py-faster-rcnn データベース追加方法

名城大学メカトロニクス工学科

ロボットシステムデザイン研究室

2017 年 11 月 27 日

目次

1. はじめに

人工知能の一つに Deep Learning と呼ばれる処理が存在する. Deep Learning とは, 予め学習内容を与えることで新規にデータを取り入れた際, 学習内容に則って答えを導き出すと言った手法である. Deep Learning には, 既存のエンジンである Caffe が用意されており, Caffe を用いることで誰もが簡単に Deep Learning を扱うことができる. そして, Caffe で物体認識を行う場合, py-faster-rcnn を用いる. py-faster-rcnn を用いることで, 容易に Deep Learnig で物体認識を扱うことができる. しかし, 独自のデータベースを扱いたい場合の参考資料が少ない. 本内容では, Deep Learning エンジンの一つである Caffe の中の py-faster-rcnn のデータベースの生成方法について述べる. また, py-faster-rcnn の demo.py が正常動作にしていると想定する.

2. 開発環境

使用した PC の開発寛容を表 1 に示す.

OS	Ubuntu 16.04
CPU	Core i7(3.6GHz-4.2H)
コア数	4 コア/ 8 スレッド
メインメモリ	32GB
GPU	GeForce GTX 1080Ti
Caffe	py-faster-rcnn
CUDA	CUDA8.0
cuDNN	cuDNN_v5

3. データベース生成方法

3.1.pascal_voc2007

pascal_voc2007 は caffemodel のダウンロードを行うのみである.

以下に pascal_voc20007 のデータベース取得手順を示す.

```
$ cd ~/py-faster-rcnn
$ ./data/scripts/fetch_faster_rcnn_models.sh
```

以上で完了とする.

3.2. coco2014

cocoは画像ファイルと注釈ファイルをダウンロードし，学習させる必要がある．また,PCのハードに依存するがダウンロードデータ時間が数十分から数時間かかる可能性がある．加えて，ダウンロード後の学習時間が私の PC では GPU 駆動で約 36 時間かかった．以下に coco2014 のデータベース取得手順を示す．

coco2014 の画像ファイルと注釈ファイルのダウンロード

```
$ cd ~/py-faster-rcnn
$ ./data/scripts/fetch_imagenet_models.sh
$ cd data
$ wget http://msvocds.blob.core.windows.net/coco2014/train2014.zip
$ wget http://msvocds.blob.core.windows.net/coco2014/val2014.zip
$ wget http://msvocds.blob.core.windows.net/coco2014/test2014.zip
$ wget -O zipfile.py 'http://www.cs.gunma-u.ac.jp/~nagai/wiki/index-
write.php?plugin=attach&pcmd=open&file=zipfile.py&refer=py-faster-
rcnn%20%B3%D0%A4%A8%BD%F1%A4%AD'
$ python zipfile.py -e train2014.zip COCO2014/images/
$ python zipfile.py -e val2014.zip   COCO2014/images/
$ python zipfile.py -e test2014.zip  COCO2014/images/
$ ln -s COCO2014 coco
$ wget http://msvocds.blob.core.windows.net/annotations-1-0-
3/instances_train-val2014.zip
$ wget http://msvocds.blob.core.windows.net/annotations-1-0-
3/person_keypoints_train+val5k2014.zip
$ wget http://msvocds.blob.core.windows.net/annotations-1-0-
3/captions_train-val2014.zip
$ wget http://msvocds.blob.core.windows.net/annotations-1-0-
4/image_info_test2014.zip
$ python zipfile.py -e instances_train-val2014.zip ./coco/
$ python zipfile.py -e person_keypoints_train+val5k2014.zip ./coco/
$ python zipfile.py -e image_info_test2014.zip ./coco/
$ python zipfile.py -e captions_train-val2014.zip ./coco/
$ wget
http://dl.dropboxusercontent.com/s/o43o90bna78omob/instances_minival2014.j
son.zip
```

```
$ wget
http://dl.dropboxusercontent.com/s/s3tw5zcg7395368/instances_valminusminiv
al2014.json.zip
$ unzip instances_minival2014.json.zip
$ unzip instances_valminusminival2014.json.zip
$ mv instances_minival2014.json
instances_valminusminival2014.json ./coco/annotations/
$ cd ..
```

coco2014 の caffemodel の作成するために run.sh を用意する.

```
$ gedit run.sh
```

以下を追加する

```
1  #!/bin/bash
2  find ./data -follow -name "*.pkl" -exec rm {} ¥;
3
4  GPU=0
5
6  #NET=ZF
7  #NET=VGG_CNN_M_1024
8  NET=VGG16
9
10 #DATASET=pascal_voc
11 #DATASET=pascal_voc_2012
12 DATASET=coco
13
14 EXPDIR=ishida  #
15
16 HOST=' hostname'
17 (time ./experiments/scripts/faster_rcnn_end2end.sh $GPU $NET $DATASET --
-set EXP_DIR $EXPDIR) 2>&1
```

coco2014 を学習させ, coco2014 の caffemodel ファイルを生成する.

```
$ sh run.sh
```

生成した caffmodel ファイルのディレクトリ移動後，ファイル名変更

```
$ mv
ouput/ishida/coco_2014_faster/vgg16_faster_rcnn_iter_490000.caffemodel
data/faster_rcnn_models/
$ mv data/faster_rcnn_models/vgg16_faster_rcnn_iter_490000.caffemodel
data/faster_rcnn_models/coco_vgg16_faster_rcnn_final.caffemodel
$ cp tools/demo.py tools/demo_coco.py
$ gedit tools/demo_coco.py
```

tools/demo_coco.py を以下のように修正する

```
...
27  CLSSSES_VOC = ( '__background__',
...
34  CLASSES_COCO = ( '__background__',
35      'person', 'bicycle', 'car', 'motorcycle',
36      'airplane', 'bus', 'train', 'truck', 'boat',
37      'traffic light', 'fire hydrant', 'stop sign',
38      'parking meter', 'bench', 'bird', 'cat',
39      'dog', 'horse', 'sheep', 'cow',
40      'elephant', 'bear', 'zebra', 'giraffe',
41      'backpack', 'umbrella', 'handbag', 'tie',
42      'suitcase', 'frisbee', 'skis', 'snowboard',
43      'sports ball', 'kite', 'baseball bat', 'baseball glove',
44      'skateboard', 'surfboard', 'tennis racket', 'bottle',
45      'wine glass', 'cup', 'fork', 'knife',
46      'spoon', 'bowl', 'banana', 'apple',
47      'sandwich', 'orange', 'broccoli', 'carrot',
48      'hot dog', 'pizza', 'donut', 'cake',
49      'chair', 'couch', 'potted plant', 'bed',
50      'dining table', 'toilet', 'tv', 'laptop',
51      'mouse', 'remote', 'keyboard', 'cell phone',
52      'microwave', 'oven', 'toaster', 'sink',
53      'refrigerator', 'book', 'clock', 'vase',
54      'scissors', 'teddy bear', 'hair drier', 'toothbrush' )
55
```

```
56  # set VOC by default
57  CLASSES = CLASEES_VOC
...
132 parser.add_argument( ‘--dateset’, dest=’dataset’, help=’The
Dataset to use’,
133                      choices=[“voc”,”coco”], default=’voc’)
...
144 if args.dataset ==”voc”:
145    prototxt = os.path.join(cfg.MODELS_DIR,NETS[args.demo_net][0],
146              ‘faster_rcnn_alt_opt’,’faster_rcnn_test.pt’)
147    caffemodel = os.path.join(cfg.DATA_DIR,’faster_rcnn_models’,
148               NETS[args.demo_net][1])
149 elif args.dataset == “coco”:
150    CLASSES = CLASSES_COCO
151    prototxt =
os.path.join(cfg.MODELS_DIR,”../coco”,NETS[args.demo_net][0],
152              ‘faster_rcnn_end2end’,’test.prototxt’)
153    caffemodel = os.path.join(cfg.DATA_DIR,’faster_rcnn_models’,
154               ”coco_vgg16_faster_rcnn_final.caffemodel”)
...
```

動作確認

```
$ ./tools/demo_coco.py  --dataset coco
```

以上で完了とする.

3.3.独自データベースの生成

独自でデータベースを作成する場合, 数百枚~数万枚の画像ファイルと画像ファイルと同じ数の注釈ファイルを用意する必要がある. 今回は, 財布を認識したい場合のデータベースの生成方法を以下に述べる.

3.3.1. 独自の学習データを準備

data/wallet/results/set1/2007/Main

data/wallet/set1/Annotations/*.xml

data/wallet/set1/ImageSets/Main/{trainval, test}}.txt

data/wallet/set1/PNGImages/*.png

PNGImages には財布画像を 400 枚用意した. Annotations には, xml ファイルで, wallet のラベル名と財布がある座標を以下のように記載する.

wallet_0.xml

```
<?xml version=" 1.0"  encoding=" utf-8" ?>
<annotation>
	<filename>wallet_0.png</filename>
	<object>
		<name>wallet</name>
		<pose>Unspecified</pose>
		<truncated>1</truncated>
		<difficult>0</difficult>
		<bndbox>
			<xmin>111</xmin>
			<ymin>68</ymin>
			<xmax>205</xmax>
			<ymax>210</yamx>
		</bndbox>
	</object>
</annotation>
```

trainval.txt と test.txt を生成するためのスクリプト生成する.

