**INSTITUTO TECNÓLOGICO DE LA PAZ**

**DIVISIÓN DE ESTUDIOS PROFESIONALES**

**Reporte semanal de residencias Semana 05**

**“Reconocimiento y seguimiento de objetos en entornos controlados”**

**CARRERA**

**Ingeniería en Sistemas Computacionales**

**PRESENTA**

**17310793 – Eloy Antonio Clemente Rosas**

La Paz, Baja California Sur, México, 26 de Febrero del 2021

# Introducción

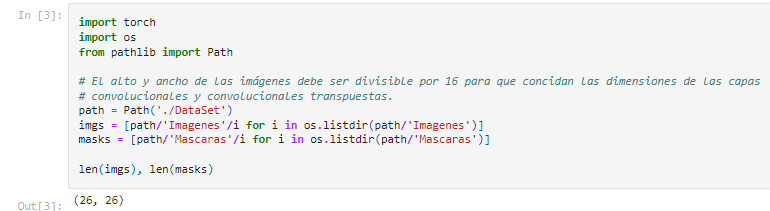
El presente documento contiene las actividades, y aprendizajes realizados durante la quinta semana de mi residencia profesional. En esta semana la red neuronal quedo lista y entrenada con fotos de prueba, obteniendo un buen reconocimiento, continuando comenzamos ensamblando la raspberry y conectándola a un motor para después con un programa indicar la velocidad y dirección de giro de un motor.

En el documento comentare un poco de los códigos de esta semana y sobre el cómo está conectada la raspberry, además de comentar un poco sobre la raspberry, sus características y sobre la forma ideal para extraer la red neuronal ya entrenada.

**Desarrollo**

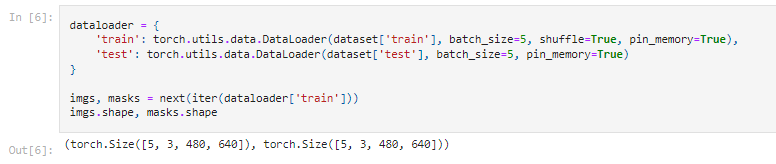
El código de la red neuronal es el siguiente:

<https://github.com/EloyFtw/Residencia-profesional/blob/main/new/Entrenamiento%20de%20la%20red%20neuronal.ipynb>

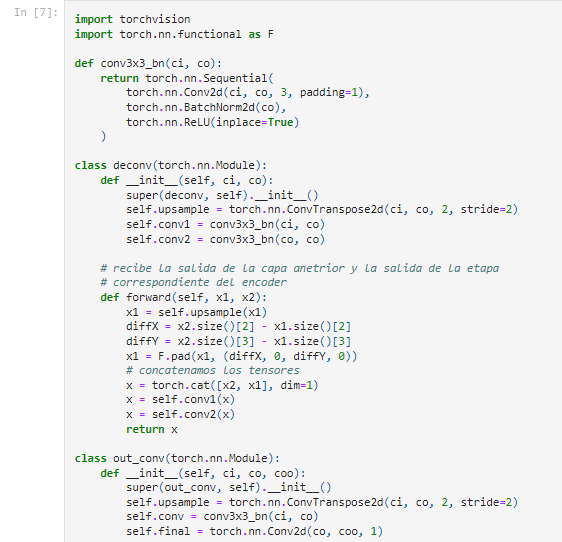
Primeramente es importante cargar el directorio con las imágenes.

También es importante darle las configuraciones a la red, y especificar cuantas imagenes son para pruebas y cuantas para entrenamiento.

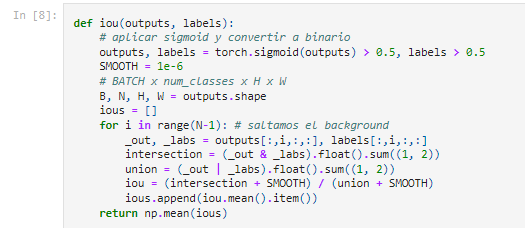
Ahora asignamos los batches.



