**Manual de Entrenamiento para Detección de Objetos con Tiny YOLOv3:**

1. **Bajar darknet:**

git clone<https://github.com/AlexeyAB/darknet.git>

cd darket

abrir makefile

cambiar a 1 los campos: GPU, OPENCV, CUDNN

Volver a hacer make clean y make

1. **Hacer minería de datos para elegir las imágenes para entrenar la red.** 
   1. Filtrar imágenes
   2. Aplicar aumentación (cambios de modelo de color, zoom, rotaciones, etc.
   3. Usar imágenes similares al objeto que se busca detectar.
2. **Bajar herramienta para etiquetar imagenes:**

Matlab Label Tool

**La herramienta permite:**

1. Generar archivos de texto por cada imagen con las coordenadas en formato (x, y, w, h) y con las coordenadas centradas, que son el formato específico de YOLO
2. Generar archivos de texto para las imágenes que se usarán para el entrenamiento y la validación.
3. Generar los bounding boxes para diferentes clases (empezando con la clase 0)

1. **Las imágenes a entrenar y de validación deben estar en el mismo directorio dataset en sus respectivas carpetas**

**Ejemplo de visualización de los archivos:**

* **darknet**
  + **“Carpeta”** (Donde van a estar tus archivos)
    - **weights** (carpeta donde se guardan los pesos)
    - **dataset** 
      * **train** (carpeta con las imágenes y labels para entrenar)
      * **val** (carpeta con imágenes y labels para validar)
    - **cfg**
      * **obj.data:** Todas las direcciones son relativas a la carpeta darket ej
    - **obj.names**
    - **test.txt:** Direcciones de las imágenes para validar, con direcciones relativas a la carpeta de darket ej:

Carpeta/val/19.jpg

* + - **train.txt:** Direcciones de las imágenes para entrenar, con direcciones relativas a la carpeta de darknet ej:

Carpeta/train/3.jpg

**Creaciòn de documentos test.txt y train.txt con el**

**SCRIPT DE PYTHON ADJUNTO procces.py:**

Colocar script en carpeta correspondiente con las imàgenes

EJEMPLO

Al colocar procces.py en la carpeta de validaciòn, descomentar dos veces el espacio correspondiente a la carpeta test.txt

Correr script

Archivo Creado - Se imprime Files Generated

1. **Preparar los archivos de configuración:**
   1. **cfg/obj.data**

Este archivo es para decir dónde se encuentran las imágenes que se van a entrenar, el número de clases y sus correspondientes nombres y donde se guardarán los backups para el entrenamiento. EL formato es el siguiente:

classes= 2 (Cambiar al numero de clases)

train = train.txt

valid = test.txt

names = obj.names

backup = backup/

* 1. **cfg/obj.names**

Este archivo contiene los nombres de las clases que vamos a entrenar. Si nuestra clase 0 son boyas, entonces la primera línea del archivo debe ser boya. Si la clase 1 son postes, entonces la segunda línea debe ser poste. Y así sucesivamente:

boya

poste

* 1. **cfg/yolo.cfg**

Podemos basarnos en el archivo **yolov3-tiny.cfg** y sólo modificarlo para adaptarlo a nuestras necesidades.

**Línea 6:** batch=64

Es el número de imágenes que se estrenarán en cada paso del entrenamiento. Este número se puede ajustar en caso de que se marquen errores de memoria.

**Línea 7:** subdivisions=8

El batch se dividirá en 8 para disminuir el consumo de memoria de la GPU, este número también se puede cambiar en caso de errores de memoria a un número mayor o a un número menor si la GPU lo soporta.

**Línea 127 y 171:** filters=(classes + 5) \* 3

No poner la fórmula, solo es representativa. Si entrenamos con una clase; filters=30, con dos clases; filters=35, y así sucesivamente.

**Línea 135 y 177:** classes=2

Número de clases con las que entrenaremos.

1. **Bajar pesos convulucionales:**

wget <https://pjreddie.com/media/files/yolov3-tiny.weights>

1. **Entrenar:**

./darknet detector train cfg/obj.data cfg/yolov3-tiny.cfg yolov3-tiny.conv.15 15

Checar detección de pesos con

./darknet detector map Roboboat2019/cfg/obj.data Roboboat2019/cfg/config.cfg Roboboat2019/weights/tiny3\_68000.weights

**mAP y F1 en lo más alto posible, 70% ya es algo aceptable**

1. **Testear:**

Usar imagenes de la carpeta test para observar como se comporta el algoritmo.

**batch=1 y subdivisions=1** cuando se quiera testear imágenes

Usar un comando como el siguiente:

./darknet detect cfg/yolo.cfg yolo.weights data/dog.jpg

Otros argumentos que se pueden agregar al comando:

-thresh [num]

-gpus 0,1,2,3

-nogpu

**Re entrenar Boyas y Markers de la competencia Roboboat 2019**

Roboboat2019 Object Detection

<https://docs.google.com/document/d/1wRIgsbQirxZRY_ttYvne-zEjMMjBTyACWFIlVH-ucTQ/edit>

Descargar RoboBot 2019 con toda la estructura previamente mencionada lista tanto como para testear y entrenar

* **darknet**
  + **“Roboboat2019”**