Sistema para la detección de uso de mascarilla utilizando técnicas de redes neuronales convolucionales.

Chiza Llambo, Juan Carlos

Vicerrectorado de Investigación, Innovación y Transferencia de Tecnologías

Centro de Posgrados

Maestría en Electrónica Mención Redes Industriales

.

Trabajo de Titulación, Previo a la Obtención del Título de Magister en Electrónica Mención Redes Industriales

Ing. Eddie Galarza Zambrano Mgs.

28 / Abril / 2022

Latacunga

AGENDA

- Introducción, objetivos e hipótesis
- Desarrollo del proyecto
- Redes neuronales artificiales convolucionales
- Base de datos y entorno de programación Entorno
- Pruebas y resultados
- Conclusiones
- Recomendaciones

Introducción

El presente trabajo tiene como finalidad diseñar un sistema que permita verificar en las personas el uso de mascarillas con ayuda de visión artificial y redes neuronales convolucionales, por esta razón y a las condiciones presentes de la pandemia, desarrollar un sistema que identifique el uso de mascarillas utilizando la clasificación de imágenes y el método de aprendizaje profundo (DL) para clasificar y distinguir dos posibilidades básicas si están o no utilizando mascarillas.

Principales aplicaciones del las redes neuronales artificiales:

- Reconocimiento de imágenes
- Clasificación de tejido invasivo de cáncer de seno
- Detección y clasificación de la mitosis
- Detección de bacterias Escherichia Coli en verduras frescas
- Reconocimiento de caracteres manuscritos

Objetivos

GENERAL:

Diseñar e implementar un sistema de detección de uso de mascarillas utilizando técnicas de redes neuronales convolucionales.

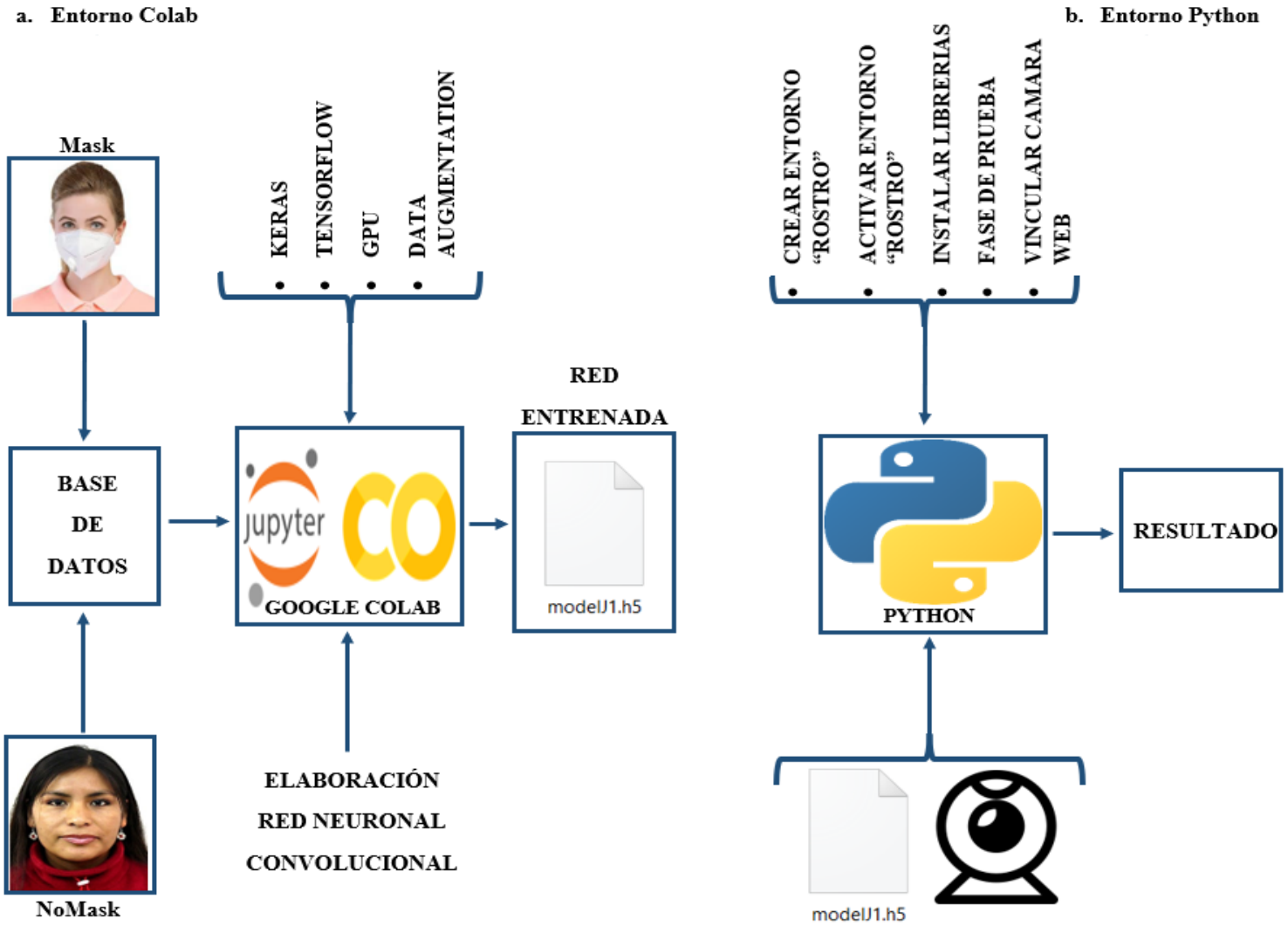
ESPECÍFICOS:

- Obtener información necesaria de redes neuronales convolucionales.
- Adquirir conocimientos sobre el procesamiento y las características de las imágenes digitales.
- Estudiar el funcionamiento de la librería aprendizaje profundo DL en entorno de programación de software libre.
- Recolectar imágenes de rostros con y sin mascarillas para la Data base de entrenamiento, prueba y validación con su respectivo etiquetado.
- Diseñar la red neuronal convolucional.
- Entrenar al sistema de detección de uso de mascarillas para que pueda reconocer el uso de tapa boca.

Hipótesis

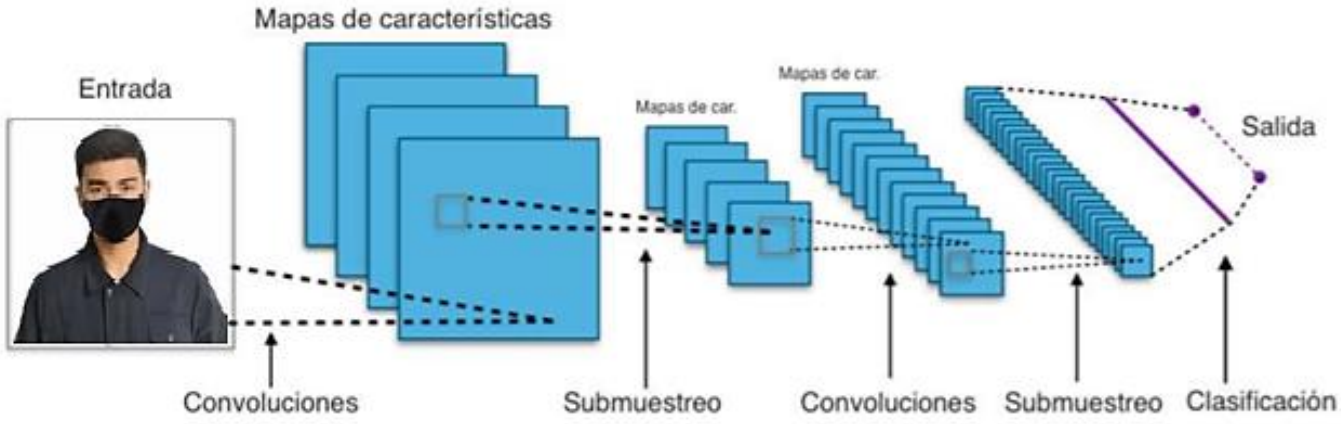
La implementación del sistema de detección de objetos en imágenes utilizando técnicas de redes neuronales convolucionales permitirá identificar si una persona utiliza tapa boca.

