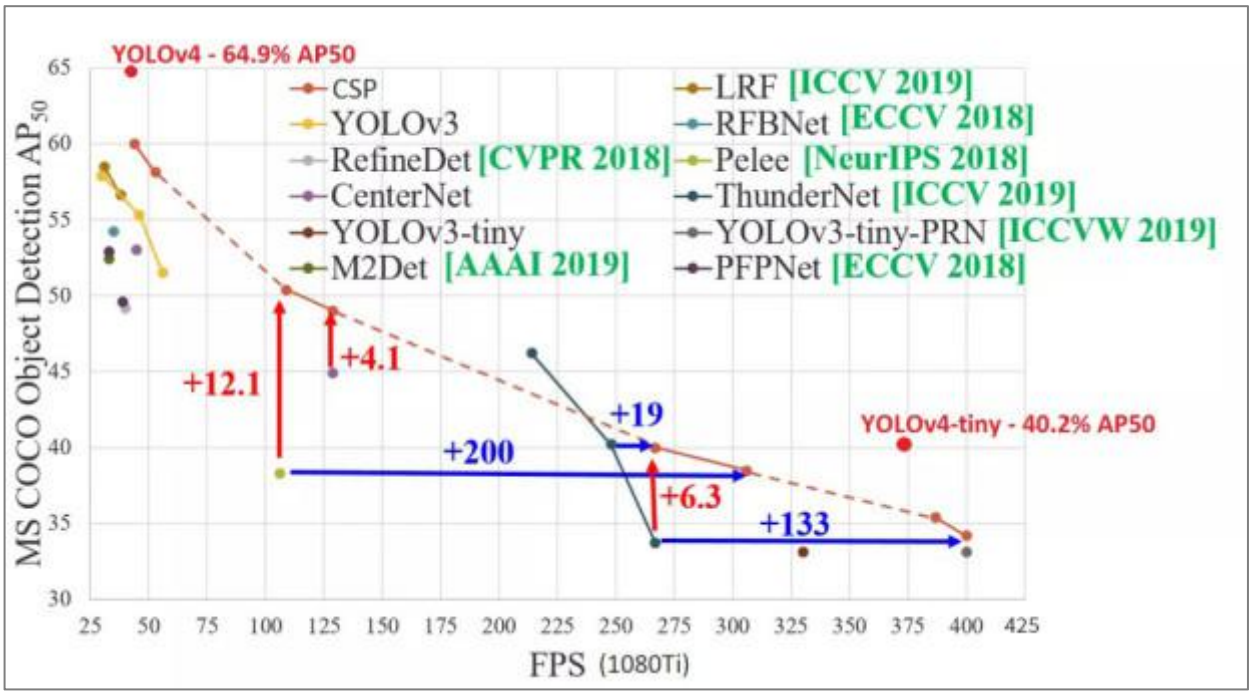
Path:
dofbot_ws/src/dofbot_garbage_yolov4_tiny/Garbage_sorting.ipynb
dofbot_ws/src/dofbot_garbage_yolov4_tiny/Garbage_sorting_single.ipynb

For the garbage sorting gameplay, users can also learn and train their own models to recognize and predict.