train_test.py

```
1  import numpy as np
2  import os
3  import os.path
4
5  sourcedir =  ‘data/wallet/set1/PNGImages/’
6  targetdir =  ‘data/Kanban/set1/ImageSets/Main’
7  trainfile =  ‘trainbal.txt’
8  testfile =  ‘test.txt’
9  trainRatio = 0.8
10
11 # get file list
12 filelist = os.listdir(sourcedir)
13 filelist = np.array(filelist)
14
15 # random permutation
16 numfile = filelist.shape[0]
17 randindex = np.random.permutation(nunfile)
18
19 # number or files
20 numtrain = np.floor(numfile*trainRation)
21
22 # open target files
23 trainpath = os.path.join(targetdir,trainfile)
24 testpath = os.path.join(targetdir,testfile)
25 ftrain = open(trainpath,’ w’ )
26 ftest = open(testpath,  ‘w’ )
27 cnt = 1
28
29 # write to target files
30 for index in randindex:
31     splits = filelist[index].split( “.” )
32
33     if cnt <= numtrain:
34             ftrain.write(splits[0]+” ¥n” )
```

```
35    else:
36            ftest.write(splits[0]+"¥n")
37    cnt = cnt + 1
38
39  ftrain.close()
40  ftest.close()
```

trainval.txt と test.txt にはランダムに選択した画像名の一覧を記載してあるか確認する.

trainval.txt

```
wallet_0
wallet_3
wallet_156
…
```

test.txt

```
wallet_5
wallet_325
wallet_134
…
```

3.3.2. lib/datasets/wallet.py 作成

lib/datasets/wallet.py からコピーし, 財布用に修正

```
$ cp lib/datasets/pascal_voc.py lib/datasets/wallet.py
$ gedit lib/datasets/wallet.py
```

lib/datasets/wallet.py を以下のように修正する.

```
22 class wallet(imdb)
23  def __init__(self,image_set, setname, year, devkit_path=None):
24    imdb.__init__(self, image_set + '_' + setname)
…
26  self._setname = setname
…
30  self._data_path = os.path.join(self._devkit_path, self._setname)
```

```
31  self._classes = ( '__background__' , #always index 0
32                        'wallet'
33                  )
...
35  self._image_ext =  'png'
...
65  image_path = os.path.join(self._data_path,  'PNGImages' ,index +
self_image_ext)
...
89  return os.path.join(cfg.DATA_DIR,  'wallet' )
...
97  cache_file = os.path.join(self.cache_path,  'wallet_' + self.name +
 '_gt_roidb.pkl' )
...
232 path = os.path.join(
233    self._devkit_path,,
234     'results' ,
235    self._setname,
236    self._year,
237     'Main' ,
238    filename)
...
260 annopath = os.parh.join(
261    self._devkit_path,
262    self._setname,
263     'Annotations' ,
263     '{:s}.xml' )
264 imagesetfile = os.path.join(
265    self._devkit_path,
267    self._setname,
268     'ImageSets' ,
269     'Main' ,
270    self._image_set +  '.txt' )
```

3.3.3. lib/datasets/factory.py に追加

```
$ gedit lib/datasets/factory.py
```

lib/datasets/factory.py に以下を加える.

```
15 from datasets.wallet.py import wallet
…
18 for setname in [ ‘set1’ ,’ set2’ ,’ set3’ ]:
19     for split in [ ‘trainval’ ,’ test’ ]:
20             name =  ‘wallet_{}_{}’ .format(setname,split)
21             __set[name] = (lambda split=split, setname=setname:
wallet(split, setname,  ‘2007’ ))
```

3.3.4. experiments/scripts/faster_rcnn_end2end.sh に追加

```
$ gedit experiments/scripts/faster_rcnn_end2end.sh
```

experiments/scripts/faster_rcnn_end2end.sh に以下を加える.

```
26 wallet)
27     TRAIN_IMDB=” wallet_set1_trainval”
28     TEST_IMDB=” wallet_set1_test”
29     PT_DIR=” wallet”
39     ITERS=70000
40     ;;
```

3.3.5. run.sh の修正

3.2 章で作った run.sh を独自データ用に修正を加える

```
$ gedit run.sh
```

run.sh ファイル以下のように修正する.

```
1  #!/bin/bash
2  find ./data -follow -name ″*.pkl″ -exec rm {} ¥;
3
4  GPU=0
5
6  #NET=ZF
7  #NET=VGG_CNN_M_1024
```

```
8  NET=VGG16
9
10  DATASET=wallet
11 #DATASET=pascal_voc
12 #DATASET=pascal_voc_2012
13 #DATASET=coco
14
15 EXPDIR=ishida
16
17 HOST=’ hostname’
18(time ./experiments/scripts/faster_rcnn_end2end.sh $GPU $NET $DATASET --
set EXP_DIR $EXPDIR) 2>&1
```