El siguiente código es para la configuración de las capas de la red neuronal.



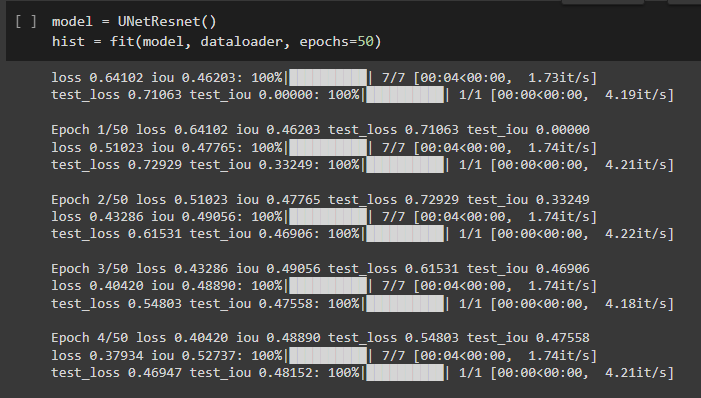




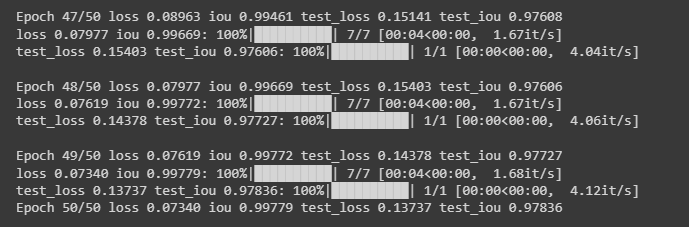
El método del código, es para el entrenamiento de la red neuronal.

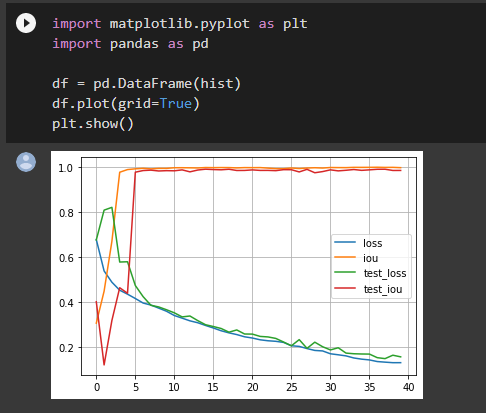


Para tener una mayor precisión el entrenamiento de la red neuronal se realizó a 50 épocas evitando un sobre entrenamiento de esta misma.



