# YOLO OBJECT DETECTION FOR A SINGLE OBJECT
# MAKING A DOG DETECTOR

To get Darknet YOLO training to work, we need the following files

- Object bounding box file for each image (.txt)
- Class name file for all the category names (.names)
- Training dataset file for the list of training images (train.txt)
- Validating dataset file for the list of validating images (test.txt)
- Configuration file for YOLO neural network specification (.cfg)
- Data location file for finding all the data information (.data)

**STEPS TO FOLLOW:**

1. Obtain the dataset - I created a dataset of 248 images of dogs and other random pictures
2. Use B-BOX LABELLING TOOL to label the dogs in the images and get annotations in the format:

   [top-left-x] [top-left-y] [bottom-right-x] [bottom-right-y]
3. YOLO needs the annotations the format:

   [object-class-id] [centre-x] [centre-y] [width] [height]
4. used the script convert.py to make the conversions
5. Now we need to split the dataset into train and validation sets and get the files: train_dogs.txt and test_dogs.txt, the process.py script does that with 10% data sent to validation set.
6. I make the dog.names file containing the names of the classes- 'dog'
7. I make the dog.data file containing:

   *classes = 1*
   *train = path to train_dogs.txt file*
   *valid = path to test_dogs.txt file*
   *names = path to dog.names*
   *backup = backup ([path] is where the intermediate .weights and final .weights files will be written)*

8. Creating the configuration (.cfg) file :

   we will copy **cfg/yolov2-voc.cfg** into **dogs.cfg** and make suitable modifications :

   - The meaning of batch and subdivisions is how many mini batches you split your batch in.
     - batch=64, loading 64 images for this “iteration”.
     - subdivision=8, split batch into 8 “mini-batches”, so 64/8 = 8 images per “mini batch” and this get sent to the GPU for process.
     - That will be repeated 8 times until the batch is completed and a new iteration will start with 64 new images. When batching you are averaging over more images, the intent is not only to speed up the training process but also to generalise the training more. If your GPU have enough memory you can reduce the subdivision to load more images into the GPU to process at the same time.
     - **Even though I trained the model on crestle.com using batch=64 and subdivisions=8, the training of the final model has been done using batch=1 and subdivisions=1 as it was trained on a CPU.**

‣

   - Other changes required are to the classes and filters specifications:
     - number_of_filters = (number_of_classes + 5)*5
     - change number of classes
     - in line 237 change number of filters to 30
     - in line 244 change number of classes to 1

**TRAINING**

For training we use convolutional weights that are pre-trained YOLO on Imagenet. We use weights from the Extraction model, i.e. **darknet19_448.conv.23**

Based on the paths to the various files, the command would be something like this -

***./darknet detector train cfg/dogs.data cfg/dogs.cfg darknet19_448.conv.23***

- Here we basically pass all the parameters that the function train_detector requires)
  - train_detector() is present in examples/detector.c/
- similarly for testing, we pass the above parameters and pass the image file

- we can also need to pass a threshold otherwise all bounding boxes would be displayed

## SCOPE OF ERRORS

- The dataset images size should be greater than or equal to 416x416 (which is the size of input images as used by Yolo cfg file). If it isn't the training will not give any b-boxes. It might also show nan and IOU will be very less due to this reason. In this case, the average loss will come down drastically but the model won't learn anything.
- the annotations should be correct
- the images should have .jpg file extension
- full path of required fields should be specified in the dogs.data file
- don't use images having no label category in the training dataset
- if GPU available, use batch_size=64 with a specified subdivisions size.
- if training on CPU, use batch_size=1