DESARROLLO DEL PROYECTO



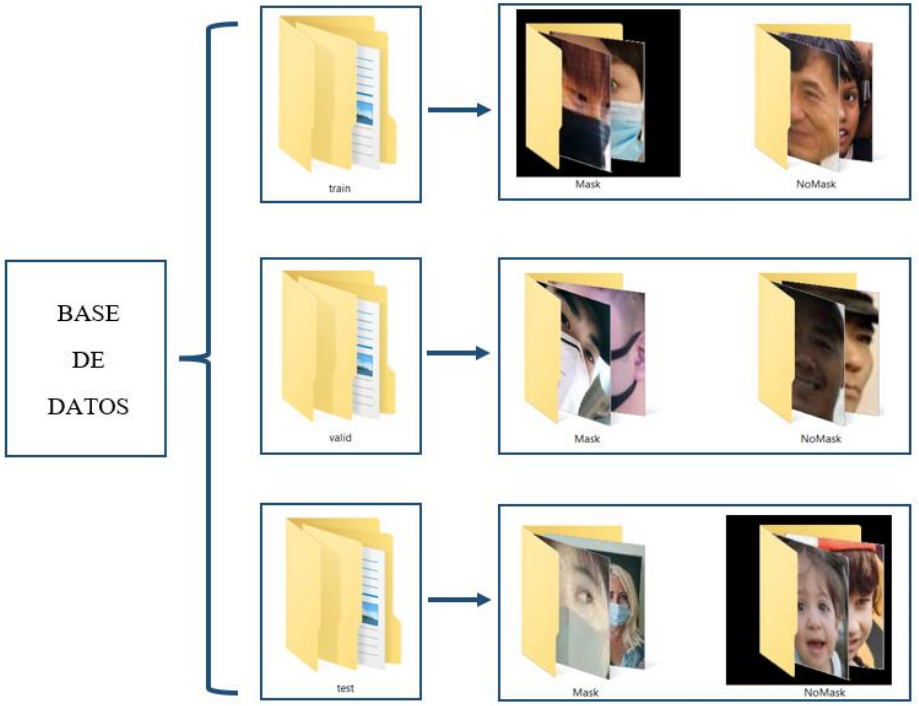
RNC

Las redes neuronales convolucionales su principal característica es que presentan una arquitecta multicapa, en donde capa está sujetada a una cantidad determinada de convoluciones con activaciones no lineales (Relu, Tanh o sigmoidal) y pooling (submuestreo) al final, tiene una serie de capas completamente conectadas como una red de perceptrón multicapa. Este tipo de conexiones multicapas convolucionales es típicamente una analogía de la corteza cerebral de un mamífero



BASE DE DATOS

- Total de 3194 imágenes con y sin mascarilla.
- Dividida en tres carpetas *Train, Valid y Test* con el *70%, 20% y 10%* respectivamente.



ENTORNO GOOGLE COLAB

Google Colab o Google Colaboratory es un entorno de programación en línea que permite a cualquier cliente digitar y ejecutar comandos de Python en el navegador.

Es un entorno adecuado para labores de aprendizaje profundo (Deep Learning), estudio de datos y educación.

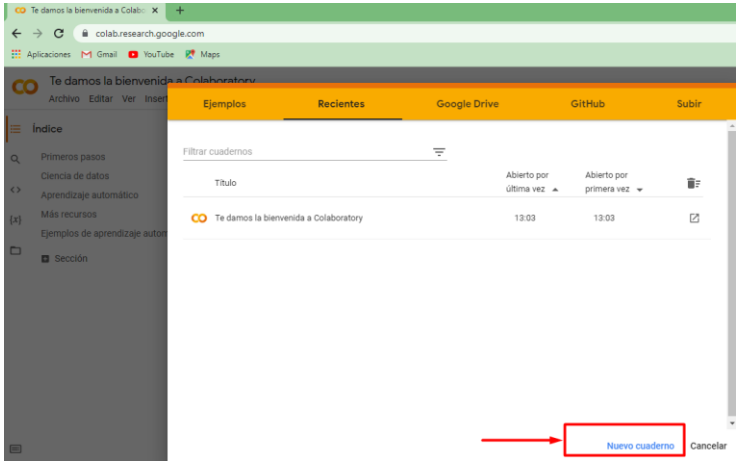
Colab es un servicio portátil que no necesita instalación, ideal para el entrenamiento de una red neuronal, estos entornos de ejecución o acelerador de hardware pueden ser:

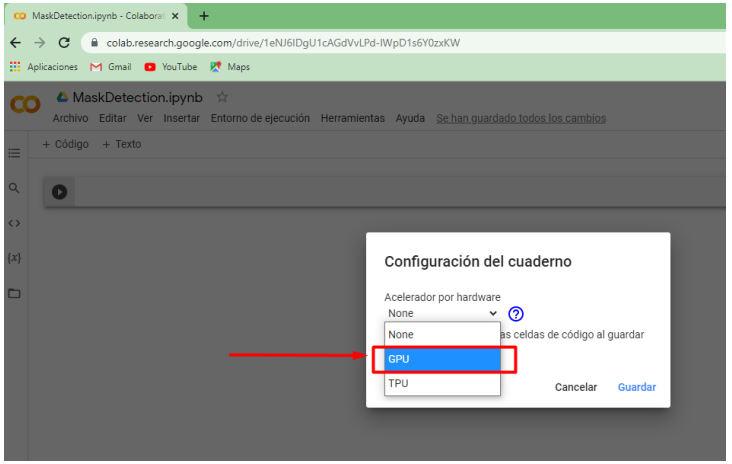
- GPUs (Graphics Processor Unit)
- TPUs (Tensor Processing Unit)

Los mismos que acortan el tiempo de entrenamiento a diferencia de usar una tarjeta convencional CPUs (Central Processing Unit).

