Universidad Internacional de la Rioja (UNIR)

ESIT

Máster Universitario en Inteligencia Artificial

Detección de uso correcto de tapabocas empleando aprendizaje automático

Trabajo Fin de Máster

presentado por: Walter Iván Leturia Rodríguez

Dirigido por: Carlos Agüero Iglesia

Ciudad: Trujillo - Perú

Fecha: 22 de septiembre de 2021

Índice de contenidos

Índice de ilustraciones

Índice de tablas

Resumen

La pandemia causada por el virus COVID-19 ha causado estragos alrededor del mundo y la primera barrera de defensa es el uso correcto de tapabocas. El presente trabajo presenta una herramienta de software basada en aprendizaje automático para la detección de uso correcto, incorrecto y no uso de tapabocas. Se utilizaron algoritmos de detección rápida para la identificación de: rostro, nariz y boca. Asimismo, se entrenó e implementó una red neuronal de convolución para determinar la presencia o ausencia de tapabocas. Con las características extraídas, se realizan evaluaciones lógicas las cuales logran determinar los 3 casos de detección propuestos con una precisión de 97%.

Palabras Clave: aprendizaje automático, detección de objetos, procesamiento de imágenes

Abstract

COVID-19 pandemic has caused havoc around the world and the first defense barrier is the correct use of face masks. This work introduces a software tool based on machine learning for the detection of correct, incorrect and non-use of face masks. Rapid detection algorithms were used to identify face, nose and mouth. In addition, a convolutional neural network was trained and implemented to determine the presence or absence of face masks. With the extracted features, logical evaluations are carried out which can determine the 3 proposed detection cases, achieving a 97% precision score.

Keywords: machine learning, object detection, image processing

Chapter 1

Introducción

1.1 Motivación

Es de conocimiento público la pandemia que se originó en el año 2020 y continúa causando estragos en este año 2021; con la propagación del virus COVID-19 el ritmo y rutina de las personas no es el mismo.

Según la OMS [World Health Organization, 2020a], el coronavirus genera un grupo de enfermedades que van desde **resfriado común** hasta las más graves como: **neumonía, síndrome respiratorio de oriente medio** (MERS) y **síndrome respiratorio agudo grave** (SARS).

¿Qué es COVID-19? COVID-19 es la **enfermedad infecciosa causada por el coronavirus**, que se ha descubierto más recientemente. Tanto este nuevo virus como la enfermedad que provoca eran desconocidos antes de que estallara el brote en Wuhan (China) en diciembre 2019 [World Health Organization, 2020b]

Desde el día 11 de marzo de 2020, este virus (COVID-19) ha sido calificado como pandemia - *"Enfermedad epidémica que se extiende a muchos países o que ataca a casi todos los individuos de una localidad o región"* [Real Academia Española, 2020]. Sin embargo, esto no quiere decir que en todas las ciudades nos encontramos en la misma etapa de avance de la enfermedad. En el continente europeo están pasando por un quinto rebrote de esta enfermedad, mientras que en la mayoría de los países latinoamericanos aún se encuentran atravesando los estragos del primer y segundo brote.

Los países que fueron ejemplo a seguir en este periodo de lucha contra el nuevo virus COVID-19 fueron Australia, Japón e Israel [BBC News Mundo, 2020]. Resaltamos la gestión de estos países la cual se basó en **confinamiento, distanciamiento social y**

correcto uso de tapabocas.

En América Latina; Brasil, con un total de 591 000 muertes, es el país que tiene el mayor número de muertes en el mundo, después de Estados Unidos; Perú tiene más de 199 000 muertes confirmadas y Colombia más de 126 000. Además, las tasas de pruebas en la etapa inicial estuvieron por debajo de las realizadas en otras partes del mundo [Roser et al., 2021].

La implementación y cumplimiento de protocolos de bioseguridad es esencial para la contención de este virus. Es de conocimiento que el mal uso de tapabocas/mascarillas está directamente relacionado al contagio, asimismo el incumplimiento de los protocolos de distanciamiento establecidos por la OMS. Sin embargo, en Perú se hace caso omiso a los protocolos de distanciamiento y es frecuente observar el uso incorrecto de tapabocas o inclusive el inexistente uso de éstos.

1.2 Planteamiento del trabajo

El presente trabajo busca desarrollar, a través de reconocimiento y análisis de imágenes, un método de detección del uso correcto de tapabocas. Con el propósito de, proporcionar una herramienta de prevención ante la propagación del virus COVID-19; que en nuestro país fue motivo para que se declare el estado de emergencia sanitaria y estado de emergencia nacional [1], que entre otras regulaciones dispone la promoción y vigilancia de uso de mascarillas.

Para lograr el objetivo mencionado, se aplicarán técnicas de aprendizaje automático y recursos (conjuntos de datos) recopilados de fuentes varias. Es decir, se utilizarán conjuntos de imágenes obtenidas de internet para el entrenamiento de una red neuronal

[1] Marco normativo que establece disposiciones a causa de la COVID-19:

- Mediante Decreto Supremo N° 008-2020-SA, publicado el 19 de febrero de 2021, el Ministerio de Salud declaró la Emergencia Sanitaria a nivel nacional por la existencia del COVID-19; prorrogado por Decreto Supremo N° 025-2021-SA a partir del 03 de setiembre de 2021 por 180 días calendario.
- Mediante Decreto Supremo N° 044-2020-PCM, publicado el 15 de marzo de 2020, se declaró el Estado de Emergencia Nacional por las graves circunstancias que afectan la vida de las personas a consecuencia de la COVID-19; luego con Decreto Supremo N° 184-2020-PCM, publicado el 30 de noviembre de 2020, se declaró nuevamente el Estado de Emergencia Nacional que continua vigente, según prórroga dispuesta por Decreto Supremo N° 149-2021-PCM, publicado el 22 de agosto de 2021.

de convolución (para la detección del objeto de estudio "tapabocas") en conjunto con post-procesamiento de imagen para la verificación de su **uso correcto**.

Se utilizarán datos propios generados a partir del conjunto de datos **Mask Dataset** [MakeML, 2020], el cual proporciona un total de 853 imágenes de tapabocas etiquetadas con las clases de **"With mask", "Without Mask", "Mask weared incorrect"**. Este conjunto de imágenes cuenta con una licencia *Public Domain*, la cual define que no está ligada a copyright y está disponible para su uso. De este dataset no se tendrá en cuenta las etiquetas, se utilizará únicamente las imágenes, debido a que se realizará un modelo de red neuronal de convolución.

Se busca poder identificar incluyendo diferentes tipos de tapabocas, para esto se empleará el repositorio **MaskTheFace** [Anwar and Raychowdhury, 2020], el cual basado en visión computacional permite convertir conjuntos de imágenes de rostros en conjuntos de imágenes de rostros con tapabocas. Además, permite utilizar tipos de protector como: tipo quirúrgico, N95, KN95, de tela y de gas. Así como texturas, variaciones de color e intensidad. Este proyecto es de licencia MIT, lo cual otorga el permiso de modificar, copiar, publicar, distribuir o vender partes del mismo. Este proyecto se utilizará con la finalidad de tener variedad de tapabocas y lograr la identificación correcta de los mismos, permitirá en resumen generar un conjunto de datos más extenso y enfocado a la realidad problemática.

El presente trabajo buscará satisfacer 3 estados de reconocimiento posibles ante el uso de un tapabocas: uso correcto de tapabocas, uso incorrecto de tapabocas y no uso de tapabocas. Las pruebas se realizarán en un entorno local, en el cual estará configurada una cámara a través de un puerto serial.

1.3 Estructura de la memoria

En el capítulo 2 "Contexto y estado del arte" se realiza un repaso de los trabajos destacados en el estado del arte referente a la detección de objetos y específicamente a la detección de mascarillas, donde se describen investigaciones en las cuales se aplicaron modelos de redes neuronales para desarrollar el objetivo principal de las mismas - la detección de tapabocas. Se mencionan publicaciones que hacen uso de transferencia de aprendizaje y redes neuronales profundas para el la detección de tapabocas.

Los objetivos que direccionan este trabajo y la metodología elegida para cumplir con

los mismos están descritos en el capítulo 3 ”Objetivos y metodología de trabajo”. De estos objetivos parten los requisitos que van a satisfacer la herramienta propuesta, éstos se describen en el capítulo 4 ”Requisitos del proyecto”. De los requisitos se delimita y orienta el desarrollo posterior, descrito en el capítulo 5 ”Desarrollo del trabajo”.

Finalmente, en el capítulo 6 ”Discusión de resultados” se analizan los resultados obtenidos en los escenarios de pruebas. Las conclusiones están desarrolladas en el capítulo 7 ”Conclusiones y trabajo futuro”.

Chapter 2

Contexto y Estado del Arte

Con la evolución de la tecnología, el reconocimiento digital ha tomado protagonismo en tareas que antes requerían un mayor esfuerzo e inclusive la necesidad de personal para llevarse a cabo. Tareas como: control de asistencia de personal, identificación de objetos anómalos en líneas de producción e inclusive traducción de textos a partir de una imagen.

2.1 Reconocimiento de objetos

La clasificación de imágenes o reconocimiento de objetos consiste en identificar un objeto específico en una imagen o vídeo [MathWorks, 2020]. El reconocimiento de objetos tiene como objetivos principales la representación y ”comprensión” por una máquina.

Es una técnica dentro del campo de visión artificial que se emplea para identificar objetos dentro de imágenes o vídeos. Gracias a esta técnica los computadores ”adquieren” cierto nivel de entendimiento del contenido de una imagen.

Con el avance de la tecnología se ha logrado aplicar la detección de objetos desde tareas de detección realizadas por un vehículo autónomo hasta rastreo de objetos enfocado a detección de un balón en partidos de fútbol [Pixel Solutionz, 2020].

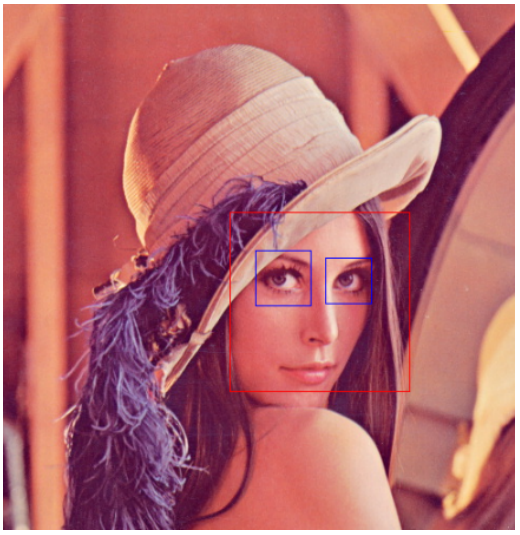
2.1.1 Clasificadores en cascada

Los clasificadores en cascada consideran regiones rectangulares adyacentes de ubicación específica en una ventana de detección. Suma las intensidades de píxeles en cada región y calcula la diferencia entre las misma. Con esto se puede categorizar subsecciones de una imagen.

El trabajo de **clasificadores en cascada** [Viola and Jones, 2001], presenta un método

de clasificación enfocado en detección rápida. El clasificador se construye seleccionando un número pequeño de características importantes, empleando una variación de AdaBoost. El ensemble de los filtros coloca los filtros más discriminatorios en las primeras capas, esto permite identificar rápidamente si el objeto no está presente en la ventana de selección y seguir iterando.

Figura 2.1: Detección de rostros utilizando Haar Cascades [OpenCV Docs, 2021a]



2.1.2 Aprendizaje profundo

Las redes neuronales profundas se han convertido en una herramienta estándar al momento de resolver problemas relacionados a visión computacional. Las redes neuronales profundas se componen de cálculos realizados por muchas capas [Witten et al., 2017]. Este tipo de diseño presenta buenos resultados tanto de precisión como tiempo de detección cuando se busca clasificar muchas clases de objetos [Krizhevsky et al., 2012].

El éxito de este modelo de redes neuronales está en aumento desde que se presentó AlexNet [Krizhevsky et al., 2012] y la red neuronal de convolución para la clasificación de imágenes que presentó en el año 2012 en el reto ImageNet.

Posteriores ganadores del desafío: VGG [Simonyan and Zisserman, 2015], GoogLeNet [Szegedy et al., 2015], ResNet [He et al., 2015], SENet [Hu et al., 2019]. La más reciente ganadora de este desafío es PNASNet-5 (2018) [Liu et al., 2018]. La tendencia en estos modelos es el aumento en cuanto cantidad de capas y disminución del ratio de error.

La principal desventaja de los modelos basados en aprendizaje profundo es la dificultad de interpretación y explicación que estos presentan [Richman and Wüthrich, 2021]. Otra

desventaja es que el entrenamiento de redes neuronales requiere de una gran cantidad de recursos de computación [Goyal and Benjamin, 2014]

Los principales dataset empleados en la evaluación de modelos de aprendizaje profundo:

- Microsoft COCO [Lin et al., 2014]. Cuenta con 91 clases para objetos comunes, 80 clases de objetos, 330000 imágenes (220000 etiquetadas) y 1.5 millones de instancias.
- CIFAR-10 [Krizhevsky, 2009]. 10 clases y 60000 imágenes a color.
- CIFAR-100 [Krizhevsky, 2009]. 100 clases agrupadas en 20 superclases. 600 imágenes por clase.
- ImageNet [Deng et al., 2009]. Conjunto de datos organizado según la jerarquía WordNet. Existen más de 100000 synset de los cuales 80000 son sustantivos, este dataset cuenta con aproximadamente 1000 imágenes para cada synset.
- MNIST [LeCun and Cortes, 2010]. Conjunto de datos de dígitos/números escritos a mano. 60000 y 10000 ejemplos para entrenamiento y validación.

2.2 Aprendizaje automático

Es un método de análisis de datos que automatiza la construcción de modelos analíticos [SAS, 2021]. Pertenece a la rama de inteligencia artificial y se basa en la idea de que los sistemas son capaces de aprender de los datos, identificando patrones y tomando decisiones con la mínima intervención humana posible. Este proceso es comparable al de minería de datos, con la diferencia de que en vez de extraer los datos para comprensión humana se extraen para comprensión del ordenador [TechTarget, 2017].

Según su naturaleza los algoritmos de aprendizaje automático pueden clasificarse en supervisado y no supervisado. Los algoritmos de aprendizaje supervisado requieren de una colección de datos etiquetada (datos históricos previamente clasificados) mientras que los algoritmos basados en aprendizaje no supervisado son capaces de extraer inferencias de las colecciones.

2.3 Procesamiento de imágenes

De acuerdo con [Gonzalez and Woods, 2008], las principales áreas del procesamiento de imagen digital son: mejorar la información para interpretación humana y procesar la información para sistemas autónomos. Una imagen puede representarse en un plano de dos dimensiones, donde cada punto (píxel) tiene una coordenada x y una coordenada y. Siguiendo este enunciado, podemos ejecutar operaciones matemáticas en la imagen para mejorar/anular características.

En cuanto a librerías que permiten realizar transformaciones/procesamiento en imágenes, las más conocidas son las siguientes:

- Scikit-image [van der Walt et al., 2014]. Está construido en C. Utiliza arreglos de NumPy para transformar las imágenes. Se caracteriza por ser rápido y cuenta con una gran variedad de métodos para manipulación de imagen.

- OpenCV [Bradski, 2000]. Primera versión en el año 2000. Enfocado en procesamiento de imagen, detección de rostros y objetos, entre otros. Escrito en C++. Cuenta con soporte para Python y puede trabajar con librerías en conjunto como NumPy, SciPy y Matplotlib. De código abierto, enfocado a visión por computador y permitir procesamiento de imagen sin esfuerzo.

- Mahotas [Coelho, 2013]. Permite utilizar más de 100 funciones avanzadas dentro de las cuales tiene soporte para: extracción de características haralick, patrones binarios, watershed, transformaciones morfológicas, convolución, entre otros. Permite trabajar con 2D y 3D.

- SciPy [Virtanen et al., 2020]. Morfología de imágenes, medir objetos, filtros lineales y no lineales, interpolación, efectos, segmentación, entre otros. Usado en su defecto para problemas que involucran soluciones matemáticas o de cálculo.

- Pillow [Clark, 2015]. Operaciones punto a punto, filtros, manipulación, entre otros. Permite un amplio soporte en cuanto formatos de imagen.

- Matplotlib [Hunter, 2007]. Mayormente usado en visualización 2D y procesamiento de imagen. No tiene un amplio soporte en cuanto a formatos de imagen pero es eficiente en cuanto extracción de información en imágenes.

2.4 OpenCV

Del inglés *Open Source Computer Vision Library.* Es una librería de código abierto, enfocada en visión computarizada y comúnmente utilizada en proyectos de aprendizaje automático [Bradski, 2000]. Cuenta con soporte a múltiples lenguajes de programación, dentro de ellos Python, contiene más de 2500 algoritmos optimizados que facilitan el procesamiento de imágenes e identificación de objetos.

Esta librería proporciona una vasta cantidad de herramientas las cuales son útiles para realizar procesamiento de imágenes, detección de rostros, seguimiento de objetos, identificación de objetos, entre otros.

En la imagen 2.2 se utilizan filtros de tipo umbral [OpenCV Docs, 2021b], dentro de las cuales resalta la eficacia de los filtros de umbral adaptativos.