La precisión es de 97.83% con una pérdida de 0.07340

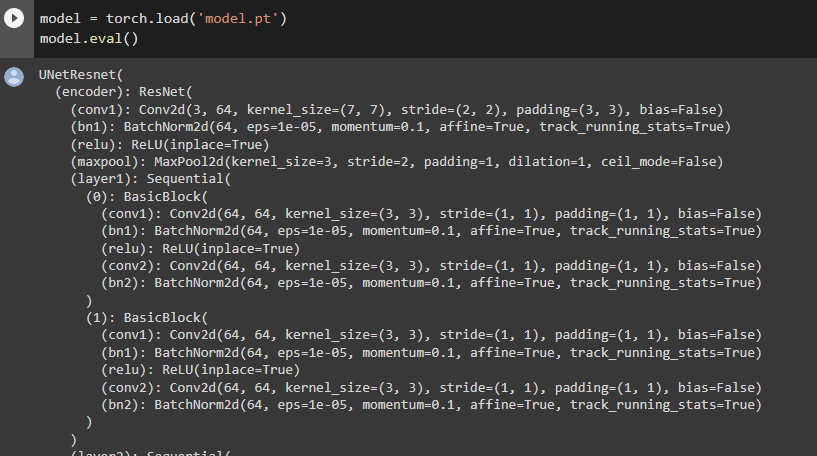


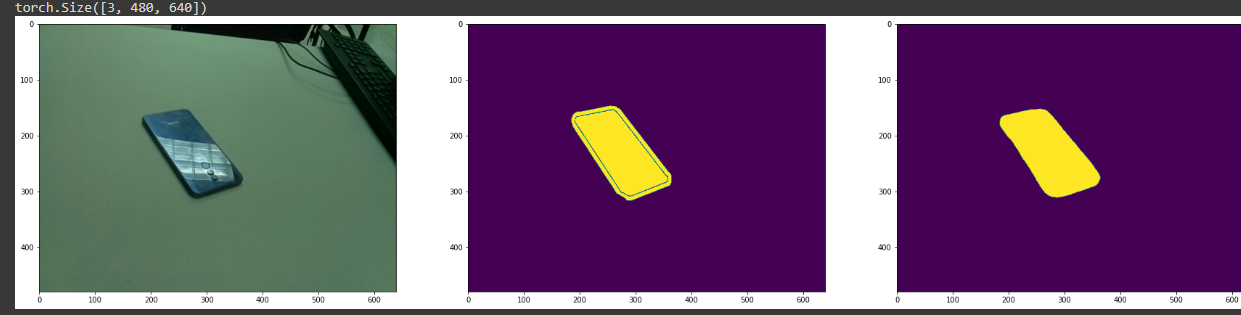
La grafica resultante por el entrenamiento es la siguiente:

Para exportar el modelo y poder usarlo se usó el siguiente comando que guarda un punto de la red neuronal en donde no es necesario instanciar, un modelo ya entrenado solamente desde Pytorch se puede cargar el modelo y usarse.



Para exportar el modelo en otro archivo es necesario, tener el archivo del modelo.



Este es el resultado de la prueba. En la primera imagen esta la imagen real, en la segunda la máscara creada por mí, y en la tercera la máscara realizada por la red neuronal.

**Preparación de imágenes**

Para poder entrenar la red y hacer pruebas es necesario convertir las imágenes que están en formato .png o en .jpg a formato npy, que es un arreglo de Numpy.

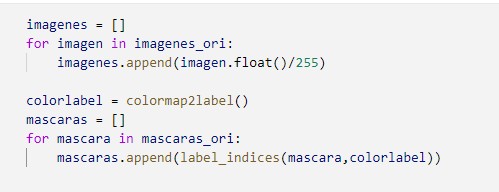
Esta sección del código crea un arreglo donde se indican los valores de colores que va a tener cada clase, es decir, guarda en un arreglo los colores rgb que va a tener cada mascara para identificar a cada objeto, es decir, este algoritmo almacena en cada posición del arreglo la clase a la que pertenece cada objeto y se guarda con valores de 1, 2, o 3 según el índice de la clase del objeto.



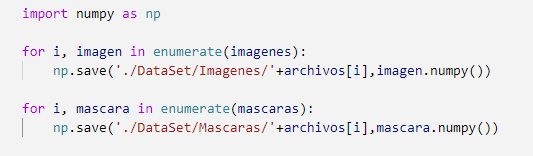
En esta sección del código se usa para almacenar el nombre de las imágenes y el nombre de las máscaras, el nombre de la máscara y el nombre de la imagen original deben coincidir.



Es importante normalizar las imágenes para después cargarlo dentro del entrenamiento.

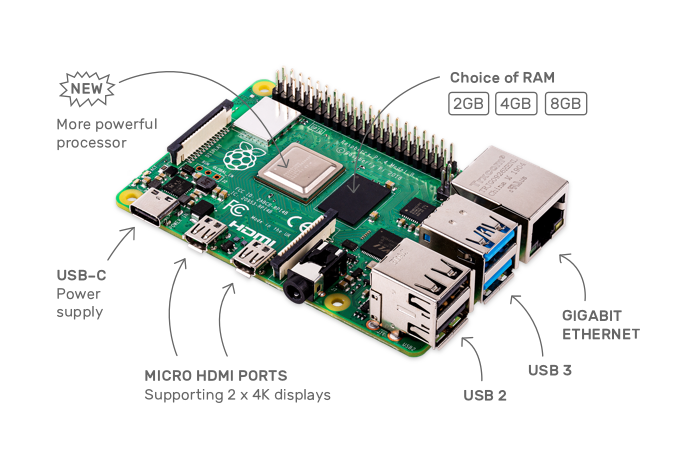


Finalmente se crea el archivo con la extensión npy.



