**INSTITUTO TECNÓLOGICO DE LA PAZ**

**DIVISIÓN DE ESTUDIOS PROFESIONALES**

**Reporte semanal de residencias Semana 08**

**“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, 23 de Marzo del 2021

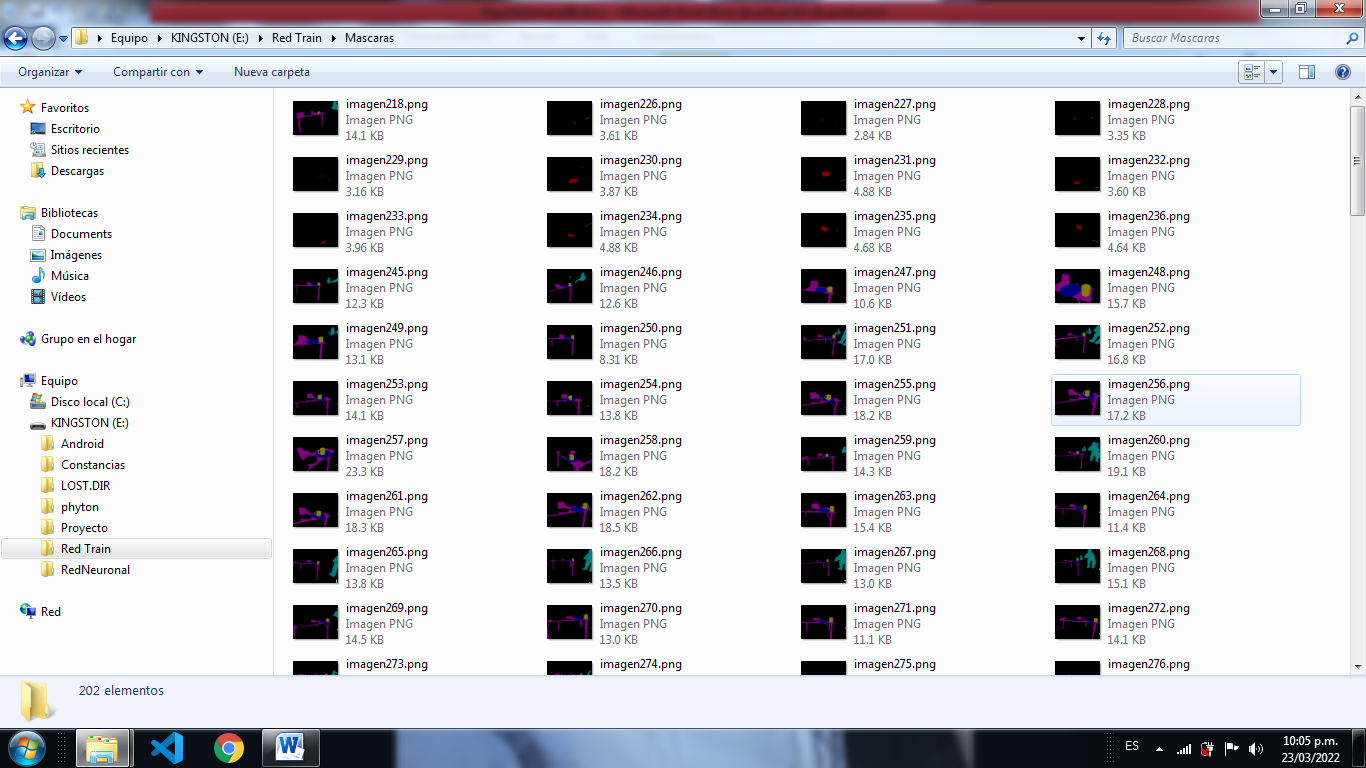
# Introducción

El presente documento contiene las actividades, y aprendizajes realizados durante la octava semana en mi residencia profesional. Me tomo tiempo segmentar las imágenes pero pude finalizar la segmentación. Esta semana trate de implementar las fotos ya mascaradas y las fotos reales para el entrenamiento de la red neuronal de lo que será el producto final, voy a pasar todo el procedimiento.

