

CSDN新首页上线啦，邀请你来立即体验！(http://blog.csdn.net/)



博客 (//blog.csdn.net/?ref=toolbar) 学院 (//edu.csdn.net?ref=toolbar)

下载 (//download.csdn.net?ref=toolbar) GitChat (//gitbook.cn/?ref=csdn)

更多 ▾

Q



weixin_113250416
//write.blog.csdn.net/posted
ref=toolbar&source=csdnblo

213

[源码分析]Text-Detection-with-FRCN

原创

2017年11月21日 17:58:39

213

Text-Detection-with-FRCN (https://github.com/jugg1024/Text-Detection-with-FRCN)项目是基于py-faster-rcnn (https://github.com/rbgirshick/py-faster-rcnn)项目在场景文字识别领域的扩展。对Text-Detection-with-FRCN的理解过程，本质上是对py-faster-rcnn的理解过程。我个人认为，初学者，尤其是对caffe还不熟悉的时候，在理解整个项目的过程中，会有以下困惑：

- 1.程序入口
- 2.数据是如何准备的？
- 3.整个网络是如何构建的？
- 4.整个网络是如何训练的？

那么，接下来，以我的理解，结合论文和源代码，一步步进行浅析。

一.程序入口

训练阶段：

入口一：/py-faster-rcnn/experiments/scripts/faster_rcnn_end2end.sh
- - >

入口二：/py-faster-rcnn/tools/train_net.py

在train_net中：

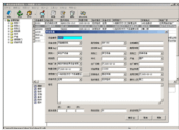
- 1.定义数据格式，获得imdb,roidb；
- 2.开始训练网络。

```
[python]
1. train_net(args.solver, roidb, output_dir, pretrained_model, max_iters)
```

train_net定义在py-faster-rcnn/lib/fast_rcnn/train.py中
- - >

入口三：/py-faster-rcnn/lib/fast_rcnn/train.py

在train_net函数中：



设备管理软件



捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

工作流程管理系统 数据可视化分析

仓库管理系统 开源网店系统



+ 关注

(http://blog.csdn.net/u013250416)

码云

原创

粉丝

喜欢

未开通
(https://gite
utm_sourc

59

8

11

他的最新文章

更多文章 (http://blog.csdn.net/u013250416)

[论文笔记]Single Shot Text Detector with Regional Attention (http://blog.csdn.net/u013250416/article/details/78667404)

[目标检测]SSD: Single Shot MultiBox Detector (http://blog.csdn.net/u013250416/article/details/78666965)

[论文笔记]Arbitrary-Oriented Scene Text Detection via Rotation Proposals (http://blog.csdn.net/u013250416/article/details/78597557)

立即体

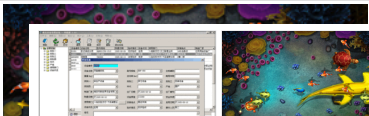
验



内容举报



返回顶部





```
[python]
1. roidb = filter_roidb(roidb)
2. sw = SolverWrapper(solver_prototxt, roidb, output_dir, pretrained_model=pretrained_model)
3. model_paths = sw.train_model(max_iters)
4. return model_paths
```

这样，就开始对整个网络进行训练了。

在solver_prototxt中，定义了train_prototxt。在train_prototxt中，定义了各种层，这些层组合起来，形成了训练网络的结构。

- ->

入口四： /py-faster-rcnn/models/coco_text/VGG16/faster_rcnn_end2end/train.prototxt

先举例说明形式：

1.自定义Caffe Python layer

```
[python]
1. layer {
2.   name: 'input-data'
3.   type: 'Python'
4.   top: 'data'
5.   top: 'im_info'
6.   top: 'gt_boxes'
7.   python_param {
8.     module: 'roi_data_layer.layer'
9.     layer: 'RoIDataLayer'
10.    param_str: "'num_classes': 2"
11.  }
12. }
```

在自定义的caffe python layer中：

type为'python'；

python_param中：

module为模块名，通常也是文件名。module: 'roi_data_layer.layer'：说明这一层定义在roi_data文件夹下面的layer中

layer为模块里的类名。layer:"RoIDataLayer"：说明该类的名字为'RoIDataLayer'

param_str为传入该层的参数。

2.caffe中原有的定义好的层，一般用c++定义。

```
[python]
1. layer {
2.   name: "conv1_1"
3.   type: "Convolution"
4.   bottom: "data"
5.   top: "conv1_1"
6.   param {
7.     lr_mult: 0
8.     decay_mult: 0
9.   }
10.  param {
11.    lr_mult: 0
12.    decay_mult: 0
```



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

工作流管理系统 数据可视化分析

仓库管理系统 开源网店系统

广告

在线课程

0

腾讯云服务器架构实现介绍 ()

讲师：董晓杰

容器技术在5G同城的实践 (http://edu.csdn.net/huiyi/series_detail/73?utm_source=blog9)

容器技术在5G同城的实践 (http://edu.csdn.net/huiyi/series_detail/73?utm_source=blog9)

他的热门文章

- Javascript 私有变量 (http://blog.csdn.net/u013250416/article/details/47609537) 3500
- [Python]爬虫利用Selenium实现批量加载a及模拟自动翻页爬取东方财富网公司公 2985
- [Python]利用高德地图api实现经纬度与地址的批量转换 (http://blog.csdn.net/u013250416/article/details/71178156) 2739

浅谈Javascript的匿名函数中的this对象 (h

设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

内容举报

返回顶部



```
12.         delay_mult. v
13.     }
14.     convolution_param {
15.         num_output: 64
16.         pad: 1
17.         kernel_size: 3
18.     }
19. }
```

在目录：`/py-faster-rcnn/caffe-fast-rcnn/include/caffe/layers`文件夹下面，可以看到`conv_layer.hpp`的头文件定义。

了解了layer的表示方法，接下来，看一下，整个网络是如何构建的。

整个网络可以分为四个部分：

- 1.Conv layers。首先使用一组基础的conv+relu+pooling层提取image的feature maps。该feature maps被共享用于后续RPN层和全连接层。
- 2.Region Proposal Networks。RPN网络用于生成region proposals。该层通过softmax判断anchors属于foreground或者background，再利用bounding box regression修正anchors来获得精确的proposals。
- 3.Roi Pooling。该层收集输入的feature maps和proposals，送入后续全连接层判定目标类别。
- 4.Classification。利用proposal feature maps计算proposal的类别，同时再次利用bounding box regression获得检测框最终的精确位置。

介绍到这里，相信大家对于整个程序的运行流程有了初步的了解。接下来，来看看具体的实现细节。首先，从数据的准备入手。

二.数据是如何准备的？

入口一：`/py-faster-rcnn/tools/train_net.py`

在`train_net`中：

获得`imdb,roidb`：`imdb, roidb = combined_roidb(args.imdb_name)`

进入位于`/py-faster-rcnn/tools/train_net.py`，`combined_roidb`中：

```
[python]
1. def combined_roidb(imdb_names):
2.     def get_roidb(imdb_name):
3.         imdb = get_imdb(imdb_name)
4.         print 'Loaded dataset `{:s}` for training'.format(imdb.name)
5.         imdb.set_proposal_method(cfg.TRAIN.PROPOSAL_METHOD)
6.         print 'Set proposal method: {:s}'.format(cfg.TRAIN.PROPOSAL_METHOD)
7.         roidb = get_training_roidb(imdb)
8.         return roidb
9.
10.    roidbs = [get_roidb(s) for s in imdb_names.split('+')]
11.    roidb = roidbs[0]
12.    if len(roidbs) > 1:
13.        for r in roidbs[1:]:
14.            roidb.extend(r)
15.        imdb = datasets.imdb.imdb(imdb_names)
16.    else:
17.        imdb = get_imdb(imdb_names)
18.    return imdb, roidb
```



开源商城系统 商城系统源码
工作流管理系统 数据可视化分析
仓库管理系统 开源网店系统

[训练测试过程记录]Text-Detection-with-FRCN (<http://blog.csdn.net/u013250416/article/details/78457624>)

Saliency Detection with Multi-Scale Superpixels对应的源码 (<http://download.csdn.net/detail/skye1221/9947383>)

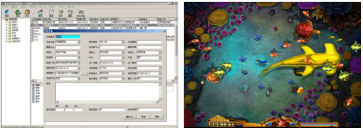
2

Java 源码：`http://bldownload.csdn.net/WaitingData/669910001.txt`

TLD(Topicality, Locality, Delineation)算法学习与源码解析(http://blog.csdn.net/qq_25667033)ker源码分

内容举报

返回顶部



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码
工作流管理系统 数据可视化分析
仓库管理系统 开源网店系统

广告



先看imdb是如何产生的，然后看如何借助imdb产生roidb:
进入位于 /py-faster-rcnn/lib/datasets/factory.py , get_imdb中：

```
[python]
1. def get_imdb(name):
2.     """Get an imdb (image database) by name."""
3.     if not __sets.has_key(name):
4.         raise KeyError('Unknown dataset: {}'.format(name))
5.     return __sets[name]()
```

由此可见，get_imdb函数的实现原理：_sets是一个字典，字典的key是数据集的名称，字典的value是一个lambda表达式（即一个函数指针）。
在前面的文章中提到过，这里已经将coco_text数据集转化为pascal_voc数据集的格式。因此，这里使用的数据集的名称为pascal_voc。

在faster_rcnn_end2end.sh中，定义了：

```
[python]
1. case $DATASET in
2.     pascal_voc)
3.         TRAIN_IMDB="voc_2007_trainval"
4.         TEST_IMDB="voc_2007_test"
5.         PT_DIR="coco_text"
6.         ITERS=70000
7.     ;;
```

看下面这段代码：

```
[python]
1. # Set up voc_<year>_<split> using selective search "fast" mode
2. for year in ['2007', '2012']:
3.     for split in ['train', 'val', 'trainval', 'test']:
4.         name = 'voc_{}_{}'.format(year, split)
5.         __sets[name] = (lambda split=split, year=year: pascal_voc(split, year))
```

所以，这里实际执行的是pascal_voc函数。

进入位于 /py-faster-rcnn/lib/datasets/pascal_voc.py。可以看到，pascal_voc是一个类，这里是调用了该类的构造函数，返回的也是该类的一个实例，所以，imdb实际上就是pascal_voc类的一个实例。
那么，来看这个类的构造函数是如何实现的，以及输入的图片数据在里面是如何组织的。
该类的构造函数如下：设置了imdb的一些属性，比如图片的路径，图片名称的索引，没有放入真实的图片数据。

```
[python]
1. class pascal_voc(imdb):
```

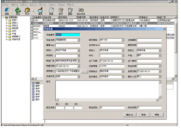
⚠

内容举报


⬆TOP

返回顶部





设备管理软件



捕鱼游戏源码

望京soho 穷人贷款 希腊房价
开源商城系统 商城系统源码
 workflows管理 数据可视化分析
仓库管理系统 开源网店系统

广告

```

1. class PascalVOC(object):
2.     def __init__(self, image_set, year, devkit_path=None):
3.         imdb.__init__(self, 'voc_' + year + '_' + image_set)
4.         self._year = year
5.         self._image_set = image_set
6.         # self._devkit_path = self._get_default_path() if devkit_path is None \
7.         #                 else devkit_path
8.         self._devkit_path = os.path.join(cfg.ROOT_DIR, '..', 'datasets', 'train_data')
9.         self._data_path = os.path.join(self._devkit_path, 'formatted_dataset')
10.        self._classes = ('__background__', # always index 0
11.                         'text')
12.        self._class_to_ind = dict(zip(self.classes, xrange(self.num_classes)))
13.        self._image_ext = '.jpg'
14.        self._image_index = self._load_image_set_index()
15.        # Default to roidb handler
16.        self._roidb_handler = self.selective_search_roidb
17.        self._salt = str(uuid.uuid4())
18.        self._comp_id = 'comp4'
19.
20.        # PASCAL specific config options
21.        self.config = {'cleanup'      : True,
22.                      'use_salt'     : True,
23.                      'use_diff'     : False,
24.                      'matlab_eval'  : False,
25.                      'rpn_file'     : None,
26.                      'min_size'     : 2}
27.
28.        assert os.path.exists(self._devkit_path), \
29.               'VOCdevkit path does not exist: {}'.format(self._devkit_path)
30.        assert os.path.exists(self._data_path), \
31.               'Path does not exist: {}'.format(self._data_path)

```

在pascal_voc的构造函数中，定义了imdb的结构，那么roidb与imdb有什么关系呢？

回到 /py-faster-rcnn/tools/train_net.py的combined_roidb中：

```

[python]
1.  imdb = get_imdb(imdb_name)
2.  print 'Loaded dataset `{:s}` for training'.format(imdb.name)
3.  imdb.set_proposal_method(cfg.TRAIN.PROPOSAL_METHOD)
4.  print 'Set proposal method: {:s}'.format(cfg.TRAIN.PROPOSAL_METHOD)
5.  roidb = get_training_roidb(imdb)
6.  return roidb

```

其中，set_proposal_method方法在/py-faster-rcnn/lib/datasets/imdb.py中：

```

[python]
1.  def set_proposal_method(self, method):
2.      method = eval('self.' + method + '_roidb')
3.      self.roidb_handler = method

```

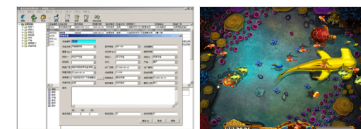
所以set_proposal_method是用于设置生成proposal的方法。



内容举报



返回顶部



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

工作流程管理系统 数据可视化分析

仓库管理系统 开源网店系统



get_training_roidb方法在/py-faster-rcnn/lib/fast-rcnn/train.py中：

```
[python]
1. def get_training_roidb(imdb):
2.     """Returns a roidb (Region of Interest database) for use in training."""
3.     if cfg.TRAIN.USE_FLIPPED:
4.         print 'Appending horizontally-flipped training examples...'
5.         imdb.append_flipped_images()
6.         print 'done'
7.
8.     print 'Preparing training data...'
9.     rdl_roidb.prepare_roidb(imdb)
10.    print 'done'
11.
12.    return imdb.roidb
```

get_training_roidb方法中包含了两个方法：append_flipped_images() 和prepare_roidb()方法。

a) append_flipped_images()：对imdb中涉及到的图像做了一个水平镜像，使得trainval中的图片的数量加倍。

b) prepare_roidb()：定义roidb的相关信息。

其中，append_flipped_images()方法定义在/py-faster-rcnn/lib/datasets/imdb.py中：

```
[python]
1. def append_flipped_images(self):
2.     num_images = self.num_images
3.     widths = self._get_widths()
4.     for i in xrange(num_images):
5.         boxes = self.roidb[i]['boxes'].copy()
6.         oldx1 = boxes[:, 0].copy()
7.         oldx2 = boxes[:, 2].copy()
8.         boxes[:, 0] = widths[i] - oldx2 - 1
9.         boxes[:, 2] = widths[i] - oldx1 - 1
10.        assert (boxes[:, 2] >= boxes[:, 0]).all()
11.        entry = {'boxes' : boxes,
12.                 'gt_overlaps' : self.roidb[i]['gt_overlaps'],
13.                 'gt_classes' : self.roidb[i]['gt_classes'],
14.                 'flipped' : True}
15.        self.roidb.append(entry)
16.        self._image_index = self._image_index * 2
```

prepare_roidb()方法定义在/py-faster-rcnn/lib/roi_data_layer/roidb.py中：

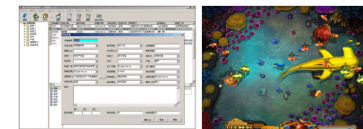
```
[python]
1. def prepare_roidb(imdb):
2.     """Enrich the imdb's roidb by adding some derived quantities that
3.     are useful for training. This function precomputes the maximum
4.     bounding box overlap that each box has with every other box in the
5.     dataset, as well as the area of the maximum overlap and the set of
6.     other boxes that it overlaps with.
7.     """
```