# RaspBerry pi 4

Es, básicamente, un ordenador. A pesar de su placa reducida, su bajo coste y su sistema operativo de código abierto Raspbian, no deja de ser un ordenador adaptado a unas necesidades de programación muy básicas.



Su tamaño, potencia y versatilidad los convierte en unos aliados valiosísimos para quienes quieren avanzar en la construcción y diseño de robots. En este sentido, es importante informarse bien sobre las especificaciones de cada Raspberry Pi, puesto que dependiendo del modelo podemos hacer que varíe mucho el resultado de nuestras investigaciones.

# Puente H

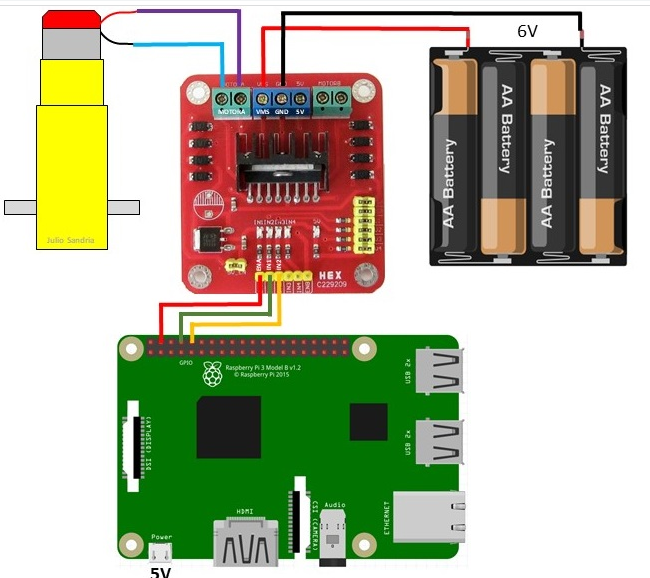
El puente H es un mecanismo electrónico que se encarga de invertir el giro de un motor usando un elemento básico en electrónica como lo es el TRANSISTOR.

El puente H juega un papel muy importante en el área de la robótica y la automatización. Por ejemplo cambiar el giro de un motor.

# Motor de corriente directa

Esta semana fue un reto hacer andar el motor de corriente directa para mí al no tener tanto conocimiento sobre estos

Siguiendo el siguiente diagrama e instrucciones de: <http://robotica.sandria.org/2017/12/controlar-motor-cd-con-python-en-raspberry-pi-y-modulo-puente-h/>

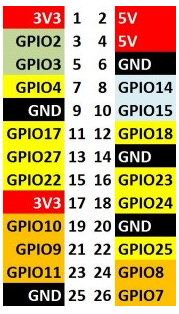
Se realizó la conexión de la raspberry con el puente h, el motor y la fuente de alimentación.

1. **import** RPi.GPIO as GPIO

# Para la numeracion logica se usar BCM

1. GPIO.setmode(GPIO.BCM)
2. # Para la numeracion de tablero de usa BOARD
3. GPIO.setmode(GPIO.BOARD)

Estos son los puertos de la raspberry de manera lógica y física.



Para Girar el motor en un sentido se usa el siguiente comando indicando el pin y usando High para alta intensidad de corriente y low para baja, es decir, high encendido, low apagado.

GPIO.output(8, GPIO.HIGH)

GPIO.output(10, GPIO.LOW)

Para girar el sentido contrario se hace invirtiendo esto.