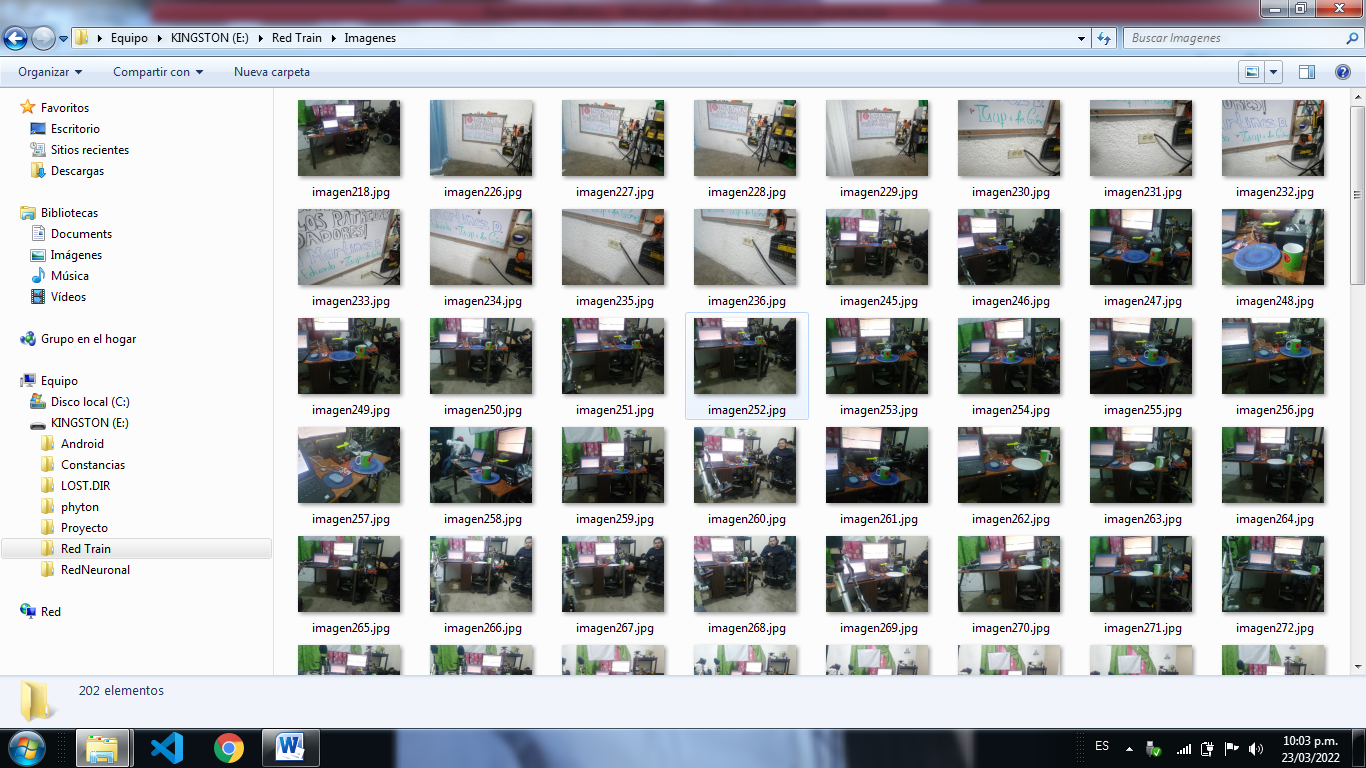
El documento contiene captura de las imágenes, los scripts usados para ajustar las imágenes, y el código usado para el entrenamiento de la red neuronal.

# Desarrollo de actividades

Para el entrenamiento de la red neuronal es necesario que todas las imágenes estén segmentadas de esta forma.