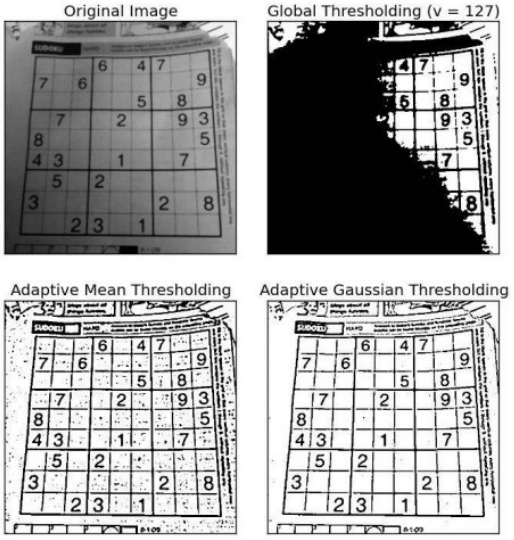


Figura 2.2: Variaciones de filtro umbral [OpenCV Docs, 2021b]

2.4.1 Filtros morfológicos

En el contexto de procesamiento de imagen, la morfología es una herramienta utilizada para extraer componentes y/o resaltar características. Operaciones como *filtrado, aclareo y poda* se realizan utilizando expresiones matemáticas.

Las transformaciones morfológicas son operaciones simples que se basan en la figura de la imagen [OpenCV, 2021]. Los filtros morfológicos requieren de dos entradas (arreglos dimensionales): la imagen a transformar y el kernel de convolución (define la naturaleza de la transformación).

Dentro de las transformaciones morfológicas disponibles en OpenCV contamos con:

- **Erosión**: Erosiona los límites del objeto en primer plano. Se desliza a través de la imagen y considerará la salida de 1 (imagen binaria) solo sí todos los píxeles seleccionados son 1.
- **Dilatación**: Funciona de manera contraria a erosión. Se desliza a través de la imagen y considerará la salida de 1 (imagen binaria) si cualquiera de los píxeles seleccionados es 1.
- **Apertura**: Consiste en la combinación de erosión seguido de dilatación.
- **Clausura**: Es inverso a apertura, consiste en la combinación de dilatación seguido de erosión.
- **Gradiente Morfológico**: Es la diferencia entre dilatación y erosión de una imagen.
- **Top Hat**: Es la diferencia entre la imagen de entrada (original) y la apertura de imagen.
- **Black Hat**: Es la diferencia entre la clausura de la imagen y la imagen de entrada (original).

En la figura 2.3 se ha recopilado la aplicación de cada uno de las transformaciones morfológicas previamente descritas.

(a) Erosión

(b) Dilatación

(c) Apertura

(d) Clausura

(e) Gradiente morfológico

(f) Top Hat

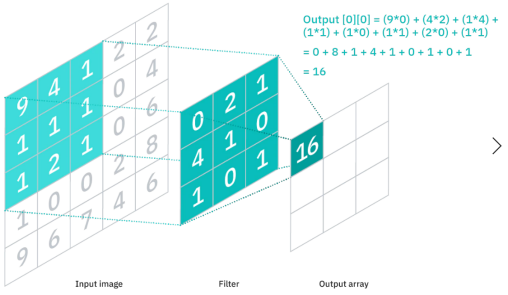
(g) Black Hat

Figura 2.3: Transformaciones morfológicas [OpenCV, 2021]

2.5 Redes neuronales de convolución

Se distinguen principalmente de otras redes por su rendimiento superior al trabajar con **imágenes, texto o voz** [IBM Cloud Education, 2020]. Cuentan con 3 tipos de capas: capa de convolución, pooling (agrupación) y fully-connected. La primera capa en una red de este tipo es la de convolución y la última una fully-connected. A mayor cantidad de capas, mayor complejidad y mayor nivel de detalle en la detección.

Figura 2.4: Operación de convolución [IBM Cloud Education, 2020]

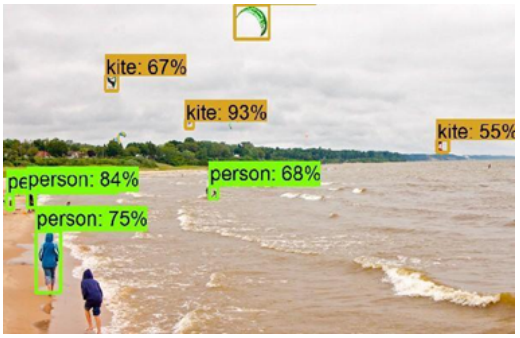


2.6 Tensorflow

Tensorflow es una plataforma de código abierto para aprendizaje automático, centrado en la simplicidad y facilidad de uso. Facilita la construcción, entrenamiento y despliegue de modelos de detección de objetos [Abadi et al., 2015].

El repositorio "Model Garden for Tensorflow" [Yu et al., 2020] cuenta con diferentes modelos de redes neuronales a disposición. Su objetivo es publicar buenas prácticas de desarrollo entre los usuarios de Tensorflow.

Figura 2.5: Modelos y ejemplos con Tensorflow [Yu et al., 2020]



2.7 Keras

Keras viene de la palabra *cuerno* en griego. Se desarrolló inicialmente como parte del esfuerzo de investigación del proyecto ONEIROS (Sistema Operativo de Robot Inteligente Neuro-electrónico Abierto).

Keras es una librería de código abierto , provee herramientas para trabajar con redes neuronales artificiales. Sigue las mejores prácticas para reducir la carga cognitiva, minimiza el número de acciones requeridas por el usuario para la implementación de redes neuronales [Chollet et al., 2015].

Keras es una API de alto nivel, escrita en Python y permite su ejecución en Tensorflow, CNTK [Seide and Agarwal, 2016] o Theano [Team et al., 2016]. Desarrollada basándose en permitir una experimentación rápida *"Ser capaz de ir de la idea al resultado lo más rápido posible es la clave para hacer una buena investigación"*.

Se rige bajo los principios de:

- Facilidad de uso
- Modularidad
- Fácil expansibilidad

Keras brinda soporte para redes convolucionales y redes recurrentes, permite también la combinación de las mismas.

Figura 2.6: Keras, ejemplo de modelo secuencial

```
# Define Sequential model with 3 layers
model = keras.Sequential(
    [
        layers.Dense(2, activation="relu", name="layer1"),
        layers.Dense(3, activation="relu", name="layer2"),
        layers.Dense(4, name="layer3"),
    ]
)
# Call model on a test input
x = tf.ones((3, 3))
y = model(x)
```

2.8 Conjunto de datos

Un dataset o conjunto de datos puede ser descrito como datos tabulados en un sistema de almacenamiento de datos estructurados [KeepCoding, 2020]. Dentro de los conjuntos de datos estos se pueden encontrar en formato de: archivo, carpeta, bases de datos y web.

2.9 Etiquetado de datos

El etiquetado del conjunto de datos si bien no es obligatoria (debido a que es posible encontrar conjuntos de datos en internet), suele ser una tarea manual. En la cual se especifica, comúnmente entre recuadros, los objetos que estamos identificando.

Dentro de las herramientas disponibles para realizar esta tarea tenemos:

- Amazon SageMaker Ground Truth
- Label Studio
- Sloth
- Labelbox

Figura 2.7: Etiquetado de tapabocas, elaboración propia

2.10 Detección de tapabocas

Dentro del contexto del presente trabajo "detección de tapabocas empleando técnicas de aprendizaje automático", encontramos diferentes enfoques a las herramientas propuestas. Las cuales varían una de las otras en: las técnicas empleadas (como transferencia de aprendizaje), los conjuntos de datos utilizados, el enfoque de detección, entre otros.

A continuación, se presentarán los trabajos más recientes en este ámbito, haciendo hincapié en sus hallazgos, limitaciones y pasos futuros.

2.10.1 Machine learning for face mask detection in image and video

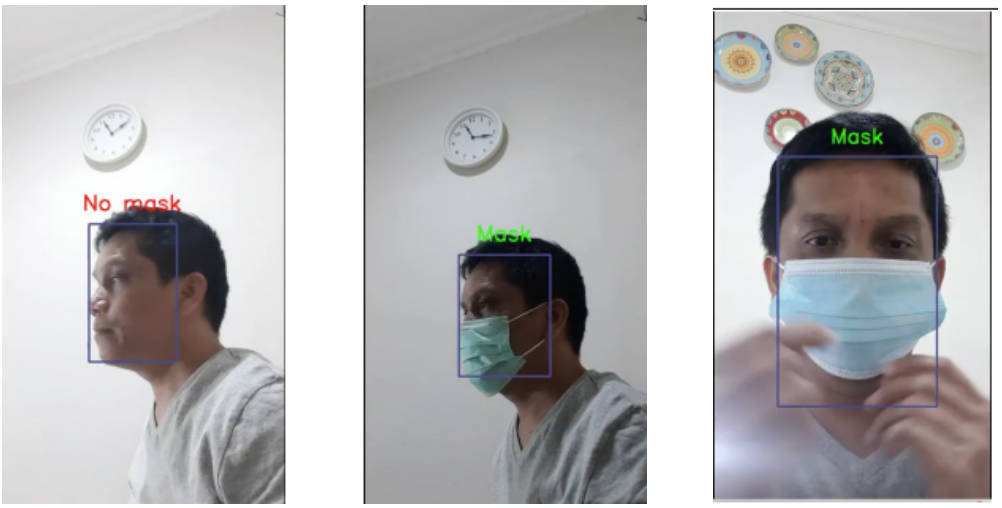
El trabajo presentado por Ramot Lubis [Lubis, 2020], principalmente motivado debido a la coyuntura COVID-19 y la necesidad del uso de tapabocas, resalta por los siguientes puntos:

Ha logrado conseguir un porcentaje de **99.4% de confianza** en cuanto a la detección de uso de tapabocas o no. Este escenario fue satisfecho gracias al dataset que utilizó, el cual estaba conformado por 90000 imágenes.

Dentro de su investigación contempla la correcta identificación de un rostro, para la cual ha logrado conseguir tanto una identificación lateral como frontal. Describe la diferencia y acentúa en cuanto una detección de rostros y una comparación de rostros, comenta que el primero está diseñado para detectar la presencia de un rostro, ignorando atributos tales como género, edad y vello facial mientras que el segundo está orientado a la indagación sobre estos atributos. Comenta también que en ambos diseños se pueden obtener **falsos positivos y falsos negativos**, en el diseño de detección de rostros esto se da detectando u obviando rostros, en el segundo fallando en cuanto a la correcta identificación de una persona a través del rostro.

El dataset utilizado para el entrenamiento de su modelo de red neuronal se compuso de 5000 imágenes de rostros con tapabocas y 90000 rostros sin tapabocas. Utilizó la función de keras para realizar la separación del conjunto de entrenamiento y de pruebas.

De este trabajo resalto la separación de conceptos para la identificación de tapabocas (detección de rostro y tapabocas) y la comparativa entre entrenamiento basado en CPU y GPU, concluye que la diferencia es de 0.5% por lo cual no es una medida que deba ser tomada en cuenta al evaluar los resultados.



(a) Estado de detección 01 (b) Estado de detección 02 (c) Estado de detección 03

Figura 2.8: Resultados de detección [Lubis, 2020]

2.10.2 Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment

El trabajo de Singh et al [Singh et al., 2021] se encuentra también motivado ante coyuntura COVID-19 y la necesidad del uso de tapabocas en lugares públicos, esta investigación resalta los siguientes puntos:

Se hizo una revisión utilizando dos modelos diferentes para lograr su objetivo, un modelo con YOLOv3 y un modelo con Faster R-CNN. Para esto los conjuntos de datos utilizados contemplan dos clases, personas utilizando mascarillas y personas sin utilizar mascarillas.

Uno de sus objetivos es detectar múltiples rostros con tapabocas o sin tapabocas dentro de la misma imagen, por lo que describe que realizarán recortes y pasarán estos datos como entrada de red neuronal (se realizará una detección por cada una de las imágenes recortadas). Utilizando la técnica de **YOLOv3**, la detección de objetos es con ”sliding windows”; esta técnica consiste en crear imágenes de menor tamaño al original, se obtiene bajo la técnica de recorte.

Una de sus conclusiones es que se puede desarrollar una red neuronal de convolución simple para lograr cumplir con el objetivo de detección. Sin embargo, menciona que se limitará a la detección simple de 1 solo elemento, es decir podrá reconocer únicamente a una sola persona, y resalta que carecerá de la capacidad para reconocer múltiples personas

y el uso o no de tapabocas en las mismas.

Utilizando Faster R-CNN, esta técnica consiste en entrenar un modelo de detección utilizando áreas previamente etiquetadas (donde se encuentra el objeto a predecir), esta técnica emplea algoritmos de búsqueda selectiva basados en CPU. Esto incrementa el tiempo de entrenamiento.

Su aplicación tiene como resultado visual dibujar rectángulos de colores verde y rojo para identificar a las personas que cuentan con mascarilla y las que no, respectivamente. Además, guarda los porcentajes de detección.

Como conclusiones finales comenta que el modelo basado en F-RCNN es ligeramente mejor. Sin embargo, en un entorno real de detección recomienda la implementación con YOLOv3 debido a que tiene en cuenta los tiempos de inferencia (detección) siendo 0.045s con el modelo YOLOv3 y 0.15s en Faster R-CNN, resaltando que los tiempos de detección empleando una F-RCNN son mayores y tendrían que ser optimizados para su pase a producción. En cuanto a precisión el modelo de YOLOv3 cuenta con 55% en promedio y el modelo de F-RCNN cuenta con 62% en promedio.

(a) Detección con YOLOv3 (b) Detección con F-RCNN

Figura 2.9: Resultados de detección con YOLOv3 y F-RCNN [Singh et al., 2021]

2.10.3 A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic

El trabajo presentado por Mohamed L. et al. [Loey et al., 2021] tiene como principal motivación la importancia del uso de tapabocas en lugares públicos, resalta la efectividad que el uso correcto de los mismos ante la propagación del virus. Comenta sobre el software basado en inteligencia artificial que se usa actualmente en el sistema de videocámaras de vigilancia del metro de París, el cual tiene como objetivo brindar data estadística a las autoridades ante posibles brotes de COVID-19.

En esta investigación se utilizó transferencia de aprendizaje profundo - *deep transfer learning* y clasificadores clásicos.

Se realizó una extracción de características a manos del primer modelo (deep transfer learning, Resnet50) y un posterior procesamiento con modelos clásicos de aprendizaje automático (comparativa entre árboles de decisión - decision trees, máquinas de vectores de soporte - svm y modelos de conjunto - ensemble).

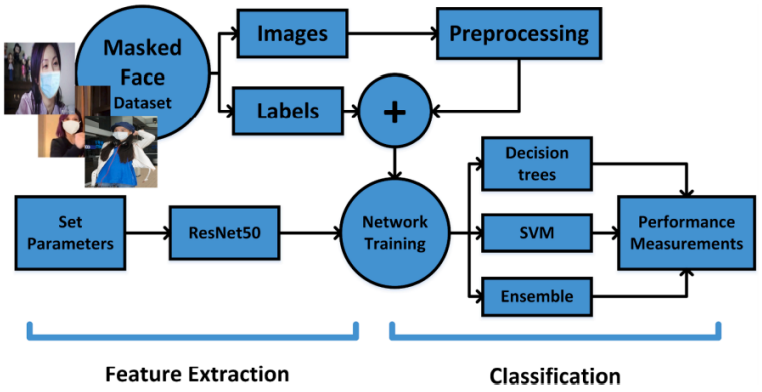
El primer conjunto de datos utilizado para esta investigación es "Real-World Masked Face Dataset (RMFD)", consiste en 5000 rostros con tapabocas y 90 000 rostros sin tapabocas. Del conjunto de datos se utilizó para esta investigación las 5000 imágenes de rostros con tapabocas y un total de 10 000 imágenes de rostros sin tapabocas, según comentan para equilibrar el dataset. El segundo conjunto de datos utilizado es "Simulated Masked Face Dataset (SMFD)", consiste en 1570 imágenes de las cuales la mitad son rostros con tapabocas y la otra mitad sin tapabocas. El tercer conjunto de datos utilizado es "Labeled Faces in the Wild (LFW)", consiste en 13 000 imágenes de celebridades con tapabocas colocado (simulado). De los conjuntos de datos el primero y segundo (RMFD y SMFD) fueron utilizados para entrenamiento, validación y pruebas. El tercero LFW fue utilizado únicamente para pruebas (benchmark)

Los conjuntos de datos fueron procesados con diferentes técnicas de entrenamiento y validación, de los 3 modelos clásicos para el procesamiento posterior a la extracción de características de parte de la red a la que se le aplicó transferencia de aprendizaje (Resnet50), el que obtuvo mejores resultados en cuanto precisión (99.64% accuracy) y menor cantidad de tiempo fue el modelo al cual se aplicó **SVM** máquinas de vectores de soporte. Estos resultados son de aplicar SVM al dataset RMFD. Con el dataset SMFD se

obtuvo un total de 99.49% y el test con LFW mostró resultados de 100% de precisión.

En esta investigación se logra una precisión superior en comparación con los trabajos relevantes mencionados, clasificando 2 categorías (con tapabocas y sin tapabocas).

Figura 2.10: Modelo transferencia de aprendizaje profundo [Loey et al., 2021]



2.10.4 Deep learning based safe social distancing and face mask detection in public areas for COVID-19 safety guidelines adherence

El trabajo presentado por Yadav [Yadav, 2020], tiene la motivación de ser una herramienta de monitoreo y prevención ante la propagación del virus COVID-19.

Está enfocado no solo a la detección del uso correcto de tapabocas, sino también al distanciamiento social, haciendo referencia a los 2 metros de distancia entre persona y persona - medida establecida por la OMS (siendo esta la segunda barrera de protección ante la propagación de este virus).

Para lograr sus objetivos propuestos, la investigación emplea una combinación de una red neuronal *ligera* (MobileNetV2), un detector de disparo único (Single Shot Detector) y la técnica de transferencia de aprendizaje. La solución desarrollada se apoya en la aplicación de modelos basados en redes neuronales para analizar el protocolo de transmisión en tiempo real RTSP usando OpenCV y MobileNet V2.