GPIO.output(8, GPIO.LOW)

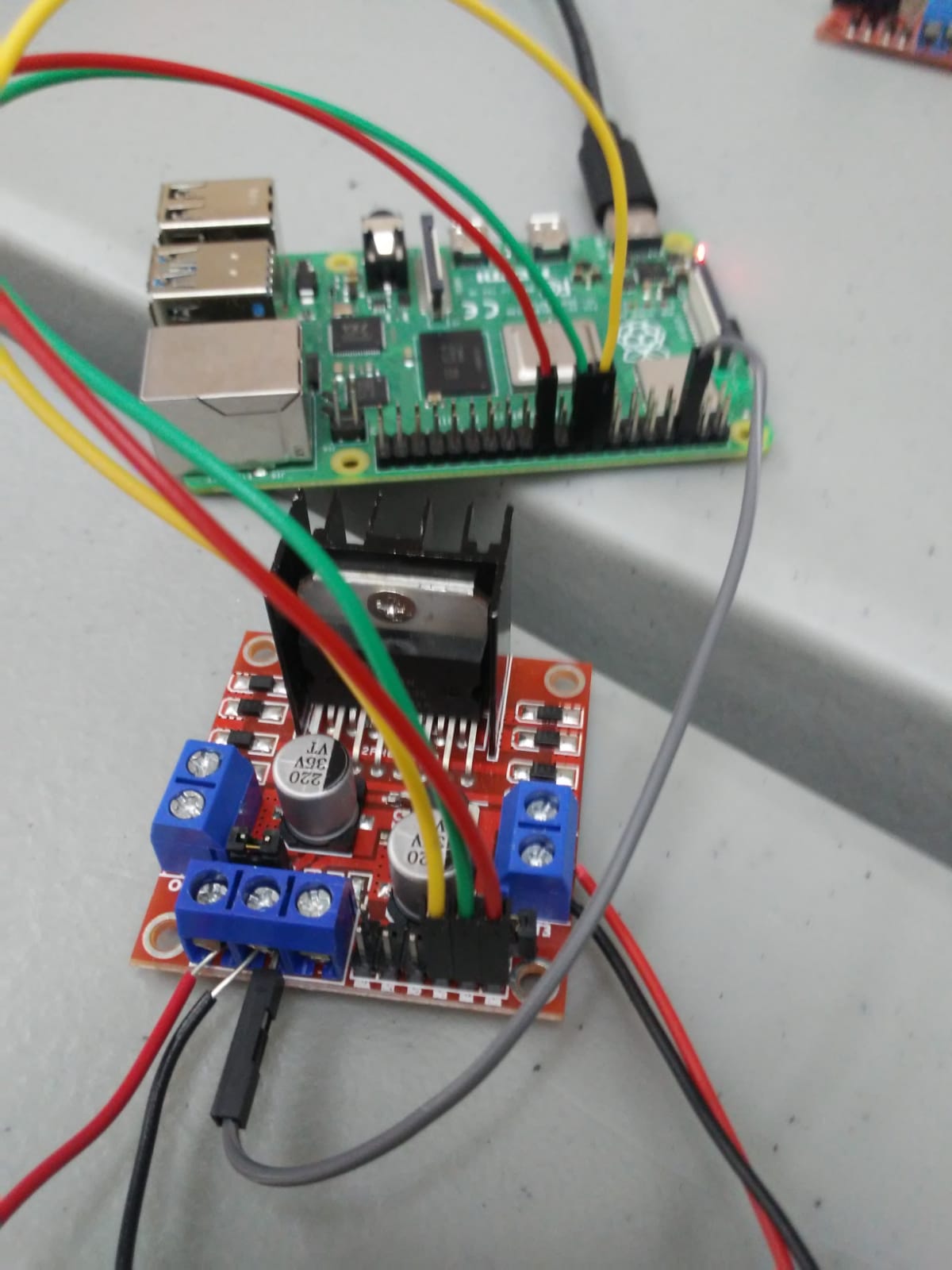
GPIO.output(10, GPIO.HIGH)

Finalmente se usar

GPIO.cleanup()

Para liberar la memoria de los puertos y detener el giro.

Fotos de los dispositivos conectados



Liga del video del motor funcionando.

<https://github.com/EloyFtw/Residencia-profesional/blob/main/new/VID-20220109-WA0000.mp4>