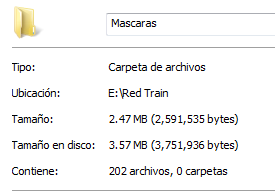
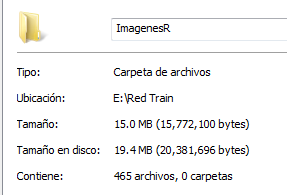


Imágenes originales, resultan ser 202 fotos.



El código usado en esta sección está en la siguiente ruta: <https://github.com/EloyFtw/Residencia-profesional/blob/main/Programas/25-03-2022/preparacionImg-1.ipynb>

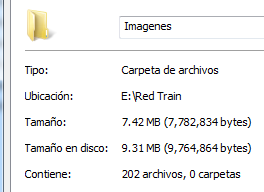
Al final se puede ver que son más mascaras que imágenes porque en su mayoría salieron movidas o mal formadas.



Para eliminar las imágenes a las cuales no se les hizo una máscara use el siguiente script



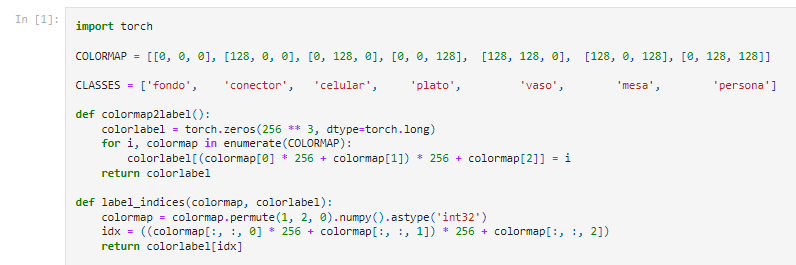
Que lo que hace es revisar todos los nombres de imágenes y crea una copia en el siguiente directorio:



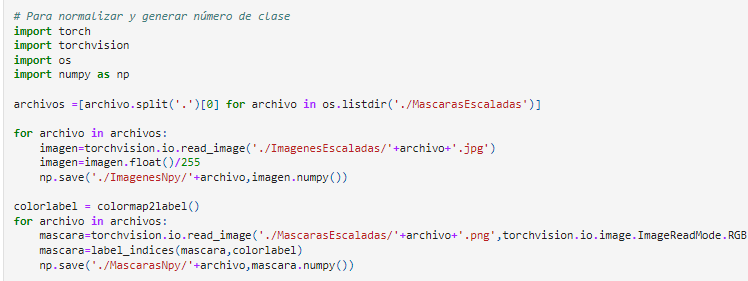
Con el siguiente script se normalizan las imágenes y máscaras a un tamaño de la mitad de tamaño: de 480, 640 a 240, 320.

El resultado son las imágenes ya escaladas.

En el siguiente pedazo de código se establecen los índices a los que pertenece cada clase a las que va a pertenecer cada objeto.



Finalmente con el siguiente script se convierte el archivo de imagen a arreglo de Numpy con el índice que se le indico anteriormente a cada color en el mapa de colores:



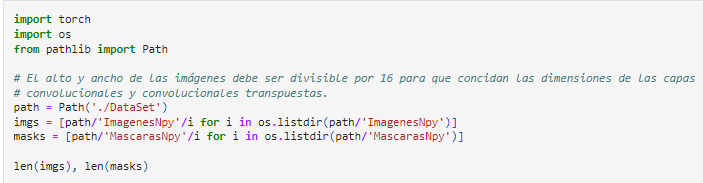
Se guardan en el siguiente directorio y son archivos de arreglo Numpy.