La solución final de detección descrita en la investigación sigue los siguientes pasos:

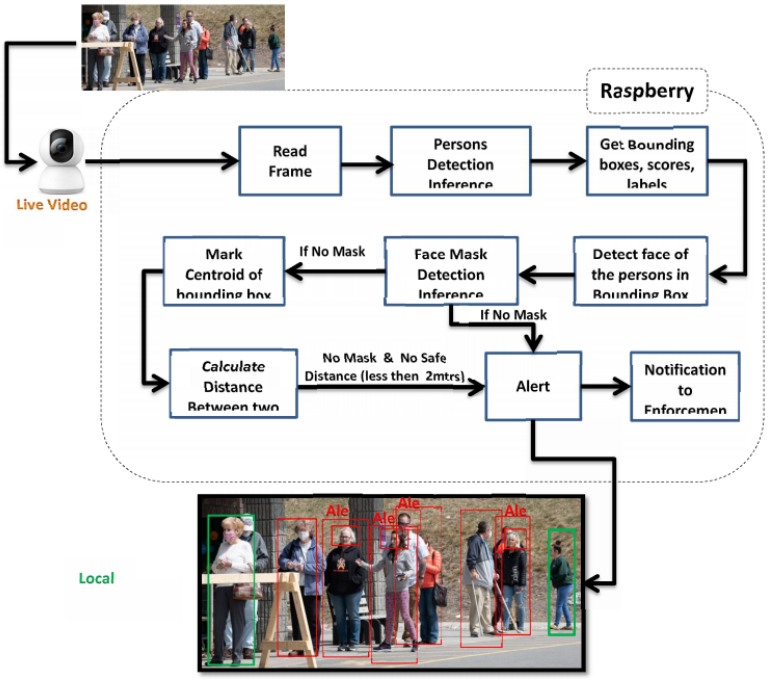
- Detección de personas (con las cuales delimita su posición).
- Detección de rostros dentro de los cuadros detectados como personas.
- Detección de tapabocas

- Compuerta lógica para determinar si cuenta con un tapabocas
- Si existiera un tapabocas calcula la distancia del próximo tapabocas
- Si no cuenta con tapabocas y no existe una distancia segura de 2 metros termina en una alerta

El conjunto de datos empleado para el modelo de clasificación está compuesto de 3165 imágenes, el cual está dividido en 2 clases (con tapabocas y sin tapabocas), se realizó una división de 80-20 respecto a entrenamiento y validación. El modelo fue entrenado teniendo en cuenta los siguientes hiper parámetros, ratio de aprendizaje de 0.001, 20 épocas, lotes de 32 y el optimizador Adam. Para la detección de aproximación se basó en la librería OpenCV y la medida de distancia euclidiana entre 2 puntos (tapabocas).

El modelo final de la investigación logra porcentajes de precisión que se encuentran dentro del rango de 85% y 95%, un porcentaje considerablemente bueno teniendo en cuenta que este modelo en su etapa de inferencia se encuentra instalado en una tarjeta raspberry pi4.

Figura 2.11: Metodología propuesta de la investigación 2.10.4 [Yadav, 2020]



Chapter 3

Objetivos y metodología de trabajo

Este capítulo identifica los objetivos que ha contemplado el presente proyecto durante el desarrollo de la herramienta software para la detección de uso correcto de tapabocas.

3.1 Objetivo General

Desarrollar un software para el reconocimiento de uso correcto, uso incorrecto y no uso de tapabocas empleando aprendizaje automático para el reconocimiento de imágenes y técnicas de procesamiento de imagen.

3.2 Objetivos específicos

- Realizar un análisis del estado del arte en sistemas de reconocimiento de objetos y técnicas de procesamiento de imagen, así como los criterios de aceptación que deberá tener el algoritmo de detección de uso correcto, incorrecto y no uso de tapabocas.

- Analizar y preparar los conjuntos de datos con los que se entrenarán los modelos de convolución para detección de tapabocas.

- Implementar técnicas de procesamiento de imágenes y rostros para delimitar las zonas de detección relevantes involucradas en la detección del uso correcto, uso incorrecto y no uso de tapabocas.

- Modelar y entrenar una red neuronal de convolución para la correcta detección de uso de tapabocas.

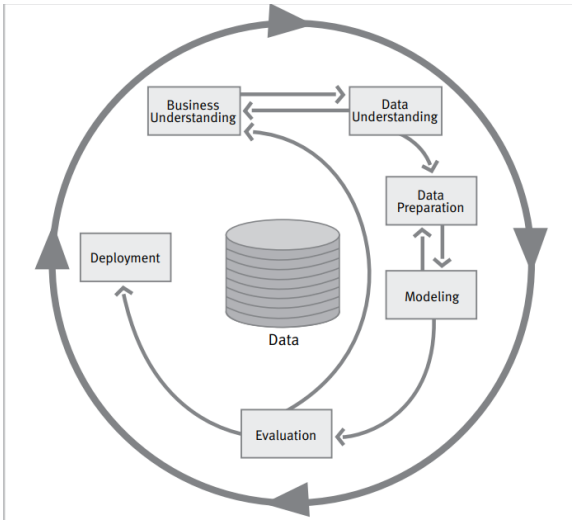
- Establecer las reglas que satisfacen los estados de salida del sistema en cuanto a identificación de rostro y uso de tapabocas.

3.3 Metodología del trabajo

Con la finalidad de lograr los objetivos planteados en el presente trabajo, se utilizó una de las metodologías más comunes para proyectos de esta índole (aprendizaje automático).

CRISP-DM [Chapman et al., 2000], del inglés "CRoss-Industry Standard for Data Mining". Los procesos que aborda CRISP-DM están desarrollados bajo 6 fases, presentes en la figura 3.1. La descripción de las fases y como se adaptaron para el desarrollo del presente proyecto está detallado en los siguientes apartados.

Figura 3.1: Fases del modelo CRISP-DM [Chapman et al., 2000]



3.3.1 Entendimiento del tema

Esta etapa establece adecuadamente los criterios específicos que deben de cumplirse para alcanzar el objetivo general, siendo "Desarrollar un software para el reconocimiento de uso correcto, uso incorrecto y no uso de tapabocas empleando aprendizaje automático para el reconocimiento de imágenes y técnicas de procesamiento de imagen", haciendo énfasis en el punto de "detección del uso correcto, uso incorrecto y no uso de tapabocas".

Los objetivos y requisitos del proyecto son delimitados por los puntos estudiados en la realidad problemática y estado del arte. Donde se entiende la importancia de este proyecto asimismo las herramientas desarrolladas a la actualidad y técnicas empleadas para la resolución de este problema, presentes en los puntos 2.10.1 y 2.10.2 correspondientes a "Machine learning for face mask detection in image and video" y "Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment".

Entendido el punto anterior, los objetivos específicos planteados para lograr alcanzar el objetivo general se encuentran en la sección Objetivos específicos.

Se estableció un capítulo adicional, el capítulo 4 "Requisitos del proyecto", el cual describe los criterios de aceptación del presente trabajo desarrollado. Incluyendo un desglose en diagramas de flujo para una representación esquemática del proceso que se implementó para lograr la detección del uso correcto, uso incorrecto y no uso de tapabocas.

3.3.2 Entendimiento de la información

Esta etapa consiste en la recolección de datos que contengan información necesaria y disponible para la ejecución del proyecto.

Para lograr el objetivo de "detección del uso correcto, uso incorrecto y no uso de tapabocas" se desglosó la detección en compuertas lógicas 4.1. Los conjuntos de datos se recopilaron teniendo en cuenta que serán utilizados para lograr el reconocimiento o no de un tapabocas presente en un rostro, con una salida de Verdadero o Falso. La comprobación de "uso correcto" se lleva a cabo en una etapa posterior, donde se tuvo en cuenta una verificación a través de procesamiento de imagen (haar cascades).

Los conjuntos de datos así como utilitarios necesarios para el posterior desarrollo fueron recopilados de librerías y proyectos bajo licencia MIT, por ende son de libre distribución y modificación bajo ninguna condición.

3.3.3 Preparación de datos

Del conjunto mencionado en la sección 1.2 "Planteamiento del trabajo" - MaskDataset se recopilaron las imágenes consideradas relevantes (ignorando las clases etiquetadas como mask_weared_incorrect) para el entrenamiento de la red neuronal de convolución. Se aplicaron transformaciones de recorte, alejamiento y variaciones en brillo y contraste con la finalidad de aumentar la cantidad disponible de imágenes (en el dataset recopilado se podía encontrar +1 clase por imagen).

Finalmente del conjunto de datos MaskDataset, una vez aplicado recorte de imagen y transformaciones, de las 853 imágenes disponibles se obtuvo un total de 1508 imágenes.

Este conjunto de datos resultante cuenta con dos clases.

- **with_mask**: Compuesto por imágenes de personas (rostros) y un tapabocas (uso correcto).
- **without_mask**: Compuesto por imágenes de rostros en general.

La relación de imágenes está en 1:1 respecto a with_mask y without_mask.

3.3.4 Modelado

Esta etapa se rige a dos partes.

La primera parte consiste en la identificación de uso de tapabocas, para el cual se entrenará un modelo de red neuronal de convolución empleando las clases obtenidas de la preparación de los datos (with_mask y without_mask). Por la casuística se desarrollará un modelo de convolución donde en la última capa (DENSE) se utilizará una función de activación la cual permita una única salida, las variaciones realizadas (cantidad de capas, filtros y funciones de activación) al modelo de convolución (en búsqueda de un modelo equilibrado en cuanto a **precisión y pérdida**) se desglosan en la sección 5.3.2 "Diseño de Red Neuronal" dentro del capítulo 5 "Desarrollo del trabajo".

La segunda parte consiste en la identificación de uso correcto de tapabocas, para la cual se utilizarán técnicas de identificación rápida de objetos (haar cascades), esta compuerta lógica permitirá reconocer si un tapabocas se encuentra correctamente o incorrectamente colocado. Para esta etapa se considerarán haar cascades individuales para reconocimiento de nariz y boca.

3.3.5 Evaluación

Se contemplarán dos tipos de evaluaciones para el presente proyecto (pruebas durante desarrollo y pruebas de integración).

Evaluación de red neuronal de convolución durante desarrollo, la cual se hará de manera constante durante el desarrollo de la red neuronal de convolución. Esta métrica se obtendrá únicamente como referente, debido a que los resultados en desarrollo e inferencia pueden variar. Además, en estas pruebas solo se estaría evaluando la clase de without_mask y sería negativamente (si el resultado de la red neuronal es negativo).

Evaluación de detección de haar cascades en desarrollo, se ajustarán los parámetros que utilizan las detecciones rápidas, teniendo en cuenta un ambiente controlado iluminado y una distancia máxima entre objeto a reconocer y cámara de 40cm.

Pruebas de integración, esta etapa se llevará a cabo con la evaluación de dos tipos de tapabocas. El escenario de pruebas A se realizará con tapabocas de tipo quirúrgico o higiénico y el escenario de pruebas B se realizará con tapabocas de tipo FPP2 o FPP3. En ambos escenarios se contemplarán métricas de precisión, exhaustividad, f1 y exactitud. Asimismo, una prueba que contemple el tiempo de clasificación.

3.3.6 Producción

La etapa de producción de este proyecto consiste en la consolidación del modelo entrenado. Exportar en un formato de producción la red neuronal <.h5> asimismo la compilación del proyecto <.py>.

En esta etapa se elabora la memoria del proceso, pruebas y conclusiones involucradas en el desarrollo de este proyecto.

Chapter 4

Requisitos del proyecto

Los estados reconocibles por el proyecto a desarrollar buscan contemplar el uso correcto de tapabocas, el uso incorrecto de tapabocas y el no uso de tapabocas. Estos estados están descritos en la tabla 4.1

Estado	Descripción	Criterio
-1	Estado no reconocible, posible error	Error de comunicación puerto serial, error interno
0	No uso de tapabocas	Detección de tapabocas negativo
1	Uso incorrecto de tapabocas	Detección de tapabocas exitoso. Está a la vista nariz, boca o barbilla
2	Uso correcto de tapabocas	Detección de tapabocas exitoso, nariz boca y barbilla cubiertos

Tabla 4.1: Estados posibles dentro del programa

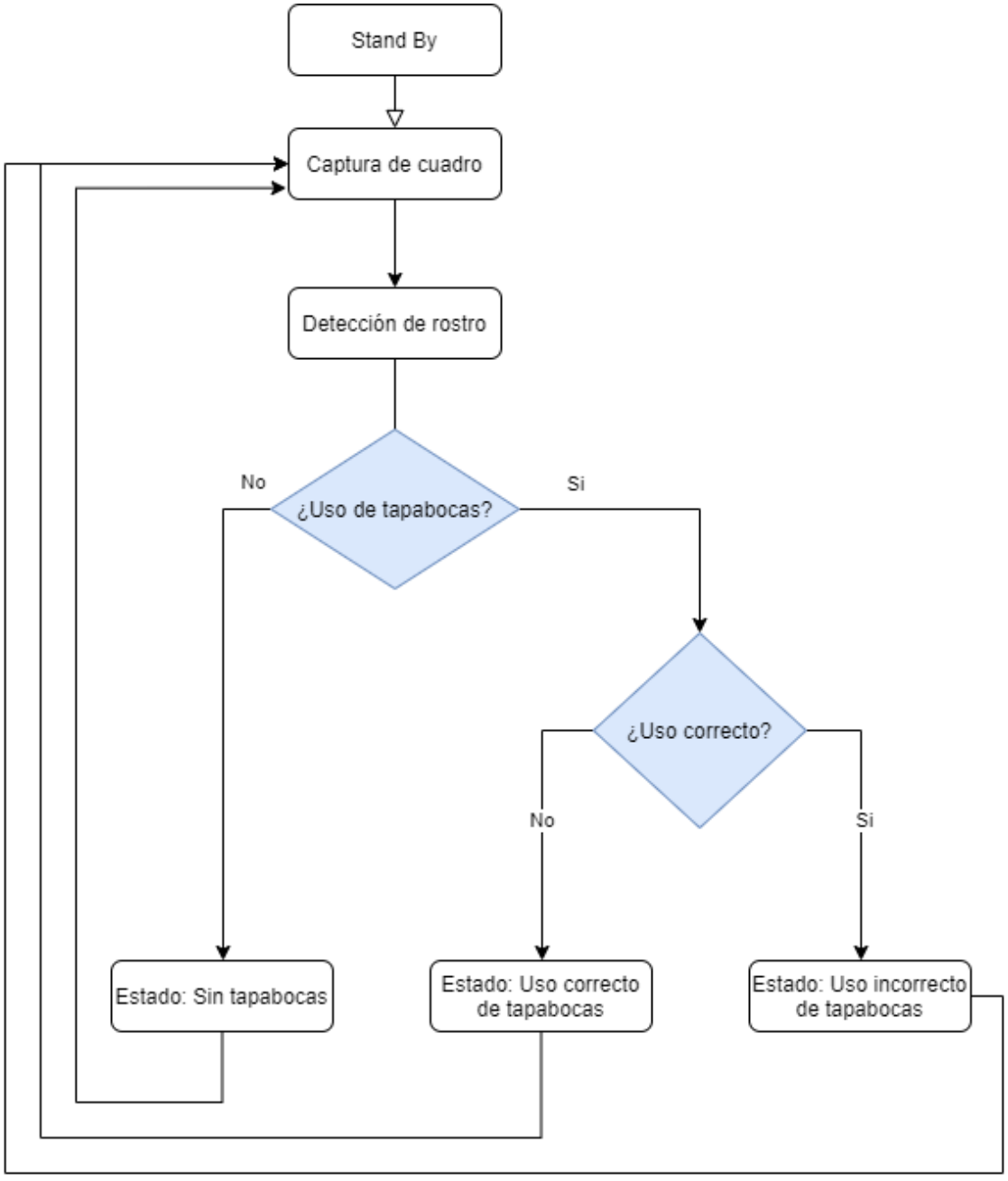
El presente proyecto desglosará los estados de detección deseados, mencionadas en la tabla 4.1, con la finalidad de identificar los algoritmos y técnicas que serán usados. Estos algoritmos y técnicas se encuentran descritos en la tabla 4.2

Para facilitar la comprensión de las salidas propuestas en la tabla 4.1 y los módulos que serán desarrollados dentro del capítulo 5 ”Desarrollo del trabajo”, se realizó un diagrama de flujo de datos, el cual describe 3 estados de salida y las compuertas lógicas que determinarán cada una de estas. La extracción de características se realizará apoyado principalmente en una red neuronal de convolución y la aplicación de diferentes haar cascades. Este diagrama se encuentra disponible en la imagen 4.1.

Etapa	Algoritmo / Técnica	Fuente
Captura de cuadro	Recorte de imagen (rostro)	Haar Cascade
Detección de partes de rostro	Procesamiento de imagen a través de filtros morfológicos (detección de nariz y boca)	Haar Cascades
Detección de tapabocas	Entrenamiento de una red neuronal de convolución	Autoría propia
Detección de uso correcto	Secuencia de condiciones lógicas	Autoría propia
Detección de uso incorrecto	Secuencia de condiciones lógicas	Autoría propia
Detección de no uso	Secuencia de condiciones lógicas	Autoría propia

Tabla 4.2: Técnicas a realizar

Figura 4.1: Diagrama de flujo de datos



Chapter 5

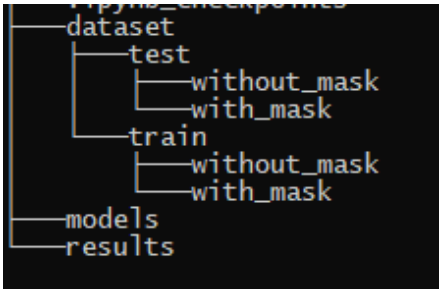
Desarrollo del trabajo

En el presente capítulo se detalla los pasos realizados para la culminación de este proyecto.

