Guanghan / **darknet**
forked from pjreddie/darknet

Convolutional Neural Networks http://pjreddie.com/darknet/

281 commits **1** branch **0** releases **2** contributors

Branch: master ▾ New pull request Create new file Upload files Find file Clone or download ▾

This branch is 52 commits ahead, 201 commits behind pjreddie:master.

Clone with HTTPS ⓘ Use SSH
Use Git or checkout with SVN using the web URL.

```
https://github.com/Guanghan/darknet.git
```

Open in Desktop Download ZIP

Guanghan Update README.md

cfg	Create yolo_2class_box11.cfg	
data	Create training_list.txt	
scripts	Convert annotation to darknet format	Dec 22, 2015
src	Update yolo.c	Jan 24, 2016
.gitignore	inet label script	Jun 13, 2015
LICENSE	add license	Jul 10, 2015
Makefile	shortcut layers, msr networks	Dec 14, 2015
README.md	Update README.md	Sep 21, 2016

README.md

#Darknet# Darknet is an open source neural network framework written in C and CUDA. It is fast, easy to install, and supports CPU and GPU computation.

For more information see the Darknet project website.

For questions or issues please use the Google Group.

#About This Fork#

1. This fork repository adds some additional niche in addition to the current darkenet from pjreddie. e.g.

   (1). Read a video file, process it, and output a video with boundingboxes.

   (2). Some util functions like image_to_Ipl, converting the image from darknet back to Ipl image format from OpenCV(C).

(3). Adds some python scripts to label our own data, and preprocess annotations to the required format by darknet.

...More to be added

2. This fork repository illustrates how to train a customized neural network with our own data, with our own classes.

   The procedure is documented in README.md.

   Or you can read this article: Start Training YOLO with Our Own Data.

#DEMOS of YOLO trained with our own data# Yield Sign: https://youtu.be/5DJVLV3P47E

Stop Sign: https://youtu.be/0CQMb3NGlMk

The cfg that I used is here: darknet/cfg/yolo_2class_box11.cfg

The weights that I trained can be downloaded here: (UPDATED 1/13/2016) yolo_2class_box11_3000.weights

The pre-compiled software with source code package for the demo: darknet-video-2class.zip

You can use this as an example. In order to run the demo on a video file, just type:

./darknet yolo demo_vid cfg/yolo_2class_box11.cfg model/yolo_2class_box11_3000.weights /video/test.mp4

If you would like to repeat the training process or get a feel of YOLO, you can download the data I collected and the annotations I labeled.

images: images.tar.gz

labels: labels.tar.gz

The demo is trained with the above data and annotations.

#How to Train With Customized Data and Class Numbers/Labels#

1. Collect Data and Annotation

   (1). For Videos, we can use video summary, shot boundary detection or camera take detection, to create static images.

   (2). For Images, we can use BBox-Label-Tool to label objects. The data I used for the demo was downloaded from Google Images, and hand-labeled by my intern employees. (Just kidding, I had to label it myself. Damn it...) Since I am training with only two classes, and that the signs have less distortions and variances (compared to person or car, for example), I only trained around 300 images for each class to get a decent performance. But if you are training with more classes or harder classes, I suggest you have at least 1000 images for each class.

2. Create Annotation in Darknet Format

   (1). If we choose to use VOC data to train, use scripts/voc_label.py to convert existing VOC annotations to darknet format.

   (2). If we choose to use our own collected data, use scripts/convert.py to convert the annotations.

   At this step, we should have darknet annotations(.txt) and a training list(.txt).

   Upon labeling, the format of annotations generated by BBox-Label-Tool is:

   class_number

   box1_x1 box1_y1 box1_x2 box1_y2

   box2_x1 box2_y1 box2_x2 box2_y2

   ....

   After conversion, the format of annotations converted by scripts/convert.py is:

   class_number box1_x1_ratio box1_y1_ratio box1_width_ratio box1_height_ratio

   class_number box2_x1_ratio box2_y1_ratio box2_width_ratio box2_height_ratio

   ....

Note that each image corresponds to an annotation file. But we only need one single training list of images. Remember to put the folder "images" and folder "annotations" in the same parent directory, as the darknet code look for annotation files this way (by default).

You can download some examples to understand the format:

before_conversion.txt

after_conversion.txt

training_list.txt

3. Modify Some Code

   (1) In src/yolo.c, change class numbers and class names. (And also the paths to the training data and the annotations, i.e., the list we obtained from step 2.)

   ```
   If we want to train new classes, in order to display correct png Label files, we also need to moidify and run
   [data/labels/make_labels] (https://github.com/Guanghan/darknet/blob/master/data/labels/make_labels.py)
   ```

   (2) In src/yolo_kernels.cu, change class numbers.

   (3) Now we are able to train with new classes, but there is one more thing to deal with. In YOLO, the number of parameters of the second last layer is not arbitrary, instead it is defined by some other parameters including the number of classes, the side(number of splits of the whole image). Please read the paper

   ```
   (5 x 2 + number_of_classes) x 7 x 7, as an example, assuming no other parameters are modified.

   Therefore, in [cfg/yolo.cfg](https://github.com/Guanghan/darknet/blob/master/cfg/yolo.cfg), change the "output"
   in line 218, and "classes" in line 222.
   ```

   (4) Now we are good to go. If we need to change the number of layers and experiment with various parameters, just mess with the cfg file. For the original yolo configuration, we have the pre-trained weights to start from. For arbitrary configuration, I'm afraid we have to generate pre-trained model ourselves.

4. Start Training

   Try something like:

   ./darknet yolo train cfg/yolo.cfg extraction.conv.weights

#Windows Version#

I also have a windows version for darknet available:

http://guanghan.info/projects/YOLO/darknet-windows.zip

But you need to use Visual Studio 2015 to open the project.

Here is a quick hand-on guide:

1. Open VS2015. If you don't have it, you can install it for free from the offcial microsoft website.
2. Open: darknet-windows\darknet\darknet.sln
3. Comiple
4. Copy the exe file from:

   darknet-windows\darknet\x64\Debug\darknet.exe

   to the root folder:

   darknet-windows\darknet.exe
5. Open cmd

   Run: darknet yolo test [cfg_file] [weight_file] [img_name]

6. The image will be output to:

   darknet-windows\prediction.png

   darknet-windows\resized.png

#Contact# If you find any problems regarding the procedure, contact me at gnxr9@mail.missouri.edu.

Or you can join the aforesaid Google Group; there are many brilliant people asking and answering questions out there.