内容举报



返回顶部



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

workflow管理系统 数据可视化分析

仓库管理系统 开源网店系统

广告

```
4.         overlap, taken over ground-truth boxes, between each ROI and
5.         each ground-truth box. The class with maximum overlap is also
6.         recorded.
7.         """
8.         sizes = [PIL.Image.open(imdb.image_path_at(i)).size
9.                   for i in xrange(imdb.num_images)]
10.        roidb = imdb.roidb
11.        for i in xrange(len(imdb.image_index)):
12.            roidb[i]['image'] = imdb.image_path_at(i)
13.            roidb[i]['width'] = sizes[i][0]
14.            roidb[i]['height'] = sizes[i][1]
15.            # need gt_overlaps as a dense array for argmax
16.            gt_overlaps = roidb[i]['gt_overlaps'].toarray()
17.            # max overlap with gt over classes (columns)
18.            max_overlaps = gt_overlaps.max(axis=1)
19.            # gt class that had the max overlap
20.            max_classes = gt_overlaps.argmax(axis=1)
21.            roidb[i]['max_classes'] = max_classes
22.            roidb[i]['max_overlaps'] = max_overlaps
23.            # sanity checks
24.            # max overlap of 0 => class should be zero (background)
25.            zero_inds = np.where(max_overlaps == 0)[0]
26.            assert all(max_classes[zero_inds] == 0)
27.            # max overlap > 0 => class should not be zero (must be a fg class)
28.            nonzero_inds = np.where(max_overlaps > 0)[0]
29.            assert all(max_classes[nonzero_inds] != 0)
```

由此可见，roidb是imdb的一个成员变量，roidb是一个list（list的每个元素对应一张图片）。其中，list中的每个元素是一个字典，字典中存放的key包括：boxes, gt_overlaps, gt_classes, flipped, seg_areas, image, width, height, max_classes, max_overlaps。至此，就利用我们提供的数据集，准备好了roidb的相关信息。

那么，真正读取数据到内存的地方是在哪儿呢？

在py-faster-rcnn/lib/roi_data_layer/layer.py文件中：

在RoIDataLayer类的forward(self, bottom, top)函数中，

利用blobd = self._get_next_minibatch(roidb, num_classes)，产生了需要的blobs。

_get_next_minibatch函数调用了minibatch.py文件中的get_minibatch(roidb, num_classes)函数。

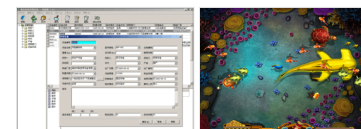
get_minibatch函数又调用了同为minibatch.py文件中的_get_image_blob(roidb, scale_inds)函数。

[python]

```
1. def _get_image_blob(roidb, scale_inds):
2.     """Builds an input blob from the images in the roidb at the specified
3.     scales.
4.     """
5.     num_images = len(roidb)
6.     processed_ims = []
7.     im_scales = []
```

内容举报

返回顶部



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价
开源商城系统 商城系统源码
工作流管理系统 数据可视化分析
仓库管理系统 开源网店系统

广告

```
8.         for i in xrange(num_images):
9.             im = cv2.imread(roidb[i]['image'])
10.            if roidb[i]['flipped']:
11.                im = im[:, ::-1, :]
12.            target_size = cfg.TRAIN.SCALES[scale_inds[i]]
13.            im, im_scale = prep_im_for_blob(im, cfg.PIXEL_MEANS, target_size,
14.                                           cfg.TRAIN.MAX_SIZE)
15.            im_scales.append(im_scale)
16.            processed_ims.append(im)
```

通过cv2.imread，实现了将图片读取到内存。

在_get_image_blob函数中，可以看到图片会被缩放到预先定义的size。其中，短边为cfg.TRAIN.SCALES，长边最长不能超过cfg.TRAIN.MAX_SIZE。

回到get_minibatch函数中，可以看到：

```
[python]
1.  # Get the input image blob, formatted for caffe
2.  im_blob, im_scales = _get_image_blob(roidb, random_scale_inds)
3.
4.  blobs = {'data': im_blob}
5.  ....
6.  blobs['im_info'] = np.array(
7.      [[im_blob.shape[2], im_blob.shape[3], im_scales[0]]],
8.      dtype=np.float32)
```

也就是，对于一副任意大小的P×Q图像（假设P为短边，Q为长边），首先reshape到M×N，其中M由cfg.TRAIN.SCALES决定，N由cfg.TRAIN.MAX_SIZE决定。blob中的data为reshape后的图像。im_info = [M,N,scale_factor]则保存了此次缩放的所有信息。

至此，介绍了数据是如何准备和进入网络的。

三.整个网络是如何构建的？

网络主要分为四个部分：

- 1.Conv layers
- 2.Region Proposal Networks
- 3.Rol Pooling
- 4.Classification

1.Conv layers

Conv layers包含了conv, pooling, relu三种层。

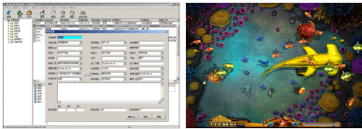
⚠

内容举报

⬆

TOP

返回顶部



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

workflow管理系统 数据可视化分析

仓库管理系统 开源网店系统



👍

0

☰

🔖

💬

🔗



在Conv layers中：

- 1) 所有的conv层都是：kernel_size = 3, pad = 1
- 2) 所有的pooling层都是：kernel_size = 2, stride = 2

在Faster-rcnn中所有的conv layers中，其对所有的卷积都做了扩边处理（pad=1），导致原图变为(M+2)*(N+2)大小，再做 3 × 3 卷积后输出M × N。正是这种设置，导致Conv layers中的conv层不改变输入和输出矩阵大小。

类似，在所有的pooling层中，kernel_size=2, stride=2, 这样每个经过pooling层的M × N矩阵，都会变成(M/2) × (N/2)大小。综上所述，在整个Conv layers中，conv和relu层不改变输入输出的大小，只有pooling层使得输出的长宽都变为输入的1/2。因此，一个M × N大小的矩阵经过Conv layers固定变为(M/16) × (N/16)。这样Conv layers生成的feature map都可以和原图对应起来。注意：以下如无特殊声明，说的原图都是指的reshape后的M × N大小的图像。

2.Region Proposal Networks

- 1) 使用n*n的滑动窗口在最后一个共享卷积层上提取信息

论文原文：“To generate region proposals, we slide a small network over the convolutional feature map output by the last shared convolutional layer. This small network takes as input an n × n spatial windowthe input convolutional feature map.”
也就是使用n*n的滑动窗口在最后一个共享卷积层上提取信息。论文后面提到n=3。
这一部分的实现，对应rpn_conv/3×3卷积。

在train.prototxt中的定义为：

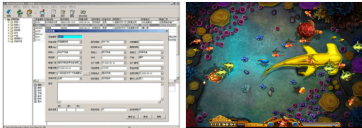
```
[python]
1. layer {
2.   name: "rpn_conv/3x3"
3.   type: "Convolution"
4.   bottom: "conv5_3"
5.   top: "rpn/output"
6.   param { lr_mult: 1.0 }
7.   param { lr_mult: 2.0 }
8.   convolution_param {
9.     num_output: 512
10.    kernel_size: 3 pad: 1 stride: 1
11.    weight_filler { type: "gaussian" std: 0.01 }
12.    bias_filler { type: "constant" value: 0 }
13.  }
14. }
```

- 2) 得到box-classification层和box-regression层

论文原文：“This feature is fed into two sibling fully-connected layers—a box-regression layer (reg) and box-classification layer (cls).”
“This architecture is naturally implemented with an n × n convolutional layer followed by two sibling 1 × 1 convolutional layers (for reg and cls, respectively).”

