. 8, nº 1, pp. 244-256, 2017.
- [6] L. Rouhiainen, *Inteligencia artificial: 101 cosas que debes saber hoy sobre nuestro futuro*, Barcelona: Alienta Editorial, 2018.
- [7] B. Mahesh, «Machine Learning Algorithms - A Review,» *International Journal of Science and Research (IJSR)*, vol. 9, nº 1, pp. 381-386, 2020.

- [8] V. Verdhan, *Computer Vision Using Deep Learning: Neural Network Architectures with Python and Keras*, New York: Apress, 2021.
- [9] N. Subramanian, O. Elharrouss, S. Al-Maadeed y M. Chowdhury, «A review of deep learning-based detection methods for COVID-19,» *Computers in Biology and Medicine*, vol. 143, nº 105233, 2022.
- [10] F. Berzal, *Redes neuronales & Deep learning*, Granada: Universidad de Granada, 2018.
- [11] H. H. Aghdam y E. J. Heravi, *Guide to Convolutional Neural Networks*, Spain: Springer, 2017.
- [12] J. Nagi, F. Ducatelle, G. Di Caro, D. Ciresan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber y L. M. Gambardella, «Max-Pooling Convolutional Neural Networks for Vision-based Hand Gesture Recognition,» de *2011 IEEE International Conference on Signal and Image Processing Applications*, Switzerland, 2011.
- [13] A. Géron, *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Unsupervised learning techniques*, Sebastopol: O'Reily Media, 2019.
- [14] C. Aggarwal, *Neural Networks and Deep Learning: A Textbook*, Basingstoke: Springer, 2018.
- [15] ImageNet, [En línea]. Available: <https://image-net.org/>. [Último acceso: 2022 04 25].
- [16] I. Zafar, G. Tzanidou, R. Burton, N. Patel y L. Araujo, *Hands-On Convolutional Neural Networks with TensorFlow: Solve computer vision problems with modeling in TensorFlow and Python*, Birmingham: Packt Publishing, 2018.
- [17] S. Targ, D. Almeida y K. Lyman, «Resnet in Resnet: Generalizing Residual Architectures,» de *ICLRW 2016*, San Juan, 2016.

- [18] J. Redmon, S. Divvala, R. Girshick y A. Farhadi, «You Only Look Once: Unified, Real-Time Object Detection,» de *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, 2016.
- [19] B. Khasoggi, E. Ermatita y S. Samsuryadi, «Efficient mobilenet architecture as image recognition on mobile and embedded devices,» *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 16, nº 1, pp. 389-394, 2019.
- [20] Z. Wang, G. Wang, B. Huang, Z. Xiong, Q. Hong, H. Wu, P. Yi, K. Jiang, N. Wang, Y. Pei, H. Chen, Y. Miao, Y. Huang y J. Liang, «Masked Face Recognition Dataset and Application,» de *National Engineering Research Center for Multimedia Software*, Wuhan, 2020.
- [21] S. Singh, U. Ahuja, M. Kumar, K. Kumar y M. Sachdeva, «Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment,» *Multimedia Tools and Applications*, vol. 80, nº 13, pp. 19753-19768, 2021.
- [22] A. Syed, E. Shirin, K. Muhammed y Q. Syed, «Real-Time Face Mask Detection in Deep Learning using Convolution Neural Network,» de *10th IEEE International Conference on Communication Systems and Network Technologies (CSNT)*, 2021.
- [23] S. Sethi, M. Kathuria y T. Mamta, «A Real-Time Integrated Face Mask Detector to Curtail Spread of Coronavirus,» *Computer Modeling in Engineering & Sciences*, vol. 127, nº 2, pp. 389-409, 2021.
- [24] M. Deshmukh, G. Deshmukh, P. Pawar y P. Deore, «Covid-19 Mask Protocol Violation Detection Using Deep Learning and Computer Vision,» *International Research Journal of Engineering and Technology (IRJET)*, vol. 8, nº 6, pp. 3292-3295, 2021.



- [25] B. Bhattarai, Y. Raj Pandeya y J. Lee, «Deep Learning-based Face Mask Detection Using Automated GUI for COVID-19,» de *6th International Conference on Machine Learning Technologies*, 2021.
- [26] Larxel, «Face Mask Detection,» Kaggle, 22 05 2022. [En línea]. Available: <https://www.kaggle.com/datasets/andrewmvd/face-mask-detection>. [Último acceso: 22 03 2022].
- [27] A. Jangra, «Face Mask Detection 12K Images Dataset,» Kaggle, 25 05 2020. [En línea]. Available: <https://www.kaggle.com/datasets/ashishjangra27/face-mask-12k-images-dataset/metadata>. [Último acceso: 22 03 2022].
- [28] A. Pham-Hoang-Nam, V. Le-Thi-Tuong, L. Phung-Khanh y N. Ly-Tu, «Densely Populated Regions Face Masks Localization and Classification Using Deep Learning Models,» de *Proceedings of the Sixth International Conference on Research in Intelligent and Computing*, 2022.
- [29] V. Kumar, «Face Mask Detection,» Kaggle, 19 05 2021. [En línea]. Available: <https://www.kaggle.com/vijaykumar1799/face-mask-detection>. [Último acceso: 11 03 2022].
- [30] V. Kumar, «Face Mask Detection: Building a face mask classifier,» Kaggle, 19 05 2021. [En línea]. Available: <https://www.kaggle.com/datasets/vijaykumar1799/face-mask-detection>. [Último acceso: 28 04 2022].
- [31] J. Yu y W. Zhang, «Face Mask Wearing Detection Algorithm Based on Improved YOLO-v4,» *Sensors*, vol. 21, n° 9, p. 3263, 2021.
- [32] E. Aydemir, M. Yalcinkaya, P. Barua, M. Baygin, O. Faust, S. Dogan, S. Chakraborty, T. Tuncer y R. Acharya, «Hybrid Deep Feature Generation for Appropriate Face Mask Use Detection,» *International Journal of Environmental Research and Public Health*, vol. 19, n° 4, p. 1939, 2022.

- [33] C. Soto-Paredes y J. Sulla-Torres, «Hybrid Model of Quantum Transfer Learning to Classify Face Images with a COVID-19 Mask,» *International Journal of Advanced Computer Science and Applications*, vol. 12, n° 10, pp. 826-836, 2021.
- [34] «Real-World Masked Face Dataset,» National Engineering Research Center for Multimedia Software, 12 01 2021. [En línea]. Available: <https://github.com/X-zhangyang/Real-World-Masked-Face-Dataset>. [Último acceso: 11 03 2022].
- [35] B. Wang, Y. Zhao y P. Chen, «Hybrid Transfer Learning and Broad Learning System for Wearing Mask Detection in the COVID-19 Era,» *IEEE Transactions on Instrumentation and Measurement*, vol. 70, 2021.
- [36] S. Rudraraju, N. Suryadevara y A. Negi, «Face Mask Detection at the Fog Computing Gateway,» de *Proceedings of the 2020 Federated Conference on Computer Science and Information Systems*, 2020.
- [37] D. Jones y C. Christoforou, «Mask Recognition with Computer Vision in the Age of a Pandemic,» *The International FLAIRS Conference Proceedings*, vol. 34, n° 1, pp. 1-6, 2021.
- [38] X. Jiang, T. Gao, Z. Zhu y Y. Zhao, «Real-Time Face Mask Detection Method Based on YOLOv3,» *Electronics*, vol. 10, n° 7, p. 837, 2021.
- [39] T. Karras, S. Laine y T. Aila, «A Style-Based Generator Architecture for Generative Adversarial Networks,» *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, n° 12, pp. 4217-4228, 2021.
- [40] A. Pham-Hoang-Nam, V. Le-Thi-Tuong, L. Phung-Khanh y N. Ly-Tu, «Densely Populated Regions Face Masks Localization and Classification Using Deep Learning Models,» de *Proceedings of the Sixth International Conference on Research in Intelligent and Computing*, 2022.

- [41] A. Das, M. Wasif Ansari y R. Basak, «Covid-19 Face Mask Detection Using TensorFlow, Keras and OpenCV,» de *2020 IEEE 17th India Council International Conference (INDICON)*, 2020.
- [42] G. Kaur, R. Sinha, P. Tiwari, S. Yadav, P. Pandey, R. Raj, A. Vashisth y M. Rakhra, «Face mask recognition system using CNN model,» *Neuroscience Informatics*, vol. 2, n° 3, p. 100035, 2022.
- [43] S. Yadav, «Deep Learning based Safe Social Distancing and Face Mask Detection in Public Areas for COVID-19 Safety Guidelines Adherence,» *International Journal for Research in Applied Science and Engineering Technology*, vol. 8, n° 7, pp. 1368-1375, 2020.
- [44] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov y L.-C. Chen, «MobileNetV2: Inverted Residuals and Linear Bottlenecks,» *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.
- [45] R. Araki, T. Onishi, T. Hirakawa, T. Yamashita y H. Fujiyoshi, «MT-DSSD: Deconvolutional Single Shot Detector Using Multi Task Learning for Object Detection, Segmentation, and Grasping Detection,» de *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020.
- [46] Vibhuti, N. Jindal, H. Singh y P. Rana, «Face mask detection in COVID-19: a strategic review,» *Multimedia Tools and Applications*, vol. 81, n° 28, pp. 40013-40042, 2022.
- [47] A. Cabani, «MaskedFace-Net,» Github, 28 05 2021. [En línea]. Available: <https://github.com/cabani/MaskedFace-Net>. [Último acceso: 11 03 2022].
- [48] «NVLabs/ffhq-dataset: Flickr-Faces-HQ Dataset (FFHQ),» Github, 08 03 2022. [En línea]. Available: <https://github.com/NVLabs/ffhq-dataset>. [Último acceso: 27 04 2022].

- [49] S. Ge, J. Li, Q. Ye y Z. Luo, «Detecting Masked Faces in the Wild with LLE-CNNs,» de *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [50] S. Yang, P. Luo, C. Loy y X. Tang, «WIDER FACE: A Face Detection Benchmark,» de *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [51] M. Mahmud, S. Kaiser, M. McGinnity y A. Hussain, «Deep Learning in Mining Biological Data,» *Cognitive Computation*, n° 13, pp. 1-33, 2021.
- [52] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama y T. Darrell, «Caffe: Convolutional Architecture for Fast Feature Embedding,» de *MM 2014 - Proceedings of the 2014 ACM Conference on Multimedia*, 2014.
- [53] H. Goyal, K. Sidana, C. Singh y A. Jain, «A real time face mask detection system using convolutional neural network,» de *Multimedia Tools and Applications*, 2022.
- [54] V. Agarwal, «Face Detection Models: Which to Use and Why?,» Medium, 02 07 2020. [En línea]. Available: <https://towardsdatascience.com/face-detection-models-which-to-use-and-why-d263e82c302c>. [Último acceso: 22 04 2022].
- [55] A. Rosebrock, «Deep learning: How OpenCV's blobFromImage works,» PyImageSearch, 06 11 2017. [En línea]. Available: <https://pyimagesearch.com/2017/11/06/deep-learning-opencvs-blobfromimage-works/>. [Último acceso: 27 04 2022].
- [56] S. Raschka y V. Mirjalili, *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn and TensorFlow*, Birmingham: Packt, 2017.
- [57] C. Wai Zhao, J. Jegatheesan y S. Chee Loon, «Exploring IOT Application Using Raspberry Pi,» *International Journal of Computer Networks and Applications*, vol. 2, n° 1, pp. 27-34, 2015.

- [58] S. Militante y N. Dionisio, «Real-Time Facemask Recognition with Alarm System using Deep Learning,» de *2020 11th IEEE Control and System Graduate Research Colloquium (ICSGRC)*, Shah Alam, Malaysia, 2020.

## ANEXOS

### Anexo 1. Experimentos para los modelos del caso de estudio 1

Modelo	A		B		C	
	Precisión	Pérdida	Precisión	Pérdida	Precisión	Pérdida
1	0.9953	0.0363	0.9878	0.0829	0.9888	0.0751
2	0.9890	0.0811	0.9878	0.0995	0.9878	0.1003
3	0.9859	0.1064	0.9878	0.1113	0.9867	0.1276
4	0.9859	0.1318	0.9878	0.1226	0.9857	0.1529
5	0.9906	0.0396	0.9878	0.1342	0.9847	0.1675
6	0.9937	0.0561	0.9878	0.1452	0.9816	0.1183
7	0.9890	0.0895	0.9857	0.0955	0.9806	0.1574
8	0.9875	0.1172	0.9867	0.1159	0.9806	0.1764
9	0.9890	0.1661	0.9867	0.1302	0.9786	0.2343
10	0.9890	0.1826	0.9827	0.1218	0.9745	0.0865
11	0.9718	0.1200	0.9878	0.1079	0.9847	0.1264
12	0.9906	0.1217	0.9867	0.1199	0.9857	0.1639
13	0.9906	0.1443	0.9847	0.1037	0.9857	0.2000
14	0.9906	0.1661	0.9827	0.1311	0.9847	0.2241
15	0.9906	0.0908	0.9827	0.1477	0.9847	0.2374
16	0.9906	0.0610	0.9878	0.1113	0.9847	0.2469
17	0.9906	0.0740	0.9878	0.1136	0.9847	0.2541
18	0.9906	0.0565	0.9857	0.1342	0.9847	0.2596
19	0.9875	0.0967	0.9878	0.1452	0.9847	0.2644
20	0.9859	0.1180	0.9878	0.1036	0.9847	0.2688
21	0.9859	0.1395	0.9878	0.1237	0.9888	0.1102
22	0.9859	0.1450	0.9878	0.1302	0.9888	0.1456
23	0.9859	0.1489	0.9827	0.1030	0.9878	0.1817
24	0.9859	0.1619	0.9827	0.1203	0.9867	0.2160

## ANEXOS

25	0.9843	0.1713	0.9827	0.1302	0.9837	0.0950
26	0.9843	0.1827	0.9827	0.1218	0.9857	0.1284
27	0.9843	0.1908	0.9878	0.1070	0.9857	0.1382
28	0.9843	0.1939	0.9878	0.1124	0.9857	0.1698
29	0.9843	0.1964	0.9878	0.1027	0.9847	0.1946
30	0.9843	0.1987	0.9878	0.1164	0.9867	0.2136
	Promedio	0.9875	Promedio	0.9862	Promedio	0.9848
	Desviación Estándar	0.0042	Desviación Estándar	0.0021	Desviación Estándar	0.0031

## ANEXOS

## Anexo 2. Experimentos para validar la Parte 1 de MaskedFaceNet

<b>Modelo</b>	<b>Validación 1</b>				
<b>Entrenamiento</b>	<b>Precisión</b>	<b>Pérdida</b>	<b>Entrenamiento</b>	<b>Precisión</b>	<b>Pérdida</b>
1	0.9958	0.0256	16	0.9958	0.0541
2	0.9958	0.0283	17	0.9958	0.0599
3	0.9958	0.0378	18	0.9958	0.0250
4	0.9958	0.0534	19	0.9958	0.0388
5	0.9958	0.0669	20	0.9958	0.0494
6	0.9958	0.0752	21	0.9958	0.0578
7	0.9969	0.0215	22	0.9958	0.0319
8	0.9969	0.0335	23	0.9958	0.0510
9	0.9969	0.0436	24	0.9958	0.0621
10	0.9958	0.0325	25	0.9958	0.0675
11	0.9958	0.0454	26	0.9969	0.0285
12	0.9958	0.0571	27	0.9958	0.0467
13	0.9958	0.0638	28	0.9958	0.0613
14	0.9958	0.0338	29	0.9958	0.0705
15	0.9958	0.0443	30	0.9958	0.0268
			<b>Promedio</b>	0.9960	
			<b>Desviación Estándar</b>	0.0004	



## ANEXOS

## Anexo 3. Experimentos para validar la Parte 2 de MaskedFaceNet

<b>Modelo</b>	<b>Validación 2</b>				
<b>Entrenamiento</b>	<b>Precisión</b>	<b>Pérdida</b>	<b>Entrenamiento</b>	<b>Precisión</b>	<b>Pérdida</b>
1	0.9979	0.0336	16	0.9948	0.0498
2	0.9969	0.0477	17	0.9979	0.0532
3	0.9969	0.0719	18	0.9979	0.0727
4	0.9969	0.0781	19	0.9969	0.0794
5	0.9958	0.0690	20	0.9969	0.0787
6	0.9958	0.0692	21	0.9969	0.1186
7	0.9969	0.0573	22	0.9969	0.0387
8	0.9969	0.1373	23	0.9927	0.0465
9	0.9885	0.0638	24	0.9979	0.0341
10	0.9979	0.0530	25	0.9958	0.0436
11	0.9979	0.0619	26	0.9969	0.0409
12	0.9979	0.0777	27	0.9969	0.0485
13	0.9969	0.1439	28	0.9969	0.0820
14	0.9948	0.0713	29	0.9969	0.0357
15	0.9979	0.0644	30	0.9958	0.0344
			<b>Promedio</b>	0.9964	
			<b>Desviación Estándar</b>	0.0019	

## ANEXOS

## Anexo 4. Experimentos para validar la Parte 3 de MaskedFaceNet

<b>Modelo</b>	<b>Validación 3</b>				
<b>Entrenamiento</b>	<b>Precisión</b>	<b>Pérdida</b>	<b>Entrenamiento</b>	<b>Precisión</b>	<b>Pérdida</b>
1	0.9979	0.0152	16	0.9948	0.0196
2	0.9969	0.0133	17	0.9937	0.0433
3	0.9958	0.0254	18	0.9958	0.0368
4	0.9990	0.0017	19	0.9979	0.0192
5	0.9990	0.0026	20	0.9969	0.0201
6	0.9990	0.0027	21	0.9969	0.0144
7	0.9948	0.0121	22	0.9979	0.0255
8	0.9979	0.0239	23	0.9979	0.0305
9	0.9979	0.0172	24	0.9990	0.9990
10	0.9958	0.0416	25	0.9979	0.0097
11	0.9958	0.0482	26	0.9990	0.0075
12	0.9979	0.0060	27	0.9990	0.0244
13	0.9990	0.0023	28	0.9979	0.0058
14	0.9990	0.0034	29	0.9969	0.0179
15	0.9958	0.0167	30	0.9979	0.0119
			<b>Promedio</b>	0.9974	
			<b>Desviación Estándar</b>	0.0015	