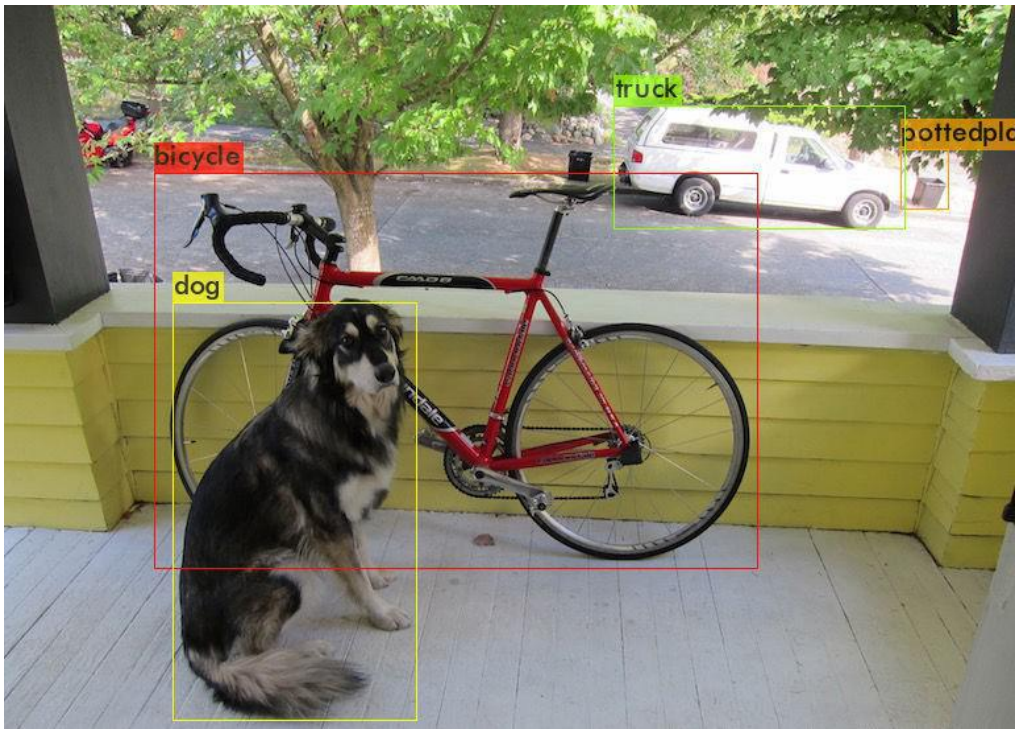
1.Yolov4-tiny introduction

Yolov4-tiny official website: https://github.com/AlexeyAB/darknet



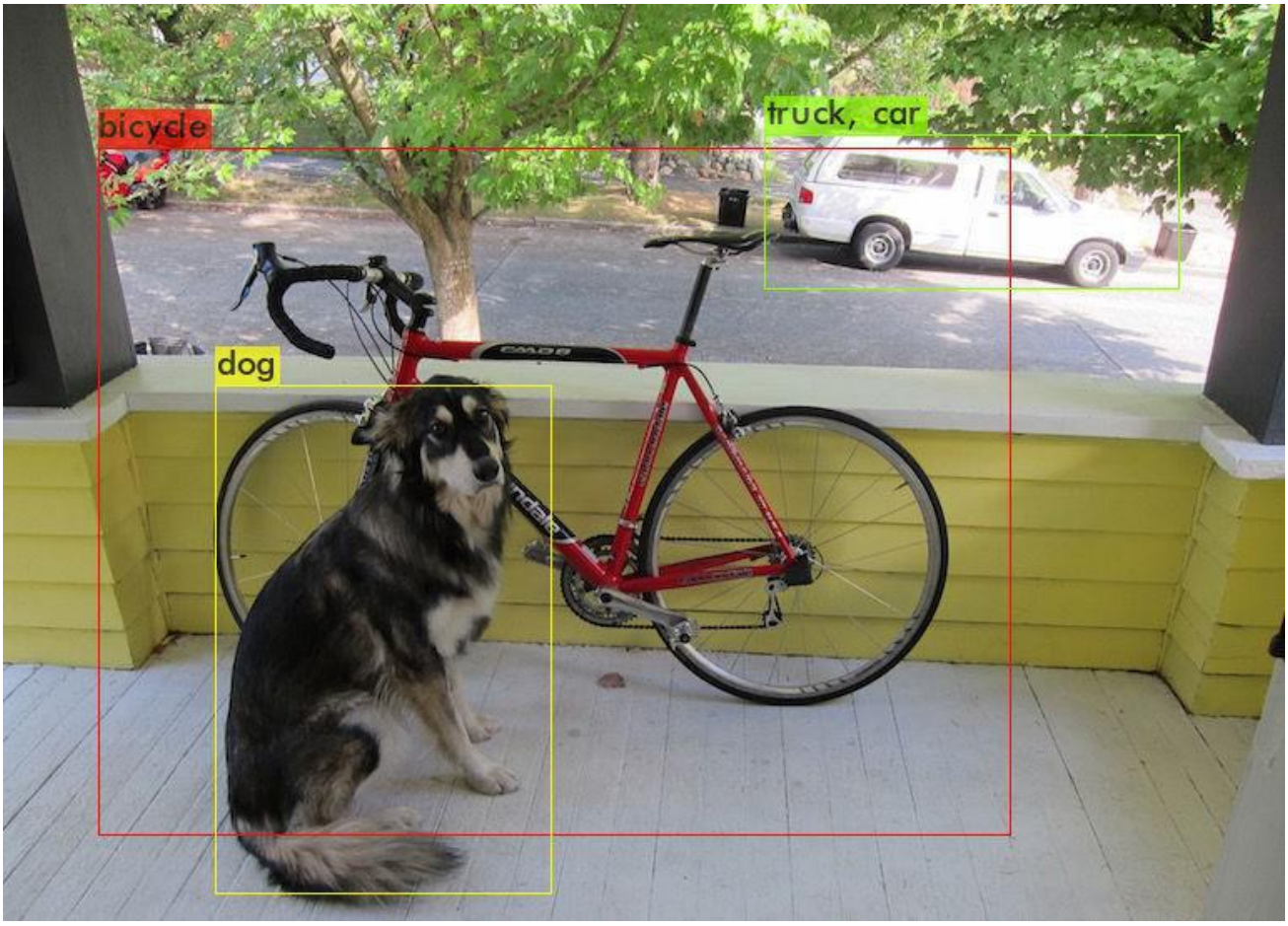
Comparison of YOLOv4 and YOLOv4-Tiny detection results.

- YOLOv4 detection results




```
Done! Loaded 162 layers from weights-file
data/dog.jpg: Predicted in 27.039000 milli-seconds.
bicycle: 92%
dog: 98%
truck: 92%
pottedplant: 33%
```

- YOLOv4-Tiny detection results




```
Done! Loaded 38 layers from weights-file
data/dog.jpg: Predicted in 2.609000 milli-seconds.
bicycle: 29%
dog: 72%
truck: 82%
car: 46%
```

We can know that the detection accuracy of Yolov4-tiny is lower than that of Yolov4, but it takes less time.

2. Environmental requirements

```
tensorflow-gpu==2.2.0
lxml
matplotlib
pandas
Pillow
scikit-learn
seaborn
tqdm
imgaug
```

3.Model training process

(1)Folder structure

garbage_data: Store the data set

garbage_data/image: Target source file

garbage_data/JPEGImages: Data set pictures (as many as possible)

garbage_data/texture: Background image (as many as possible)

garbage_data/train.txt: The label file corresponding to the data set picture

garbage_data/GetData.py: Get the data set code

font: Store font package

img: Store test images

logs: Store the test log and the final training model last1.h5.

model_data: Store the pre-trained model (weight file), custom label file (corresponding to the target source file), and yolo model parameter anchors.

nets and utils: Some library files of yolo

(2)Training steps

Training code source network, original link: https://github.com/bubbliiiing/yolov4-tiny-tf2

- Make a data set

The name of the image and the label file should be one-to-one correspondence. The label format in the train.txt file is as follows:

```
./garbage_data/JPEGImages/0.jpg 113,163,293,298,9
# Image path          y, x, y + w, x + h ,label
```

How to make a data set?

Method 1: Take some photos first, use the tagging tool to tag the target on each photo, create a new train.txt file under the garbage_data folder, and write the target information.

Method 2: Put the background picture (as many as possible) in the garbage_data/texture folder, modify the GetData.py code according to the requirements, and execute the GetData.py program file to generate the data set (as many as possible).

- Add weight file

The latest weight files can be downloaded from the Internet by themselves. The weight files yolov4_tiny_weights_coco.h5 and yolov4_tiny_weights_voc.h5 have been provided under the garbage_data file.

- Make your own classes--->garbage.txt

```
Zip_top_can
Old_school_bag
Newspaper
```

```
Book
Toilet_paper
......
```

- Modify train.py file

Modify the code according to your needs.

```
# Label location
annotation_path = 'garbage_data/train.txt'
# Get the location of classes and anchors
classes_path = 'model_data/garbage.txt'
anchors_path = 'model_data/yolo_anchors.txt'
# The location of the pre-trained model
weights_path = 'model_data/yolov4_tiny_weights_coco.h5'
# Obtain classes and anchor
class_names = get_classes(classes_path)
anchors = get_anchors(anchors_path)
# Categories
num_classes = len(class_names)
num_anchors = len(anchors)
# The location of the trained model is saved
log_dir = 'logs/'
# Input image size
input_shape = (416,416)
# Initial epoch value
Init_epoch = 0
# Freeze training epoch value
Freeze_epoch = 50

batch_size = 16

#Maximum learning rate
learning_rate_base = 1e-3

Epoch = 100
```

- Start training

According to the above process, after the operation is completed, directly run the train.py file to start training the model

(3)Custom model detection

- Modify yolov.py file

```
class YOLO(object):
    _defaults = {
        "model_path": 'model_data/garbage.h5',
        "anchors_path": 'model_data/yolo_anchors.txt',
        "classes_path": 'model_data/garbage.txt',
        "score" : 0.5,
        "iou" : 0.3,
        "eager" : False,

        "model_image_size" : (416, 416)
    }
... ...
self.font_path = 'font/Block_Simplified.TTF'
```

- Perform py file for detection

predict_img.py: Image detection.
predict_video.py: Video detection.

4. Single garbage sorting

- Import head file

```
#!/usr/bin/env python
# coding: utf-8
import Arm_Lib
import cv2 as cv
import threading
from time import sleep
import ipywidgets as widgets
from IPython.display import display
from single_garbage_identify import single_garbage_identify
```

- Create an instance, initialize parameters

```
# Create and get target instance
single_garbage = single_garbage_identify()
# Initialization mode
model = "General"
```

- Initialize the robot arm position

```
import Arm_Lib
arm = Arm_Lib.Arm_Device()
joints_0 = [90, 135, 0, 0, 90, 30]
arm.Arm_serial_servo_write6_array(joints_0, 1000)
```

- Create control

```
button_layout = widgets.Layout(width='320px', height='60px', align_self='center')
output = widgets.Output()

exit_button = widgets.Button(description='Exit', button_style='danger', layout=button_layout)
imgbox = widgets.Image(format='jpg', height=480, width=640,
layout=widgets.Layout(align_self='center'))
controls_box = widgets.VBox([imgbox, exit_button], layout=widgets.Layout(align_self='center'))
```

- Exit process

```
def exit_button_Callback(value):
    global model
    model = 'Exit'
    with output: print(model)
exit_button.on_click(exit_button_Callback)
```

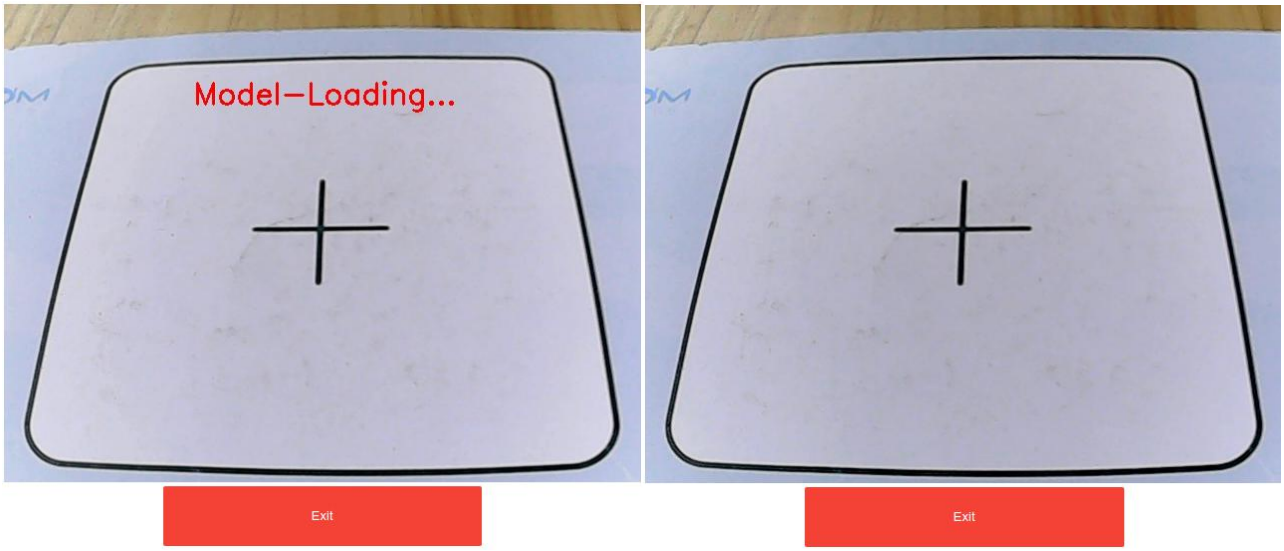
- Main process

```
def camera():
    # Open camera
    capture = cv.VideoCapture(0)
    while capture.isOpened():
        try:
            _, img = capture.read()
            img = cv.resize(img, (640, 480))
            img = single_garbage.single_garbage_run(img)
            if model == 'Exit':
                cv.destroyAllWindows()
                capture.release()
                break
            imgbox.value = cv.imencode('.jpg', img)[1].tobytes()
        except KeyboardInterrupt:capture.release()
```

- Start

```
display(controls_box,output)
threading.Thread(target=camera, ).start()
```

It takes some time to load the model, please be patient.
Place the box with trash on the center of the cross of the box facing the robotic arm. The robotic arm will continue to accurately identify 10 times and it will be automatically sorted.



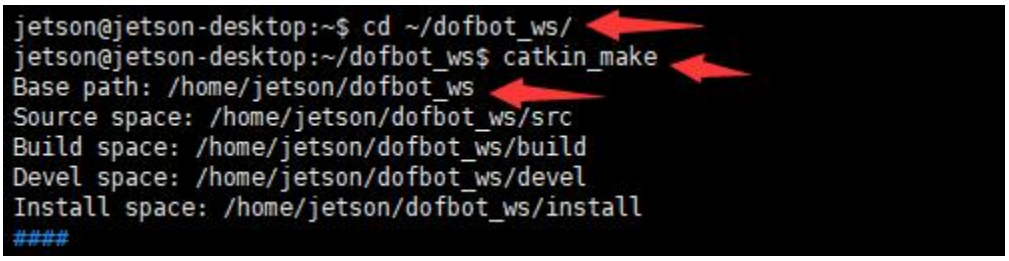
5. Garbage sorting

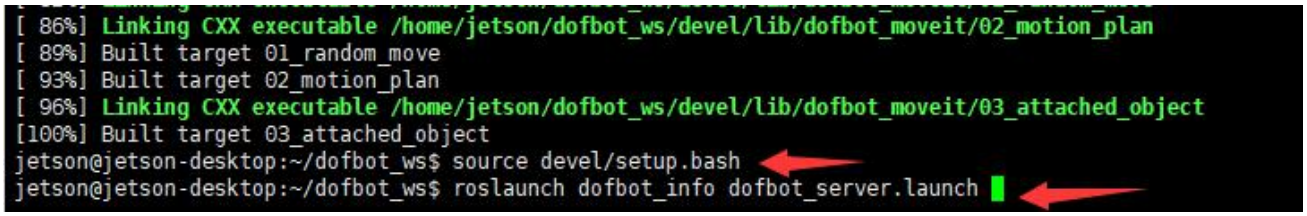
Before starting garbage function, you must start the reverse solution server and keep running.