ENTORNO GOOGLE COLAB

Creación y configuración de entorno





Fragmento de programación

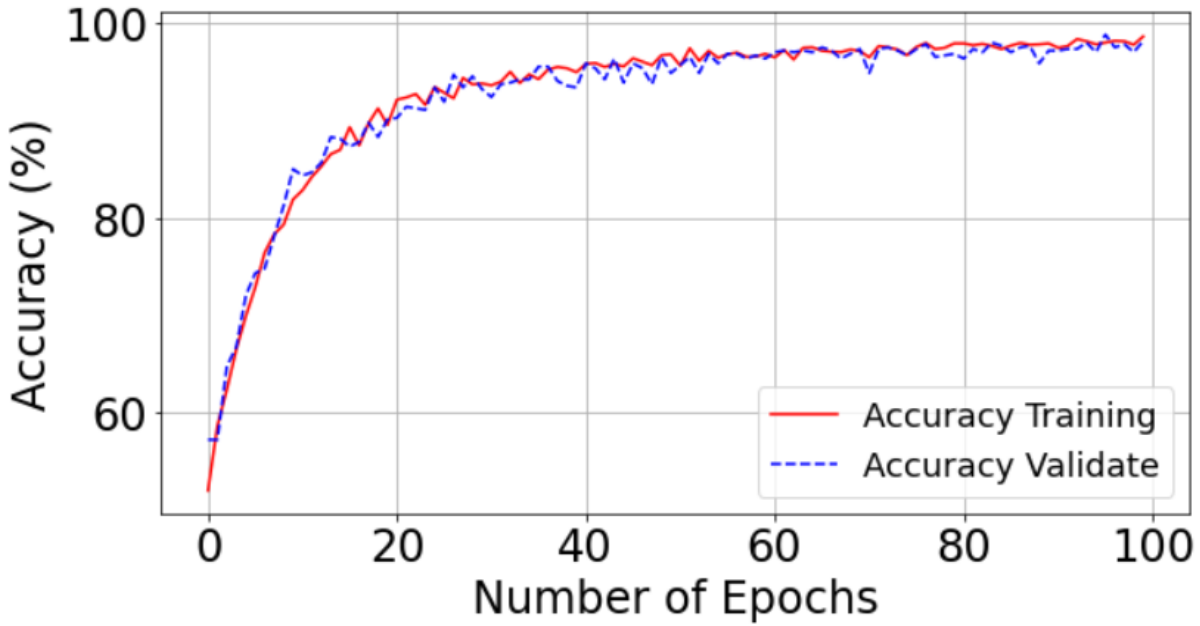
```
from keras.applications.vgg16 import VGG16

image_input = Input(shape=(width_shape, height_shape, 3))
model2 = VGG16(input_tensor=image_input, include_top=True,weights='imagenet')
#model2.summary()
last_layer = model2.get_layer('block5_pool').output
x= Flatten(name='flatten')(last_layer)
x = Dense(128, activation='relu', name='fc1')(x)
x = Dense(128, activation='relu', name='fc2')(x)
out = Dense(num_classes, activation='softmax', name='output')(x)
model = Model(image_input, out)
#model.summary()
# congelar todas las capas excepto las densas
for layer in model.layers[:-3]:
  layer.trainable = False
model.summary()

model.compile(loss='categorical_crossentropy',optimizer='adadelta',metrics=['accuracy'])
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels.h5
553467904/553467096 [==============================] - 8s 0us/step
Model: "model"
```

ENTORNO GOOGLE COLAB

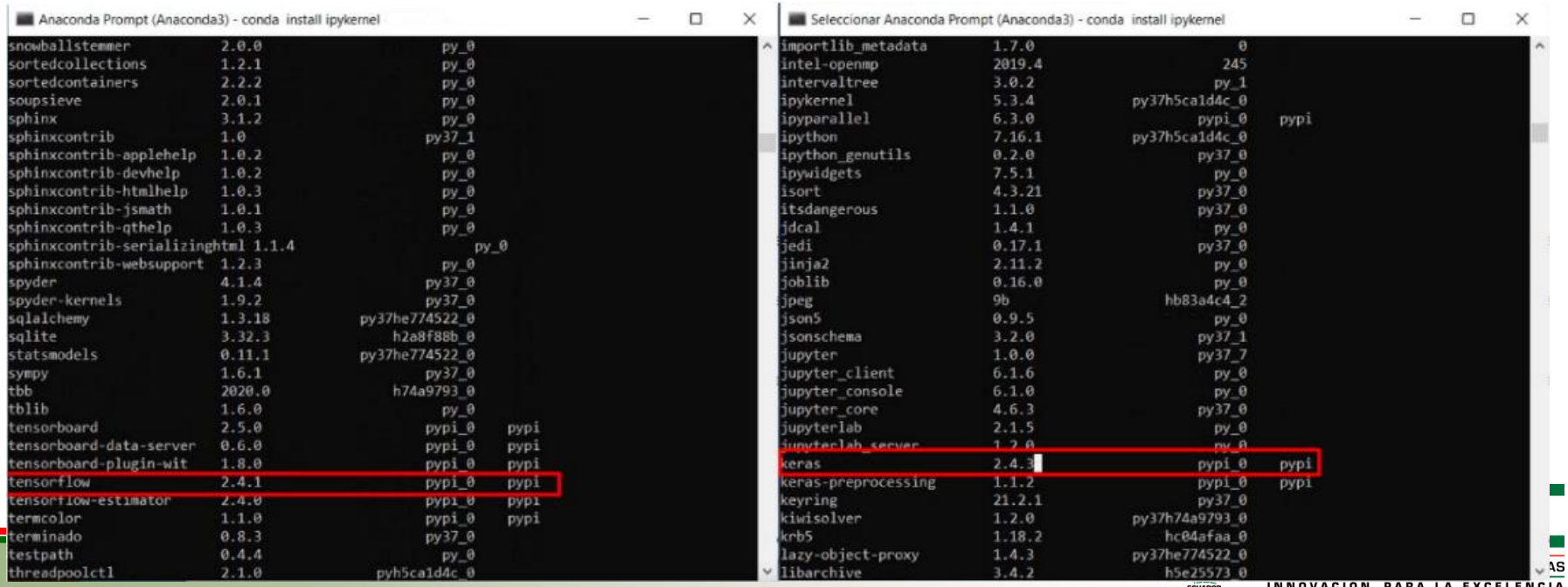
Entrenamiento del Sistema

Después de acciones de prueba y error se define el número de épocas (ciclo de entrenamiento) en 100 para mejorar la precisión y acercar la curva de predicción al valor de 1 que es equivalente al 100% como se observa en la Figura



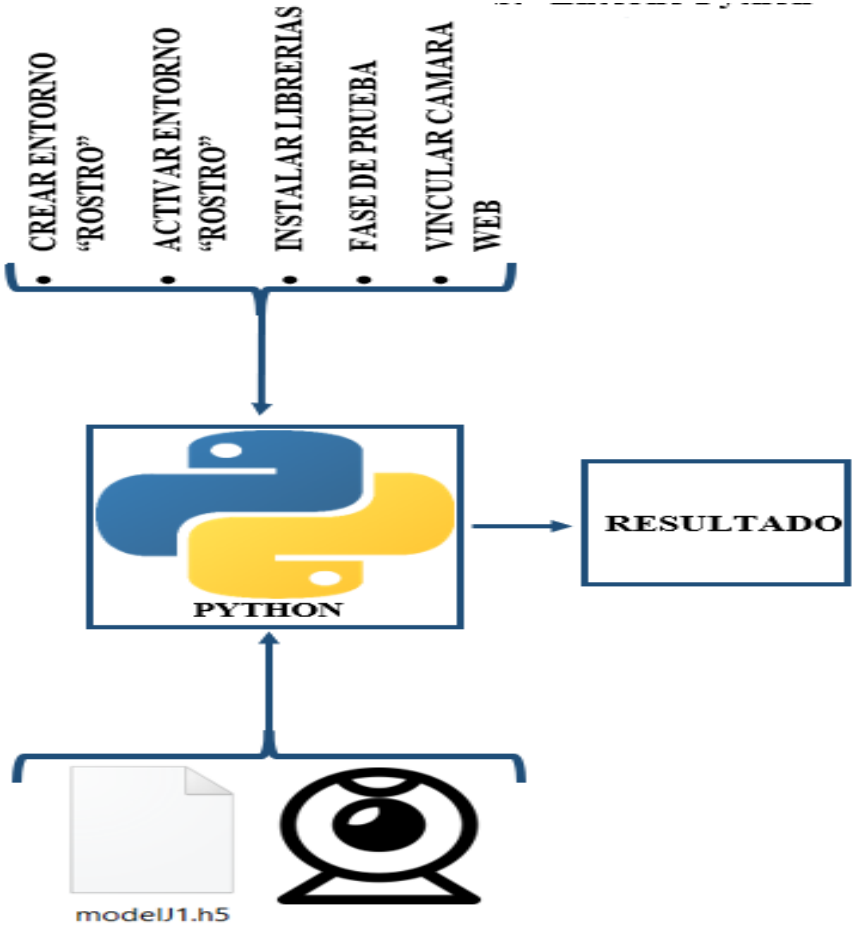
ENTORNO PYTHON

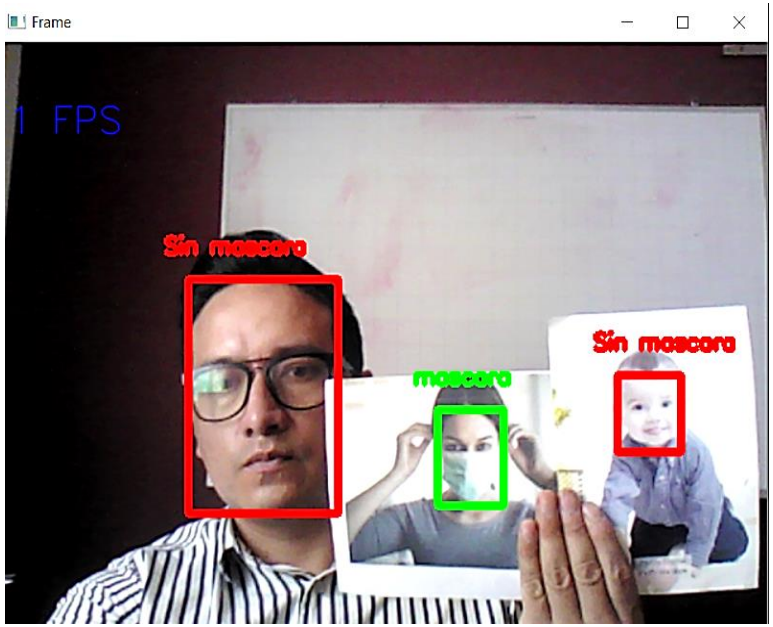
Python es un entorno de programación que se encuentra presente en diversas aplicaciones y para diferentes plataformas como iOS, Android, Linux, Windows o Mac. Python es un lenguaje de programación orientado a objetos, funcional e imperativo, por lo que se denomina un lenguaje multiparadigma y multiplataforma en donde predomina por su código legible y limpio. Se instala la paquetería y las versiones necesarias.



ENTORNO PYTHON

Esquema y respuesta del sistema de detección





PRUEBAS Y RESULTADOS

Matriz de confusión

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

$$Presición = \frac{TP}{TP + FP}$$

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

True Positives (TP)

True Negative (TN)

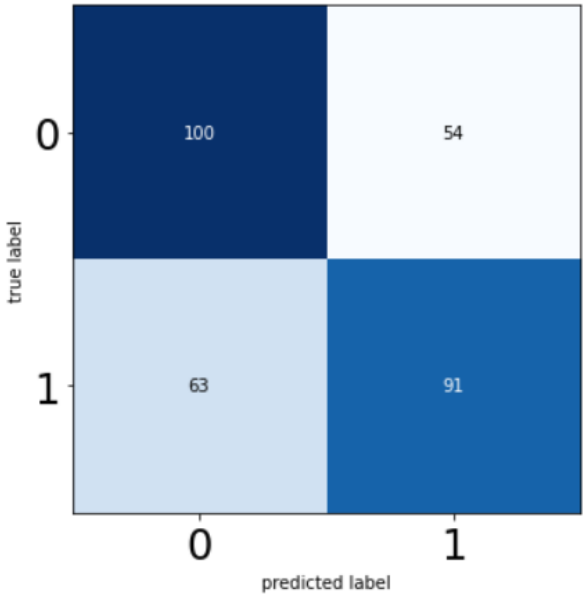
False Positives (FP)

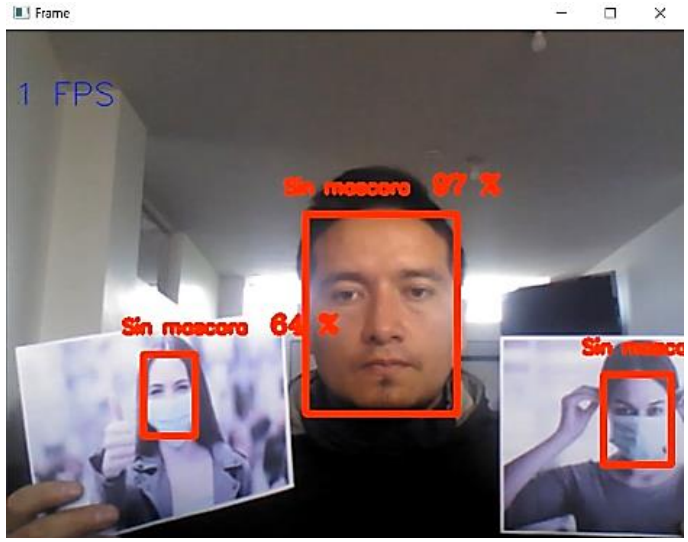
False Negative (FN)