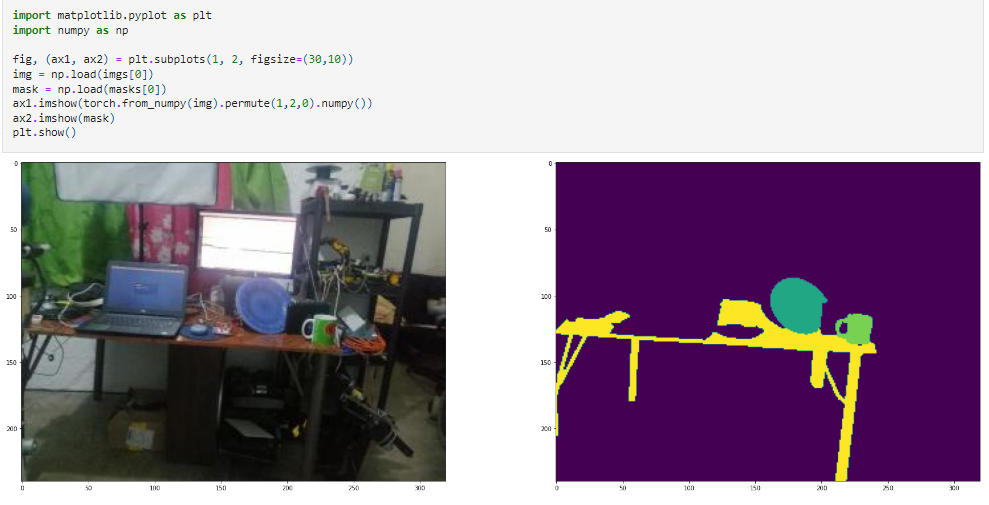


# Entrenamiento.

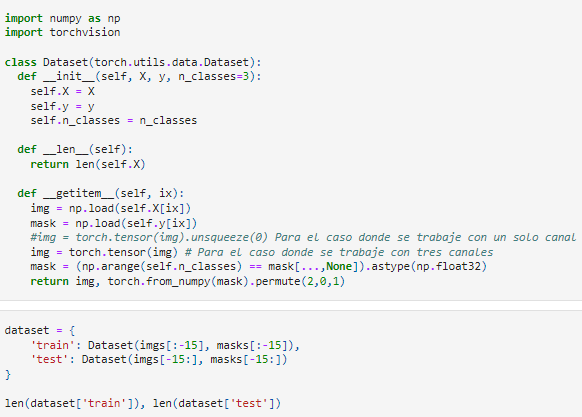
Con esto ya es posible realizar el entrenamiento el código para este intento está en el siguiente link: <https://github.com/EloyFtw/Residencia-profesional/blob/main/Programas/25-03-2022/PyTorch%20Segmentaci%C3%B3n%20Sem%C3%A1ntica.ipynb>

Primero se debe cargar los archivos Npy de las imágenes y las máscaras.



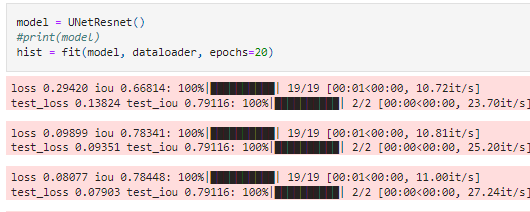
Aquí se puede ver un ejemplo de una imagen.

Aquí me falto señalar el número de clases que se van a usar en este caso son 7 contando el fondo en la configuración de los arreglos para la red, son 202 imágenes y donde 187 para entrenamiento y 15 para pruebas.

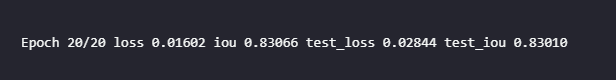


No aplique alguna configuración adicional en esta versión.

Aquí comienza el entrenamiento:



Obteniendo como resultado esta precisión y pérdida:



Todo perfecto hasta hacer pruebas donde en su mayoría no mostraba ni siquiera las máscaras originales que anteriormente si mostraba, es decir, las máscaras están bien pero la red está mal configurada.