5.1 Estructura del proyecto

La estructura de archivos para desarrollar los objetivos fue la siguiente

Figura 5.1: Estructura de archivos



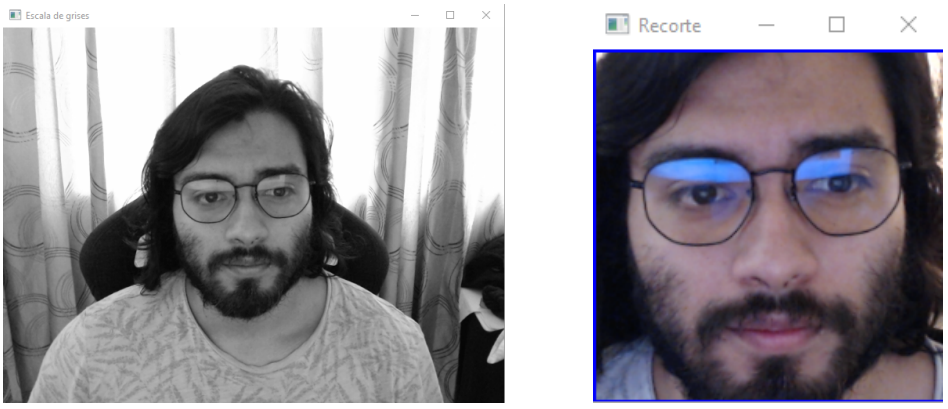
Donde se organizaron los archivos e imágenes de la siguiente forma:

- **dataset:** Conjunto de imágenes recopiladas para el entrenamiento de la red neuronal de convolución, el cual está dividido en dos carpetas (train y test). Cada una contiene carpetas que corresponden a las clases de *without_mask* y *with_mask*
- **models:** Donde se almacenan los modelos neuronales entrenados.
- **results:** Resultados obtenidos dentro de la ejecución de los programas.

5.2 Detección de rostros

Para la detección de rostros se empleó la librería de OpenCV apoyada de un haar cascade, en este caso el haar cascade frontalface. Con la finalidad de detectar en primera instancia un rostro. Se tuvo en cuenta también la aplicación de filtros y convoluciones para facilitar la detección de los rostros, los filtros aplicados en la captura de imagen fueron los siguientes:

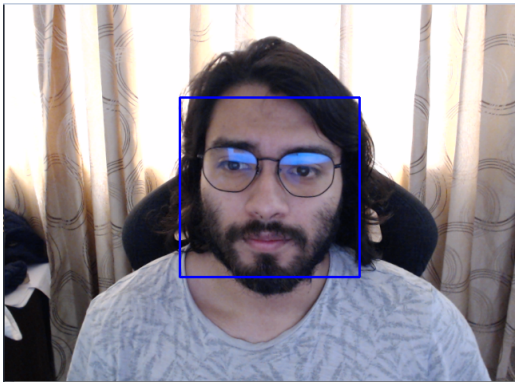
- **Blanco y Negro:** Su aplicación dentro del proyecto es de preprocesamiento. Para facilitar la detección con el haar cascade empleado.
- **Recorte:** Se aplica un recorte en el área resultante del filtro haar, con la finalidad de obtener un plano de rostro antes de ser procesado por el modelo de convolución.
- **Voltear:** Voltear la imagen. Únicamente por preferencia.



(a) Aplicación de filtro Blanco y Negro (b) Recorte de Imagen

Figura 5.2: Detección de rostro

Figura 5.3: Resultado



5.3 Detección de tapabocas - Red Neuronal de convolución

Para el entrenamiento de la red neuronal se utilizó Keras, en la versión proporcionada por Tensorflow. Definido previamente en la imagen 4.1 el flujo cuenta con una primera condicional, la cual es ¿Uso de tapabocas? Para lo cual, se abordó de la siguiente manera

5.3.1 Conjunto de datos

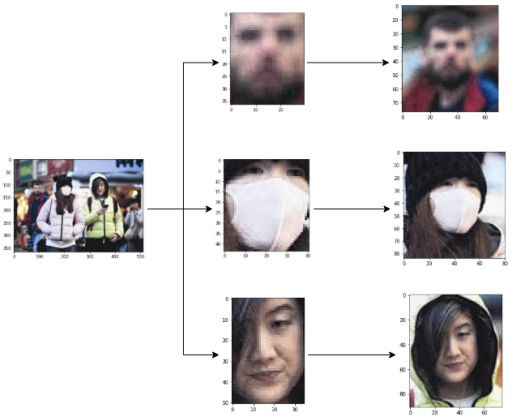
Se utilizó como base el conjunto de datos MaskDataset, el cual fue descargado en su versión MaskPascalVOC. La descompresión de este formato resulta en dos carpetas:

- **annotations**: Contiene archivos en formato XML, los cuales describen el nombre de la imagen de referencia y las clases que se encuentran identificadas en la imagen *coordenadas*
- **images**: Conjunto de 853 imágenes en formato PNG

Con una función propia se explora este dataset y contabiliza únicamente las imágenes de rostros con tapabocas (with_mask) y sin tapabocas (without_mask). Se ignora la clase con tapabocas colocado incorrectamente (mask_weared_incorrect). Esto reduce el total de clases encontradas de 4072 a 3949. De este total de imágenes 3232 pertenecen a la categoría “with_mask” y 717 a “without_mask”.

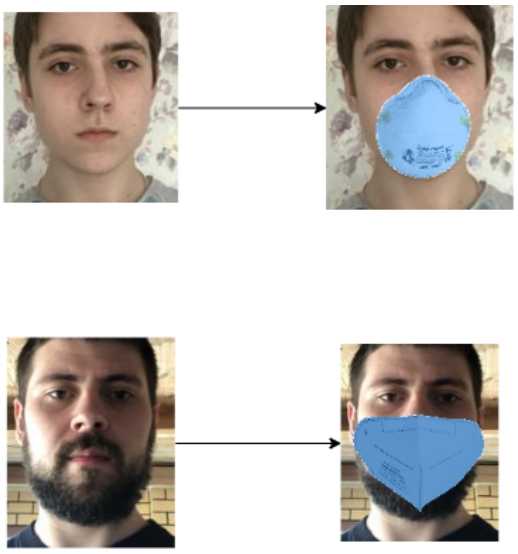
Una vez contabilizadas las imágenes se explora el nuevo dataset, aplicando una función de recorte sobre cada imagen incluyendo un espaciado (padding) con la finalidad de tener un resultado mejor delimitado. Cada recorte obtenido fue almacenado en una carpeta con el formato “tipo/index.jpg” - ej. with_mask/1.jpg

Figura 5.4: Transformación de imagen



De esta operación se realizó un descarte manual posterior, donde se omitieron imágenes borrosas y/o consideradas no aptas para el entrenamiento. Además, con la finalidad de enriquecer el dataset se utilizó el proyecto MaskTheFace en algunos rostros sin tapabocas.

Figura 5.5: Colocar tapabocas



De estas transformaciones se realizó una segunda limpieza. Finalmente, se recopiló un conjunto de datos de 1508 imágenes en formato *jpg* con diferentes variaciones de tamaño.

Separación de conjunto de datos		
Conjunto de datos 1	Entrenamiento	1206
	Validación	302

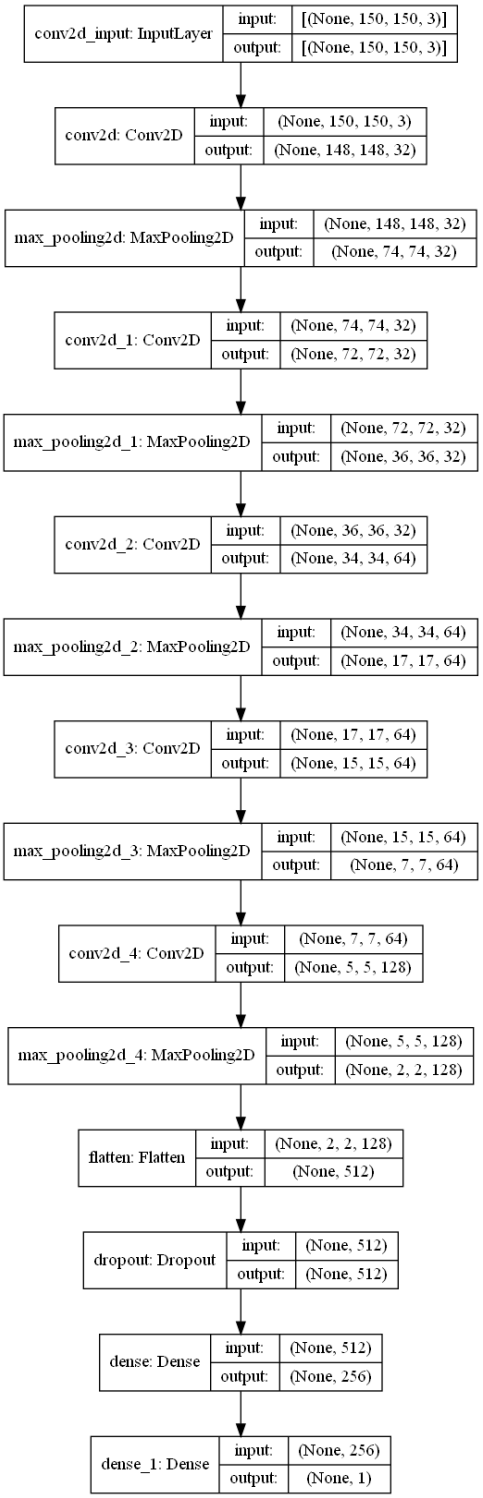
Tabla 5.1: Separación de conjunto de datos

Para la separación del conjunto de datos se empleó la función de keras 'train_test_split' con un valor de 80-20 para entrenamiento y validación. Se tuvo en cuenta también utilizar el parámetro "shuffle" en el conjunto de datos, con la finalidad de evitar diferencias sistemáticas en el modelo de entrenamiento.

5.3.2 Diseño de Red Neuronal

El diseño de red neuronal con el que se obtuvo mejores resultados se encuentra en la imagen 5.6

Figura 5.6: Modelo de red neuronal de convolución



Para esta red neuronal varió la cantidad de capas y cantidad de neuronas dentro de cada una, además se tuvo en cuenta las siguientes técnicas de optimización:

- **Dropout:** Técnica para ”omitir” neuronas durante el entrenamiento, se utiliza para evitar overfitting.
- **Image Augmentation:** Técnica para variar el conjunto de datos de entrenamiento.

Debido a que buscamos obtener un resultado de activación ”verdadero” o falso” se trabajó un modelo binario. Es decir la última capa de la neurona cuenta con una activación sigmoidal. Lo cual traduce su salida en 2 posibles valores (0 y 1) para el cual uno corresponde a ”with_mask” y el otro a ”without_mask”

5.3.3 Entrenamiento de Red Neuronal

Se empleó la técnica ”data augmentation” al conjunto de datos de entrenamiento. Donde se aplicaron las siguientes transformaciones:

- **Escalado**
- **Acercamiento**
- **Alejamiento**
- **Recorte**
- **Invertir**

Para el conjunto de validación no se realizaron variaciones.
El tamaño de entrada de la red neuronal (150x150) se definió en la sección 5.3.2.
Parámetros de entrenamiento:

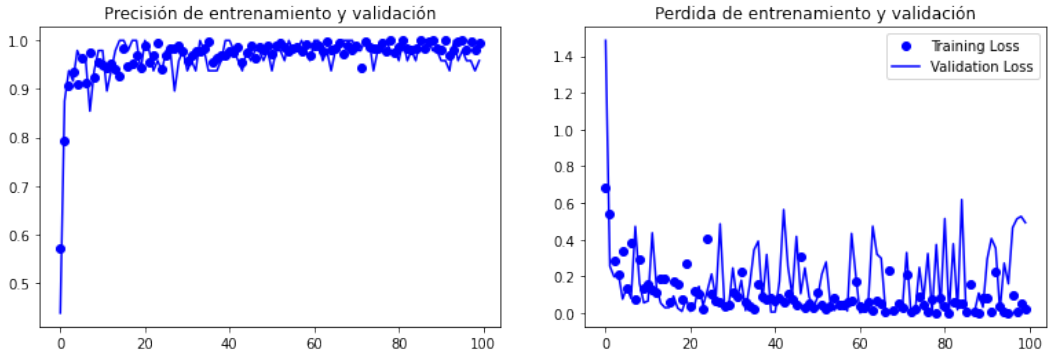
- **Pasos por epoch:** 20
- **Epochs:** 100
- **Tamaño de Batch:** 64

Las métricas observadas en cada epoch de la red neuronal fueron el valor de pérdida (loss) y el de precisión (accuracy) 5.7.

Para evaluar la eficiencia del entrenamiento se compararon las métricas previamente mencionadas. Obteniendo los gráficos de precisión de entrenamiento y validación 5.8a y pérdida de entrenamiento y validación 5.8b.

Figura 5.7: Entrenamiento de red neuronal

```
Epoch 1/100
20/20 - 7s - loss: 0.6853 - accuracy: 0.5719 - val_loss: 1.4851 - val_accuracy: 0.4375
Epoch 2/100
20/20 - 1s - loss: 0.5420 - accuracy: 0.7941 - val_loss: 0.2551 - val_accuracy: 0.8750
Epoch 3/100
20/20 - 1s - loss: 0.2857 - accuracy: 0.9062 - val_loss: 0.1979 - val_accuracy: 0.9375
Epoch 4/100
20/20 - 1s - loss: 0.2116 - accuracy: 0.9344 - val_loss: 0.2188 - val_accuracy: 0.9167
Epoch 5/100
20/20 - 1s - loss: 0.3404 - accuracy: 0.9094 - val_loss: 0.0764 - val_accuracy: 0.9792
Epoch 6/100
20/20 - 1s - loss: 0.1382 - accuracy: 0.9625 - val_loss: 0.1399 - val_accuracy: 0.9583
Epoch 7/100
20/20 - 1s - loss: 0.3803 - accuracy: 0.9125 - val_loss: 0.0802 - val_accuracy: 0.9583
Epoch 8/100
20/20 - 1s - loss: 0.0740 - accuracy: 0.9750 - val_loss: 0.4722 - val_accuracy: 0.8542
```



(a) Precisión de entrenamiento y validación (b) Pérdida de entrenamiento y validación

Figura 5.8: Precisión y pérdida de entrenamiento y validación

5.4 Identificación de uso correcto de tapabocas

El paso posterior a la detección de un tapabocas presente en un recorte de imagen de rostro es el de identificar el correcto uso del mismo. Se considera un uso correcto de tapabocas o mascarilla cuando este cubre en su totalidad el área de la boca y nariz [Fundación Cardiovascular de Colombia, 2020].

El enunciado anterior define los dos procesos que se abarcan en esta sección:

- **Detección de nariz**
- **Detección de boca**

Ambos fueron resueltos empleando haar cascades.

La configuración de parámetros se realizó de manera manual, configurando lo siguiente:

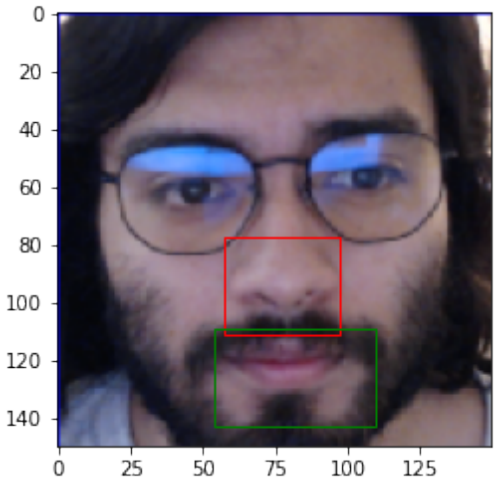
- **scaleFactor**: Parámetro que especifica que tanto será reducido el tamaño de imagen por cada transformación

- **minNeighbors**: La cantidad de vecinos que cada candidato debe tener para ser seleccionado.

Los mejores resultados se obtuvieron con scaleFactor con un valor de 2 y minNeighbors con un valor de 4. Como salida de estas detecciones se obtiene la representación de la imagen 5.9. Donde se detecta exitosamente nariz y boca (rojo y verde respectivamente).

Debido a la naturaleza de la detección esta verificación es incluyente, es decir para que la posición del tapabocas sea considerada correcta no debe ser detectados dentro del recorte de rostro ni nariz ni boca.

Figura 5.9: Detección de nariz y boca empleando haar cascades



5.5 Escenarios de detección

Una vez culminados los pasos anteriores se realizó la integración de los módulos de: procesamiento de imagen, detección y recorte de rostro, identificación de tapabocas y resultado de detección. Siguiendo el diagrama especificado en la figura 4.1

Este proyecto está orientado a una detección de rostro frontal. Para todos los escenarios de prueba a continuación se tuvo en cuenta una buena iluminación y una cercanía de 50cm 15+-.

Las puntos que se tuvieron en cuenta para la evaluación de ambos escenarios fueron los siguientes:

- **Detección de rostros:** Detección de rostros, resultados válidos y no válidos.

- **Duración de estados:** Duración dentro de los estados de detección del proyecto, con tapabocas sin tapabocas y con tapabocas mal colocado.
- **Transiciones entre estados:** El tiempo y errores causados por la transición de un estado al otro.
- **Clasificación:** Clasificación dentro de los estados del sistema.

Evaluación de escenario

Se realizará la matriz de confusión para las 3 clases presentes en cada escenario de detección. Donde se contemplarán las siguientes métricas de evaluación:

- Precisión

$$precision = \frac{TP}{TP + FP}$$

- Exhaustividad

$$recall = \frac{TP}{TP + FN}$$

- F1

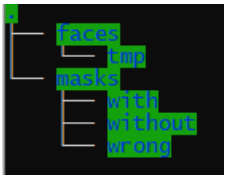
$$f1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

- Exactitud

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

La organización de carpetas y archivos se realiza mediante la estructura de ejemplo mostrada en la imagen 5.10. Donde las detecciones de rostro se estarán guardando en la carpeta *faces* y las detecciones se guardarán por clase en la carpeta *masks*, posteriormente una revisión manual determinará si la clasificación fue correcta. Cada archivo se guarda con un timestamp.

Figura 5.10: Organización de carpetas para pruebas



5.5.1 Escenario de pruebas A

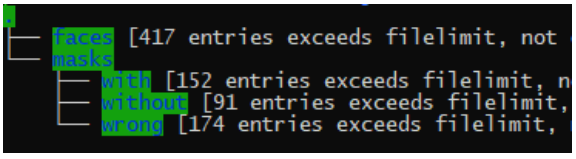
Este escenario contempla modelo de tapabocas de tipo higiénicas y quirúrgicas.

La duración de prueba en este escenario sumó un total de **26 segundos**. Y la distribución fue la siguiente

- Estado 01 - Con mascarilla 9 Segundos
- Estado de transición 4 Segundos
- Estado 02 - Con mascarilla mal colocada 7 segundos
- Estado de transición 1 segundo
- Estado 03 - Sin mascarilla 5 segundos

La distribución de las imágenes dentro de las carpetas para los escenarios de prueba fue la siguiente:

Figura 5.11: Resultado - Escenario de pruebas A



Detección de rostros

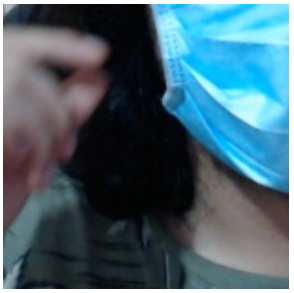
Rostros detectados válidos y no válidos

	Válido	No Válido	
Rostros	413	4	417

Tabla 5.2: Rostros detectados - Prueba A

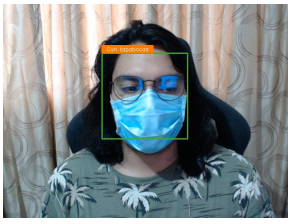
Imagen de referencia de un rostro válido y no válido se encuentran en la imagen 5.12

(a) Rostro válido

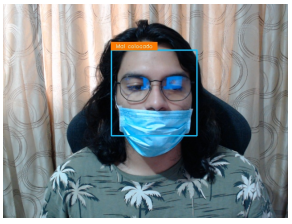


(b) Rostro no válido

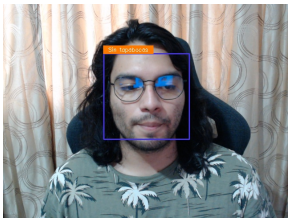
Figura 5.12: Prueba A - Rostros válidos y no válidos



(a) Estado 01

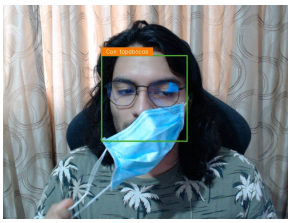


(b) Estado 02

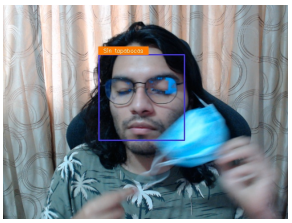


(c) Estado 03

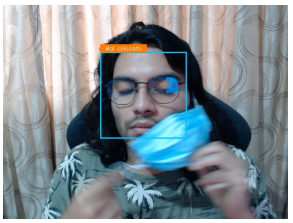
Figura 5.13: Prueba A - Estados de detección



(a) Transición 01



(b) Transición 02



(c) Transición 03

Figura 5.14: Prueba A - Errores de transición

Clasificación de estados

Matriz de confusión de los resultados obtenidos en la prueba

	Con tapabocas	Mal colocado	Sin tapabocas	
Con tapabocas	136	16	0	152
Mal colocado	15	156	3	174
Sin tapabocas	2	0	89	91

Tabla 5.3: Matriz de confusión - Prueba A

Errores causados por transición entre estados

- Estado 01 v Estado 02: 13 errores de los 16 contabilizados
- Estado 02 v Estado 01: 2 errores de los 2 contabilizados
- Estado 02 v Estado 03: 3 errores de los 3 contabilizados

Métricas - Con tapabocas

$$TP = 136$$

$$FP = 15 + 2 = 17$$

$$FN = 16$$

$$TN = 156 + 89 + 3 + 0 = 248$$

Precisión

$$precision = \frac{136}{136 + 17}$$

$$precision \simeq 0.89$$

Exhaustividad

$$recall = \frac{136}{136 + 16}$$

$$recall \simeq 0.89$$

F1

$$f1 = 2 \cdot \frac{0.89 \cdot 0.89}{0.89 + 0.89}$$

$$f1 = 2 \cdot \frac{0.7921}{1.78}$$

$$f1 \simeq 0.89$$

Exactitud

$$accuracy = \frac{136 + 248}{136 + 248 + 17 + 16}$$

$$accuracy = \frac{384}{417}$$

$$accuracy \simeq 0.921$$

Métricas - Con tapabocas mal colocado

$$TP = 156$$

$$FP = 16 + 0 = 16$$

$$FN = 15 + 3 = 18$$

$$TN = 136 + 0 + 89 + 2 = 227$$

Precisión

$$precision = \frac{156}{156 + 16}$$

$$precision \simeq 0.90$$

Exhaustividad

$$recall = \frac{156}{156 + 18}$$

$$recall \simeq 0.89$$

F1

$$f1 = 2 \cdot \frac{0.90 \cdot 0.89}{0.90 + 0.89}$$

$$f1 = 2 \cdot \frac{0.801}{1.79}$$

$$f1 \simeq 0.89$$

Exactitud

$$accuracy = \frac{156 + 227}{156 + 227 + 16 + 18}$$

$$accuracy = \frac{383}{417}$$

$$accuracy \simeq 0.918$$

Métricas - Sin tapabocas

$$TP = 89$$

$$FP = 0 + 3 = 3$$

$$FN = 2 + 0 = 2$$

$$TN = 136 + 16 + 15 + 156 = 323$$

Precisión

$$precision = \frac{89}{89 + 3}$$

$$precision \simeq 0.97$$

Exhaustividad

$$recall = \frac{89}{89 + 2}$$

$$recall \simeq 0.98$$

F1

$$f1 = 2 \cdot \frac{0.97 \cdot 0.98}{0.97 + 0.98}$$

$$f1 = 2 \cdot \frac{0.9506}{1.95}$$

$$f1 \simeq 0.975$$

Exactitud

$$accuracy = \frac{89 + 323}{89 + 323 + 3 + 2}$$

$$accuracy = \frac{412}{417}$$

$$accuracy \simeq 0.988$$

5.5.2 Escenario de pruebas B

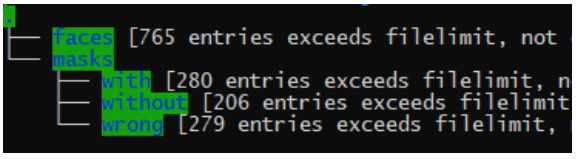
Este escenario contempla modelo de tapabocas de tipo FPP2 y FPP3.

La duración de prueba en este escenario sumó un total de **50 segundos**. Y la distribución fue la siguiente

- Estado 01 - Con mascarilla 20 Segundos
- Estado de transición 1 Segundos
- Estado 02 - Con mascarilla mal colocada 16 segundos
- Estado de transición 1 segundo
- Estado 03 - Sin mascarilla 12 segundos

La distribución de las imágenes dentro de las carpetas para los escenarios de prueba fue la siguiente:

Figura 5.15: Resultado - Escenario de pruebas B



Detección de rostros

Rostros detectados válidos y no válidos

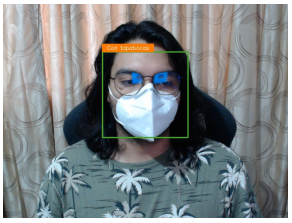
	Válido	No Válido	
Rostros	765	0	765

Tabla 5.4: Rostros detectados - Prueba B

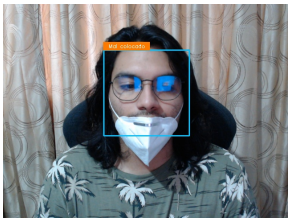
Imagen de referencia de un rostro válido y no válido se encuentran en la imagen 5.16

(a) Rostro válido

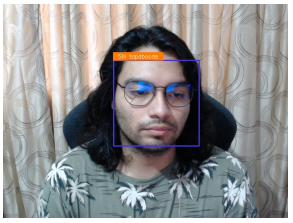
Figura 5.16: Prueba B - Rostros válidos y no válidos



(a) Estado 01



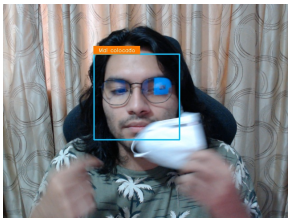
(b) Estado 02



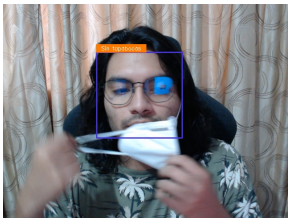
(c) Estado 03

Figura 5.17: Prueba B - Estados de detección

(a) Transición 01



(b) Transición 02



(c) Transición 03

Figura 5.18: Prueba B - Errores de transición

Clasificación de estados

Matriz de confusión de los resultados obtenidos en la prueba

	Con tapabocas	Mal colocado	Sin tapabocas	
Con tapabocas	280	0	0	280
Mal colocado	11	266	2	279
Sin tapabocas	0	4	202	206

Tabla 5.5: Matriz de confusión - Prueba B

Errores causados por transición entre estados

- Estado 01 v Estado 02: 7 errores de los 11 contabilizados
- Estado 02 v Estado 03: 2 errores de los 2 contabilizados
- Estado 02 v Estado 02: 1 errores de los 4 contabilizados

Métricas - Con tapabocas

$$TP = 280$$

$$FP = 11 + 0 = 11$$

$$FN = 0 + 0 = 0$$

$$TN = 266 + 2 + 4 + 202 = 474$$

Precisión

$$precision = \frac{280}{280 + 11}$$

$$precision \simeq 0.96$$

Exhaustividad

$$recall = \frac{280}{280 + 0}$$

$$recall = 1$$

F1

$$f1 = 2 \cdot \frac{0.96 \cdot 1}{0.96 + 1}$$

$$f1 = 2 \cdot \frac{0.96}{1.96}$$

$$f1 \simeq 0.979$$

Exactitud

$$accuracy = \frac{280 + 474}{280 + 474 + 11 + 0}$$

$$accuracy = \frac{754}{765}$$

$$accuracy \simeq 0.985$$

Métricas - Con tapabocas mal colocado

$$TP = 266$$

$$FP = 0 + 4 = 4$$

$$FN = 11 + 2 = 13$$

$$TN = 280 + 0 + 0 + 202 = 482$$

Precisión

$$precision = \frac{266}{266 + 4}$$

$$precision \simeq 0.98$$

Exhaustividad

$$recall = \frac{266}{266 + 13}$$

$$recall \simeq 0.95$$

F1

$$f1 = 2 \cdot \frac{0.98 \cdot 0.95}{0.98 + 0.95}$$

$$f1 = 2 \cdot \frac{0.931}{1.93}$$

$$f1 \simeq 0.964$$

Exactitud

$$accuracy = \frac{266 + 482}{266 + 482 + 4 + 13}$$

$$accuracy = \frac{748}{765}$$

$$accuracy \simeq 0.97$$

Métricas - Sin tapabocas

$$TP = 202$$

$$FP = 2 + 0 = 2$$

$$FN = 0 + 4 = 4$$

$$TN = 280 + 0 + 11 + 266 = 557$$

Precisión

$$precision = \frac{202}{202 + 2}$$

$$precision \simeq 0.99$$

Exhaustividad

$$recall = \frac{202}{202 + 4}$$

$$recall \simeq 0.98$$

F1

$$f1 = 2 \cdot \frac{0.99 \cdot 0.98}{0.99 + 0.98}$$

$$f1 = 2 \cdot \frac{0.9702}{1.97}$$

$$f1 \simeq 0.984$$

Exactitud

$$accuracy = \frac{202 + 557}{202 + 557 + 2 + 4}$$

$$accuracy = \frac{759}{765}$$

$$accuracy \simeq 0.992$$

5.5.3 Resumen de resultados

En esta sección se describen los dos escenarios de prueba, teniendo en cuenta sus resultados por clase (con tapabocas, mal colocado y sin tapabocas) y la precisión, exhaustividad, f1 y exactitud alcanzada.

Escenario de prueba A

	Precisión	Exhaustividad	F1	Exactitud
Con tapabocas	0.89	0.89	0.89	0.921
Mal colocado	0.90	0.89	0.89	0.918
Sin tapabocas	0.97	0.98	0.975	0.988

Macro F1

$$MacroF1 = \frac{0.89 + 0.89 + 0.975}{3}$$

$$MacroF1 \simeq 0.918$$

Escenario de prueba B

	Precisión	Exhaustividad	F1	Exactitud
Con tapabocas	0.96	1	0.979	0.985
Mal colocado	0.98	0.95	0.964	0.97
Sin tapabocas	0.99	0.98	0.984	0.992

Macro F1

$$MacroF1 = \frac{0.979 + 0.964 + 0.984}{3}$$

$$MacroF1 \simeq 0.975$$

5.5.4 Tiempos de detección

El total de imágenes detectadas y analizadas con el algoritmo será igual al de rostros detectados. No se tendrá en cuenta errores de clasificación (sin embargo se considerará el tiempo de transición), únicamente será contrastado el tiempo total y la cantidad de imágenes resultantes.

Además, considerar que este algoritmo, para propósito de pruebas está dedicando milésimas a la escritura y lectura de los archivos. Es decir, que en un escenario de producción el tiempo de inferencia podría ser menor.

Escenario de pruebas A

El total de rostros detectados fue 417 y el tiempo de ejecución del mismo fue de 26 segundos aproximadamente.

- Estado 01 - Con mascarilla 8.90 Segundos - 143 imágenes
- Estado de transición 01 3.48 Segundos - 57 imágenes
- Estado 02 - Con mascarilla mal colocada 6.70 segundos - 108 imágenes
- Estado de transición 02 1.39 segundos - 23 imágenes
- Estado 03 - Sin mascarilla 5.29 segundos - 86 imágenes

En la tabla 5.6 se realiza la división del tiempo total entre la cantidad de imágenes clasificadas.

	Imágenes	Tiempo	Resultado
Estado 01	143	8.90	0.062
Transición 01	57	3.48	0.061
Estado 02	108	6.70	0.062
Transición 02	23	1.39	0.060
Estado 03	86	5.29	0.061

Tabla 5.6: Tiempos de detección escenario de pruebas A

Escenario de pruebas B

El total de rostros detectados fue 765 y el tiempo de ejecución del mismo fue de 50 segundos aproximadamente.

- Estado 01 - Con mascarilla 19.722 segundos - 283 imágenes
- Estado de transición 01 1.301 segundos - 22 imágenes
- Estado 02 - Con mascarilla mal colocada 15.210 segundos - 240 imágenes
- Estado de transición 02 1.363 segundos - 22 imágenes
- Estado 03 - Sin mascarilla 12.146 segundos - 198 imágenes

En la tabla 5.7 se realiza la división del tiempo total entre la cantidad de imágenes clasificadas.

	Imágenes	Tiempo	Resultado
Estado 01	283	19.722	0.069
Transición 01	22	1.301	0.059
Estado 02	240	15.210	0.063
Transición 02	22	1.363	0.061
Estado 03	198	12.146	0.061

Tabla 5.7: Tiempos de detección escenario de pruebas B

Chapter 6

Discusión de resultados

En el presente capítulo del proyecto se analizarán e interpretarán los resultados obtenidos en las pruebas realizadas en el capítulo 5 ”Desarrollo del trabajo”, específicamente en las subsecciones 5.5.3 y 5.5.4 correspondientes a ”Resumen de resultados” y ”Tiempos de detección”.

6.1 Evaluación de modelo

6.1.1 Precisión

La métrica de precisión de un modelo permite medir la calidad en cuanto a la clasificación.

Las medidas de precisión obtenidas para las clases ”Con tapabocas”, ”Mal colocado” y ”Sin tapabocas” en el escenario A son de 0.89, 0.90 y 0.97, respectivamente. En el escenario B son de 0.96, 0.98 y 0.99.

La métrica de precisión más elevada se obtiene en la clase **sin tapabocas** en ambos escenarios. Sin embargo, el presente proyecto resalta que existe una diferencia sustancial en las medidas obtenidas del escenario B en las clases ”Con tapabocas” y ”Mal colocado”.

El modelo presenta una precisión ponderada de **0.92** en cuanto a la clasificación de **tapabocas de tipo quirúrgico y/o higiénico**. Y **0.97** en cuanto a la clasificación de **tapabocas de tipo FPP2 y FPP3**

6.1.2 Exhaustividad

La métrica de exhaustividad de un modelo permite medir la eficiencia en cuanto a la correcta identificación. Mide la sensibilidad (en cuanto porcentaje el modelo desarrollado

permite identificar la información relevante).

Las medidas de exhaustividad obtenidas para las clases: ”Con tapabocas”, ”Mal colocado” y ”Sin tapabocas” en el escenario A son de 0.89, 0.89 y 0.98, respectivamente. En el escenario B son de 1, 0.95 y 0.98.

El modelo presenta una precisión ponderada de **0.92** en cuanto a la clasificación de **tapabocas de tipo quirúrgico y/o higiénico**. Y **0.97** en cuanto a la clasificación de **tapabocas de tipo FPP2 y FPP3**.

