Hand Detection and Recognition via Faster R-CNN

- If you encounter issues feel free to contact me : **yunqiuxu1991@gmail.com**

---

1. Setup the Model

- Download the model

```
git clone https://github.com/YunqiuXu/HandDetection.git
```

- Change GPU architecture

```
cd HandDetection/lib
vi setup.py # line 136
```

GPU model	Architecture
TitanX (Maxwell/Pascal)	sm_52
GTX 960M	sm_50
GTX 1080 (Ti)	sm_61
Grid K520 (AWS g2.2xlarge)	sm_30
Tesla K80 (AWS p2.xlarge)	sm_37

- Build the Cython modules

```
make clean
make
cd ..
```

- Set the Python COCO API

```
cd data
```

```
git clone https://github.com/pdollar/coco.git
cd coco/PythonAPI
make
cd ../../..
```

- Download pretrained model: vgg16 + resnet

```
cd data
mkdir imagenet_weights
cd imagenet_weights
wget -v http://download.tensorflow.org/models/vgg_16_2016_08_28.tar.gz
wget -v
http://download.tensorflow.org/models/resnet_v1_101_2016_08_28.tar.gz
tar -xzvf vgg_16_2016_08_28.tar.gz
tar -xzvf resnet_v1_101_2016_08_28.tar.gz
mv vgg_16.ckpt vgg16.ckpt
mv resnet_v1_101.ckpt res101.ckpt
rm -rf *.tar.gz
cd ../..
```

2. Preprocess the Data

- Do these operations first

```
cd data
mkdir voc_2007_trainval+voc_2012_trainval
unzip VOCdevkit2007.zip
```

- You can either use your own data or use the crawler we provide to collect images from google-image:

```
cd crawler
vi config.json # change save directory and keywords
python google_image_crawler.py
cd ..
```

- Put your images into `data/VOCdevkit2007/VOC2007/JPEGImages/`, go into this folder, then rename and resize the images by running

```
cd VOCdevkit2007/VOC2007
```

```
./rename.sh # 000001.jpg
python resize.py JPEGImages/*.jpg # 480 * 256
cd ..
```

- Label the images using labelIMG(https://github.com/tzutalin/labelImg). Change the save path to `old_annotations` to store the generated .xml files.
- Change the annotations to VOC2007 format(see below)

```
python labelIMG_to_VOC2007.py
rm -rf old_annotations/*.xml
```

```
<annotation>
  <folder>VOC2007</folder>
  <filename>000003.jpg</filename>
  <source>
    <database>The VOC2007 Database</database>
    <annotation>PASCAL VOC2007</annotation>
  </source>
  <size>
    <width>480</width>
    <height>256</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>YES</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>146</xmin>
      <ymin>95</ymin>
      <xmax>232</xmax>
      <ymax>230</ymax>
    </bndbox>
  </object>
</annotation>
```

- Split the dataset by running

```
python build_main.py
cd ../../..
```

- Following is the architecture of dataset, you can get more details from folder `SampleDataset`

```
- VOC2007/
    - Annotations/
        - 000001.xml
        - 000002.xml
        ...
    - ImageSets/
```

```
        - Main/
            - test.txt
            - train.txt
            - trainval.txt
            - val.txt
    - JPEGImages/
        - 000001.jpg
        - 000002.jpg
        ...
    ...
```

3. Train and Test the Model

- We provide 2 models to perform hand detection:
  - Modified VGG16
  - Original ResNet
- Before training, you should build these folders first:

```
unzip output.zip
unzip result.zip
```

- If you have split the dataset, you can train the model by running

```
# ./train.sh [GPU_ID] [NET]
# ./train.sh 0 res101
./train.sh 0 vgg16
```

The default number of iterations is 70000 and can be modifyed by changing line 22 of `experiments/scripts/train_faster_rcnn.sh`. The models will be saved in folder `output`

- Before testing, you should make sure the number of iterations in `experiments/scripts/test_faster_rcnn.sh` matches the model (e.g. if you want to test model `res101_faster_rcnn_iter_20000`, you need to set ITERATIONS as 20000). Then you can test the model by running following scripts, the performance (e.g. mAP, loss) will be printed on command line and the prediction will be saved in folder `result`

```
# ./test.sh [GPU_ID] [NET]
# ./test.sh 0 res101
./test.sh 0 vgg16
```

4. Test the Model on New Dataset

- If you have trained the model and want to test it on other datasets without splitting, you can follow Part 2 to build dataset, the only difference is running `build\_main_testonly.py` instead of `build_main.py`
- Then you can run `./test.sh 0 vgg16` to make prediction

5. Visualize the Prediction

- After testing you can draw predicted bounding box via following steps:

```
cp result/YES.txt data/VOCdevkit2007/VOC2007/drawing
cp result/NO.txt data/VOCdevkit2007/VOC2007/drawing
cd data/VOCdevkit2007/VOC2007
bash draw_bounding_box.sh
```

- Then images with bounding boxes will be built in folder `drawing`, here are some examples: