Assignment

In this assignment, you will first go through a YOLOv4 tutorial, which has been included below for convienence. After working through the tutorial, you will train a YOLOv4-tiny model using a small custom dataset and also answer some questions. The assignment begins after the tutorial section.

This ansignment uses Google Colab. Google Colab (or Google Colaboratory) is a cloud based platform which allows you to write and execute python code from a browser using Jupyter Notebooks. Google Colab provides free access to GPUs, so you can train the YOLO model in this assignment even if you do not have access to a GPU. You just need a Google account to use Goole Colab.

To train the YOLO model, we will use Darknet. Darknet is an open source framework for executing and training CNNs and YOLO models. It is maintained by some of the authors of the YOLO models. Darknet is written in C and can be a bit challenging to install with GPU support so we will build and run it using Google Colab. Instructions for this are given in the tutorial section.

First save a copy of this notebook so you can modify it: `File->Save A Copy`. You can save is it as `Assignment3_LASTNAME`.

Tutorial

Running a YOLOv4 Object Detector with Darknet in the Cloud! (GPU ENABLED)

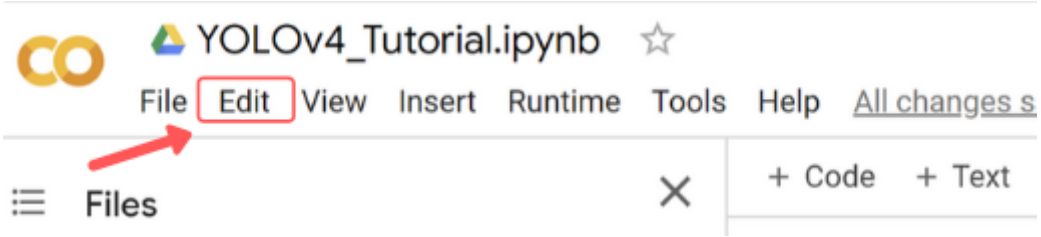
This tutorial will help you build YOLOv4 easily in the cloud with GPU enabled so that you can run object detections in milliseconds!

Step 1: Enabling GPU within your notebook

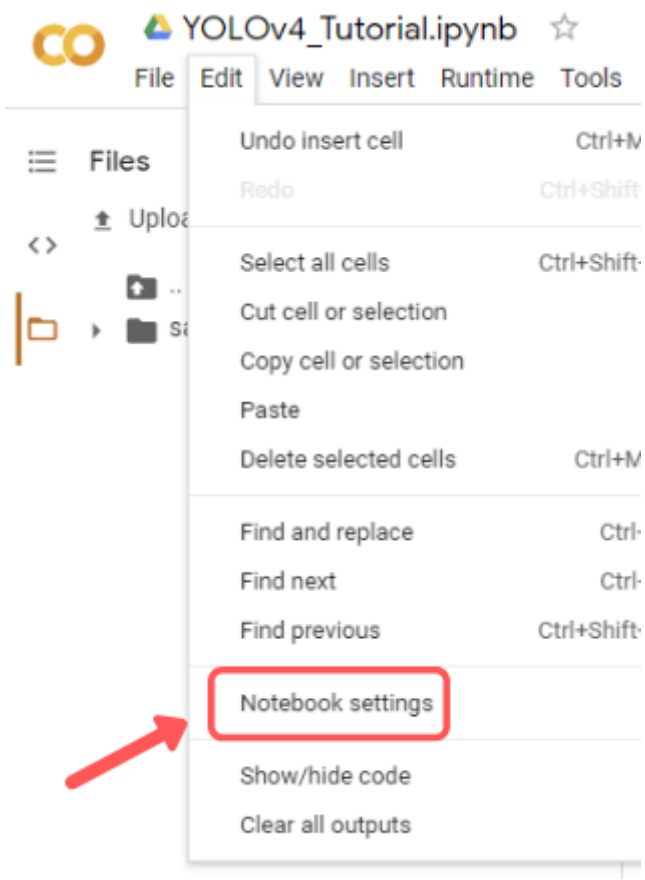
You will want to enable GPU acceleration within your Colab notebook so that your YOLOv4 system will be able to process detections over 100 times faster than CPU.

Steps:

i) Click **Edit** at top left of your notebook



ii) Click **Notebook Settings** within dropdown



Step 2: Cloning and Building Darknet

The following cells will clone darknet from AlexeyAB's famous repository, adjust the Makefile to enable OPENCV and GPU for darknet and then build darknet.

Do not worry about any warnings when you run the '!make' cell!

```
# clone darknet repo
!git clone https://github.com/AlexeyAB/darknet
```

```
Cloning into 'darknet'...
remote: Enumerating objects: 15420, done.
remote: Total 15420 (delta 0), reused 0 (delta 0), pack-reused 15420
Receiving objects: 100% (15420/15420), 14.05 MiB | 24.09 MiB/s, done.
Resolving deltas: 100% (10362/10362), done.
```

Hardware accelerator

```
# change makefile to have GPU and OPENCV enabled
%cd darknet
!sed -i 's/OPENCV=0/OPENCV=1/' Makefile
!sed -i 's/GPU=0/GPU=1/' Makefile
!sed -i 's/CUDNN=0/CUDNN=1/' Makefile
!sed -i 's/CUDNN_HALF=0/CUDNN_HALF=1/' Makefile
```

```
/content/darknet
```

```
# verify CUDA
!/usr/local/cuda/bin/nvcc --versiondetec
```

```
nvcc fatal   : Unknown option '--versiondetec'
```

```
# make darknet (builds darknet so that you can then use the darknet executable file to run or train object detectors)
!make
```

```
                    from ./src/yolo_layer.h:5,
                    from ./src/yolo_layer.c:1:
/usr/local/cuda/include/cuda_runtime_api.h:4811:39: note: expected ‘void **’ but argument is of type ‘float **’
 extern __host__ cudaError_t CUDARTAPI cudaHostAlloc(void **pHost, size_t size, unsigned int flags);
                                       ^~~~~~~~~~~~~
./src/yolo_layer.c: In function ‘process_batch’:
./src/yolo_layer.c:426:25: warning: variable ‘best_match_t’ set but not used [-Wunused-but-set-variable]
                     int best_match_t = 0;
                         ^~~~~~~~~~~~
./src/yolo_layer.c: In function ‘forward_yolo_layer’:
./src/yolo_layer.c:707:11: warning: unused variable ‘avg_anyobj’ [-Wunused-variable]
     float avg_anyobj = 0;
           ^~~~~~~~~~
./src/yolo_layer.c:706:11: warning: unused variable ‘avg_obj’ [-Wunused-variable]
```

```
     float avg_obj = 0;
           ^~~~~~~
./src/yolo_layer.c:705:11: warning: unused variable ‘avg_cat’ [-Wunused-variable]
     float avg_cat = 0;
           ^~~~~~~
./src/yolo_layer.c:704:11: warning: unused variable ‘recall75’ [-Wunused-variable]
     float recall75 = 0;
           ^~~~~~~~
./src/yolo_layer.c:703:11: warning: unused variable ‘recall’ [-Wunused-variable]
     float recall = 0;
           ^~~~~~
./src/yolo_layer.c:702:11: warning: unused variable ‘tot_ciou_loss’ [-Wunused-variable]
     float tot_ciou_loss = 0;
           ^~~~~~~~~~~~~
./src/yolo_layer.c:701:11: warning: unused variable ‘tot_diou_loss’ [-Wunused-variable]
     float tot_diou_loss = 0;
           ^~~~~~~~~~~~~
./src/yolo_layer.c:698:11: warning: unused variable ‘tot_ciou’ [-Wunused-variable]
     float tot_ciou = 0;
           ^~~~~~~~
./src/yolo_layer.c:697:11: warning: unused variable ‘tot_diou’ [-Wunused-variable]
     float tot_diou = 0;
           ^~~~~~~~
./src/yolo_layer.c:696:11: warning: unused variable ‘tot_giou’ [-Wunused-variable]
     float tot_giou = 0;
           ^~~~~~~~
./src/yolo_layer.c:668:12: warning: unused variable ‘n’ [-Wunused-variable]
     int b, n;
            ^
gcc -Iinclude/ -I3rdparty/stb/include -DOPENCV `pkg-config --cflags opencv4 2> /dev/null || pkg-config --cflags opencv` -DGP
./src/gaussian_yolo_layer.c: In function ‘make_gaussian_yolo_layer’:
./src/gaussian_yolo_layer.c:72:38: warning: passing argument 1 of ‘cudaHostAlloc’ from incompatible pointer type [-Wincompat
     if (cudaSuccess == cudaHostAlloc(&l.output, batch*l.outputs * sizeof(float), cudaHostRegisterMapped)) l.output_pinned =
                                      ^
In file included from /usr/local/cuda/include/cuda_runtime.h:96:0,
                 from include/darknet.h:41,
                 from ./src/gaussian_yolo_layer.h:5,
                 from ./src/gaussian_yolo_layer.c:7:
/usr/local/cuda/include/cuda_runtime_api.h:4811:39: note: expected ‘void **’ but argument is of type ‘float **’
 extern __host__ cudaError_t CUDARTAPI cudaHostAlloc(void **pHost, size_t size, unsigned int flags);
                                       ^~~~~~~~~~~~~
```

```
./src/gaussian_yolo_layer.c:79:38: warning: passing argument 1 of ‘cudaHostAlloc’ from incompatible pointer type [-Wincompat
    if (cudaSuccess == cudaHostAlloc(&l.delta, batch*l.outputs * sizeof(float), cudaHostRegisterMapped)) l.delta_pinned = 1
```

Step 3: Download pre-trained YOLOv4 weights

YOLOv4 has been trained already on the coco dataset which has 80 classes that it can predict. We will grab these pretrained weights so that we can run YOLOv4 on these pretrained classes and get detections.

```
!wget https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.weights
```

```
--2022-06-01 02:27:17--  https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.weights
Resolving github.com (github.com)... 192.30.255.113
Connecting to github.com (github.com)|192.30.255.113|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/75388965/ba4b6380-889c-11ea-9751-f994f59
--2022-06-01 02:27:17--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/75388965/ba4b6380-889c-11
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, .
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 257717640 (246M) [application/octet-stream]
Saving to: ‘yolov4.weights’

yolov4.weights      100%[===================>] 245.78M  14.8MB/s    in 13s

2022-06-01 02:27:31 (18.3 MB/s) - ‘yolov4.weights’ saved [257717640/257717640]
```

Step 4: Define Helper Functions

These three functions are helper functions that will allow you to show the image in your Colab Notebook after running your detections, as well as upload and download images to and from your Cloud VM.

```
 def imShow(path):
  import cv2
  import matplotlib.pyplot as plt
  %matplotlib inline

  image = cv2.imread(path)
  height, width = image.shape[:2]
  resized_image = cv2.resize(image,(3*width, 3*height), interpolation = cv2.INTER_CUBIC)

  fig = plt.gcf()
  fig.set_size_inches(18, 10)
  plt.axis("off")
  plt.imshow(cv2.cvtColor(resized_image, cv2.COLOR_BGR2RGB))
  plt.show()

# use this to upload files
def upload():
  from google.colab import files
  uploaded = files.upload()
  for name, data in uploaded.items():
    with open(name, 'wb') as f:
      f.write(data)
      print ('saved file', name)

# use this to download a file
def download(path):
  from google.colab import files
  files.download(path)
```

Step 5: Run Your Detections with Darknet and YOLOv4!