6.1.3 F1

La métrica de F1 combina las métricas de precisión y recall en un solo valor. Se consideró esta métrica debido a que el presente trabajo tiene la premisa de que tanto la precisión como exhaustividad son de igual importancia.

Las medidas de F1 obtenidas para las clases ”Con tapabocas”, ”Mal colocado” y ”Sin tapabocas” en el escenario A son de 0.89, 0.89 y 0.975, respectivamente. En el escenario B son de 0.979, 0.964 y 0.984.

El modelo presenta un F1 ponderado de **0.91** en cuanto a la clasificación de **tapabocas de tipo quirúrgico y/o higiénico**. Y **0.97** en cuanto a la clasificación de **tapabocas de tipo FPP2 y FPP3**.

6.1.4 Exactitud

La métrica de exactitud mide la cantidad de aciertos que ha tenido el modelo.

Las medidas de exactitud obtenidas para las clases ”Con tapabocas”, ”Mal colocado” y ”Sin tapabocas” en el escenario A son de 0.921, 0.918 y 0.988, respectivamente. En el escenario B son de 0.985, 0.97 y 0.992.

El modelo presenta una exactitud ponderada de **0.942** en cuanto a la clasificación de **tapabocas de tipo quirúrgico y/o higiénico**. Y **0.982** en cuanto a la clasificación de **tapabocas de tipo FPP2 y FPP3**.

6.2 Evaluación de tiempos de detección

Los resultados obtenidos de la tabla 5.6 ”Tiempos de detección escenario de pruebas A” en las clasificaciones ”Estado 01”, ”Transición 01”, ”Estado 02”, ”Transición 02”, ”Estado 03” son 0.062, 0.061, 0.062, 0.060, 0.061 segundos por imagen, respectivamente. Demostrando

una varianza no significativa en los tiempos por imagen entre categorías y un tiempo promedio de identificación de **0.061**

Los resultados obtenidos de la tabla 5.7 "Tiempos de detección escenario de pruebas A" en las clasificaciones "Estado 01", "Transición 01", "Estado 02", "Transición 02", "Estado 03" son 0.069, 0.059, 0.063, 0.061, 0.061 segundos por imagen, respectivamente. Este escenario tiene una varianza de 0.01 entre las clases Estado 01 y Transición 01, sin embargo presenta un tiempo de respuesta de clasificación de **0.062** en promedio por imagen.

6.3 Resultados finales

En cuanto a clasificación, se considera que si bien se cuenta con un buen resultado en ambos escenarios, este modelo presenta un mejor desempeño en la correcta clasificación en el escenario B **tapabocas de tipo FPP2 y FPP3**. Donde ha logrado resultados superiores en cada una de las métricas comparadas con una varianza ponderada que oscila entre 0.04 - 0.06 (según la métrica).

En cuanto a tiempo de detección, si bien hubo una diferenciación entre clases relativamente significativa en el escenario de pruebas B (0.01) respecto al escenario de pruebas A (0.002), los **tiempos ponderados fueron de 0.061 y 0.062** para el escenario de pruebas A - tapabocas de tipo quirúrgico y/o higiénico y B tapabocas de tipo FPP2 y FPP3, respectivamente. Concluyendo, que no existen diferencias significativas.

Chapter 7

Conclusiones y trabajo futuro

7.1 Conclusiones

La herramienta software desarrollada en este proyecto permite, a partir de imágenes capturadas por una cámara, determinar si una persona cuenta con un tapabocas bien colocado, mal colocado o si no cuenta con tapabocas. Esto con el objetivo de ser una herramienta útil para la prevención ante la propagación del virus SARS-CoV-2.

Para la preparación del conjunto de datos se revisaron diferentes fuentes disponibles con licencias MIT, lo cual permite su libre distribución y modificación. Se tuvo en cuenta una preparación de dataset con la técnica de *Data Augmentation* utilizando variaciones como *escalado, acercamiento, alejamiento, recorte e invertir*. Finalmente la distribución se realizó en carpetas (train y validation) las cuales contenían las clases (with y without), teniendo en cuenta 80-20 para entrenamiento y validación lo cual produjo una distribución de 1206 imágenes y 302 respectivamente.

Las técnicas de procesamiento de imagen que se utilizaron para delimitar la zona de detección (rostro) en la aplicación fueron la aplicación de filtros de blanco y negro y recorte de imagen combinado con haar cascade, la capa haar fue empleada con la finalidad de realizar una detección rápida, esta capa fue ajustada con los parámetros de *vecinos cercanos* y *factor de escalamiento*, con la finalidad de evitar falsos positivos (múltiples detecciones dentro de la misma imagen). La salida resultante de la aplicación de la capa haar consiste de un arreglo de puntos/coordenadas, los cuales delimitan la zona de detección de manera lógica y visual. La delimitación visual es apreciada mediante el rectángulo de detección que se sobrepone. La delimitación lógica se realiza en combinación con la técnica de recorte y es aprovechada en las capas posteriores de este proyecto, sirve de entrada para la red

neuronal de convolución que identificará la presencia o no de tapabocas.

En cuanto a la construcción y validación de la red neuronal, se iteraron diferentes modelos realizando variaciones tanto en las capas como en la cantidad de filtros y neuronas que conformaban el modelo. El modelo que obtuvo mejores resultados se encuentra adjunto como diagrama en la imagen 5.6. Las métricas de entrenamiento se encuentran disponibles en la figura 5.8. Finalmente las clases reconocibles son 0 para uso de tapabocas y 1 para no uso de tapabocas.

Desarrollada la red neuronal el paso posterior es la integración dentro de un algoritmo para reconocer no solo el uso de tapabocas sino el uso correcto. La regla utilizada para definir el uso correcto o incorrecto de tapabocas se da bajo la premisa de "nariz y boca deben quedar correctamente protegidas" es decir que no se visualicen. Para esto se aplicaron dos capas haar de detección, una para narices y otra para bocas. Se ajustaron utilizando también los parámetros de *vecinos cercanos* y *factor de escalamiento* con valores equivalentes a los utilizados en la capa haar para detección de rostro.

En cuanto a la evaluación del algoritmo final de detección. Este proyecto ha logrado alcanzar exactitudes de *accuracy* $\simeq 0.98$ en ambos escenarios probados (tapabocas de tipo quirúrgico/higiénico y tapabocas de tipo FPP2/FPP3), demostrando la robustez del algoritmo final de detección. Resaltando este modelo presenta mejores resultados de detección cuando se trata de tapabocas de tipo FPP2 y FPP3, donde no cuenta con falsos positivos al momento de detectar este tipo de tapabocas. Concluyo que esto se debe al conjunto de datos de entrenamiento, en su mayoría fueron imágenes con este tipo de tapabocas.

El presente proyecto ha logrado implementar satisfactoriamente un estado de detección útil en cuanto a la prevención ante la propagación del virus COVID-19, el estado "Uso incorrecto de tapabocas", el cual logra ser detectado de manera eficiente tanto en tapabocas de tipo quirúrgico/higiénico como con tapabocas de tipo FPP2 o FPP3. En comparativa con los antecedentes analizados para el desarrollo de este proyecto el aporte diferencial de investigación es la identificación de este estado. En cuanto a velocidad de detección, este proyecto ha logrado tiempos de inferencia promedio de $\simeq$0.06s, no logró superar los tiempos de detección con los que se obtuvo en el modelo de implementación con YOLOv3 (0.045s) en la investigación 2.10.2 "Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment" sin embargo, logra superar los tiempos obtenidos a través del modelo implementado con R-CNN que tenía un tiempo de inferencia de 0.15s.

7.2 Trabajo Futuro

Este trabajo ha sido realizado para una correcta identificación de tapabocas en una imagen de rostro de perspectiva frontal. Sería una mejora considerable realizar una identificación considerando también una perspectiva lateral. Esto se puede llevar a cabo teniendo en cuenta haar cascades de detección lateral de rostro.

Las pruebas en un escenario real se encuentran fuera del alcance de este trabajo. Sin embargo; teniendo en cuenta la naturaleza de detección de este algoritmo y las pruebas que se realizaron de manera local, el mejor escenario de integración que puede solventar es similar al de una cabina de desinfección.

A su vez, se podría realizar una integración teniendo en cuenta parámetros de ajuste (tales como vecinos cercano y factor de escalamiento) para, según las condiciones donde se implemente, lograr una detección correcta.

Considerar también, la detección del tipo de mascarilla, la cual para este proyecto no se encuentra dentro del alcance. La detección del tipo de mascarilla podría servir de indicador en cuanto al factor de bioseguridad requerido dentro de un escenario en específico. Así como otras medidas de bioseguridad que se pueden detectar a través de visión por computador, así como la mencionada en la investigación 2.10.4 ”Deep learning based safe social distancing and face mask detection in public areas for COVID-19 safety guidelines adherence” referente al distanciamiento en cuanto proximidad de persona a persona.

Bibliography

[Abadi et al., 2015] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Recuperado 18 de abril de 2021, a partir de `www.tensorflow.org`.

[Anwar and Raychowdhury, 2020] Anwar, A. and Raychowdhury, A. (2020). Masked face recognition for secure authentication. `https://github.com/aqeelanwar/MaskTheFace`.

[BBC News Mundo, 2020] BBC News Mundo (2020). 3 países que fueron ejemplo de lucha contra el coronavirus y ahora ven una "segunda ola" de contagios. Recuperado 15 de abril de 2021, a partir de `https://www.bbc.com/mundo/noticias-53541458`.

[Bradski, 2000] Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

[Chapman et al., 2000] Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., and Wirth, R. (2000). Crisp-dm 1.0 step-by-step data mining guide. Technical report, The CRISP-DM consortium.

[Chollet et al., 2015] Chollet, F. et al. (2015). Keras. `https://keras.io`.

[Clark, 2015] Clark, A. (2015). Pillow (pil fork) documentation.

[Coelho, 2013] Coelho, L. P. (2013). Mahotas: Open source software for scriptable computer vision. *Journal of Open Research Software*, 1.

[Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

[Fundación Cardiovascular de Colombia, 2020] Fundación Cardiovascular de Colombia (2020). ¿Cómo usar de forma correcta el tapabocas? Recuperado 16 de junio de 2021, a partir de `http://www.fcv.org/corp/noticias/prensa/164-como-usar-de-forma-correcta-el-tapabocas`.

[Gonzalez and Woods, 2008] Gonzalez, R. and Woods, R. (2008). *Digital Image Processing*. Pearson/Prentice Hall.

[Goyal and Benjamin, 2014] Goyal, S. and Benjamin, P. (2014). Object recognition using deep neural networks: A survey.

[He et al., 2015] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.

[Hu et al., 2019] Hu, J., Shen, L., Albanie, S., Sun, G., and Wu, E. (2019). Squeeze-and-excitation networks.

[Hunter, 2007] Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95.

[IBM Cloud Education, 2020] IBM Cloud Education (2020). Convolutional neural networks. Recuperado 18 de abril de 2021, a partir de `https://www.ibm.com/cloud/learn/convolutional-neural-networks`.

[KeepCoding, 2020] KeepCoding (2020). ¿qué son los datasets y dónde conseguirlos? Recuperado 10 de mayo de 2021, a partir de `https://keepcoding.io/blog/que-son-datasets`.

[Krizhevsky, 2009] Krizhevsky, A. (2009). Learning multiple layers of features from tiny images.

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

[LeCun and Cortes, 2010] LeCun, Y. and Cortes, C. (2010). MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/.

[Lin et al., 2014] Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., and Dollár, P. (2014). Microsoft coco: Common objects in context. cite arxiv:1405.0312Comment: 1) updated annotation pipeline description and figures; 2) added new section describing datasets splits; 3) updated author list.

[Liu et al., 2018] Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.-J., Fei-Fei, L., Yuille, A., Huang, J., and Murphy, K. (2018). Progressive neural architecture search.

[Loey et al., 2021] Loey, M., Manogaran, G., Taha, M. H. N., and Khalifa, N. E. M. (2021). A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the covid-19 pandemic. *Measurement*, 167:108288.

[Lubis, 2020] Lubis, R. (2020). MACHINE LEARNING (CONVOLUTIONAL NEURAL NETWORKS) FOR FACE MASK DETECTION IN IMAGE AND VIDEO. Recuperado 10 de mayo de 2021, a partir de `http://eprints.binus.ac.id/id/eprint/36805`.

[MakeML, 2020] MakeML (2020). Mask dataset. Recuperado 10 de mayo de 2021, a partir de `https://makeml.app/datasets/mask`.

[MathWorks, 2020] MathWorks (2020). ¿Qué es el reconocimiento de objetos? Tres cosas que es necesario saber. Recuperado 18 de abril de 2021, a partir de `https://la.mathworks.com/solutions/image-video-processing/object-recognition.html`.

[OpenCV, 2021] OpenCV (2021). Morphological transformations. Recuperado 18 de abril de 2021, a partir de `https://docs.opencv.org/master/d9/d61/tutorial_py_morphological_ops.html`.

[OpenCV Docs, 2021a] OpenCV Docs (2021a). Face detection using haar cascades. Recuperado 18 de abril de 2021, a partir de `https://docs.opencv.org/4.5.3/d2/d99/tutorial_js_face_detection.html`.

[OpenCV Docs, 2021b] OpenCV Docs (2021b). Image thresholding. Recuperado 18 de abril de 2021, a partir de `https://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html`.

[Pixel Solutionz, 2020] Pixel Solutionz (2020). Application of object detection in real life. Recuperado 16 de junio de 2021, a partir de https://www.pixelsolutionz.com/application-object-detection-real-life.

[Real Academia Española, 2020] Real Academia Española (2020). Recuperado 15 de abril de 2021, a partir de https://dle.rae.es.

[Richman and Wüthrich, 2021] Richman, R. and Wüthrich, M. V. (2021). Localglmnet: interpretable deep learning for tabular data.

[Roser et al., 2021] Roser, M., Ritchie, H., Ortiz-Ospina, E., and Hasell, J. (2021). Coronavirus Pandemic (COVID-19). *Our World in Data*. Recuperado 21 de setiembre de 2021, a partir de https://ourworldindata.org/coronavirus.

[SAS, 2021] SAS (2021). Machine learning what it is and why it matters. Recuperado 16 de junio de 2021, a partir de https://www.sas.com/en_us/insights/analytics/machine-learning.html.

[Seide and Agarwal, 2016] Seide, F. and Agarwal, A. (2016). Cntk: Microsoft's open-source deep-learning toolkit. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 2135, New York, NY, USA. Association for Computing Machinery.

[Simonyan and Zisserman, 2015] Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition.

[Singh et al., 2021] Singh, S., Ahuja, U., Kumar, M., Kumar, K., and Sachdeva, M. (2021). Face mask detection using yolov3 and faster r-cnn models: Covid-19 environment. *Multimedia Tools and Applications*. Recuperado a partir de https://doi.org/10.1007/s11042-021-10711-8.

[Szegedy et al., 2015] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9.

[Team et al., 2016] Team, T. T. D., Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., Bastien, F., Bayer, J., Belikov, A., Belopolsky, A., Bengio,

Y., Bergeron, A., Bergstra, J., Bisson, V., Snyder, J. B., Bouchard, N., Boulanger-Lewandowski, N., Bouthillier, X., de Brébisson, A., Breuleux, O., Carrier, P.-L., Cho, K., Chorowski, J., Christiano, P., Cooijmans, T., Côté, M.-A., Côté, M., Courville, A., Dauphin, Y. N., Delalleau, O., Demouth, J., Desjardins, G., Dieleman, S., Dinh, L., Ducoffe, M., Dumoulin, V., Kahou, S. E., Erhan, D., Fan, Z., Firat, O., Germain, M., Glorot, X., Goodfellow, I., Graham, M., Gulcehre, C., Hamel, P., Harlouchet, I., Heng, J.-P., Hidasi, B., Honari, S., Jain, A., Jean, S., Jia, K., Korobov, M., Kulkarni, V., Lamb, A., Lamblin, P., Larsen, E., Laurent, C., Lee, S., Lefrancois, S., Lemieux, S., Léonard, N., Lin, Z., Livezey, J. A., Lorenz, C., Lowin, J., Ma, Q., Manzagol, P.-A., Mastropietro, O., McGibbon, R. T., Memisevic, R., van Merriënboer, B., Michalski, V., Mirza, M., Orlandi, A., Pal, C., Pascanu, R., Pezeshki, M., Raffel, C., Renshaw, D., Rocklin, M., Romero, A., Roth, M., Sadowski, P., Salvatier, J., Savard, F., Schlüter, J., Schulman, J., Schwartz, G., Serban, I. V., Serdyuk, D., Shabanian, S., Étienne Simon, Spieckermann, S., Subramanyam, S. R., Sygnowski, J., Tanguay, J., van Tulder, G., Turian, J., Urban, S., Vincent, P., Visin, F., de Vries, H., Warde-Farley, D., Webb, D. J., Willson, M., Xu, K., Xue, L., Yao, L., Zhang, S., and Zhang, Y. (2016). Theano: A python framework for fast computation of mathematical expressions.

[TechTarget, 2017] TechTarget (2017). Aprendizaje automático (machine learning). Recuperado 16 de junio de 2021, a partir de `https://searchdatacenter.techtarget.com/es/definicion/Aprendizaje-automatico-machine-learning`.

[van der Walt et al., 2014] van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., Yu, T., and the scikit-image contributors (2014). scikit-image: image processing in Python. *PeerJ*, 2:e453.

[Viola and Jones, 2001] Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I.

[Virtanen et al., 2020] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas,

J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

[Witten et al., 2017] Witten, I. H., Frank, E., Hall, M. A., and Pal, C. J. (2017). Chapter 10 - deep learning. In Witten, I. H., Frank, E., Hall, M. A., and Pal, C. J., editors, *Data Mining (Fourth Edition)*, pages 417–466. Morgan Kaufmann, fourth edition edition.