3.3.6. wallet 用のモデルファイルを用意

```
$ cp -rp models/pascal_voc models/wallet
$ gedit models/wallet/VGG16/faster_rcnn_end2end/train.prototxt
```

pascal_voc はクラス数が 21 に対して，今回は背景と財布でクラス数が 2 なので num_classes と num_output を以下のように修正し. solver.prototxt の train.prototxt 参照先の修正をする.

models/wallet/VGG16/faster_rcnn_end2end/train.prototxt

```
11  param_str: “’ num_classes’ : 2”      # class number c
…
530 param_str: “’ num_classes’ : 2”      # class number c
…
620  num_output: 2              # class number c
…
643  num_outpu: 8               # class number c * 4
```

```
$ gedit models/wallet/VGG16/faster_rcnn_end2end/test.prototxt
```

models/wallet/VGG16/faster_rcnn_end2end/test.prototxt

```
567  num_output: 2              # class number c
…
```

```
592 num_output: 8                # class number c * 4
```

```
$ gedit models/wallet/VGG16/faster_rcnn_end2end/solver.prototxt
```

models/wallet/VGG16/faster_rcnn_end2end/solve.prototxt

```
1  train_net: “models/wallet/VGG16/faster_rcnn_end2end/train.prototxt”
```

3.3.7. run.sh を実行し, caffemodel 作成(補足 4.1, 4.2)

```
$ sh run.sh
```

生成した caffmodel ファイルのディレクトリ移動後, ファイル名変更

```
$ mv ouput/ishida/trainval_set1/vgg16_faster_rcnn_iter_70000.caffemodel
data/faster_rcnn_models/
$ mv data/faster_rcnn_models/vgg16_faster_rcnn_iter_70000.caffemodel
data/faster_rcnn_models/wallet_vgg16_faster_rcnn_final.caffemodel
```

3.3.8. 動作確認

認識対象画像を data/demo ディレクトリに保存し, 今回は walletsample.jpg を data/demo ディレクトリに保存します.

```
$ cp tools/demo.py tools/demo_wallet.py
$ gedit tools/demo_wallet.py
```

tools/demo_wallet.py を以下のように修正

```
29  CLASSES = ( ‘__background__’ ,
30                ‘wallet’ )
…
32  NETS = { ‘vgg16’ :( ‘VGG16’ ,
33                ‘wallet_vgg16_faster_rcnn_final.caffemodel’ ),
…
118 protoxt = os.path.join(cfg.MODELS_DIR,
 ‘../wallet’ ,NETS[args.demo_net][0],
119                        ‘faster_rcnn_end2end’ ,  ‘test.prototxt’ )
…
```

```
146 im_names = [ ‘004545.jp’,’ 000456.jpg’,’ walletsample.jpg’ ]
```

動作確認(補足 4.3).

```
$ python /tools/demo_wallet.py
```

以上で完了とする.

4. 補足事項

補足事項としては実際に起きたエラー内容とその対策を述べる.

4.1.独自データベースを生成する前に, 学習前ネットワークをダウンロードしてあるかを確認する. ダウンロードしていない場合以下を実行する.

```
$ cd ~/py-faster-rcnn
$ ./data/scripts/fetch_imagenet_models.sh
```

4.2.numpy.1.12 は float がサポートされていないそのため, 以下のようなエラーが起きた場合

```
Type Error : slice indices must be integers or None or have an
__index__method
```

対策として numpy.1.11.2 バージョンにするか, 以下の手順を行う.

```
$ gedit lib proposal_target_layer.py
```

lib/proposal_target_layer.py に以下を追加する.

```
126 start=int(start)
127 end=int(end)
…
166 _rois_per_this_image=int(fg_rois_per_this_image)
```

4.3.独自データベースを生成後, 動作確認を行ったとき以下のようなエラーが起きた場合

```
$ Cannot copy param 0 weights from layer ‘rpn_conv/3×3’; shape mismatch
…
```

対策として独自データベースの test.prototxt のパスが通っていないので通す必要がある.

5. 参考資料

COCO データベースセット
https://soralab.space-ichikawa.com/2016/11/py-faster-rcnn-coco/

py-faster-rcnn　覚え書き
https://www.cs.gunma-u.ac.jp/~nagai/wiki/index.php?py-faster-rcnn%20%B3%D0%A4%A8%BD%F1%A4%AD#m4d938ba

py-faster_rcnn 独自データ生成
http://hirotaka-hachiya.hatenablog.com/entry/2017/05/18/174011