$$F1 - Score = \frac{2 * (Recall * Precisión)}{Recall + Precisión}$$

PRUEBAS Y RESULTADOS

PARA 25 ÉPOCAS

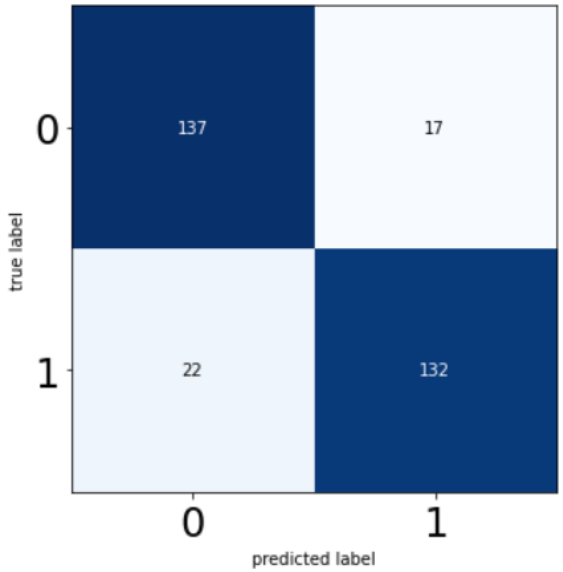


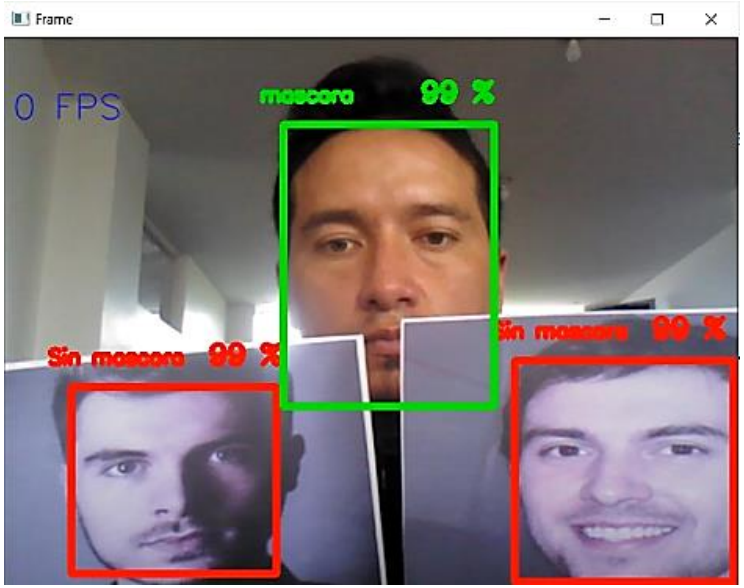


	PRECISIÓN	SUPPORT
0	0.6135	154
1	0.6276	154
Accurracy	0.6201	308

PRUEBAS Y RESULTADOS

PARA 50 ÉPOCAS

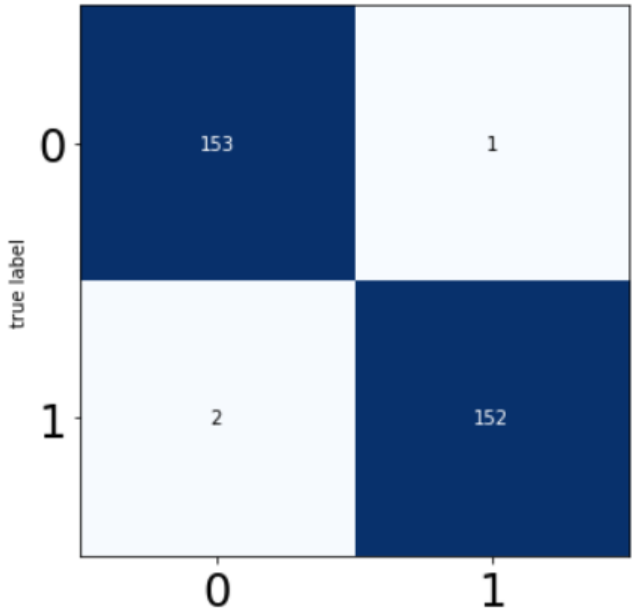


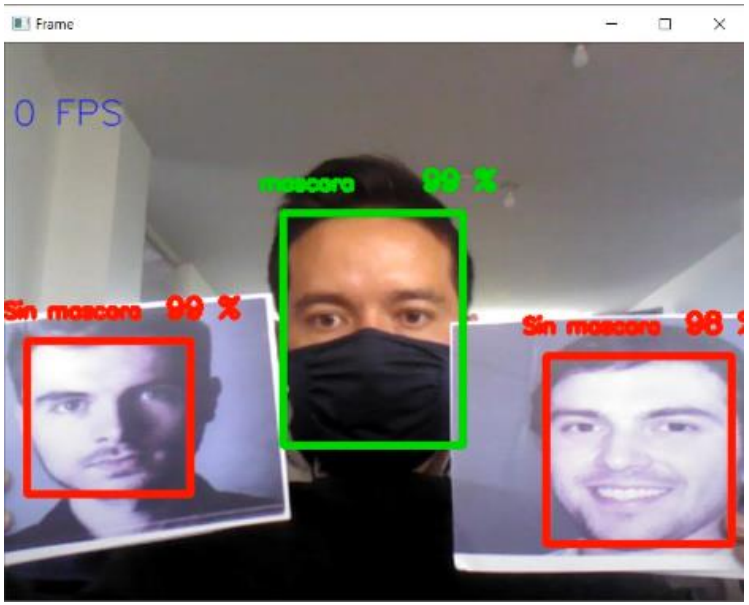


	PRECISIÓN	SUPPORT
0	0.8616	154
1	0.8859	154
Accurracy	0.8734	308

PRUEBAS Y RESULTADOS

PARA 100 ÉPOCAS





	PRECISIÓN	SUPPORT
0	0.9871	154
1	0.9935	154
Accurracy	0.9903	308

CONCLUSIONES

- El sistema se desarrolló en dos etapas, la etapa de entrenamiento la cual se realizó totalmente en la plataforma Google Colab con una base de datos de 3194 imágenes divididas en train, valid y test con el 70, 20 y 10% respectivamente y la segunda etapa se desarrolló en software libre Anaconda Navigator la cual permite aplicaciones, entornos y paqueterías dedicados al desarrollo en lenguaje Python, como editor de código se utiliza Jupyter Notebook y sus diferentes paqueterías para el desarrollo de la programación.
- Una de los beneficios primordiales al instante de ejecutar un modelo de Machine Learning y Deep Learning con Google Colab es que posibilita la implementación de una GPU que es muchísimo más veloz que en una CPU. La GPU resulta muchísimo más potente en hacer cálculos matemáticos por ejemplo descenso de gradiente, multiplicaciones matriciales o el Backpropagation de una red neuronal, es decir, algoritmos que anteriormente tomaban días en entrenar, pero ahora se resuelven en horas.
- La implementación del algoritmo para la detección de uso de mascarilla en tiempo real utilizando procesamiento de imágenes y redes neuronales convolucionales opera de manera efectiva debido a que permite extraer y categorizar gracias al entrenamiento realizado en Google Colab, para esto la imagen de entrada debe estar en dimensiones de 224 x 224.

CONCLUSIONES

- La arquitectura de una red neuronal está basada en la red escogida para entrenar, para este proyecto la red es la Vgg-16 que contiene 16 capas (3 capas densas y 13 capas convolucionales) las mismas que son preentrenadas con la base de datos de ImageNet para una clasificación de 1000 clases, debido a su estructura implica menos memoria, menos requerimientos computacional y por lo tanto menos tiempo de entrenamiento.
- La red neuronal convolucional implementada consta de 16 capas plenamente conectadas entre sí, para la cual se ocupa el procedimiento de aprendizaje supervisado. Los resultados se evaluaron por medio de la matriz de confusión, en la cual se obtuvo un porcentaje de predicción del 99.03%, un porcentaje elevado para la categorización de uso de mascarilla.

RECOMENDACIONES

- Se requiere de una base de datos la cual contenga un gran número de imágenes y las dos categorías mascarilla y sin mascarilla. Además, estas imágenes deben ser claras e indicar la región de interés a predecir, para este caso las imágenes recolectadas son imágenes que contienen solo el rostro, esto optimiza los puntos de interés que debe considerar la red neuronal al entrenar.
- Es necesario el uso de un equipo que contenga una tarjeta gráfica, de ser posible una NVIDIA CUDA, debido que una GPU resume el tiempo de entrenamiento y los procesos computacionales matemáticos de la red neuronal.
- En la situación de usar cámaras web de más grande resolución para el análisis de imágenes, se debe considerar que la cámara no presente configuraciones de alta resolución, debido que al exponerlo a exteriores las imágenes transmitidas están saturadas en color imposibilitando la ubicación del rostro y la predicción.

MUCHAS GRACIAS