[World Health Organization, 2020a] World Health Organization (2020a). OMS | Infecciones por coronavirus. Recuperado 15 de abril de 2021, a partir de `http://www.who.int/topics/coronavirus_infections/es`.

[World Health Organization, 2020b] World Health Organization (2020b). Preguntas y respuestas sobre la enfermedad por coronavirus (COVID-19). Recuperado 15 de abril de 2021, a partir de `https://www.who.int/es/emergencies/diseases/novel-coronavirus-2019/advice-for-public/q-a-coronaviruses`.

[Yadav, 2020] Yadav, S. (2020). 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*, 8(7):1368–1375.

[Yu et al., 2020] Yu, H., Chen, C., Du, X., Li, Y., Rashwan, A., Hou, L., Jin, P., Yang, F., Liu, F., Kim, J., and Li, J. (2020). TensorFlow Model Garden. `https://github.com/tensorflow/models`.

Appendix A

Apéndices

A.1 Pruebas

A continuación se adjuntarán imágenes resultado de las pruebas realizadas, estas imágenes tienen de descripción un timestamp.

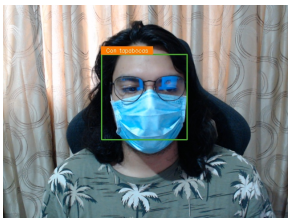
A.1.1 Resultados Prueba A

Imágenes resultantes del caso de prueba A, tapabocas tipo higiénicas y quirúrgicas.

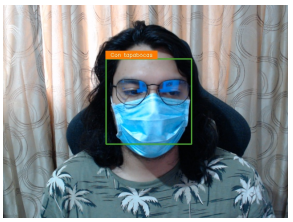
Con tapabocas

3 imágenes de las 152 imágenes resultantes de la prueba.

(a) 17:17:27.460622



(b) 17:17:30.157684

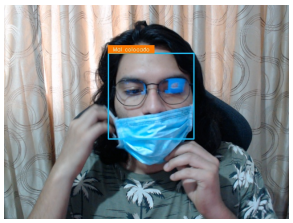


(c) 17:17:35.099781

Figura A.1: Prueba A - Con tapabocas

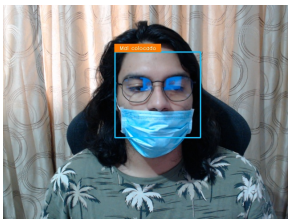
Mal colocado

3 imágenes de las 174 imágenes resultantes de la prueba.



(a) 17:17:38.086811

(b) 17:17:41.312138

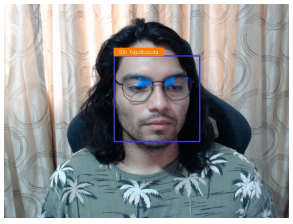


(c) 17:17:45.290596

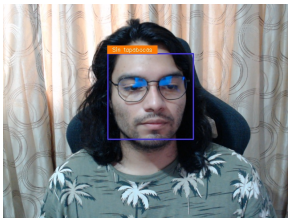
Figura A.2: Prueba A - Con tapabocas mal colocado

Sin tapabocas

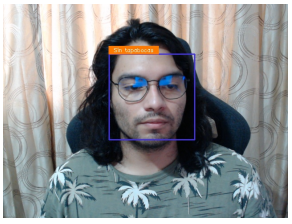
3 imágenes de las 91 imágenes resultantes de la prueba.



(a) 17:17:49.548853



(b) 17:17:51.038022

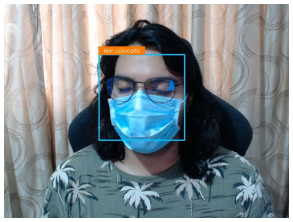


(c) 17:17:51.331020

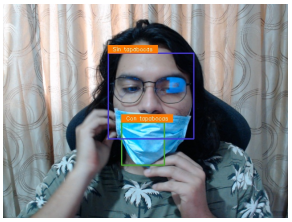
Figura A.3: Prueba A - Sin tapabocas

Irregularidades

3 imágenes de las 36 imágenes con irregularidades identificadas dentro de la prueba.



(a) 17:17:26.942139



(b) 17:17:38.500480

(c) 17:17:38.676951

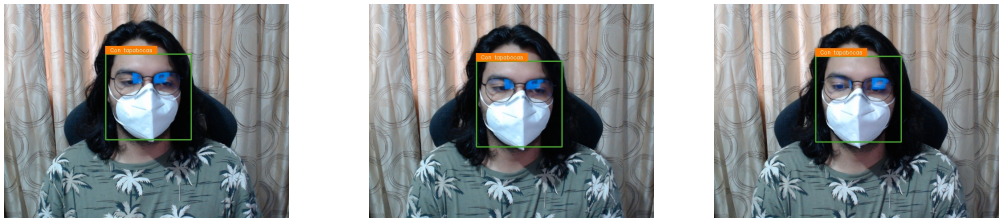
Figura A.4: Prueba A - Irregularidades

A.1.2 Resultados Prueba B

Imágenes resultantes del caso de prueba A, tapabocas tipo FPP2 y FPP3.

Con tapabocas

3 imágenes de las 280 imágenes resultantes de la prueba.

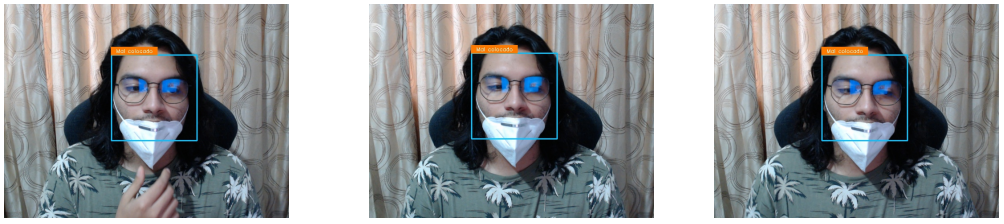


(a) 17:40:12.812493 (b) 17:40:20.069242 (c) 17:40:28.775307

Figura A.5: Prueba B - Con tapabocas

Mal colocado

3 imágenes de las 279 imágenes resultantes de la prueba.

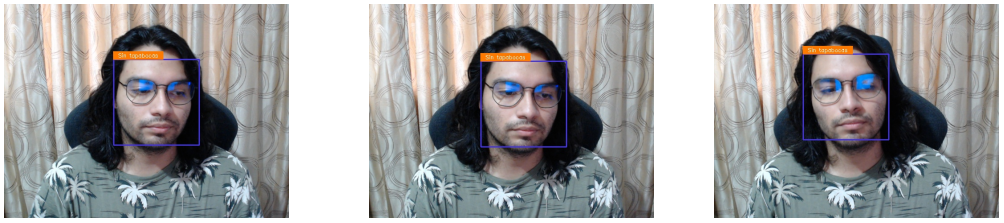


(a) 17:40:31.873975 (b) 17:40:43.943422 (c) 17:40:46.328690

Figura A.6: Prueba B - Con tapabocas mal colocado

Sin tapabocas

3 imágenes de las 206 imágenes resultantes de la prueba.

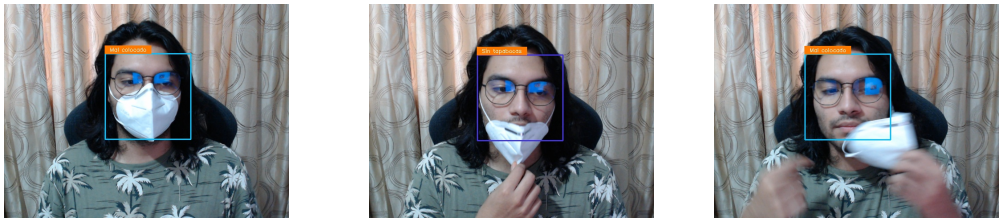


(a) 17:40:50.310595 (b) 17:40:56.475069 (c) 17:40:59.838543

Figura A.7: Prueba B - Sin tapabocas

Irregularidades

3 imágenes de las 17 imágenes con irregularidades identificadas dentro de la prueba.

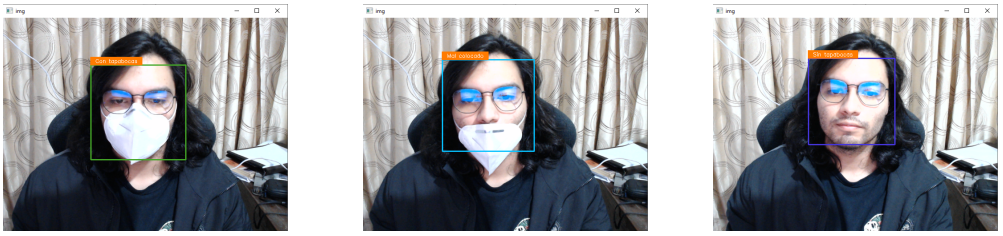


(a) 17:40:13.385996 (b) 17:40:31.316702 (c) 17:40:48.328174

Figura A.8: Prueba B - Irregularidades

A.2 Casos de detección

Casos de detección contemplados por el sistema.



(a) Con tapabocas (b) Mal colocado (c) Sin tapabocas

Figura A.9: Estados de detección del sistema

Detección de uso correcto de tapabocas empleando aprendizaje automático

Walter Iván Leturia Rodríguez

Universidad Internacional de la Rioja, Logroño (España)

22 de septiembre de 2021

RESUMEN

La pandemia por el virus COVID-19 ha causado estragos alrededor del mundo y la primera barrera de defensa es el uso correcto de tapabocas. El presente trabajo presenta una herramienta de software basada en aprendizaje automático para la detección de uso correcto, incorrecto y no uso de tapabocas. Se utilizaron algoritmos de detección rápida para la identificación de: rostro, nariz y boca. Asimismo, se entrenó e implementó una red neuronal de convolución para determinar la presencia o ausencia de tapabocas. Con las características extraídas, se realizan evaluaciones lógicas las cuales logran determinar los 3 casos de detección propuestos con una precisión de 97 %.

PALABRAS CLAVE

aprendizaje automático, detección de objetos, procesamiento de imagen

I. INTRODUCCIÓN

Es de conocimiento público la pandemia que se originó en el año 2020 y continúa causando estragos en este año 2021, con la propagación del virus COVID-19, el ritmo y rutina de las personas no es el mismo. La implementación y cumplimiento de protocolos de bioseguridad es esencial para la contención de este virus. El mal uso de tapabocas/mascarillas está directamente relacionado al contagio, asimismo el incumplimiento de los protocolos de distanciamiento establecidos por la OMS [1]. Sin embargo, en Perú se hace caso omiso a los protocolos de distanciamiento y es frecuente observar el uso incorrecto de tapabocas o inclusive el inexistente uso de estos.

El presente trabajo se descompone en 2 fases principales: identificación de uso de tapabocas en un rostro e identificación de correcto uso de tapabocas en un rostro. Se proporciona un modelo que contempla el uso correcto, uso incorrecto y el no uso de tapabocas. Para lograr estos objetivos, se hacen uso de filtros morfológicos, detección rápida (haar) y la aplicación de un modelo de clasificación de imágenes, entrenado mediante técnicas de aprendizaje automático, para la identificación de tapabocas.

El modelo fue entrenado con diferentes tipos de tapabocas, incluyendo: higiénicos, quirúrgicos, FPP2 y FPP3. Las pruebas fueron realizadas en un entorno local, separado en dos grupos donde se obtuvieron métricas para evaluar: precisión, exhaustividad, F1, exactitud y tiempo de detección.

El presente artículo está organizado con la siguiente estructura: El capítulo II *Estado del arte* realiza un repaso de los trabajos destacados referente a la identificación de tapabocas utilizando técnicas de aprendizaje automático para la identificación de su uso en la coyuntura COVID.

Los objetivos y metodología que direccionaron este trabajo están desarrollados en el capítulo III *Objetivos y metodología*.

Los resultados obtenidos se encuentran en el capítulo V *Resultados o evaluación* y la discusión de los mismos en el capítulo VI *Discusión*

o análisis de resultados. Finalmente, el capítulo VII *Conclusiones* realiza la recopilación de conclusiones del trabajo desarrollado.

II. ESTADO DEL ARTE

Con la evolución de la tecnología, el reconocimiento digital ha tomado protagonismo en tareas que antes requerían un mayor esfuerzo e inclusive la necesidad de personal para llevarse a cabo. Tareas como: control de asistencia de personal, identificación de objetos anómalos en líneas de producción e inclusive traducción de textos a partir de una imagen.

A. Aprendizaje automático

Es un método de análisis de datos que automatiza la construcción de modelos analíticos [2]. Pertenece a la rama de inteligencia artificial y se basa en la idea de que los sistemas son capaces de aprender de los datos, identificando patrones y tomando decisiones con la mínima intervención humana posible. Este proceso es comparable al de minería de datos, con la diferencia de que en vez de extraer los datos para comprensión humana se extraen para comprensión del ordenador [3].

Según su naturaleza los algoritmos de aprendizaje automático pueden clasificarse en supervisados y no supervisados. Los algoritmos de aprendizaje supervisado requieren de una colección de datos etiquetada (datos históricos previamente clasificados) mientras que los algoritmos basados en aprendizaje no supervisado son capaces de extraer inferencias de las colecciones.

B. Reconocimiento de Objetos

La clasificación de imágenes o reconocimiento de objetos, consiste en identificar un objeto específico en una imagen o vídeo [4]. El objetivo principal que busca satisfacer el reconocimiento de objetos es permitir ser representado y "comprendido" por una máquina.

Con el avance de la tecnología se ha logrado aplicar la detección de objetos desde tareas de detección realizadas por un vehículo autónomo hasta rastreo de objetos enfocado a detección de un balón en partidos de fútbol [5]. Una de las formas para reconocer objetos, priorizando la velocidad es el método de clasificadores simples en cascada [6].

El aprendizaje profundo ha revolucionado la manera de resolver problemas que involucran visión artificial y detección de objetos [7]. Esta tendencia aumenta desde el éxito de AlexNet [8], al utilizar una red neuronal de convolución en el reto ImageNet. Los modelos recientes sugieren que el éxito de estas redes está en aumentar la cantidad de capas [9, 10, 11, 12]. Utilizar redes neuronales convolucionales es la mejor opción al momento de abordar problemas que involucran imágenes y vídeos en el campo de la inteligencia artificial.

C. Procesamiento de imágenes

De acuerdo con [13], las principales áreas del procesamiento de imagen digital son: mejorar la información para interpretación humana y procesar la información para sistemas autónomos. Una imagen puede representarse en un plano de dos dimensiones, donde cada punto (píxel) tiene una coordenada x y una coordenada y. Siguiendo este enunciado, podemos ejecutar operaciones matemáticas en la imagen para mejorar/anular características. Para procesamiento de imágenes con python existen diversas librerías [14, 15, 16, 17, 18, 19] las cuales facilitan la aplicación de: filtros, métodos y kernels; para la transformación de imágenes [20, 21].

D. Reconocimiento de tapabocas

Los trabajos que empleen detección de imágenes en el campo de aprendizaje automático, como se ha mencionado en la sección anterior, son abordados en su mayoría a través de redes neuronales de convolución y aprendizaje profundo. Esto se debe a la principal ventaja que este diseño de redes sustenta ante estos escenarios de clasificación. Específicamente en el escenario de reconocimiento de tapabocas ante la coyuntura COVID-19 los trabajos desarrollados resaltan la necesidad de identificación de tapabocas en una persona (específicamente en

un rostro) [22, 23, 24], otros trabajos añaden nuevas variables como distanciamiento social [25]. Los trabajos mencionados utilizan utilizan modelos basados en: CNN, RCNN, transferencia de aprendizaje y aprendizaje profundo.

III. OBJETIVOS Y METODOLOGÍA

A. Objetivo General

Desarrollar un software para el reconocimiento de uso correcto, incorrecto y no uso de tapabocas empleando aprendizaje automático para el reconocimiento de imágenes y técnicas de procesamiento de imagen.

B. Objetivos específicos

- Realizar un análisis del estado del arte en sistemas de reconocimiento de objetos y técnicas de procesamiento de imagen, así como los criterios de aceptación que deberá tener el algoritmo de detección de uso correcto, incorrecto y no uso de tapabocas.
- Analizar y preparar los conjuntos de datos con los que se entrenarán los modelos de convolución para detección de tapabocas.
- Implementar técnicas de procesamiento de imágenes y rostros para delimitar las zonas de detección relevantes involucradas en la detección del uso correcto, uso incorrecto y no uso de tapabocas.
- Modelar y entrenar una red neuronal de convolución para la correcta detección de uso de tapabocas.
- Establecer las reglas que satisfacen los estados de salida del sistema en cuanto a identificación de rostro y uso de tapabocas.

C. Metodología del trabajo

Con la finalidad de lograr los objetivos planteados en el presente trabajo, se utilizó una de las metodologías más comunes para proyectos de ésta índole (aprendizaje automático).

CRISP-DM, del inglés *CRoss-Industry Standard for Data Mining*. Los procesos que aborda CRISP-DM están desarrollados bajo 6 fases.

- Entendimiento del tema; establecer los objetivos del proyecto e identificar los requerimientos de desarrollo
- Entendimiento de la información; recopilación de conjuntos de datos y herramientas disponibles para el desarrollo del proyecto
- Preparación de datos; realizar las transformaciones necesarias a los datos
- Modelado; enfocado al entrenamiento del modelo de clasificación
- Evaluación; escenarios de prueba necesarios y las métricas para determinar si el modelo es satisfactorio o no
- Producción; consolidación de la red neuronal entrenada

IV. CONTRIBUCIÓN

Un software de reconocimiento de uso correcto, incorrecto y no uso de tapabocas. Hace uso de 1 cámara de video para lograr los estados deseados. Este proyecto ha entrenado una red neuronal de convolución para la identificación de tapabocas, posteriormente extracción de características faciales con técnicas de procesamiento rápido de imágenes. Permitiendo que, a través, de una secuencia lógica se determine si existe un buen uso, mal uso o simplemente no se encuentre un tapabocas. La herramienta software ha sido desarrollada en el lenguaje Python, con las librerías de OpenCV (procesamiento de imagen) y Keras (red neuronal de convolución).

A. Captura y análisis de datos

La captura de imágenes se obtiene en formato RGB con una resolución 1080p, fueron realizadas con una cámara web ubicada a una

distancia de 50cm 15+- y un ambiente iluminado.

La lectura y procesamiento de imágenes se realiza con la librería OpenCV [15], de propósito general, útil con la transformación de imágenes [20], aplicación de filtros en cascada [26] y filtros de umbral [21].

B. Entrenamiento del modelo de clasificación

Se realizó el entrenamiento de un modelo de aprendizaje para el reconocimiento de tapabocas a partir de una imagen proporcionada. Red neuronal de convolución y clasificación binaria, las categorías son: con tapabocas, sin tapabocas.

B.1. Selección y análisis de conjuntos de datos

Las imágenes utilizadas para el entrenamiento fue una recopilación propia a partir del conjunto de datos **Mask Dataset** [27], el cual contaba con "anchor boxes" las cuales fueron utilizadas para recortar las imágenes que presentaban múltiples objetos de reconocimiento. Las clases consideradas fueron: "with_mask" y "without_mask".

Se enriqueció el conjunto de datos con la librería **MaskTheFace** [28], cambiando el tapabocas presente en las imágenes de la categoría "with_mask" con un tapabocas de tipo FPP2/FPP3.

En total se recopiló un conjunto de datos de 1508 imágenes con formato .jpg con diferentes variaciones de tamaño (píxeles). De estas, 1206 imágenes fueron utilizadas para entrenamiento y 302 para validación.

B.2. Entrenamiento

El entrenamiento del modelo de aprendizaje fue realizado con la librería **Keras** [29] que utiliza el framework TensorFlow [30].

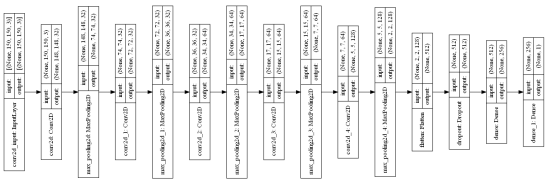
El diseño de red neuronal con el que se obtuvo mejores resultados se encuentra en la figura 1

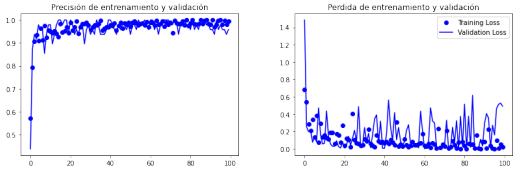
El modelo está compuesto por 5 capas convolucionales de 32*2, 64*2 y 128 filtros, con filtros de 3x3. Seguido a las capas convolucionales se encuentra una capa Flatten. Las 2 capas finales son de tipo densas, para la primera se utilizó una función de activación ReLU y para la capa final una función de activación Sigmoid.

Para regularizar la red se utilizó la capa MaxPooling ubicada después de cada convolución y una capa Dropout ubicada después de la capa Flatten del modelo.

Se consideró la función de entropía cruzada binaria como la función de pérdida más adecuada para esta red neuronal. Se utilizó con el optimizador RMSprop para el entrenamiento del modelo. Se realizaron 100 épocas (epochs) con 64 imágenes por lote (batch).

Figura 1: Modelo de red neuronal de convolución





(a) Precisión de entrenamiento y validación (b) Pérdida de entrenamiento y validación

Figura 2: Precisión y pérdida de entrenamiento y validación

B.3. Serialización

Finalizado el entrenamiento y consolidación del grafo de la red neuronal, se exporta el modelo en un formato .h5 con la función *save* de Keras. Este archivo permite la exportación del modelo de clasificación desde el módulo de integración.

C. Detección de características

Las imágenes capturadas por el puerto serial deben ser preparadas para la detección de

tapabocas y características necesarias que determinan la correcta posición del tapabocas.

C.1. Procesamiento de imagen

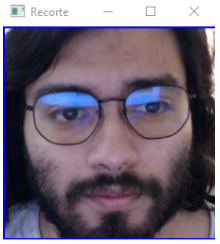
Cada imagen será procesada y tendrá una clasificación independiente (de 60 fotogramas se tendrán 60 clasificaciones). Para esto se realizaron transformaciones a la imagen con la finalidad de disminuir el tiempo de preprocesamiento (sin perder características esenciales). Apoyado en la librería OpenCV se realizaron las transformaciones de: filtro gaussiano y escala de grises.

Figura 3: Procesamiento de imagen

C.2. Detección de rostro

Para la detección de rostro se utiliza la imagen anterior en la secuencia (escalada en grises y con filtro gaussiano), a esta se le aplica un clasificador de tipo cascada, para el cual se han ajustado los parámetros de: scaleFactor y minNeighbors. Este filtro retorna 4 puntos (coordenadas) de ubicación de objeto. Por motivos funcionales, se añade un *padding* para obtener un cuadrilátero mejor delimitado.

Figura 4: Detección de rostro



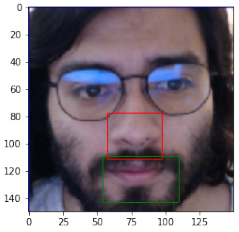
C.3. Detección de tapabocas

En caso exista una detección satisfactoria de uno o más rostros en una imagen se procesa el resultante del recorte. Este recorte es el resultante de: filtro gaussiano, escala de grises, recorte y suma de padding. El resultado viene de la aplicación de red neuronal entrenada, la cual considera: 0 - Con tapabocas y 1 Sin tapabocas.

C.4. Características de rostro

En caso exista una detección satisfactoria de tapabocas en el recorte previo, se procesa con dos filtros en cascada, uno para la detección de nariz y otro para la detección de boca. Esta salida resuelve cada una un arreglo de coordenadas, en caso no se encuentren coordenadas tanto para nariz o boca asume que el tapabocas se encuentra cubriendo correctamente el rostro, caso contrario se considera mal colocado.

Figura 5: Detección de nariz y boca empleando haar cascades



D. Estados de detección

Los estados de detección dependen de manera secuencial, las reglas lógicas establecidas son las siguientes:

- Estado 01: CT, detección de tapabocas sin detección de nariz o boca
- Estado 02: MC, detección de tapabocas y detección de nariz o boca
- Estado 03: ST, sin detección de tapabocas

De manera visual se sobrepone un rectángulo de color para cada uno de los estados y un texto informativo. Véase Figuras 6 y 7.

V. RESULTADOS

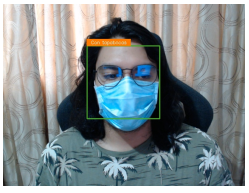
Se realizaron pruebas en 2 escenarios: Escenario A - tapabocas de tipo quirúrgico/higiénico y Escenario B - tapabocas de tipo FPP2 y FPP3

Los puntos de evaluación son: Duración de estados, Detección de rostros, Clasificación y Errores por transición de estado.

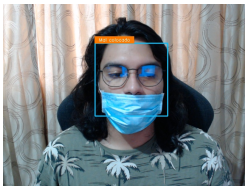
A. Evaluación A

La duración de prueba en este escenario sumó un total de **26 segundos**.

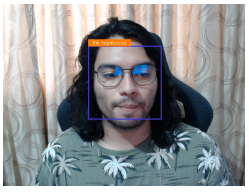
- Estado 01 - CT 9 Segundos
- Estado de transición 4 Segundos
- Estado 02 - MC 7 segundos
- Estado de transición 1 segundo
- Estado 03 - ST 5 segundos



(a) Estado 01



(b) Estado 02



(c) Estado 03

Figura 6: Prueba A - Estados de detección

A.1. Detección de rostros

Rostros detectados válidos y no válidos

	Válido	No Válido	
Rostros	413	4	417

Cuadro 1: Rostros detectados - Prueba A

A.2. Clasificación de estados

Matriz de confusión de los resultados obtenidos en la prueba

	CT	MC	ST	
CT	136	16	0	152
MC	15	156	3	174
ST	2	0	89	91

Cuadro 2: Matriz de confusión - Prueba A

A.3. Errores causados por transición entre estados

- E01 v E02: 13/16 contabilizados
- E02 v E01: 2/2 contabilizados
- E02 v E03: 3/3 contabilizados

A.4. Métricas

Tabla resultante de aplicar las métricas de **Precisión, Exhaustividad, F1 y Exactitud**

	Precis	Recall	F1	Acc
CT	0.89	0.89	0.89	0.921
MC	0.90	0.89	0.89	0.918
ST	0.97	0.98	0.975	0.988

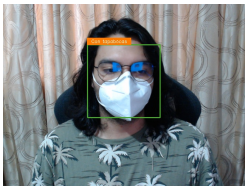
B. Evaluación B

La duración de prueba en este escenario sumó un total de **50 segundos**.

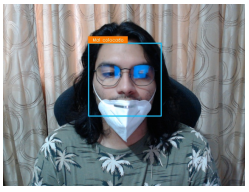
- Estado 01 - CT 20 Segundos
- Estado de transición 1 Segundos
- Estado 02 - MC 16 segundos
- Estado de transición 1 segundo
- Estado 03 - ST 12 segundos

B.1. Detección de rostros

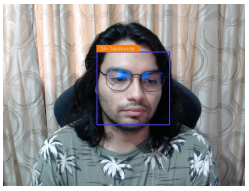
No se obtuvieron falsos positivos



(a) Estado 01



(b) Estado 02



(c) Estado 03

Figura 7: Prueba B - Estados de detección

B.2. Clasificación de estados

Matriz de confusión de los resultados obtenidos en la prueba

	CT	MC	ST	
CT	280	0	0	280
MC	11	266	2	279
ST	0	4	202	206

Cuadro 3: Matriz de confusión - Prueba B

B.3. Errores causados por transición entre estados

- E01 v E02: 7/11 contabilizados
- E02 v E03: 2/2 contabilizados
- E02 v E02: 1/4 contabilizados

B.4. Métricas

Tabla resultante de aplicar las métricas de **Precisión, Exhaustividad, F1 y Exactitud**

	Precis	Recall	F1	Acc
CT	0.96	1	0.979	0.985
MC	0.98	0.95	0.964	0.97
ST	0.99	0.98	0.984	0.992

VI. ANÁLISIS DE RESULTADOS

En cuanto a clases, la precisión más acertada se obtiene en la clase **ST** para ambos escenarios. Para las clases “CT” y “MC” el escenario B presenta mejor precisión.

En cuanto a clasificación, este modelo presenta un mejor desempeño en el escenario B. Donde ha logrado resultados superiores en cada una de las métricas comparadas con una varianza que oscila entre 0.04 - 0.06 (según la métrica).

En cuanto a tiempo de detección, se obtuvieron varianzas de: B (0.01), A (0.002). Sin embargo, los **tiempos ponderados fueron de 0.061 y 0.062** para el escenario de pruebas A - tapabocas de tipo quirúrgico y/o higiénico y B tapabocas de tipo FPP2 y FPP3 respectivamente. Concluyendo, que no existen diferencias significativas.

VII. CONCLUSIONES

La herramienta software desarrollada en este proyecto permite, a partir de imágenes capturadas por una cámara, determinar si una persona cuenta con un tapabocas bien colocado, mal colocado o si no cuenta con tapabocas. Esto con el objetivo de ser una herramienta útil para la prevención ante la propagación del virus SARS-CoV-2.

Se recopilaron conjuntos de datos a los cuales se adaptaron para el desarrollo del proyecto. En cuanto al entrenamiento se aplicó la técnica de *Data Augmentation* y la distribución para entrenamiento y validación fue de 80 % y 20 %.

Para delimitar rostro se hizo uso de filtro blanco y negro, clasificador en cascada (modificando vecinos cercanos y factor de escalado) y recorte de imagen. En este proceso se obtiene una delimitación visual (rectangulos sobrepuestos) y una delimitación lógica la cual se obtiene a través del recorte de imagen y es la entrada para la red neuronal de convolución que detecta tapabocas.

De los diferentes modelos iterados, se adjunta el diagrama y métricas del modelo con me-

jores resultados 1 2. El modelo ofrece 2 salidas: 0 - Con tapabocas y 1 - Sin tapabocas.

Reconocimiento de uso correcto, se definieron reglas lógicas que satisfacen la premisa "nariz y boca deben quedar correctamente cubiertas". Se hizo uso de clasificadores de cascada individuales.

Evaluación final, el proyecto ha logrado alcanzar exactitudes de *accuracy* $\simeq$ 0,98 en los escenarios de prueba, demostrando robustez del algoritmo final de detección. Presenta mejores resultados cuando se trata de tapabocas de tipo FPP2 y FPP3, donde no se obtuvieron falsos positivos.

Este proyecto ha logrado implementar satisfactoriamente un estado de detección útil en cuanto a la prevención ante la propagación del virus COVID-19, el estado "uso incorrecto de tapabocas". El cual es el aporte diferencial ante los antecedentes analizados [23, 22, 24, 25].

En cuanto a velocidad de detección, este proyecto ha logrado tiempos de inferencia promedio de $\simeq$0.06s, no logrando superar los tiempos de la investigación [22] que obtuvo 0.045s en su implementación con YOLOv3 pero si logra superar el tiempo del modelo implementado con R-CNN de 0.15s.

Como líneas de trabajo futuro se sugiere lo siguiente:

- Considerar la perspectiva lateral de un rostro para realizar la identificación de estados.
- Realizar pruebas de campo en un ambiente controlado (ej. una cabina de desinfección) para observar el funcionamiento integral el este software en un escenario real.
- Permitir ajustar los parámetros de los componentes basados en detección tipo cascada.
- Considerar una detección incluyendo el tipo de mascarilla.

Referencias

[1] World Health Organization. Preguntas y respuestas sobre la enfermedad por coronavirus (COVID-19), 2020. Recuperado 15 de abril de 2021, a partir de `https://www.who.int/es/emergencies/diseases/novel-coronavirus-2019/advice-for-public/q-a-coronaviruses`.

[2] SAS. Machine learning what it is and why it matters, 2021. Recuperado 16 de junio de 2021, a partir de `https://www.sas.com/en_us/insights/analytics/machine-learning.html`.

[3] TechTarget. Aprendizaje automático (machine learning), 2017. Recuperado 16 de junio de 2021, a partir de `https://searchdatacenter.techtarget.com/es/definicion/Aprendizaje-automatico-machine-learning`.

[4] MathWorks. ¿Qué es el reconocimiento de objetos? Tres cosas que es necesario saber, 2020. Recuperado 18 de abril de 2021, a partir de `https://la.mathworks.com/solutions/image-video-processing/object-recognition.html`.

[5] Pixel Solutionz. Application of object detection in real life, 2020. Recuperado 16 de junio de 2021, a partir de `https://www.pixelsolutionz.com/application-object-detection-real-life`.

[6] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, 2001.

[7] Soren Goyal and Paul Benjamin. Object recognition using deep neural networks: A survey, 2014.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural net-

works. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014. cite arxiv:1405.0312Comment: 1) updated annotation pipeline description and figures; 2) added new section describing datasets splits; 3) updated author list.

[10] A. Krizhevsky. Learning multiple layers of features from tiny images, 2009.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[12] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/, 2010.

[13] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Pearson/Prentice Hall, 2008.

[14] Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors. scikit-image: image processing in Python. *PeerJ*, 2:e453, 6 2014.

[15] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[16] Luis Pedro Coelho. Mahotas: Open source software for scriptable computer vision. *Journal of Open Research Software*, 1, July 2013.

[17] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[18] Alex Clark. Pillow (pil fork) documentation, 2015.

[19] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[20] OpenCV. Morphological transformations, 2021. Recuperado 18 de abril de 2021, a partir de `https://docs.opencv.org/master/d9/d61/tutorial_py_morphological_ops.html`.

[21] OpenCV Docs. Image thresholding, 2021. Recuperado 18 de abril de 2021, a partir de `https://docs.opencv.org/master/d7/d4d/tutorial_py_thresholding.html`.

[22] Sunil Singh, Umang Ahuja, Munish Kumar, Krishan Kumar, and Monika Sachdeva. Face mask detection using yolov3 and faster r-cnn models: Covid-19 environment. *Multimedia Tools and Applications*, 2021. Recuperado a partir de `https://doi.org/10.1007/s11042-021-10711-8`.

[23] Ramot Lubis. MACHINE LEARNING (CONVOLUTIONAL NEURAL NETWORKS) FOR FACE MASK DETECTION IN IMAGE AND VIDEO, 2020.

Recuperado 10 de mayo de 2021, a partir de http://eprints.binus.ac.id/id/eprint/36805.

[24] Mohamed Loey, Gunasekaran Manogaran, Mohamed Hamed N. Taha, and Nour Eldeen M. Khalifa. A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the covid-19 pandemic. *Measurement*, 167:108288, 2021.

[25] Shashi 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*, 8(7):1368–1375, July 2020.

[26] OpenCV Docs. Face detection using haar cascades, 2021. Recuperado 18 de abril de 2021, a partir de https://docs.opencv.org/4.5.3/d2/d99/tutorial_js_face_detection.html.

[27] MakeML. Mask dataset, 2020. Recuperado 10 de mayo de 2021, a partir de https://makeml.app/datasets/mask.

[28] Aqeel Anwar and Arijit Raychowdhury. Masked face recognition for secure authentication. https://github.com/aqeelanwar/MaskTheFace, 2020.

[29] François Chollet et al. Keras. https://keras.io, 2015.

[30] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Recuperado 18 de abril de 2021, a partir de www.tensorflow.org.