Darknet is now built and ready to run detections using YOLOv4 in the cloud! You can find out which sorts of classes the pre-trained YOLOv4 weights can detect by clicking here. [COCO CLASSES]

The object detector can be run using the following command

```
!./darknet detector test <path to .data file> <path to config> <path to weights> <path to image>
```

Darknet comes with a few images already installed in the darknet/data/ folder.

Note: After running detections OpenCV can't open the image instantly in the cloud so we must run:

```
imShow('predictions.jpg')
```

This will output the image with the detections shown. The most recent detections are always saved to 'predictions.jpg'

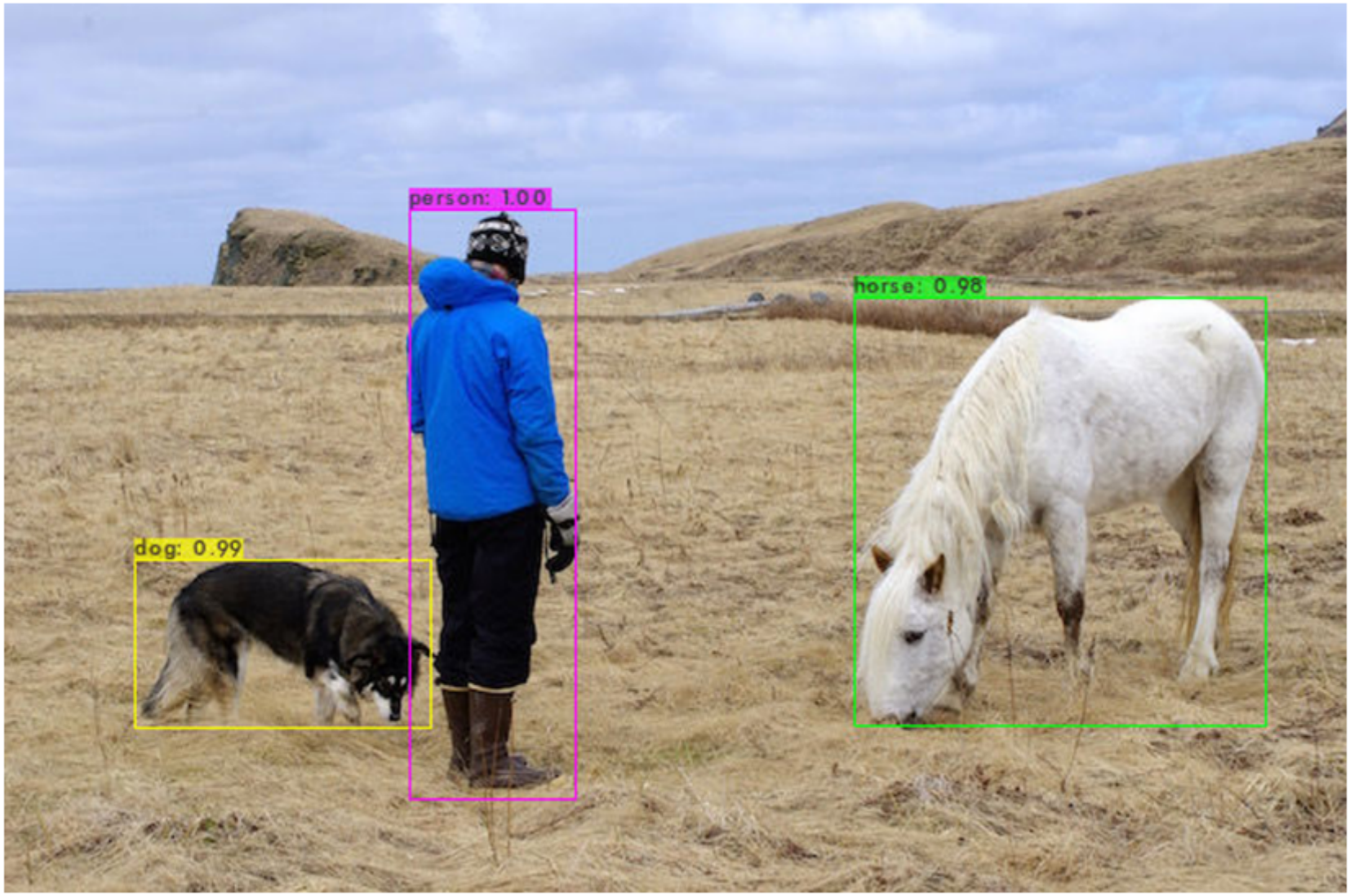
Try out the examples below for yourself!

```
# run darknet detection on test images
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights data/person.jpg
```

```
       91 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
       92 Shortcut Layer: 89,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
       93 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
       94 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
       95 Shortcut Layer: 92,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
       96 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
       97 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
       98 Shortcut Layer: 95,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
       99 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
      100 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
      101 Shortcut Layer: 98,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
      102 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
      103 route  102 87                                ->   19 x  19 x1024
      104 conv   1024       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x1024 0.757 BF
      105 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
      106 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
      107 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
      108 max               5x 5/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.005 BF
      109 route  107                                     ->   19 x  19 x 512
```

```
 110 max                9x 9/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.015 BF
 111 route  107                                        ->   19 x  19 x 512
 112 max               13x13/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.031 BF
 113 route  112 110 108 107                          ->   19 x  19 x2048
 114 conv    512       1 x 1/ 1     19 x  19 x2048 ->   19 x  19 x 512 0.757 BF
 115 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 116 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 117 conv    256       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 256 0.095 BF
 118 upsample                2x    19 x  19 x 256 ->   38 x  38 x 256
 119 route  85                                       ->   38 x  38 x 512
 120 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 121 route  120 118                                  ->   38 x  38 x 512
 122 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 123 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 124 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 125 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 126 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 127 conv    128       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 128 0.095 BF
 128 upsample                2x    38 x  38 x 128 ->   76 x  76 x 128
 129 route  54                                       ->   76 x  76 x 256
 130 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 131 route  130 128                                  ->   76 x  76 x 256
 132 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 133 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 134 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 135 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 136 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 137 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 138 conv    255       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 255 0.754 BF
 139 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.20
nms_kind: greedynms (1), beta = 0.600000
 140 route  136                                      ->   76 x  76 x 128
 141 conv    256       3 x 3/ 2     76 x  76 x 128 ->   38 x  38 x 256 0.852 BF
 142 route  141 126                                  ->   38 x  38 x 512
 143 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 144 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 145 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 146 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
```

```
# show image using our helper function
imShow('predictions.jpg')
```




```
# This stops 'Run all' at this cell by causing an error
assert False
```

```
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
<ipython-input-9-90e9831ca4f5> in <module>()
      1 # This stops 'Run all' at this cell by causing an error
----> 2 assert False

AssertionError:
```

SEARCH STACK OVERFLOW

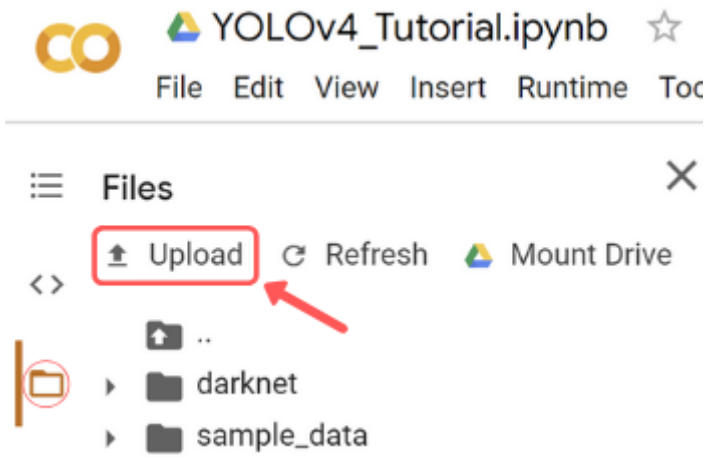
Step 6: Uploading Local or Google Drive Files to Use

You may want to run detections on more than just the images within the darknet/data/ folder. This step will show you how to upload local or Google Drive files to the cloud VM and run detections on them!

Method 1: Local Files

To upload local files just use our helper function by running 'upload()' as seen below. Click **Choose Files** and select the image from your local machine that you want to upload to the cloud VM.

If this function doesn't work for you then click the **Upload** button in the File Explorer on the left side of your notebook.



The image should save to the root directory of the cloud VM so that you can access it from the darknet command by running.

```
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights ../<your image name>
```

```
# try out the upload helper function! (I uploaded an image called highway.jpg, upload whatever you want!)
%cd ..
upload()
%cd darknet
```

```
# run darknet with YOLOv4 on your personal image! (note yours will not be called highway.jpg so change the name)
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights ../highway.jpg
imShow('predictions.jpg')
```

Method 2: Google Drive (NOT REQUIRED FOR ASSIGNMENT)

Images can also be uploaded from your Google Drive and easily have YOLOv4 detections run on them.

You will want to run the below cell to mount your google drive into the cloud VM so that you can access its contents. It is that easy!

NOTE: We will be creating a symbolic link between '/content/gdrive/My\ Drive/' and '/mydrive.

This means we are just creating a shortcut '/mydrive' to map to the contents within the folder '/content/gdrive/My\ Drive/'.

The reason for this is that sometime having the space in 'My Drive' folder path can cause issues when running certain commands. This symbolic link will stop this from happening!

Now you can run YOLOv4 with images from Google Drive using the darknet command:

```
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights /mydrive/<path to image>
```

I recommend saving images within a folder called 'images' at the root level of your Google Drive.

```
%cd ..
from google.colab import drive
drive.mount('/content/gdrive')
```

```
# this creates a symbolic link so that now the path /content/gdrive/My\ Drive/ is equal to /mydrive
!ln -s /content/gdrive/My\ Drive/ /mydrive
!ls /mydrive
```

```
# cd back into the darknet folder to run detections
%cd darknet
```

```
# run detections on image within your Google Drive!
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights /mydrive/images/street.jpg
imShow('predictions.jpg')
```

Download Files to Local Machine or Google Drive from Cloud VM

You can also easily download images from your cloud VM to save to your local machine or Google Drive.

Method 1: Local Machine

You can do it easily by using our helper function 'download()' or by right clicking the image in the File Explorer on the left side of your notebook and hitting **Download**. Files will be saved to your *Downloads* folder.

This is useful if you want to download the **'predictions.jpg'** images that the object detector outputs.

Method 2: Google Drive

A simple copy command can copy over any file to your Google Drive as it is already mounted. (you must run the mount command above if you have not already)

```
!cp <file to download> <destination to save file>
```

See example of each below!

```
# LOCAL MACHINE DOWNLOAD
# if you get an error first run then run it again and it should work
download('predictions.jpg')
```

```
# GOOGLE DRIVE DOWNLOAD
# note that I can change what the image name is saved as (I am saving it as detection1.jpg)
!cp predictions.jpg /mydrive/images/detection1.jpg
```

Step 7: Running YOLOv4 on Video in the Cloud! (NOT REQUIRED FOR ASSIGNMENT)

You can also run YOLOv4 object detections on video in your Google Colab VM. Follow the cells below to see how to run videos from your local machine or from your Google Drive!

Local Machine Video

Here is how to upload video from local machine, run detector and then download video showing detections.

```
# upload the video of your choosing! (Feel free to use the same video I do, it is in the Github repository)
upload()
```

Choose Files istockphoto…dpp_is.mp4

- **istockphoto-1258113642-640_adpp_is.mp4**(video/mp4) - 2900051 bytes, last modified: 5/28/2022 - 100% done

```
Saving istockphoto-1258113642-640_adpp_is.mp4 to istockphoto-1258113642-640_adpp_is.mp4
saved file istockphoto-1258113642-640_adpp_is.mp4
```

```
!./darknet detector demo cfg/coco.data cfg/yolov4.cfg yolov4.weights -dont_show istockphoto-1258113642-640_adpp_is.mp4 -i 0 -out_file
```

```
person: 57%
person: 37%
person: 30%

FPS:26.2        AVG_FPS:27.5

 cvWriteFrame
Objects:

handbag: 65%
backpack: 26%
person: 98%
person: 98%
person: 95%
person: 95%
person: 95%
person: 92%
person: 91%
person: 90%
person: 89%
person: 77%
person: 68%
person: 37%
person: 37%

FPS:26.0        AVG_FPS:27.5

 cvWriteFrame
Objects:

handbag: 68%
backpack: 26%
train: 37%
person: 97%
person: 97%
person: 96%
person: 92%
person: 91%
person: 91%
person: 90%
person: 88%
```

```
person: 88%
person: 86%
person: 78%
person: 78%
person: 55%
person: 36%
person: 26%

FPS:25.9        AVG_FPS:27.5

 cvWriteFrame
Objects:

handbag: 76%
train: 34%
person: 98%
person: 97%
person: 96%
```

```
# download the video with detections shown
download('results.avi')
```

Google Drive Video

Here is how to run detector on video stored in Google Drive and save video straight to Google drive as well!

Note: You will have to change the paths to where your video is stored within your Google Drive and to where you want the resulting video stored. I have a videos folder in the home directory of my Google Drive.

```
!./darknet detector demo cfg/coco.data cfg/yolov4.cfg yolov4.weights -dont_show /mydrive/videos/test.mp4 -i 0 -out_filename /mydrive/
```

Step 8: Customize YOLOv4 with the different command line flags.

Darknet and YOLOv4 have a lot of command line flags you can add to your '!./darknet detector ...' to allow it to be customizeable and flexible.

I will show a few examples of these flags that you can take advantage of! Feel free to mix and match them together to customize your detections in any way you want.

Threshold Flag

There is a flag '-thresh' you can use to add a threshold for confidences on the detections. Only detections with a confidence level above the threshold you set will be returned.

In the example below we run darknet with YOLOv4 without a threshold on the test image dog.jpg. The image returns four detections, the lowest confidence being on the pottedplant with 33%.

If we add the '-thresh 0.5' flag this will only output three detections as now pottedplant falls below the threshold and is ignored.

Check it out below!

```
# this is ran without the threshold flag set
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights data/dog.jpg
imShow('predictions.jpg')
```

```
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 8, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 1    608 x 608 x   3 ->  608 x 608 x  32 0.639 BF
   1 conv     64       3 x 3/ 2    608 x 608 x  32 ->  304 x 304 x  64 3.407 BF
   2 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   3 route  1                                     ->  304 x 304 x  64
   4 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   5 conv     32       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  32 0.379 BF
   6 conv     64       3 x 3/ 1    304 x 304 x  32 ->  304 x 304 x  64 3.407 BF
   7 Shortcut Layer: 4,  wt = 0, wn = 0, outputs: 304 x 304 x  64 0.006 BF
   8 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   9 route  8 2                                   ->  304 x 304 x 128
  10 conv     64       1 x 1/ 1    304 x 304 x 128 ->  304 x 304 x  64 1.514 BF
  11 conv    128       3 x 3/ 2    304 x 304 x  64 ->  152 x 152 x 128 3.407 BF
  12 conv     64       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x  64 0.379 BF
  13 route  11                                    ->  152 x 152 x 128
  14 conv     64       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x  64 0.379 BF
  15 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  16 conv     64       3 x 3/ 1    152 x 152 x  64 ->  152 x 152 x  64 1.703 BF
  17 Shortcut Layer: 14,  wt = 0, wn = 0, outputs: 152 x 152 x  64 0.001 BF
  18 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  19 conv     64       3 x 3/ 1    152 x 152 x  64 ->  152 x 152 x  64 1.703 BF
  20 Shortcut Layer: 17,  wt = 0, wn = 0, outputs: 152 x 152 x  64 0.001 BF
  21 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  22 route  21 12                                 ->  152 x 152 x 128
  23 conv    128       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x 128 0.757 BF
  24 conv    256       3 x 3/ 2    152 x 152 x 128 ->   76 x  76 x 256 3.407 BF
  25 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
  26 route  24                                    ->   76 x  76 x 256
  27 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
  28 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
  29 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
  30 Shortcut Layer: 27,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
  31 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
```

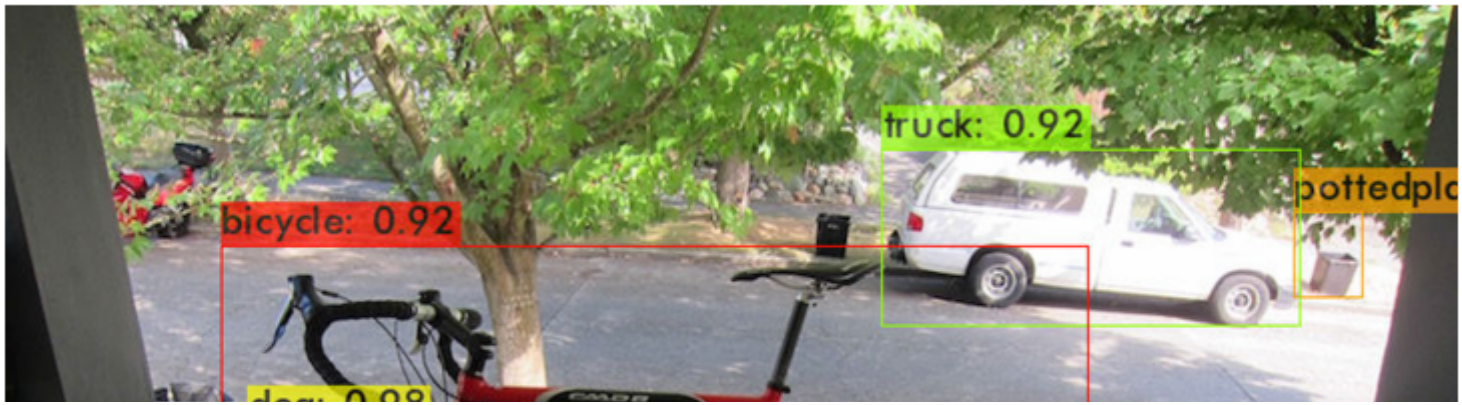
```
32 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
33 Shortcut Layer: 30,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
34 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
35 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
36 Shortcut Layer: 33,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
37 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
38 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
39 Shortcut Layer: 36,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
40 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
41 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
42 Shortcut Layer: 39,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
43 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
44 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
45 Shortcut Layer: 42,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
46 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
47 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
48 Shortcut Layer: 45,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
49 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
50 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
51 Shortcut Layer: 48,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
52 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
53 route  52 25                                   ->   76 x  76 x 256
54 conv    256       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 256 0.757 BF
55 conv    512       3 x 3/ 2     76 x  76 x 256 ->   38 x  38 x 512 3.407 BF
56 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
57 route  55                                      ->   38 x  38 x 512
58 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
59 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
60 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
61 Shortcut Layer: 58,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
62 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
63 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
64 Shortcut Layer: 61,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
65 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
66 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
67 Shortcut Layer: 64,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
68 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
69 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
70 Shortcut Layer: 67,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
71 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
72 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
```

```
 73 Shortcut Layer: 70,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 74 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 75 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 76 Shortcut Layer: 73,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 77 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 78 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 79 Shortcut Layer: 76,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 80 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 81 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 82 Shortcut Layer: 79,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 83 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 84 route  83 56                                   ->   38 x  38 x 512
 85 conv    512       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 512 0.757 BF
 86 conv   1024       3 x 3/ 2     38 x  38 x 512 ->   19 x  19 x1024 3.407 BF
 87 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 88 route  86                                      ->   19 x  19 x1024
 89 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 90 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
 91 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
 92 Shortcut Layer: 89,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
 93 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
 94 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
 95 Shortcut Layer: 92,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
 96 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
 97 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
 98 Shortcut Layer: 95,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
 99 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
100 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
101 Shortcut Layer: 98,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
102 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
103 route  102 87                                 ->   19 x  19 x1024
104 conv   1024       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x1024 0.757 BF
105 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
106 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
107 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
108 max               5x 5/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.005 BF
109 route  107                                       ->   19 x  19 x 512
110 max               9x 9/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.015 BF
111 route  107                                       ->   19 x  19 x 512
112 max              13x13/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.031 BF
113 route  112 110 108 107                        ->   19 x  19 x2048
```

```
 114 conv    512       1 x 1/ 1     19 x  19 x2048 ->   19 x  19 x 512 0.757 BF
 115 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 116 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 117 conv    256       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 256 0.095 BF
 118 upsample                2x    19 x  19 x 256 ->   38 x  38 x 256
 119 route  85                                      ->   38 x  38 x 512
 120 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 121 route  120 118                                 ->   38 x  38 x 512
 122 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 123 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 124 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 125 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 126 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 127 conv    128       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 128 0.095 BF
 128 upsample                2x    38 x  38 x 128 ->   76 x  76 x 128
 129 route  54                                      ->   76 x  76 x 256
 130 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 131 route  130 128                                 ->   76 x  76 x 256
 132 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 133 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 134 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 135 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 136 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 137 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 138 conv    255       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 255 0.754 BF
 139 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.20
nms_kind: greedynms (1), beta = 0.600000
 140 route  136                                     ->   76 x  76 x 128
 141 conv    256       3 x 3/ 2     76 x  76 x 128 ->   38 x  38 x 256 0.852 BF
 142 route  141 126                                 ->   38 x  38 x 512
 143 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 144 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 145 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 146 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 147 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 148 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 149 conv    255       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 255 0.377 BF
 150 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.10
nms_kind: greedynms (1), beta = 0.600000
 151 route  147                                     ->   38 x  38 x 256
```

```
 152 conv    512       3 x 3/ 2     38 x  38 x 256 ->   19 x  19 x 512 0.852 BF
 153 route  152 116                                ->   19 x  19 x1024
 154 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 155 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 156 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 157 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 158 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 159 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 160 conv    255       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 255 0.189 BF
 161 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 128.459
avg_outputs = 1068395
 Allocate additional workspace_size = 52.43 MB
Loading weights from yolov4.weights...
 seen 64, trained: 32032 K-images (500 Kilo-batches_64)
Done! Loaded 162 layers from weights-file
 Detection layer: 139 - type = 28
 Detection layer: 150 - type = 28
 Detection layer: 161 - type = 28
data/dog.jpg: Predicted in 54.657000 milli-seconds.
bicycle: 92%
dog: 98%
truck: 92%
pottedplant: 33%
Unable to init server: Could not connect: Connection refused

(predictions:4544): Gtk-WARNING **: 03:57:42.800: cannot open display:
```




```
# same detections but ran with the threshold flag set to 0.5 (pottedplant is no longer detected!)
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights data/dog.jpg -thresh 0.5
imShow('predictions.jpg')
```

```
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 8, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 1    608 x 608 x   3 ->  608 x 608 x  32 0.639 BF
   1 conv     64       3 x 3/ 2    608 x 608 x  32 ->  304 x 304 x  64 3.407 BF
   2 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   3 route  1                                     ->  304 x 304 x  64
   4 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   5 conv     32       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  32 0.379 BF
   6 conv     64       3 x 3/ 1    304 x 304 x  32 ->  304 x 304 x  64 3.407 BF
   7 Shortcut Layer: 4,  wt = 0, wn = 0, outputs: 304 x 304 x  64 0.006 BF
   8 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   9 route  8 2                                   ->  304 x 304 x 128
  10 conv     64       1 x 1/ 1    304 x 304 x 128 ->  304 x 304 x  64 1.514 BF
  11 conv    128       3 x 3/ 2    304 x 304 x  64 ->  152 x 152 x 128 3.407 BF
  12 conv     64       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x  64 0.379 BF
  13 route  11                                    ->  152 x 152 x 128
  14 conv     64       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x  64 0.379 BF
  15 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  16 conv     64       3 x 3/ 1    152 x 152 x  64 ->  152 x 152 x  64 1.703 BF
  17 Shortcut Layer: 14,  wt = 0, wn = 0, outputs: 152 x 152 x  64 0.001 BF
  18 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  19 conv     64       3 x 3/ 1    152 x 152 x  64 ->  152 x 152 x  64 1.703 BF
  20 Shortcut Layer: 17,  wt = 0, wn = 0, outputs: 152 x 152 x  64 0.001 BF
  21 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  22 route  21 12                                 ->  152 x 152 x 128
  23 conv    128       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x 128 0.757 BF
  24 conv    256       3 x 3/ 2    152 x 152 x 128 ->   76 x  76 x 256 3.407 BF
  25 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
  26 route  24                                    ->   76 x  76 x 256
  27 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
  28 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
  29 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
  30 Shortcut Layer: 27,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
  31 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
```

```
 32 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 33 Shortcut Layer: 30,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 34 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 35 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 36 Shortcut Layer: 33,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 37 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 38 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 39 Shortcut Layer: 36,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 40 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 41 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 42 Shortcut Layer: 39,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 43 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 44 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 45 Shortcut Layer: 42,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 46 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 47 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 48 Shortcut Layer: 45,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 49 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 50 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 51 Shortcut Layer: 48,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 52 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 53 route  52 25                                  ->   76 x  76 x 256
 54 conv    256       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 256 0.757 BF
 55 conv    512       3 x 3/ 2     76 x  76 x 256 ->   38 x  38 x 512 3.407 BF
 56 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 57 route  55                                     ->   38 x  38 x 512
 58 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 59 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 60 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 61 Shortcut Layer: 58,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 62 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 63 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 64 Shortcut Layer: 61,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 65 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 66 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 67 Shortcut Layer: 64,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 68 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 69 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 70 Shortcut Layer: 67,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 71 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 72 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
```

```
 73 Shortcut Layer: 70,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 74 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 75 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 76 Shortcut Layer: 73,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 77 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 78 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 79 Shortcut Layer: 76,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 80 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 81 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 82 Shortcut Layer: 79,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 83 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 84 route  83 56                                   ->   38 x  38 x 512
 85 conv    512       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 512 0.757 BF
 86 conv   1024       3 x 3/ 2     38 x  38 x 512 ->   19 x  19 x1024 3.407 BF
 87 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 88 route  86                                      ->   19 x  19 x1024
 89 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 90 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
 91 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
 92 Shortcut Layer: 89,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
 93 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
 94 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
 95 Shortcut Layer: 92,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
 96 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
 97 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
 98 Shortcut Layer: 95,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
 99 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
100 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
101 Shortcut Layer: 98,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
102 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
103 route  102 87                                 ->   19 x  19 x1024
104 conv   1024       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x1024 0.757 BF
105 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
106 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
107 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
108 max               5x 5/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.005 BF
109 route  107                                          ->   19 x  19 x 512
110 max               9x 9/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.015 BF
111 route  107                                          ->   19 x  19 x 512
112 max              13x13/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.031 BF
113 route  112 110 108 107                        ->   19 x  19 x2048
```

```
 114 conv    512       1 x 1/ 1     19 x  19 x2048 ->   19 x  19 x 512 0.757 BF
 115 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 116 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 117 conv    256       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 256 0.095 BF
 118 upsample                2x    19 x  19 x 256 ->   38 x  38 x 256
 119 route  85                                      ->   38 x  38 x 512
 120 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 121 route  120 118                                 ->   38 x  38 x 512
 122 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 123 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 124 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 125 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 126 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 127 conv    128       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 128 0.095 BF
 128 upsample                2x    38 x  38 x 128 ->   76 x  76 x 128
 129 route  54                                      ->   76 x  76 x 256
 130 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 131 route  130 128                                 ->   76 x  76 x 256
 132 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 133 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 134 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 135 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 136 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 137 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 138 conv    255       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 255 0.754 BF
 139 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.20
nms_kind: greedynms (1), beta = 0.600000
 140 route  136                                     ->   76 x  76 x 128
 141 conv    256       3 x 3/ 2     76 x  76 x 128 ->   38 x  38 x 256 0.852 BF
 142 route  141 126                                 ->   38 x  38 x 512
 143 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 144 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 145 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 146 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 147 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 148 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 149 conv    255       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 255 0.377 BF
 150 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.10
nms_kind: greedynms (1), beta = 0.600000
 151 route  147                                     ->   38 x  38 x 256
```

```
 152 conv    512       3 x 3/ 2     38 x  38 x 256 ->   19 x  19 x 512 0.852 BF
 153 route  152 116                                ->   19 x  19 x1024
 154 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 155 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 156 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 157 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 158 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 159 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 160 conv    255       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 255 0.189 BF
 161 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 128.459
avg_outputs = 1068395
 Allocate additional workspace_size = 52.43 MB
Loading weights from yolov4.weights...
 seen 64, trained: 32032 K-images (500 Kilo-batches_64)
Done! Loaded 162 layers from weights-file
 Detection layer: 139 - type = 28
 Detection layer: 150 - type = 28
 Detection layer: 161 - type = 28
data/dog.jpg: Predicted in 54.460000 milli-seconds.
bicycle: 92%
dog: 98%
truck: 92%
Unable to init server: Could not connect: Connection refused

(predictions:4560): Gtk-WARNING **: 03:58:04.546: cannot open display:
```

Output Bounding Box Coordinates

You can output bounding box coordinates for each detection with the flag '-ext_output'. This external outputs flag will give you a few extra details about each detection within an image.

Check it out below!

```
# darknet run with external output flag to print bounding box coordinates
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights data/person.jpg -ext_output
imShow('predictions.jpg')
```

```
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 8, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 1    608 x 608 x   3 ->  608 x 608 x  32 0.639 BF
   1 conv     64       3 x 3/ 2    608 x 608 x  32 ->  304 x 304 x  64 3.407 BF
   2 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   3 route  1                                     ->  304 x 304 x  64
   4 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   5 conv     32       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  32 0.379 BF
   6 conv     64       3 x 3/ 1    304 x 304 x  32 ->  304 x 304 x  64 3.407 BF
   7 Shortcut Layer: 4,  wt = 0, wn = 0, outputs: 304 x 304 x  64 0.006 BF
   8 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   9 route  8 2                                   ->  304 x 304 x 128
  10 conv     64       1 x 1/ 1    304 x 304 x 128 ->  304 x 304 x  64 1.514 BF
  11 conv    128       3 x 3/ 2    304 x 304 x  64 ->  152 x 152 x 128 3.407 BF
  12 conv     64       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x  64 0.379 BF
  13 route  11                                    ->  152 x 152 x 128
  14 conv     64       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x  64 0.379 BF
  15 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  16 conv     64       3 x 3/ 1    152 x 152 x  64 ->  152 x 152 x  64 1.703 BF
  17 Shortcut Layer: 14,  wt = 0, wn = 0, outputs: 152 x 152 x  64 0.001 BF
  18 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  19 conv     64       3 x 3/ 1    152 x 152 x  64 ->  152 x 152 x  64 1.703 BF
  20 Shortcut Layer: 17,  wt = 0, wn = 0, outputs: 152 x 152 x  64 0.001 BF
  21 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  22 route  21 12                                 ->  152 x 152 x 128
  23 conv    128       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x 128 0.757 BF
  24 conv    256       3 x 3/ 2    152 x 152 x 128 ->   76 x  76 x 256 3.407 BF
  25 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
  26 route  24                                    ->   76 x  76 x 256
  27 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
  28 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
  29 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
  30 Shortcut Layer: 27,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
  31 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
```

```
 32 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 33 Shortcut Layer: 30,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 34 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 35 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 36 Shortcut Layer: 33,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 37 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 38 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 39 Shortcut Layer: 36,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 40 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 41 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 42 Shortcut Layer: 39,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 43 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 44 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 45 Shortcut Layer: 42,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 46 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 47 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 48 Shortcut Layer: 45,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 49 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 50 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 51 Shortcut Layer: 48,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 52 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 53 route  52 25                                  ->   76 x  76 x 256
 54 conv    256       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 256 0.757 BF
 55 conv    512       3 x 3/ 2     76 x  76 x 256 ->   38 x  38 x 512 3.407 BF
 56 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 57 route  55                                     ->   38 x  38 x 512
 58 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 59 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 60 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 61 Shortcut Layer: 58,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 62 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 63 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 64 Shortcut Layer: 61,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 65 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 66 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 67 Shortcut Layer: 64,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 68 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 69 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 70 Shortcut Layer: 67,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 71 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 72 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
```

```
 73 Shortcut Layer: 70,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 74 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 75 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 76 Shortcut Layer: 73,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 77 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 78 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 79 Shortcut Layer: 76,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 80 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 81 conv    256       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 256 1.703 BF
 82 Shortcut Layer: 79,  wt = 0, wn = 0, outputs:  38 x  38 x 256 0.000 BF
 83 conv    256       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 256 0.189 BF
 84 route  83 56                                   ->   38 x  38 x 512
 85 conv    512       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 512 0.757 BF
 86 conv   1024       3 x 3/ 2     38 x  38 x 512 ->   19 x  19 x1024 3.407 BF
 87 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 88 route  86                                      ->   19 x  19 x1024
 89 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 90 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
 91 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
 92 Shortcut Layer: 89,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
 93 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
 94 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
 95 Shortcut Layer: 92,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
 96 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
 97 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
 98 Shortcut Layer: 95,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
 99 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
100 conv    512       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x 512 1.703 BF
101 Shortcut Layer: 98,  wt = 0, wn = 0, outputs:  19 x  19 x 512 0.000 BF
102 conv    512       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.189 BF
103 route  102 87                                 ->   19 x  19 x1024
104 conv   1024       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x1024 0.757 BF
105 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
106 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
107 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
108 max               5x 5/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.005 BF
109 route  107                                          ->   19 x  19 x 512
110 max               9x 9/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.015 BF
111 route  107                                          ->   19 x  19 x 512
112 max              13x13/ 1     19 x  19 x 512 ->   19 x  19 x 512 0.031 BF
113 route  112 110 108 107                        ->   19 x  19 x2048
```

```
 114 conv    512       1 x 1/ 1     19 x  19 x2048 ->   19 x  19 x 512 0.757 BF
 115 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 116 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 117 conv    256       1 x 1/ 1     19 x  19 x 512 ->   19 x  19 x 256 0.095 BF
 118 upsample                2x    19 x  19 x 256 ->   38 x  38 x 256
 119 route  85                                      ->   38 x  38 x 512
 120 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 121 route  120 118                                 ->   38 x  38 x 512
 122 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 123 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 124 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 125 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 126 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 127 conv    128       1 x 1/ 1     38 x  38 x 256 ->   38 x  38 x 128 0.095 BF
 128 upsample                2x    38 x  38 x 128 ->   76 x  76 x 128
 129 route  54                                      ->   76 x  76 x 256
 130 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 131 route  130 128                                 ->   76 x  76 x 256
 132 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 133 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 134 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 135 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 136 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 137 conv    256       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 256 3.407 BF
 138 conv    255       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 255 0.754 BF
 139 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.20
nms_kind: greedynms (1), beta = 0.600000
 140 route  136                                     ->   76 x  76 x 128
 141 conv    256       3 x 3/ 2     76 x  76 x 128 ->   38 x  38 x 256 0.852 BF
 142 route  141 126                                 ->   38 x  38 x 512
 143 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 144 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 145 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 146 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 147 conv    256       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 256 0.379 BF
 148 conv    512       3 x 3/ 1     38 x  38 x 256 ->   38 x  38 x 512 3.407 BF
 149 conv    255       1 x 1/ 1     38 x  38 x 512 ->   38 x  38 x 255 0.377 BF
 150 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.10
nms_kind: greedynms (1), beta = 0.600000
 151 route  147                                     ->   38 x  38 x 256
```

```
 152 conv    512       3 x 3/ 2     38 x  38 x 256 ->   19 x  19 x 512 0.852 BF
 153 route  152 116                                ->   19 x  19 x1024
 154 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 155 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 156 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 157 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 158 conv    512       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 512 0.379 BF
 159 conv   1024       3 x 3/ 1     19 x  19 x 512 ->   19 x  19 x1024 3.407 BF
 160 conv    255       1 x 1/ 1     19 x  19 x1024 ->   19 x  19 x 255 0.189 BF
 161 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 128.459
avg_outputs = 1068395
 Allocate additional workspace_size = 52.43 MB
Loading weights from yolov4.weights...
 seen 64, trained: 32032 K-images (500 Kilo-batches_64)
Done! Loaded 162 layers from weights-file
 Detection layer: 139 - type = 28
 Detection layer: 150 - type = 28
 Detection layer: 161 - type = 28
data/person.jpg: Predicted in 54.546000 milli-seconds.
dog: 99%        (left_x:   62   top_y:  265   width:  142   height:   80)
person: 100%    (left_x:  194   top_y:   98   width:   78   height:  281)
horse: 98%      (left_x:  406   top_y:  140   width:  196   height:  204)
```

Don't Show Image

You can add the flag '-dont_show' to not have the image outputted after running darknet. This doesn't really affect anything when running in Colab as the image is unable to output properly straight from darknet anyways. However, by adding the -dont_show flag you will get rid of the following warning from showing.

```
Unable to init server: Could not connect: Connection refused

(predictions:1850): Gtk-WARNING **: 17:01:00.687: cannot open display:
```

This is an important flag to have when running darknet with YOLOv4 on video as it will suppress having the video shown.

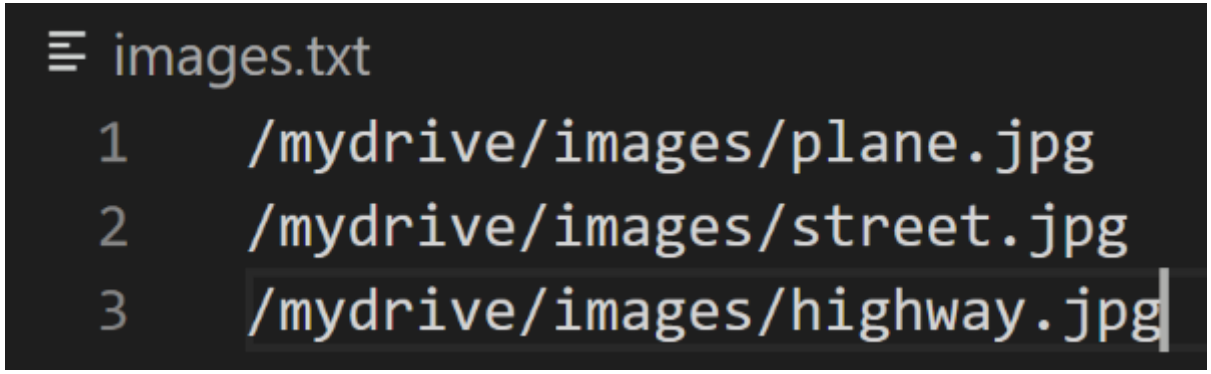
```
# running darknet with dont show flag set (no longer get warnings)
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights data/person.jpg -dont_show
```

```
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 8, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 1    608 x 608 x   3 ->  608 x 608 x  32 0.639 BF
   1 conv     64       3 x 3/ 2    608 x 608 x  32 ->  304 x 304 x  64 3.407 BF
   2 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   3 route  1                                     ->  304 x 304 x  64
   4 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   5 conv     32       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  32 0.379 BF
   6 conv     64       3 x 3/ 1    304 x 304 x  32 ->  304 x 304 x  64 3.407 BF
   7 Shortcut Layer: 4,  wt = 0, wn = 0, outputs: 304 x 304 x  64 0.006 BF
   8 conv     64       1 x 1/ 1    304 x 304 x  64 ->  304 x 304 x  64 0.757 BF
   9 route  8 2                                   ->  304 x 304 x 128
  10 conv     64       1 x 1/ 1    304 x 304 x 128 ->  304 x 304 x  64 1.514 BF
  11 conv    128       3 x 3/ 2    304 x 304 x  64 ->  152 x 152 x 128 3.407 BF
  12 conv     64       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x  64 0.379 BF
  13 route  11                                    ->  152 x 152 x 128
  14 conv     64       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x  64 0.379 BF
  15 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  16 conv     64       3 x 3/ 1    152 x 152 x  64 ->  152 x 152 x  64 1.703 BF
  17 Shortcut Layer: 14,  wt = 0, wn = 0, outputs: 152 x 152 x  64 0.001 BF
  18 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  19 conv     64       3 x 3/ 1    152 x 152 x  64 ->  152 x 152 x  64 1.703 BF
  20 Shortcut Layer: 17,  wt = 0, wn = 0, outputs: 152 x 152 x  64 0.001 BF
  21 conv     64       1 x 1/ 1    152 x 152 x  64 ->  152 x 152 x  64 0.189 BF
  22 route  21 12                                ->  152 x 152 x 128
  23 conv    128       1 x 1/ 1    152 x 152 x 128 ->  152 x 152 x 128 0.757 BF
  24 conv    256       3 x 3/ 2    152 x 152 x 128 ->   76 x  76 x 256 3.407 BF
  25 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
  26 route  24                                    ->   76 x  76 x 256
```

```
 27 conv    128       1 x 1/ 1     76 x  76 x 256 ->   76 x  76 x 128 0.379 BF
 28 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 29 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 30 Shortcut Layer: 27,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 31 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 32 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 33 Shortcut Layer: 30,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 34 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 35 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 36 Shortcut Layer: 33,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 37 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 38 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 39 Shortcut Layer: 36,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 40 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 41 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 42 Shortcut Layer: 39,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 43 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 44 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 45 Shortcut Layer: 42,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
 46 conv    128       1 x 1/ 1     76 x  76 x 128 ->   76 x  76 x 128 0.189 BF
 47 conv    128       3 x 3/ 1     76 x  76 x 128 ->   76 x  76 x 128 1.703 BF
 48 Shortcut Layer: 45,  wt = 0, wn = 0, outputs:  76 x  76 x 128 0.001 BF
```

Step 9: Multiple Images at Once

YOLOv4 object detections can be run on multiple images at once. This is done through having a text file which has the paths to several images that you want to have the detector run on.



The .txt file should be in this format. One path to an image per line.

This file is stored to my Google Drive root directory and holds the path to three images within my Google Drive images folder.

Save Results to .JSON File

Here is an example of saving the multiple image detections to a .JSON file.

```
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights -ext_output -dont_show -out result.json < /data/ImageLinks.txt
```

```
     /bin/bash: /data/ImageLinks.txt: No such file or directory
```

```
download('result.json')
```

Saving Results to a .txt File

You can also save the results of running multiple images to a text file.

```
!./darknet detector test cfg/coco.data cfg/yolov4.cfg yolov4.weights -dont_show -ext_output < data/ImageLinks.txt > result.txt
```

```
     /bin/bash: data/ImageLinks.txt: No such file or directory
```

```
download('result.txt')
```

Assignment

The assignment begins below. You will train a YOLOv4-tiny model using a small custom dataset. First the data has to be converted into a format usable by darknet.

Setup Data

Download the dataset from UR courses or from https://www.kaggle.com/datasets/tarunbisht11/yolo-animal-detection-small. Upload the dataset to the Google Collab instance. See step 6 in the tutorial above.

The data first needs to be unzipped. Upload `dataset.zip` to the `/content/` (the default folder opened in colab) folder.

Unzip `dataset.zip` with this command:

```
%cd /content/
!unzip archive.zip -d .
```

```
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_071.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_071.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_073.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_073.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_075.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_075.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_076.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_076.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_081.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_081.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_083.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_083.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_084.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_084.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_086.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_086.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_088.jpg
```

```
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_088.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_089.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_089.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_090.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_090.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_092.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_092.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_094.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_094.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_095.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_095.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_097.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_097.xml
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_098.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_and_cats_098.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_002.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_002.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_004.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_004.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_005.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_005.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_007.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_007.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_017.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_017.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_018.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_018.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_019.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_019.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_020.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_020.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_022.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_022.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_026.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_026.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_049.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_049.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_053.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_053.xml
  inflating: ./yolo-animal-detection-small/train/dogs_groups_057.jpg
  inflating: ./yolo-animal-detection-small/train/dogs_groups_057.xml
```

Next create `catdogmonkey.names` file which has the names of the object classes on each line. Save this to the `darknet/data` folder.

```
!echo "cat" > darknet/data/catdogmonkey.names
!echo "dog" >> darknet/data/catdogmonkey.names
!echo "monkey" >> darknet/data/catdogmonkey.names
```

Next create a `catdogmonkey` folder in `darknet/data`. This file gives darknet information about our training data. Our trained models will be saved in the `backup` folder.

```
!echo "classes = 3" > darknet/data/catdogmonkey.data
!echo "train =/content/yolo-animal-detection-small/train/train.txt" >> darknet/data/catdogmonkey.data
!echo "valid =/content/yolo-animal-detection-small/test/test.txt" >> darknet/data/catdogmonkey.data
!echo "names =data/catdogmonkey.names" >> darknet/data/catdogmonkey.data
!echo "backup = backup" >> darknet/data/catdogmonkey.data
```

The dataset contains bounding box information, but it is in the wrong format for darknet. Write a script to generate .txt files for each image in the train and test folders.

- Each .txt file should contain the bounding box labels for each object in the image.
- Each txt file should have the same name as the image. (ex: `cats_001.jpg` should have a `cats_001.txt` file)
- Each line of the .txt file should contain a single bounding box in this format:

  `<class_index> <relative_center_x> <relative_center_y> <relative_width> <relative_height>`

  i. Example for `dogs_and_cats_000.txt` (Verify your conversion is correct before training):

```
1 0.3122 0.5237 0.6202 0.9525
0 0.8069 0.4572 0.3860 0.8765
```

> > ii. You will have to convert to from absolute min/max boxes to center/size boxes that are relative to the image size.

- You can use either the dataset's xml files for each image or the test/train.csv files

Also generate a train.txt and a test.txt file which contains image paths in the train and test sets. The paths should be relative to the darknet folder or absolute.

Example train.txt:

```
../dataset/catdogmonkey/yolo-animal-detection-small/train/cats_001.jpg
../dataset/catdogmonkey/yolo-animal-detection-small/train/cats_002.jpg
../dataset/catdogmonkey/yolo-animal-detection-small/train/cats_003.jpg
...
```

OR (absolute path in Google Colab):

```
content/dataset/catdogmonkey/yolo-animal-detection-small/train/cats_001.jpg
content/dataset/catdogmonkey/yolo-animal-detection-small/train/cats_002.jpg
content/dataset/catdogmonkey/yolo-animal-detection-small/train/cats_003.jpg
...
```

Fill out the codeblock below:

```
# Your answer can go here.
import pandas as pd
train_dataset=pd.read_csv('train.csv')
test_dataset=pd.read_csv('test.csv')
def class_number(animal):
   if(animal=="cat"):
     return 0
```

```
    elif(animal=="dog"):
      return 1
    else:
      return 2


for index, row in train_dataset.iterrows():
  i=1
  class_index=class_number(row["class"])
  relative_width=(row["xmax"]/row["width"])-(row["xmin"]/row["width"])
  relative_height=(row["ymax"]/row["height"])-(row["ymin"]/row["height"])
  relative_center_x=((row["xmax"]/row["width"])+(row["xmin"]/row["width"]))/2
  relative_center_y=((row["ymax"]/row["height"])+(row["ymin"]/row["height"]))/2
  filename=row["filename"].strip('.jpg')
  filepath="/content/yolo-animal-detection-small/train/"
  filename_t=filename+".txt"
  filename_i=filename+".jpg"
  filepath=filepath+filename_i
  content=str(class_index)+" "+str(relative_center_x)+" "+str(relative_center_y)+" "+str(relative_width)+" "+str(relative_height)
  !echo { content } >> yolo-animal-detection-small/train/{ filename_t }
  !echo { filepath } >> yolo-animal-detection-small/train/train.txt


for index, row in test_dataset.iterrows():
  i=1
  class_index=class_number(row["class"])
  relative_width=(row["xmax"]/row["width"])-(row["xmin"]/row["width"])
  relative_height=(row["ymax"]/row["height"])-(row["ymin"]/row["height"])
  relative_center_x=((row["xmax"]/row["width"])+(row["xmin"]/row["width"]))/2
  relative_center_y=((row["ymax"]/row["height"])+(row["ymin"]/row["height"]))/2
  filename=row["filename"].strip('.jpg')
  filepath="/content/yolo-animal-detection-small/test/"
  filename_t=filename+".txt"
  filename_i=filename+".jpg"
  filepath=filepath+filename_i
  content=str(class_index)+" "+str(relative_center_x)+" "+str(relative_center_y)+" "+str(relative_width)+" "+str(relative_height)
  !echo { content } >> yolo-animal-detection-small/test/{ filename_t }
```

```
!echo { filepath } >> yolo-animal-detection-small/test/test.txt
```

In the `darknet/cfg/` folder there are `.cfg` files which stores the neural network structure. This needs to be modified to work with the custom dataset. For this assignment we will use a lower resolution of `224x224`.

Copy `yolov4-tiny-custom.cfg` and rename it to `yolov4-tiny-catdogmonkey.cfg` and make the following modifications.

1. Change `width=224` and `height=224`.
2. Change `max_batches=6000`
3. Change `steps=4800,5400`
4. For each `[yolo]` layer, change `classes=3`
5. In the `[convolutional]` layers that appear immediately before a `[yolo]` layer, change `filters = 24`. Filters is (5 + 3) * 3 = 24, because there are 5 box properties plus 3 object classes, each of which is predicted for 3 anchor boxes.

```
%cd /content/
#%cp darknet/cfg/yolov4-tiny-catdogmonkey.cfg darknet/cfg/yolov4-tiny-catdogmonkey.cfg

!sed -i 's/width=416/width=224/g' darknet/cfg/yolov4-tiny-catdogmonkey.cfg
!sed -i 's/height=416/height=224/g' darknet/cfg/yolov4-tiny-catdogmonkey.cfg
!sed -i 's/max_batches = 500200/max_batches = 6000/g' darknet/cfg/yolov4-tiny-catdogmonkey.cfg
!sed -i 's/steps=400000,450000/steps=4800,5400/g' darknet/cfg/yolov4-tiny-catdogmonkey.cfg
!sed -i 's/classes=80/classes=3/g' darknet/cfg/yolov4-tiny-catdogmonkey.cfg
!sed -i 's/filters=255/filters=24/g' darknet/cfg/yolov4-tiny-catdogmonkey.cfg

     /content
```

Training

Now we can begin training. First we will download the pretrained YOLOv4-Tiny weights to begin training from.

```
%cd /content/darknet/
!wget https://github.com/AlexeyAB/darknet/releases/download/yolov4/yolov4-tiny.conv.29
```

```
/content/darknet
--2022-05-30 03:00:26--  https://github.com/AlexeyAB/darknet/releases/download/yolov4/yolov4-tiny.conv.29
Resolving github.com (github.com)... 140.82.112.3
Connecting to github.com (github.com)|140.82.112.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/75388965/a876d846-96cd-4e18-9ba7-f384d9e
--2022-05-30 03:00:26--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/75388965/a876d846-96cd-4e
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.109.133, 185.199.111.133, 185.199.108.133, .
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 19789716 (19M) [application/octet-stream]
Saving to: ‘yolov4-tiny.conv.29’

yolov4-tiny.conv.29 100%[===================>]  18.87M  35.8MB/s    in 0.5s

2022-05-30 03:00:27 (35.8 MB/s) - ‘yolov4-tiny.conv.29’ saved [19789716/19789716]
```

Now we can begin training with the following command below. When training is complete, the weights will be saved in `darknet/backup`. Training took about an hour when tested.

```
%cd /content/darknet/
!./darknet detector train data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg yolov4-tiny.conv.29 -dont_show -map
```

```
 (next mAP calculation at 5600 iterations)

 Tensor Cores are used.
 Last accuracy mAP@0.50 = 58.51 %, best = 65.14 %
 5593: 0.243913, 0.302184 avg loss, 0.000026 rate, 0.228268 seconds, 357952 images, 0.033731 hours left
Loaded: 0.007130 seconds
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 30 Avg (IOU: 0.854115), count: 125, class_loss = 0.196247
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 37 Avg (IOU: 0.802175), count: 221, class_loss = 0.408256
 total_bbox = 2196768, rewritten_bbox = 0.788294 %
```

```
 (next mAP calculation at 5600 iterations)

 Tensor Cores are used.
 Last accuracy mAP@0.50 = 58.51 %, best = 65.14 %
 5594: 0.302437, 0.302210 avg loss, 0.000026 rate, 0.179215 seconds, 358016 images, 0.033652 hours left
Loaded: 0.038540 seconds
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 30 Avg (IOU: 0.849197), count: 113, class_loss = 0.17561
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 37 Avg (IOU: 0.808869), count: 257, class_loss = 0.58333
 total_bbox = 2197138, rewritten_bbox = 0.788344 %

 (next mAP calculation at 5600 iterations)

 Tensor Cores are used.
 Last accuracy mAP@0.50 = 58.51 %, best = 65.14 %
 5595: 0.379665, 0.309955 avg loss, 0.000026 rate, 0.197780 seconds, 358080 images, 0.033526 hours left
Loaded: 0.025781 seconds
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 30 Avg (IOU: 0.850829), count: 116, class_loss = 0.18670
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 37 Avg (IOU: 0.822082), count: 270, class_loss = 0.53732
 total_bbox = 2197524, rewritten_bbox = 0.788433 %

 (next mAP calculation at 5600 iterations)

 Tensor Cores are used.
 Last accuracy mAP@0.50 = 58.51 %, best = 65.14 %
 5596: 0.362197, 0.315179 avg loss, 0.000026 rate, 0.182783 seconds, 358144 images, 0.033456 hours left
Loaded: 0.054751 seconds
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 30 Avg (IOU: 0.857487), count: 118, class_loss = 0.21546
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 37 Avg (IOU: 0.842682), count: 242, class_loss = 0.39767
 total_bbox = 2197884, rewritten_bbox = 0.788395 %

 (next mAP calculation at 5600 iterations)

 Tensor Cores are used.
 Last accuracy mAP@0.50 = 58.51 %, best = 65.14 %
 5597: 0.306731, 0.314335 avg loss, 0.000026 rate, 0.151349 seconds, 358208 images, 0.033356 hours left
Loaded: 0.096716 seconds
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 30 Avg (IOU: 0.867827), count: 106, class_loss = 0.21304
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 37 Avg (IOU: 0.834047), count: 307, class_loss = 0.38619
 total_bbox = 2198297, rewritten_bbox = 0.788610 %

 (next mAP calculation at 5600 iterations)
```

```
 Tensor Cores are used.
 Last accuracy mAP@0.50 = 58.51 %, best = 65.14 %
 5598: 0.299782, 0.312879 avg loss, 0.000026 rate, 0.209097 seconds, 358272 images, 0.033253 hours left
Loaded: 0.017074 seconds
```

After the model is trained, you should immediately download the `_last.weights`, `_best.weights`, and `_final.weights` in the `backup` folder. The files stored your in Google Colab instance are temporary, so you should make sure you save them before they are removed. If your instance is restarted, you can just copy your saved files back into the `backup` folder.

If your session gets interupted for whatever reason, you can resume retraining from your last trained weights with this command:

```
%cd /content/darknet/
!./darknet detector train data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_last.weights -dont_
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 Prepare additional network for mAP calculation...
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
```

```
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                              1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                           ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                           ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                              1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                           ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                           ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                              ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                           ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
yolov4-tiny-catdogmonkey
```

```
%cd /content/darknet/
!./darknet detector test data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_final.weights /conte
imShow('predictions.jpg')
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max               2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                                1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                                ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                                ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                                1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                                ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                                ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
```

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                    ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                 ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_final.weights...
 seen 64, trained: 384 K-images (6 Kilo-batches_64)
```

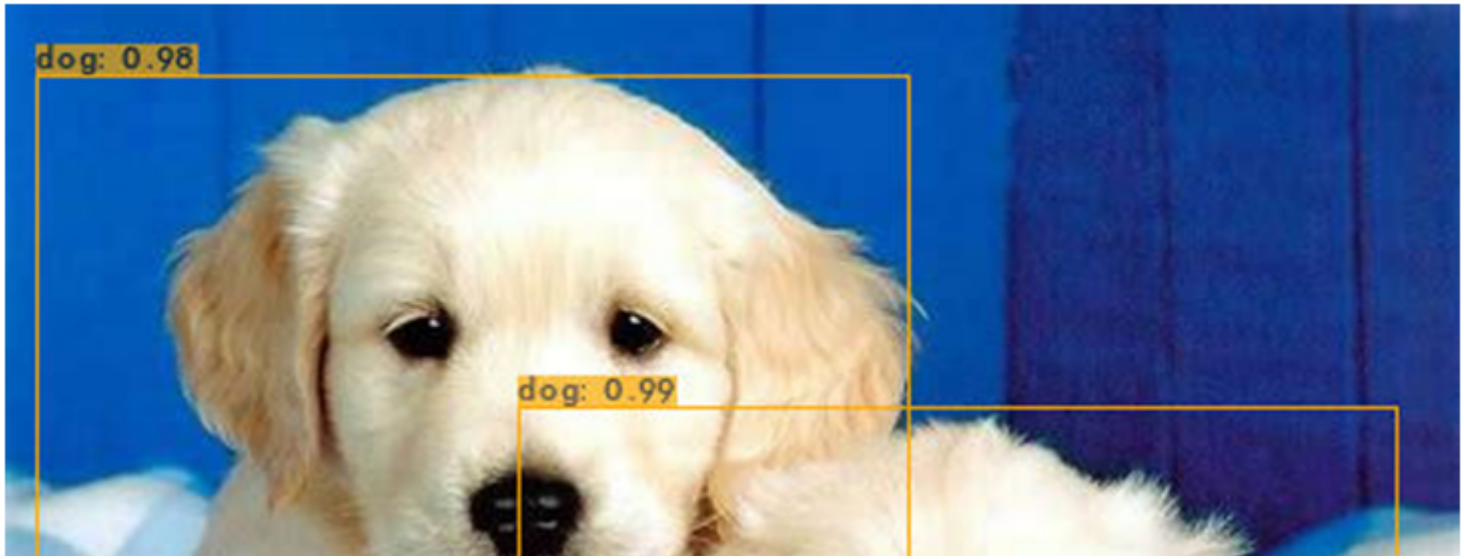
Now that the model is trained, test it out on some images from the test set.

```
/content/yolo-animal-detection-small/test/cats_000.jpg: Predicted in 3.241000 milli-seconds.
```

```
%cd /content/darknet/
!./darknet detector test data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_final.weights -dont_
imShow('predictions.jpg')
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                                1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                                ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                                ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                                1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                                ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                                ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
```

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                    ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                 ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_final.weights...
 seen 64, trained: 384 K-images (6 Kilo-batches_64)
Done! Loaded 38 layers from weights-file
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
/content/yolo-animal-detection-small/test/dogs_042.jpg: Predicted in 3.254000 milli-seconds.
dog: 98%
dog: 99%
```




```
%cd /content/darknet/
```

```
!./darknet detector test data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_final.weights -dont_
imShow('predictions.jpg')
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                                1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                                ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                                ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                                1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                                ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                                ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
```

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                     ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                  ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_final.weights...
 seen 64, trained: 384 K-images (6 Kilo-batches_64)
```

To measure the mAP of the model, run the following command:

```
 Detection layer: 37 - type = 28
```

```
%cd /content/darknet/
!./darknet detector map data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_best.weights -points
```

```
  6 route  5 4                                     ->   56 x  56 x  64
  7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
  8 route  2 7                                     ->   56 x  56 x 128
  9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
 10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
 11 route  10                                  1/2 ->   28 x  28 x  64
 12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
 13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
 14 route  13 12                                  ->   28 x  28 x 128
 15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
 16 route  10 15                                  ->   28 x  28 x 256
 17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
 18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
 19 route  18                                  1/2 ->   14 x  14 x 128
 20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
 21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
 22 route  21 20                                  ->   14 x  14 x 256
 23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
```

```
  24 route  18 23                                  ->   14 x  14 x 512
  25 max                2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                     ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                  ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_best.weights...
 seen 64, trained: 128 K-images (2 Kilo-batches_64)
Done! Loaded 38 layers from weights-file

 calculation mAP (mean average precision)...
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
116
 detections_count = 2216, unique_truth_count = 485
class_id = 0, name = cat, ap = 53.88%           (TP = 68, FP = 76)
class_id = 1, name = dog, ap = 88.20%           (TP = 88, FP = 51)
class_id = 2, name = monkey, ap = 50.10%        (TP = 82, FP = 8)

 for conf_thresh = 0.25, precision = 0.64, recall = 0.49, F1-score = 0.55
 for conf_thresh = 0.25, TP = 238, FP = 135, FN = 247, average IoU = 49.50 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.640621, or 64.06 %
Total Detection Time: 1 Seconds
```

Questions

Answer the following questions. To answer, either edit the notebook's text or code blocks.

1. What are the following statistics for your trained model with an IoU threshold (overlap threshold) of 0.5? (See the output of the map command above. Use the -iou_thresh parameter to change the threshold):

ANSWER:

overlap threshold	AP (cat)	AP (dog)	AP (monkey)	mAP
0.1	63.26%	90.25%	58.34%	70.62%
0.2	62.83%	90.25%	58.34%	70.47%
0.3	59.81%	90.25%	58.34%	69.47%
0.4	56.00%	90.25%	56.27%	67.50%
0.5	53.88%	88.20%	50.10%	64.06%
0.6	49.62%	83.43%	50.10%	61.05%
0.7	34.17%	66.16%	31.58%	43.97%
0.8	7.88%	12.56%	15.31%	11.92%
0.9	2.27%	0.00%	0.21%	0.83%

2. What is the contents of your generated `dataset/train/dogs_and_cats_002.txt` ?

ANSWER:

```
Your answer can do here.
0 0.30801687763713076 0.6814516129032258 0.4472573839662447 0.5645161290322581
1 0.7510548523206751 0.6370967741935484 0.45569620253164556 0.6935483870967742
```

```
%cd /content/darknet/
```

```
!./darknet detector test data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_best.weights /conte
imShow('predictions.jpg')
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                                1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                                ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                                ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                                1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                                ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                                ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
```

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                    ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                 ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_best.weights...
 seen 64, trained: 128 K-images (2 Kilo-batches_64)
Done! Loaded 38 layers from weights-file
 Detection layer: 30 - type = 28
```

3. Test your trained model on this Image. At what detection threshold does the model no longer detect the cat and dog? (Use the -thresh flag to change the detection threshold)

3. If vase class = 0 and dog class = 1, annotate the image with approximate bounding box coordinates (can be done visually) in the darknet labeling format.

ANSWER:

```
0 0.5 0.5 0.30 0.75

1 0.8 0.35 0.15 0.34
```

4. Find 6 images from the COCO dataset. You can search for images by class. Try to find 3 cat/dog images that you model can detect and 3 animals images that contain an object your model can't detect such as zebra or horse .Remember that the model you trained only has

limited training data so try to pick simple images with large objects.

> Fill out the following table with 6 image URLs from the COCO dataset. Search for images and press the URL button to get the image URL.

ANSWER:

Your answer can go here.

> Test your model on these images and fill out the following table. Use the -ext_output when testing to output the bounding box coordinates. Add the main object class or classes to the table for each image above. Add the bounding box with the highest confidence to the table (Detected class, confidence, x, y, width and height, correct). For the "correct?" column you can enter `{ yes, sortof, no }`. You can leave fields blank if it did not detect anything.

Img	Main objs	Detected	Conf	x	y	width	height	Correct
1 https://farm1.staticflickr.com/117/304657399_959f87a925_z.jpg	dog	dog	98%	152	45	253	509	yes
2 https://farm4.staticflickr.com/3551/3360603096_ce13ea2e94_z.jpg	cat	cat	63%	-26	2	475	458	yes
3 https://farm5.staticflickr.com/4011/4451615442_53e6a82607_z.jpg	cat	cat	77%	67	-26	444	543	yes
4 https://farm8.staticflickr.com/7223/7383901372_6bd3046b2c_z.jpg	bear	dog	53%	64	17	374	423	no
5 https://farm1.staticflickr.com/64/163712426_611deaebe0_z.jpg	zebra	cat	75%	173	135	226	290	no
6 https://farm4.staticflickr.com/3777/9145344116_730a252ed8_z.jpg	cow	dog	99%	128	89	454	218	no

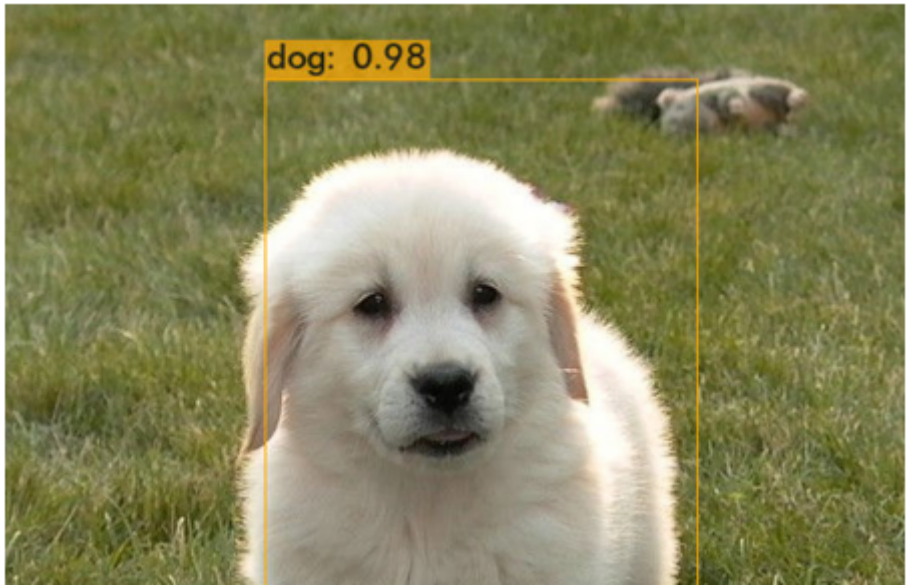
```
#Question 4 implementation


#image1
%cd /content/darknet/
!./darknet detector test data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_best.weights /conte
imShow('predictions.jpg')
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                                1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                                ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                                ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                                1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                                ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                                ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
```

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                    ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                 ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_best.weights...
 seen 64, trained: 128 K-images (2 Kilo-batches_64)
Done! Loaded 38 layers from weights-file
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
/content/Q4_Images/Image1_dog.jpg: Predicted in 3.243000 milli-seconds.
dog: 98%        (left_x:  152   top_y:   45   width:  253   height:  509)
Unable to init server: Could not connect: Connection refused

(predictions:5483): Gtk-WARNING **: 03:35:11.239: cannot open display:
```



```
#image 2
%cd /content/darknet/
!./darknet detector test data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_best.weights /conte
imShow('predictions.jpg')
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                                1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                                ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                                ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                                1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                                ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                                ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
```

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                    ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                 ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_best.weights...
 seen 64, trained: 128 K-images (2 Kilo-batches_64)
Done! Loaded 38 layers from weights-file
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
/content/Q4_Images/Image2_cat.jpg: Predicted in 3.240000 milli-seconds.
cat: 63%        (left_x:  -26   top_y:    2   width:  475   height:  458)
Unable to init server: Could not connect: Connection refused

(predictions:5503): Gtk-WARNING **: 03:35:58.454: cannot open display:
```


```
#image 3
%cd /content/darknet/
!./darknet detector test data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_best.weights /conte
imShow('predictions.jpg')
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                                1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                                ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                                ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                                1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                                ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                                ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
```

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                    ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                 ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_best.weights...
 seen 64, trained: 128 K-images (2 Kilo-batches_64)
Done! Loaded 38 layers from weights-file
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
/content/Q4_Images/Image3_cat.jpg: Predicted in 3.241000 milli-seconds.
cat: 77%        (left_x:   67   top_y:  -26   width:  444   height:  543)
Unable to init server: Could not connect: Connection refused

(predictions:5521): Gtk-WARNING **: 03:36:30.134: cannot open display:
```

```
# image 4
%cd /content/darknet/
!./darknet detector test data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_best.weights /conte
imShow('predictions.jpg')
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                                1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                                ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                                ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                                1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                                ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                                ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
```

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                    ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                 ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_best.weights...
 seen 64, trained: 128 K-images (2 Kilo-batches_64)
Done! Loaded 38 layers from weights-file
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
/content/Q4_Images/Image4_bear.jpg: Predicted in 3.242000 milli-seconds.
dog: 53%        (left_x:   64   top_y:   17   width:  374   height:  423)
Unable to init server: Could not connect: Connection refused

(predictions:5543): Gtk-WARNING **: 03:37:12.165: cannot open display:
```


```
#image 5
%cd /content/darknet/
```

```
!./darknet detector test data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_best.weights /conter
imShow('predictions.jpg')
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                                1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                                ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                                ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                                1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                                ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                                ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
```

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                    ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                 ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_best.weights...
 seen 64, trained: 128 K-images (2 Kilo-batches_64)
Done! Loaded 38 layers from weights-file
 Detection layer: 30 - type = 28
```

```
#image 6
%cd /content/darknet/
!./darknet detector test data/catdogmonkey.data cfg/yolov4-tiny-catdogmonkey.cfg backup/yolov4-tiny-catdogmonkey_best.weights /conte
imShow('predictions.jpg')
```

```
/content/darknet
 CUDA-version: 11010 (11020), cuDNN: 7.6.5, CUDNN_HALF=1, GPU count: 1
 CUDNN_HALF=1
 OpenCV version: 3.2.0
 0 : compute_capability = 750, cudnn_half = 1, GPU: Tesla T4
net.optimized_memory = 0
mini_batch = 1, batch = 1, time_steps = 1, train = 0
   layer   filters  size/strd(dil)      input                output
   0 Create CUDA-stream - 0
 Create cudnn-handle 0
conv     32       3 x 3/ 2    224 x 224 x   3 ->  112 x 112 x  32 0.022 BF
   1 conv     64       3 x 3/ 2    112 x 112 x  32 ->   56 x  56 x  64 0.116 BF
   2 conv     64       3 x 3/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.231 BF
   3 route  2                                 1/2 ->   56 x  56 x  32
   4 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   5 conv     32       3 x 3/ 1     56 x  56 x  32 ->   56 x  56 x  32 0.058 BF
   6 route  5 4                                   ->   56 x  56 x  64
   7 conv     64       1 x 1/ 1     56 x  56 x  64 ->   56 x  56 x  64 0.026 BF
   8 route  2 7                                   ->   56 x  56 x 128
   9 max                2x 2/ 2     56 x  56 x 128 ->   28 x  28 x 128 0.000 BF
  10 conv    128       3 x 3/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.231 BF
  11 route  10                                1/2 ->   28 x  28 x  64
  12 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  13 conv     64       3 x 3/ 1     28 x  28 x  64 ->   28 x  28 x  64 0.058 BF
  14 route  13 12                                ->   28 x  28 x 128
  15 conv    128       1 x 1/ 1     28 x  28 x 128 ->   28 x  28 x 128 0.026 BF
  16 route  10 15                                ->   28 x  28 x 256
  17 max               2x 2/ 2     28 x  28 x 256 ->   14 x  14 x 256 0.000 BF
  18 conv    256       3 x 3/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.231 BF
  19 route  18                                1/2 ->   14 x  14 x 128
  20 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  21 conv    128       3 x 3/ 1     14 x  14 x 128 ->   14 x  14 x 128 0.058 BF
  22 route  21 20                                ->   14 x  14 x 256
  23 conv    256       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x 256 0.026 BF
  24 route  18 23                                ->   14 x  14 x 512
  25 max               2x 2/ 2     14 x  14 x 512 ->    7 x   7 x 512 0.000 BF
  26 conv    512       3 x 3/ 1      7 x   7 x 512 ->    7 x   7 x 512 0.231 BF
  27 conv    256       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x 256 0.013 BF
  28 conv    512       3 x 3/ 1      7 x   7 x 256 ->    7 x   7 x 512 0.116 BF
  29 conv     24       1 x 1/ 1      7 x   7 x 512 ->    7 x   7 x  24 0.001 BF
  30 yolo
```

```
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
  31 route  27                                    ->    7 x   7 x 256
  32 conv    128       1 x 1/ 1      7 x   7 x 256 ->    7 x   7 x 128 0.003 BF
  33 upsample                 2x     7 x   7 x 128 ->   14 x  14 x 128
  34 route  33 23                                 ->   14 x  14 x 384
  35 conv    256       3 x 3/ 1     14 x  14 x 384 ->   14 x  14 x 256 0.347 BF
  36 conv     24       1 x 1/ 1     14 x  14 x 256 ->   14 x  14 x  24 0.002 BF
  37 yolo
[yolo] params: iou loss: ciou (4), iou_norm: 0.07, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.05
nms_kind: greedynms (1), beta = 0.600000
Total BFLOPS 1.969
avg_outputs = 86962
 Allocate additional workspace_size = 26.22 MB
Loading weights from backup/yolov4-tiny-catdogmonkey_best.weights...
 seen 64, trained: 128 K-images (2 Kilo-batches_64)
Done! Loaded 38 layers from weights-file
 Detection layer: 30 - type = 28
 Detection layer: 37 - type = 28
/content/Q4_Images/Image6_cow.jpg: Predicted in 3.238000 milli-seconds.
dog: 99%        (left_x:  128   top_y:   89   width:  454   height:  218)
Unable to init server: Could not connect: Connection refused

(predictions:5581): Gtk-WARNING **: 03:38:33.367: cannot open display:
```

5. Write a function `iou` to compute the intersection over union (IoU) of 2 bounding boxes.

```
def iou(x1_center,y1_center,w1,h1,x2_center,y2_center,w2,h2):
  b1={}
  b1['x1']=(x1_center-(w1/2))
  b1['y1']=(y1_center-(h1/2))
  b1['x2']=(x1_center+(w1/2))
  b1['y2']=(y1_center+(h1/2))

  b2={}
  b2['x1']=(x2_center-(w2/2))
  b2['y1']=(y2_center-(h2/2))
  b2['x2']=(x2_center+(w2/2))
  b2['y2']=(y2_center+(h2/2))

  x_left=max(b1['x1'],b2['x1'])
  y_top=max(b1['y1'],b2['y1'])
  x_right=min(b1['x2'],b2['x2'])
  y_bottom=min(b1['y2'],b2['y2'])

  intersection_area=abs((x_left-x_right)*(y_top-y_bottom))
  b1_area=abs((b1['x1']-b1['x2'])*(b1['y1']-b1['y2']))
  b2_area=abs((b2['x1']-b2['x2'])*(b2['y1']-b2['y2']))

  b1_union_b2=(b1_area-intersection_area)+(b2_area-intersection_area)+intersection_area
  iou=intersection_area/b1_union_b2

  return iou



iou(0.6,0.6,0.5,0.5,0.4,0.5,0.3,0.7)

     0.2777777777777779
```

6. What is the IoU of the following bounding boxes?:

```
| x    | y    | width  | height  |
|------|------|--------|---------|
| 0.6  | 0.6  | 0.5    | 0.5     |
| 0.4  | 0.5  | 0.3    | 0.7     |
```

ANSWER:

```
0.277
```

```
0.277
```

7. Which of the following bounding boxes would be supressed in the Non-Max-Suppression algorithm with an overlap threshold of 0.5:

box	class	x	y	width	height	score
A	0	0.25	0.25	0.50	0.50	0.9
B	0	0.75	0.75	0.45	0.45	0.8
C	0	0.75	0.75	0.50	0.50	0.7
D	0	0.25	0.25	0.45	0.45	0.6
E	0	0.25	0.25	0.10	0.10	0.5
F	1	0.75	0.75	0.45	0.45	0.4

ANSWER:

```
C and D
```

```
below cells has step by step solution.
```

```
#First iteration
#since A has highest score we need to compare remaining bounding boxes of same class with B,C,D,E
#AB
AB=iou(0.25,0.25,0.50,0.50,0.75,0.75,0.45,0.45)
print(f'AB:{AB}')
AC=iou(0.25,0.25,0.50,0.50,0.75,0.75,0.50,0.50)
print(f'AC:{AC}')
AD=iou(0.25,0.25,0.50,0.50,0.25,0.25,0.45,0.45)
print(f'AD:{AD}')
AE=iou(0.25,0.25,0.50,0.50,0.25,0.25,0.10,0.10)
print(f'AE:{AE}')
```

```
AB:0.001383125864453668
AC:0.0
AD:0.8099999999999998
AE:0.03999999999999998
```

```
#Since AD has higher threshold than 0.5 it will be elminated

#D will be removed

#now left over boxes are B,C,E

#out of these boxes B has highest score this needs to be compared with left over boxes.
BC=iou(0.75,0.75,0.45,0.45,0.75,0.75,0.50,0.50)
BE=iou(0.75,0.75,0.45,0.45,0.25,0.25,0.10,0.10)
print(f'BC:{BC}')
print(f'BE:{BE}')
```

```
BC:0.8099999999999998
BE:0.31274131274131295
```

```
#since BC has threshold greater than 0.8
#c is removed
```

```
CE=iou(0.25,0.25,0.45,0.45,0.25,0.25,0.10,0.10)
print(f'CE:{CE}')
print('since there are no boxes left C and  D boxes are  removed')
```

```
CE:0.0493827160493827
since there are no boxes left C and  D boxes are  removed
```

```
from google.colab import files
files.download("/content/yolo-animal-detection-small/train/train.txt")
```

Double-click (or enter) to edit

Double-click (or enter) to edit

Deliverables

Submit the following files to UR Courses.

- Submit your notebook, File->Download > download .ipynb
  - This should include your answers to the questions.
- Your generated dataset files
  - `data/train.txt`
  - `data/test.txt`
- Training files
  - `data/catdogmonkey.names`
  - `data/catdogmonkey.data`
  - `cfg/yolov4-tiny-catdogmonkey.cfg`