Input following command:

```
cd ~/dofbot_ws/    # Entering Workspace
catkin_make      # compile
source devel/setup.bash    # Update the system environment
roslaunch dofbot_info dofbot_server.launch    # Start the server terminal node
```

As shown below.





```
[ 96%] Linking CXX executable /home/jetson/dofbot_ws/devel/lib/dofbot_moveit/03_attached_object
[100%] Built target 03_attached_object
jetson@jetson-desktop:~/dofbot_ws$ source devel/setup.bash
jetson@jetson-desktop:~/dofbot_ws$ roslaunch dofbot_info dofbot_server.launch
... logging to /home/jetson/.ros/log/5d3c7692-5985-11eb-b258-355db64c8d49/roslaunch-jetson-desktop-19109.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.1.169:33229/

SUMMARY
========

PARAMETERS
 * /rosdistro: melodic
 * /rosversion: 1.14.10

NODES
  /
    dofbot_server (dofbot_info/dofbot_server)

ROS_MASTER_URI=http://localhost:11311

process[dofbot_server-1]: started with pid [19164]
```

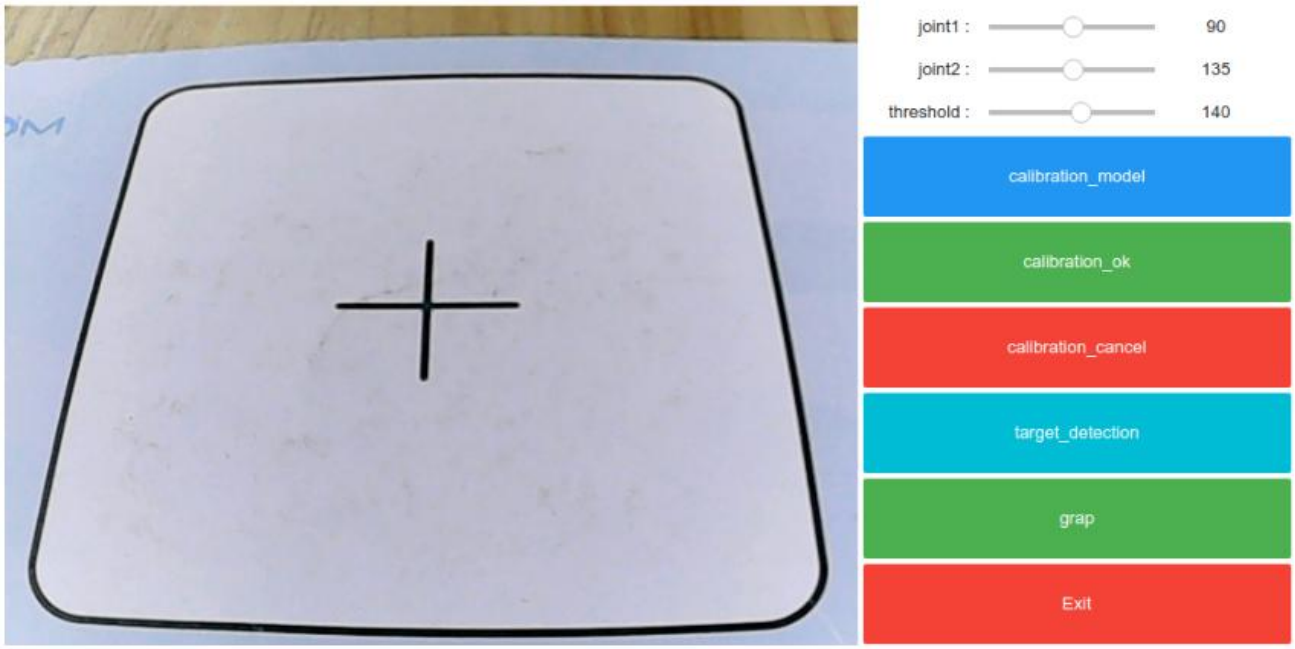
5.3 Box calibration

After the color calibration is completed, box calibration is required to obtain the position information of the block in the box.

More Details, please check [Visual positioning] course.

5.4 Running garbage sorting program

After running the program, you will see the interface as shown below.



5.1 After the [box calibration] is completed, click the [target_detection] button to load the model for identification.
5.4 When the block is recognized on map, click the [grap] button to start DOFBOT.
5.5 When the game is over, please click the [Exit] button to exit the program.