## ANEXOS

### Anexo 5. Código para ejecución del sistema tiempo real

```
# Librerías
import numpy as np
import cv2
import tensorflow as tf
from scipy.special import softmax
import time
import RPi.GPIO as GPIO

face_detection_model=
cv2.dnn.readNetFromCaffe('./caffemodel/deploy.prototxt.txt', './caffemodel/res10_300x300
_ssd_iter_140000_fp16.caffemodel') # Cargar modelo para la detección de rostro

model = tf.keras.models.load_model('./FaceMaskModel.h5') # Cargar modelo propuesto para
la clasificación del uso de cubrebocas

GPIO.setwarnings(False) # Evitar advertencias del GPIO
GPIO.setmode(GPIO.BOARD) # Modo board en RaspberryPi

# Asignación de número de pin
pinGREEN = 11
pinYL = 13
pinRED = 15

# Configurar como pin de salida
GPIO.setup(pinGREEN, GPIO.OUT)
GPIO.setup(pinYL, GPIO.OUT)
GPIO.setup(pinRED, GPIO.OUT)

all_pins = [pinGREEN, pinYL, pinRED] # Agregar pines a un arreglo
labels_info = ['Incorrect', 'Mask', 'No Mask'] # Asignar etiquetas para el uso de
cubrebocas

#Método para encender LED dependiendo del pin
def turnOnPin(pinHigh):
    pins=all_pins.copy()
    GPIO.output(pinHigh, GPIO.HIGH) # Encender pin
    low_pins= pins.remove(pinHigh) # Remover pin de la lista
    for pin in pins:
        GPIO.output(pin, GPIO.LOW) # Apagar pin

# Método para asignar color y encender LED de acuerdo a la clase predicha
def getColor(label_info):
    if label_info == "Mask":
        color=(0,255,0)
        turnOnPin(pinGREEN)

    elif label_info == 'No Mask':
        color=(0,0,255)
        turnOnPin(pinRED)

    else:
        color=(0,255,255)
```

## ANEXOS

```
turnOnPin(pinYL)

return color

# Método para la predicción del uso de cubrebocas
def face_mask_prediction(img):
    # Paso - 1 : Detección del rostro
    image = img.copy()
    h, w = image.shape[:2]
    blob = cv2.dnn.blobFromImage(image,1,(300,300),(104,117,123),swapRB=True)
    face_detection_model.setInput(blob)
    detection = face_detection_model.forward()
    for i in range(0,detection.shape[2]):
        confidence = detection[0,0,i,2]
        if confidence > 0.7:
            box = detection[0,0,i,3:7]*np.array([w,h,w,h])
            box = box.astype(int)
            pt1 = (box[0], box[1])
            pt2 = (box[2], box[3])
            # Paso - 2: Preprocesamiento de imagen
            face = image[box[1]:box[3],box[0]:box[2]]
            face_blob =
cv2.dnn.blobFromImage(face,1,(100,100),(104,117,123),swapRB=True)
            face_blob_squeeze = np.squeeze(face_blob).T
            face_blob_rotate = cv2.rotate(face_blob_squeeze,cv2.ROTATE_90_CLOCKWISE)
            face_blob_flip = cv2.flip(face_blob_rotate,1)
            img_norm = np.maximum(face_blob_flip,0)/face_blob_flip.max() #
Normalización
            # Paso - 3: Convolutional Neural Network
            img_input = img_norm.reshape(1,100,100,3)
            result = model.predict(img_input)
            # Aplicar softmax al resultado
            result = softmax(result)[0]
            confidence_index = result.argmax()
            confidence_score = result[confidence_index]
            label = labels_info[confidence_index]
            label_text = '{}'.format(label)
            # Color
            color = getColor(label)
            cv2.rectangle(image,pt1,pt2,color,2)
            cv2.putText(image,label_text,pt1,cv2.FONT_HERSHEY_PLAIN,2,color,2)

    return image

cap = cv2.VideoCapture(0, cv2.CAP_DSHOW) # Encender cámara

# Ejecutar la cámara hasta que se cumpla la condición
while True:
    ret, frame = cap.read() # Leer frame capturado
    if ret == False: # Verificar que se encuentre cámara
        break

    image = face_mask_prediction(frames) # Aplicar modelo para la clasificación del
    uso de cubrebocas al frame
    cv2.imshow('Face Mask Prediction',image) # Mostrar la predicción
```

## ANEXOS

```
if cv2.waitKey(1) == 27:  
    break  
  
# Liberar memoria de cámara y raspberry  
cap.release()  
GPIO.cleanup()  
cv2.destroyAllWindows()
```