内容举报

返回顶部



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

workflows管理 数据可视化分析

仓库管理系统 开源网店系统

广告

内容举报

返回顶部

在前面得到的rpn_conv层的基础上，分别通过 1×1 卷积，得到两个层，box-regression层和box-classification层。

在train.prototxt中的定义为：

```
[python]
1. layer {
2.   name: "rpn_cls_score"
3.   type: "Convolution"
4.   bottom: "rpn/output"
5.   top: "rpn_cls_score"
6.   param { lr_mult: 1.0 }
7.   param { lr_mult: 2.0 }
8.   convolution_param {
9.     num_output: 18 # 2(bg/fg) * 9(anchors)
10.    kernel_size: 1 pad: 0 stride: 1
11.    weight_filler { type: "gaussian" std: 0.01 }
12.    bias_filler { type: "constant" value: 0 }
13.  }
14. }
15.
16. layer {
17.   name: "rpn_bbox_pred"
18.   type: "Convolution"
19.   bottom: "rpn/output"
20.   top: "rpn_bbox_pred"
21.   param { lr_mult: 1.0 }
22.   param { lr_mult: 2.0 }
23.   convolution_param {
24.     num_output: 36 # 4 * 9(anchors)
25.     kernel_size: 1 pad: 0 stride: 1
26.     weight_filler { type: "gaussian" std: 0.01 }
27.     bias_filler { type: "constant" value: 0 }
28.   }
29. }
```

论文中提到的最小化损失的目标函数为：

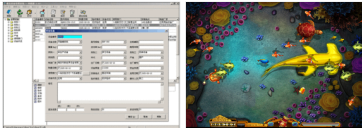
$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*)$$

目标函数中的参数p(i)，t(i)，Nreg等参数，都将通过下面的rpn.anchor_target_layer层来得到。下面，就来一一说明。

3) 使用自定义的rpn.anchor_target_layer生成anchor和其他关键信息

在train.prototxt中的定义为：

```
[python]
1. layer {
2.   name: 'rpn-data'
```



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

workflows管理 数据可视化分析

仓库管理系统 开源网店系统



内容举报

返回顶部



```
3.     type: 'Python'
4.     bottom: 'rpn_cls_score'
5.     bottom: 'gt_boxes'
6.     bottom: 'im_info'
7.     bottom: 'data'
8.     top: 'rpn_labels'
9.     top: 'rpn_bbox_targets'
10.    top: 'rpn_bbox_inside_weights'
11.    top: 'rpn_bbox_outside_weights'
12.    python_param {
13.        module: 'rpn.anchor_target_layer'
14.        layer: 'AnchorTargetLayer'
15.        param_str: "'feat_stride': 16"
16.    }
17. }
```

1.生成anchors

所谓anchors，实际是一组由rpn/generate_anchors.py生成的矩形。直接运行代码中的generate_anchors.py可以得到以下输出：

```
[python]
1.  array([[ -83.,  -39.,  100.,   56.],
2.         [-175.,  -87.,  192.,  104.],
3.         [-359., -183.,  376.,  200.],
4.         [ -55.,  -55.,   72.,   72.],
5.         [-119., -119.,  136.,  136.],
6.         [-247., -247.,  264.,  264.],
7.         [ -35.,  -79.,   52.,   96.],
8.         [ -79., -167.,   96.,  184.],
9.         [-167., -343.,  184.,  360.]])
```

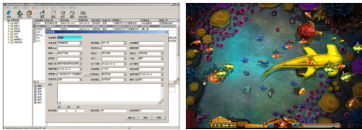
这个是rpn/output输出的feature map的(0,0)位置的anchor坐标。其中每行的4个值[x1,y1,x2,y2]代表矩阵左上角和右下角点的坐标。一共有9行，代表feature map中的每个点都会生成9个anchors。
生成的anchors有三种面积，128*128，256*256，512*512。对于每一种面积的anchors，其长宽比为1:1，1:2，2:1。

代码细节：

(1) 生成feature map的(0,0)位置的anchor坐标：
在/py-faster-rcnn/lib/rpn/generate_anchors.py中：

a) 设置base anchor的坐标为[0,0,15,15]。
Q：为什么设置将base anchor设置为[0,0,15,15]？

A：对于rpn/output输出的feature map的(0,0)位置的像素，对应的是原图像中(0,0)到(15,15)(左上角到右下角)位置的像素。因为，相比原图，feature map缩小了16倍！base anchor的面积为16*16，设置的scale为8，16，32，令边长 = base_anchor的边长*scale，就可以得到需要的anchors的面积，128*128，256*256，512*512。



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

workflow管理系统 数据可视化分析

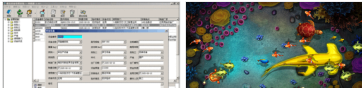
仓库管理系统 开源网店系统

广告

⚠
内容举报

⬆
TOP

返回顶部





望京soho 穷人贷款 希腊房价
开源商城系统 商城系统源码
 workflows管理 数据可视化分析
仓库管理系统 开源网店系统

广告

[python]

```
1. def generate_anchors(base_size=16, ratios=[0.5, 1, 2],
2.                       scales=2**np.arange(3, 6)):
3.     """
4.     Generate anchor (reference) windows by enumerating aspect ratios X
5.     scales wrt a reference (0, 0, 15, 15) window.
6.     """
7.
8.     base_anchor = np.array([1, 1, base_size, base_size]) - 1
9.     ratio_anchors = _ratio_enum(base_anchor, ratios)
10.    anchors = np.vstack([_scale_enum(ratio_anchors[i, :], scales)
11.                          for i in xrange(ratio_anchors.shape[0])])
12.    return anchors
```

b) 设置不同的长宽比和面积

[python]

```
1. def _ratio_enum(anchor, ratios):
2.     """
3.     Enumerate a set of anchors for each aspect ratio wrt an anchor.
4.     """
5.     w, h, x_ctr, y_ctr = _whctrs(anchor)
6.     size = w * h
7.     size_ratios = size / ratios
8.     ws = np.round(np.sqrt(size_ratios))
9.     hs = np.round(ws * ratios)
10.    anchors = _mkanchors(ws, hs, x_ctr, y_ctr)
11.    return anchors
12.
13. def _scale_enum(anchor, scales):
14.     """
15.     Enumerate a set of anchors for each scale wrt an anchor.
16.     """
17.
18.     w, h, x_ctr, y_ctr = _whctrs(anchor)
19.     ws = w * scales
20.     hs = h * scales
21.     anchors = _mkanchors(ws, hs, x_ctr, y_ctr)
22.     return anchors
```

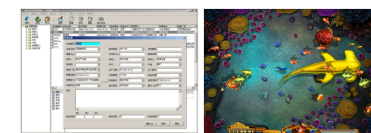
(2) 生成feature map的其他位置的anchor坐标:

在/py-faster-rcnn/lib/rpn/anchor_target_layer.py中:

a) 计算偏移量

计算偏移量的原理: 原图的大小是feature map的16倍, 因此, 计算feature map其他位置的anchors, 相对于(0,0)位置的anchors在原图的偏移

量 需要将其在feature map中相对于(0,0)的偏移量*16



设备管理软件 捕鱼游戏源码
望京soho 穷人贷款 希腊房价


内容举报
返回顶部



三，而系统会生成anchors map如下所示（图中略去了400

```
[python]
1. # 1. Generate proposals from bbox deltas and shifted anchors
2.     shift_x = np.arange(0, width) * self._feat_stride
3.     shift_y = np.arange(0, height) * self._feat_stride
4.     shift_x, shift_y = np.meshgrid(shift_x, shift_y)
5.     shifts = np.vstack((shift_x.ravel(), shift_y.ravel(),
6.                         shift_x.ravel(), shift_y.ravel())).transpose()
7.
```

b) 累积得到anchors

```
[python]
1. # add A anchors (1, A, 4) to
2. # cell K shifts (K, 1, 4) to get
3. # shift anchors (K, A, 4)
4. # reshape to (K*A, 4) shifted anchors
5. A = self._num_anchors
6. K = shifts.shape[0]
7. all_anchors = (self._anchors.reshape((1, A, 4)) +
8.               shifts.reshape((1, K, 4)).transpose((1, 0, 2)))
9. all_anchors = all_anchors.reshape((K * A, 4))
10. total_anchors = int(K * A)
```

c) 过滤掉不在原图内的anchors

```
[python]
1. # only keep anchors inside the image
2. inds_inside = np.where(
3.     (all_anchors[:, 0] >= -self._allowed_border) &
4.     (all_anchors[:, 1] >= -self._allowed_border) &
5.     (all_anchors[:, 2] < im_info[1] + self._allowed_border) & # width
6.     (all_anchors[:, 3] < im_info[0] + self._allowed_border) # height
7. )[0]
```

2.生成 rpn_labels

(1) 在原论文中，对于positive label的标准：

" We assign a positive label to two kinds of anchors:

(i) the anchor/anchors with the highest Intersection-over-Union (IoU) overlap with a ground-truth box,

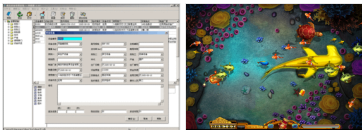
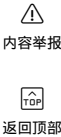
(ii) an anchor that has an IoU overlap higher than 0.7 with any ground-truth box. Note that a single ground-truth box may assign positive labels to multiple anchors."

对于negative label的标准：

We assign a negative label to a non-positive anchor if its IoU ratio is lower than 0.3 for all ground-truth boxes. Anchors that are neither



开源商城系统 商城系统源码
 workflows管理 系统 数据可视化分析
 仓库管理系统 开源网店系统



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价
 开源商城系统 商城系统源码
 workflows管理 系统 数据可视化分析
 仓库管理系统 开源网店系统





positive nor negative do not contribute to the training objective."

在源码中的实现：将positive label设置为 1，将negative label设置为0。

```
[python]
1. if not cfg.TRAIN.RPN_CLOBBER_POSITIVES:
2.     # assign bg labels first so that positive labels can clobber them
3.     labels[max_overlaps < cfg.TRAIN.RPN_NEGATIVE_OVERLAP] = 0
4.
5. # fg label: for each gt, anchor with highest overlap
6. labels[gt_argmax_overlaps] = 1
7.
8. # fg label: above threshold IOU
9. labels[max_overlaps >= cfg.TRAIN.RPN_POSITIVE_OVERLAP] = 1
10.
11. if cfg.TRAIN.RPN_CLOBBER_POSITIVES:
12.     # assign bg labels last so that negative labels can clobber positives
13.     labels[max_overlaps < cfg.TRAIN.RPN_NEGATIVE_OVERLAP] = 0
```

(2) 筛选anchors

由于anchors 的数量很多，因此，在每一张图像中，只选择256个anchors进行训练。

将正负样本的比例设为1:1，如果正样本的数量小于128个，则用负样本进行填充。

源码中的实现：源码中的实现与论文中的描述一致。同时，将多余的样本设置为 - 1。

所以：positive label：1; negative label：0; disabled label：-1.

```
[python]
1. # subsample positive labels if we have too many
2. num_fg = int(cfg.TRAIN.RPN_FG_FRACTION * cfg.TRAIN.RPN_BATCHSIZE)
3. fg_inds = np.where(labels == 1)[0]
4. if len(fg_inds) > num_fg:
5.     disable_inds = npr.choice(
6.         fg_inds, size=(len(fg_inds) - num_fg), replace=False)
7.     labels[disable_inds] = -1
8.
9. # subsample negative labels if we have too many
10. num_bg = cfg.TRAIN.RPN_BATCHSIZE - np.sum(labels == 1)
11. bg_inds = np.where(labels == 0)[0]
12. if len(bg_inds) > num_bg:
13.     disable_inds = npr.choice(
14.         bg_inds, size=(len(bg_inds) - num_bg), replace=False)
15.     labels[disable_inds] = -1
```

3.生成rpn_bbox_targets

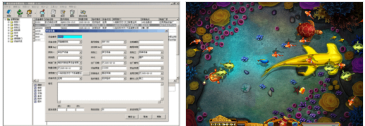
论文原文：“ti is a vector representing the 4 parameterized coordinates of the predicted bounding box, and t* i is that of the ground-truth box associated with a positive anchor.”

这里提到t(i)与t(*)都是经过parameterized的bounding box的坐标。

那么，具体是如何parameterized的呢？



0



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

workflows管理 系统 数据可视化分析

仓库管理系统 开源网店系统





$$t_x = (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a,$$

$$t_w = \log(w/w_a), \quad t_h = \log(h/h_a),$$

$$t_x^* = (x^* - x_a)/w_a, \quad t_y^* = (y^* - y_a)/h_a,$$

$$t_w^* = \log(w^*/w_a), \quad t_h^* = \log(h^*/h_a),$$

源码中的实现：

[python]

```
1. bbox_targets = np.zeros((len(inds_inside), 4), dtype=np.float32)
2. bbox_targets = _compute_targets(anchors, gt_boxes[argmax_overlaps, :])
```

[python]

```
1. def _compute_targets(ex_rois, gt_rois):
2.     """Compute bounding-box regression targets for an image."""
3.
4.     assert ex_rois.shape[0] == gt_rois.shape[0]
5.     assert ex_rois.shape[1] == 4
6.     assert gt_rois.shape[1] == 5
7.
8.     return bbox_transform(ex_rois, gt_rois[:, :4]).astype(np.float32, copy=False)
```

来看一下/py-faster-rcnn/lib/fast_rcnn/bbox_transform.py中的bbox_transform函数：

[python]

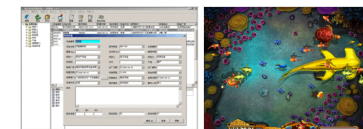
```
1. def bbox_transform_inv(boxes, deltas):
2.     if boxes.shape[0] == 0:
3.         return np.zeros((0, deltas.shape[1]), dtype=deltas.dtype)
4.
5.     boxes = boxes.astype(deltas.dtype, copy=False)
6.
7.     widths = boxes[:, 2] - boxes[:, 0] + 1.0
8.     heights = boxes[:, 3] - boxes[:, 1] + 1.0
9.     ctr_x = boxes[:, 0] + 0.5 * widths
10.    ctr_y = boxes[:, 1] + 0.5 * heights
11.
12.    dx = deltas[:, 0::4]
13.    dy = deltas[:, 1::4]
14.    dw = deltas[:, 2::4]
15.    dh = deltas[:, 3::4]
16.
17.    pred_ctr_x = dx * widths[:, np.newaxis] + ctr_x[:, np.newaxis]
18.    pred_ctr_y = dy * heights[:, np.newaxis] + ctr_y[:, np.newaxis]
19.    pred_w = np.exp(dw) * widths[:, np.newaxis]
20.    pred_h = np.exp(dh) * heights[:, np.newaxis]
21.
```



内容举报



返回顶部



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

workflow管理系统 数据可视化分析

仓库管理系统 开源网店系统



0



```
22.     pred_boxes = np.zeros(deltas.shape, dtype=deltas.dtype)
23.     # x1
24.     pred_boxes[:, 0::4] = pred_ctr_x - 0.5 * pred_w
25.     # y1
26.     pred_boxes[:, 1::4] = pred_ctr_y - 0.5 * pred_h
27.     # x2
28.     pred_boxes[:, 2::4] = pred_ctr_x + 0.5 * pred_w
29.     # y2
30.     pred_boxes[:, 3::4] = pred_ctr_y + 0.5 * pred_h
31.
32.     return pred_boxes
```

因此，返回的rpn_bbox_targets为anchor与gt bbox之间的差距值。同理，前面的rpn_bbox_pred层返回的rpn_bbox_pred也为预测的bbox与gt bbox之间的差距值。

4.生成 rpn_bbox_inside_weights

```
[python]
1.     bbox_inside_weights = np.zeros((len(inds_inside), 4), dtype=np.float32)
2.     bbox_inside_weights[labels == 1, :] = np.array(cfg.TRAIN.RPN_BBOX_INSIDE_WEIGHTS)
```

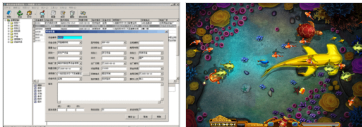
因此，rpn_bbox_inside_weights就是公式里面的p(i)，对于正样本为1，对于负样本为0。

5.生成rpn_bbox_outside_weights

```
[python]
1.     bbox_outside_weights = np.zeros((len(inds_inside), 4), dtype=np.float32)
2.     if cfg.TRAIN.RPN_POSITIVE_WEIGHT < 0:
3.         # 实现均匀取样
4.         # uniform weighting of examples (given non-uniform sampling)
5.         num_examples = np.sum(labels >= 0)
6.         positive_weights = np.ones((1, 4)) * 1.0 / num_examples
7.         negative_weights = np.ones((1, 4)) * 1.0 / num_examples
8.     else:
9.         assert ((cfg.TRAIN.RPN_POSITIVE_WEIGHT > 0) &
10.                (cfg.TRAIN.RPN_POSITIVE_WEIGHT < 1))
11.         positive_weights = (cfg.TRAIN.RPN_POSITIVE_WEIGHT /
12.                             np.sum(labels == 1))
13.         negative_weights = ((1.0 - cfg.TRAIN.RPN_POSITIVE_WEIGHT) /
14.                             np.sum(labels == 0))
15.         bbox_outside_weights[labels == 1, :] = positive_weights
16.         bbox_outside_weights[labels == 0, :] = negative_weights
```

在cfg文件里面，cfg.TRAIN.RPN_POSITIVE_WEIGHT为-1，因此这里是对正负样本的权重都除以样本总数，相当于实现了1/Nreg的功能。

写到这里，把anchor_target_layer.py所做的工作就大致讲完了。这一层是利用现有的信息进行转化，得到我们需要的信息。因此，不需要进行反向传播。



设备管理软件 捕鱼游戏源码

- 望京soho 穷人贷款 希腊房价
- 开源商城系统 商城系统源码
- workflow管理系统 数据可视化分析
- 仓库管理系统 开源网店系统

广告



下面再回头看前面提到的box-classification层和box-regression层。

4) 利用box-classification层，使用softmax判定foreground与background

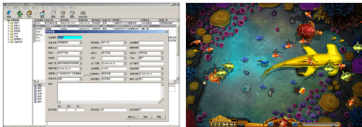
```
[python]
1. layer {
2.   name: "rpn_cls_score"
3.   type: "Convolution"
4.   bottom: "rpn/output"
5.   top: "rpn_cls_score"
6.   param { lr_mult: 1.0 }
7.   param { lr_mult: 2.0 }
8.   convolution_param {
9.     num_output: 18 # 2(bg/fg) * 9(anchors)
10.    kernel_size: 1 pad: 0 stride: 1
11.    weight_filler { type: "gaussian" std: 0.01 }
12.
13.    bias_filler { type: "constant" value: 0 }
14.  }
```

可以看到其num_output=18，也就是经过该卷积的输出图像为W×H×18大小。刚好对应了feature map的每一个点都有9个anchors，同时每个anchors又有可能是foreground和background，所有这些信息都保存为W×H×(9×2)大小的矩阵。

```
[python]
1. layer {
2.   bottom: "rpn_cls_score"
3.   top: "rpn_cls_score_reshape"
4.   name: "rpn_cls_score_reshape"
5.   type: "Reshape"
6.   reshape_param { shape { dim: 0 dim: 2 dim: -1 dim: 0 } }
7. }
8.
9. layer {
10.  name: "rpn_cls_prob"
11.  type: "Softmax"
12.  bottom: "rpn_cls_score_reshape"
13.  top: "rpn_cls_prob"
14. }
15.
16. layer {
17.  name: 'rpn_cls_prob_reshape'
18.  type: 'Reshape'
19.  bottom: 'rpn_cls_prob'
20.  top: 'rpn_cls_prob_reshape'
21.  reshape_param { shape { dim: 0 dim: 18 dim: -1 dim: 0 } }
22. }
```

⚠
内容举报

⬆
TOP
返回顶部



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价
开源商城系统 商城系统源码
工作流管理系统 数据可视化分析
仓库管理系统 开源网店系统



👍
0

☰

🔖

💬

🔗

可以看到，这里使用softmax分类获得foreground anchors，也就相当于初步提取了检测目标候选区域box（一般认为目标在foreground anchors中）。

注意：这里的分类是针对foreground和background，是一个粗略的分类，还没有判断目标的具体类别。

Q：为什么要在softmax前后都接一个reshape layer？

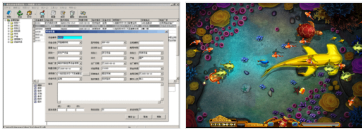
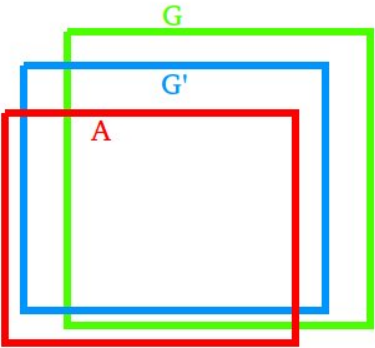
A：这里是为了便于softmax分类。具体原因要从caffe的实现形式说起。在caffe的基本数据结构blob中以如下形式保存数据：

blob=[batch_size, channel, height, width]

对应至上面保存的bg/fg anchors的矩阵，其在caffe blob中的存储形式为[1, 2*9, H, W]。而在softmax分类时需要进行fg/bg二分类，所以reshape layer会将其变为[1, 2, 9*H, W]大小。之后再将其reshape为原状。

5) bounding box regression原理

先来介绍bounding box regression原理。如下图左边所示绿色框为第一行文字的Ground Truth(GT)，红色为提取出的anchors。那么，即使红色的框被判断为文字，由于红色的框定位不准，也相当于没有正确的检测出文本行。所以，希望采用一种方法对红色的框进行微调，使得foreground anchors与GT更加接近。



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

workflow管理系统 数据可视化分析

仓库管理系统 开源网店系统

广告



内容举报



返回顶部



对于窗口一般使用四维向量(x,y,w,h)表示，分别表示窗口的中心点坐标和宽高。如上图右边所示，红色的框 A 表示原始的Foreground Anchors，绿色的框 G 代表目标的GT，我们的目标是寻找一种关系，使得输入原始的Anchor A经过映射得到一个跟真实窗口 G 更加接近的回归窗口 G'，即给定anchor A=(Ax, Ay, Aw, Ah)，GT=(Gx, Gy, Gw, Gh)，寻找一种变换，使得F(Ax, Ay, Aw, Ah)=(G'x, G'y, G'w, G'h)，其中(G'x, G'y, G'w, G'h)≈(Gx, Gy, Gw, Gh)。

那么，经过何种变换 F 才能从右图中的anchor A变为 G 呢？

1.先做平移：

$$\begin{aligned} G'_x &= A_w \cdot d_x(A) + A_x \\ G'_y &= A_h \cdot d_y(A) + A_y \end{aligned}$$

2.再做缩放：

$$\begin{aligned} G'_w &= A_w \cdot \exp(d_w(A)) \\ G'_h &= A_h \cdot \exp(d_h(A)) \end{aligned}$$

观察上面4个公式发现，需要学习的是dx(A)，dy(A)，dw(A)，dh(A)这四个变换。当输入的anchor A与GT相差较小时，可以认为这种变换是一种线性变换，那么就可以用线性回归来建模对窗口进行微调（注意，只有当anchors A和GT比较接近时，才能使用线性回归模型，否则就是复杂的非线性问题了）。对应于Faster RCNN原文，平移量(tx, ty)与尺度因子(tw, th)如下：

$$\begin{aligned} t_x &= (x - x_a)/w_a, & t_y &= (y - y_a)/h_a, \\ t_w &= \log(w/w_a), & t_h &= \log(h/h_a), \end{aligned}$$

接下来的问题就是如何通过线性回归获得dx(A)，dy(A)，dw(A)，dh(A)了。线性回归就是给定输入的特征向量X，学习一组参数W，使得经过线性回归后的值跟真实值Y非常接近，即Y=WX。对于该问题，输入X是一张经过卷积获得的feature map，定义为Φ；同时还有训练传入的GT，即(tx, ty, tw, th)。输出是dx(A)，dy(A)，dw(A)，dh(A)四个变换。那么目标函数可以表示为：

$$d_*(A) = w_*^T \cdot \Phi(A)$$

其中Φ(A)是对应anchor的feature map组成的特征向量，w是需要学习的参数，d(A)是得到的预测值（*表示x，y，w，h，也就是每一个变换对应一个上述目标函数）。为了让预测值(tx, ty, tw, th)与真实值差距最小，设计损失函数：

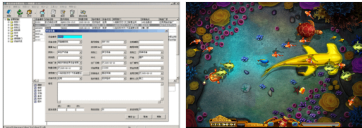
$$\text{Loss} = \sum_i^N \left(t_*^i - \hat{w}_*^T \cdot \Phi(A^i) \right)^2$$

函数优化目标为：

$$w_* = \operatorname{argmin}_{\hat{w}_*} \sum_i^N \left(t_*^i - \hat{w}_*^T \cdot \Phi(A^i) \right)^2 + \lambda \|\hat{w}_*\|^2$$

6) 对proposals进行bounding box regression

```
[python]
1. layer {
2.   name: "rpn_bbox_pred"
3.   ... "..."
```



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价
开源商城系统 商城系统源码
 workflows管理 数据可视化分析
仓库管理系统 开源网店系统

广告

内容举报

返回顶部

内容举报

返回顶部

```
3.     type: "Convolution"
4.     bottom: "rpn/output"
5.     top: "rpn_bbox_pred"
6.     param { lr_mult: 1.0 }
7.     param { lr_mult: 2.0 }
8.
9.     convolution_param {
10.        num_output: 36 # 4 * 9(anchors)
11.        kernel_size: 1 pad: 0 stride: 1
12.        weight_filler { type: "gaussian" std: 0.01 }
13.        bias_filler { type: "constant" value: 0 }
14.    }
```

可以看到，其num_output=36，即经过该卷积输出图像为 $W \times H \times 36$ ，在caffe blob存储为[1, 36, H, W]，这里相当于feature maps上，每个点都有9个anchors，每个anchors又有4个用于回归的[dx(A), dy(A), dw(A), dh(A)]变换量。

7) Proposal Layer

Proposal Layer负责综合所有[dx(A), dy(A), dw(A), dh(A)]变换量和foreground anchors，计算出精准的proposal，送入后续RoI Pooling Layer。
先来看看Proposal Layer的train.prototxt定义：

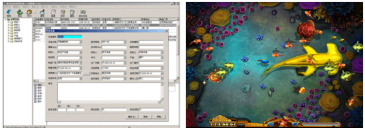
```
[python]
1. layer {
2.   name: 'proposal'
3.   type: 'Python'
4.   bottom: 'rpn_cls_prob_reshape'
5.   bottom: 'rpn_bbox_pred'
6.   bottom: 'im_info'
7.   top: 'rpn_rois'
8.   # top: 'rpn_scores'
9.   python_param {
10.    module: 'rpn.proposal_layer'
11.    layer: 'ProposalLayer'
12.    param_str: "'feat_stride': 16"
13.  }
14. }
```

Proposal Layer forward（caffe layer的前传函数）按照以下顺序依次处理：

- 1. 生成anchors，利用[dx(A), dy(A), dw(A), dh(A)]对所有的anchors做bbox regression回归（这里的anchors生成和训练时完全一致）也就是说，前面的网络是对anchor进行训练，而proposal层是用来生成anchor。
- 2. 按照输入的foreground softmax scores由大到小排序anchors，提取前pre_nms_topN(e.g. 6000)个anchors，即提取修正位置后的foreground anchors。
- 3. 限定超出图像边界的foreground anchors为图像边界（防止后续roi pooling时proposal超出图像边界）
- 4. 剔除非常小（width<threshold or height<threshold）的foreground anchors
- 5. 进行nonmaximum suppression
- 6. 再次按照nms后的foreground softmax scores由大到小排序fg anchors，提取前post_nms_topN(e.g. 300)结果作为proposal输出。

RPN网络结构就介绍到这里，总结起来就是：

生成anchors -> softmax分类器提取fg anchors -> bbox reg回归fg anchors -> Proposal Layer生成proposals



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

workflow管理系统 数据可视化分析

仓库管理系统 开源网店系统



⚠
内容举报

⬆
返回顶部

3.Rol Pooling

缩进 Rol Pooling层则负责收集proposal, 并计算出proposal feature maps, 送入后续网络。从图2中可以看到Rol pooling层有2个输入：

- 1. 原始的feature maps
- 2. RPN输出的proposal boxes (大小各不相同)

1) 为何需要Rol Pooling

先来看一个问题：对于传统的CNN (如AlexNet, VGG), 当网络训练好后输入的图像尺寸必须是固定值, 同时网络输出也是固定大小的vector or matrix。如果输入图像大小不定, 这个问题就变得比较麻烦。有2种解决办法：

- 1. 从图像中crop一部分传入网络
- 2. 将图像warp成需要的大小后传入网络



crop与warp破坏图像原有结构信息

两种办法的示意图如图, 可以看到无论采取那种办法都不好, 要么crop后破坏了图像的完整结构, 要么warp破坏了图像原始形状信息。回忆 RPN网络生成的proposals的方法: 对foreground anchors进行bound box regression, 那么这样获得的proposals也是大小形状各不相同, 即也存在上述问题。所以Faster RCNN中提出了Rol Pooling解决这个问题。

2) Rol Pooling原理

缩进

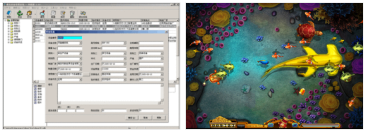
分析之前先来看看Rol Pooling Layer的train.prototxt的定义：

```
[python]
1. layer {
2.   name: "roi_pool5"
3.   type: "ROIPooling"
4.   bottom: "conv5_3"
5.   bottom: "rois"
6.   top: "pool5"
7.   roi_pooling_param {
8.     pooled_w: 7
9.     pooled_h: 7
10.    spatial_scale: 0.0625 # 1/16
11.  }
12. }
```

其中有新参数pooled_w=pooled_h=7。

Rol Pooling layer forward过程：在之前有明确提到：proposal=[x1, y1, x2, y2]是对应MxN尺度的, 所以首先使用spatial_scale参数将其映射回 (M/16)x(N/16)大小的feature maps尺度；之后将每个proposal水平和竖直都分为7份, 对每一份都进行max pooling处理。这样处理后, 即使大小不同的proposal, 输出结果都是7x7大小, 实现了fixed-length output (固定长度输出)。

Proposals



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

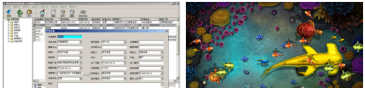
工作流管理系统 数据可视化分析

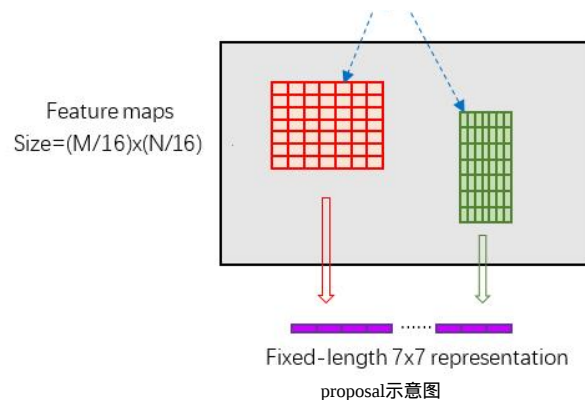
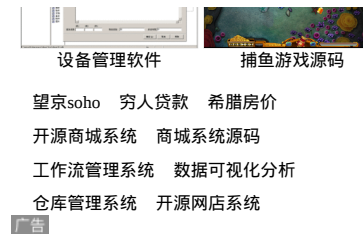
仓库管理系统 开源网店系统

广告

内容举报

返回顶部





上面是roi pooling 层的实现原理。下面，结合源代码，对具体实现细节进行解读。

rois中只保存了生成的region proposal，也就是坐标，所以，需要将这些坐标映射到原有的feature map:conv5_3上，得到映射信息，才能进行池化。

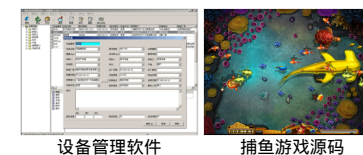
这一部分的代码在：/py-faster-rcnn/caffe-fast-rcnn/src/caffe/layers/roi_pooling_layer.cpp和

roi_pooling_layer.cu中，先来看cpu版本，也就是：roi_pooling_layer.cpp，主要代码在Forward_cpu函数中：

```
[cpp]
1. template <typename Dtype>
2. void ROIPoolingLayer<Dtype>::Forward_cpu(const vector<Blob<Dtype>*>& bottom,
3.     const vector<Blob<Dtype>*>& top) {
4.     //conv5-3信息
5.     const Dtype* bottom_data = bottom[0]->cpu_data();
6.     //rois信息
7.     const Dtype* bottom_rois = bottom[1]->cpu_data();
8.     //Number of ROIs
9.     int num_rois = bottom[1]->num();
10.    //样本大小
11.    int batch_size = bottom[0]->num();
12.    int top_count = top[0]->count();
13.
14.    //初始化top_data 和 argmax_data两个数组
15.    //caffe_set(const int N, const Dtype alpha, Dtype* argmax_data);
16.    Dtype* top_data = top[0]->mutable_cpu_data();
17.    caffe_set(top_count, Dtype(-FLT_MAX), top_data);
18.    int* argmax_data = max_idx->mutable_cpu_data();
19.    caffe_set(top_count, -1, argmax_data);
20.
21.    // For each ROI R=[batch_index x1 y1 x2 y2]: max pool over R
22.    for (int n = 0; n < num_rois; ++n) {
23.        int roi_batch_ind = bottom_rois[0];
24.        //将生成的rois的坐标映射到原来的feature map上
25.        //rois中只包含了坐标信息，而不包含feature map信息
26.        int roi_start_w = round(bottom_rois[1] * spatial_scale_);
27.        int roi_start_h = round(bottom_rois[2] * spatial_scale_);
28.        int roi_end_w = round(bottom_rois[3] * spatial_scale_);
```

内容举报

返回顶部



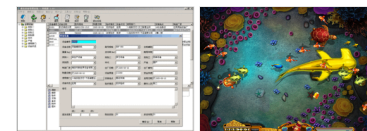
开源商城系统 商城系统源码
工作流管理系统 数据可视化分析
仓库管理系统 开源网店系统



```
28. int roi_end_w = round(bottom_rois[3] * spatial_scale_);
29. int roi_end_h = round(bottom_rois[4] * spatial_scale_);
30. CHECK_GE(roi_batch_ind, 0);
31. CHECK_LT(roi_batch_ind, batch_size);
32.
33. //每一个region在feature map上对应的大小
34. int roi_height = max(roi_end_h - roi_start_h + 1, 1);
35. int roi_width = max(roi_end_w - roi_start_w + 1, 1);
36.
37. //每一个sub region的大小
38. const Dtype bin_size_h = static_cast<Dtype>(roi_height) / static_cast<Dtype>
(pooled_height_);
39. const Dtype bin_size_w = static_cast<Dtype>(roi_width) / static_cast<Dtype>
(pooled_width_);
40.
41. const Dtype* batch_data = bottom_data + bottom[0]->offset(roi_batch_ind);
42.
43. for(int c = 0; c < channels_; ++c) {
44.     for(int ph = 0; ph < pooled_height_; ++ph) {
45.         for(int pw = 0; pw < pooled_width_; ++pw) {
46.             // Compute pooling region for this output unit:
47.             // start (included) = floor(ph * roi_height / pooled_height_)
48.             // end (excluded) = ceil((ph+1) * roi_height / pooled_height_)
49.             // floor(x):取小于等于x的整数, ceil(x):取大于x的整数
50.             //取得每一个sub region的起点终点坐标
51.             int hstart = static_cast<int>(floor(static_cast<Dtype>
(ph) * bin_size_h));
52.             int wstart = static_cast<int>(floor(static_cast<Dtype>
(pw) * bin_size_w));
53.             int hend = static_cast<int>(ceil(static_cast<Dtype>
(ph+1) * bin_size_h));
54.             int wend = static_cast<int>(ceil(static_cast<Dtype>
(pw+1) * bin_size_w));
55.
56.             hstart = min(max(hstart + roi_start_h, 0), height_);
57.             hend = min(max(hend + roi_start_h, 0), height_);
58.             wstart = min(max(wstart + roi_start_w, 0), width_);
59.             wend = min(max(wend + roi_start_w, 0), width_);
60.
61.             //剔除无效的roi
62.             bool is_empty = (hend <= hstart) || (wend <= wstart);
63.
64.             //池化区域的编号
65.             const int pool_index = ph * pooled_width_ + pw;
66.             if(is_empty){
67.                 //如果该区域无效,则将池化结果设为0
68.                 top_data[pool_index] = 0;
69.                 //将最大区域的index设为-1
70.                 argmax_data[pool_index] = -1;
71.             }
72.
73.             //进行最大池化操作
74.             //pool_index:7*7的某一个池化区域的索引, index:feature map某一点的索引
75.             for(int h = hstart; h < hend; ++h) {
76.                 for(int w = wstart; w < wend; ++w){
77.                     //计算在feature map中的索引
78.                     const int index = h * width_ + w;
```

内容举报

返回顶部



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价
开源商城系统 商城系统源码
工作流管理系统 数据可视化分析
仓库管理系统 开源网店系统



W

...

88

0

≡

W

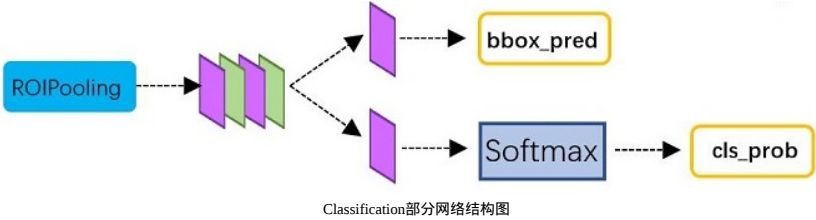
...

```
78.         const int index = n - width_ + w;  
79.         if(batch_data[index] > top_data[pool_index]){  
80.             top_data[pool_index] = batch_data[index];  
81.             argmax_data[pool_index] = index;  
82.         }  
83.     }  
84. }  
85.  
86. }  
87. }  
88.  
89. //Increment all data pointers by one channel  
90. //也就是，将指针指向下一个channel  
91. batch_data += bottom[0]->offset(0,1);  
92. top_data += top[0]->offset(0,1);  
93. argmax_data += max_idx_.offset(0,1);  
94. }  
95. //Increment ROI data pointer  
96. bottom_rois += bottom[1]->offset(1);  
97. }  
98. }
```

变量解释：
roi_height:region proposal的高度
roi_width:region proposal的宽度
将每一个region proposal都分为 7 × 7 的sub region:
对于每一个sub region:
bin_size_h: sub region的高度
bin_size_w:sub region的宽度
(wstart,hstart)为sub region左上角坐标， (wend,hend) 为sub region右下角坐标。

4.Classification

缩进Classification部分利用已经获得的proposal feature maps，通过full connect层与softmax计算每个proposal具体属于哪个类别，输出cls_prob概率向量；同时再次利用bounding box regression获得每个proposal的位置偏移量bbox_pred，用于回归更加精确的目标检测框。Classification部分网络结构如下图。



- 从RoI Pooling获取到7x7=49大小的proposal feature maps后，送入后续网络，可以看到做了如下2件事：
- 1. 通过全连接和softmax对proposals进行分类；
 - 2. 再次对proposals进行bounding box regression，获取更高精度的rect box。


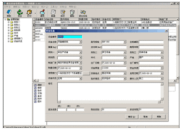
⚠

内容举报

⬆

TOP

返回顶部



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

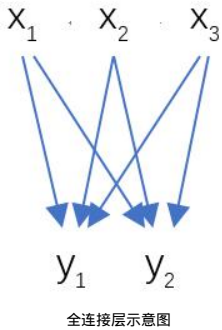
workflows管理 数据可视化分析

仓库管理系统 开源网店系统

广告



这里来看看全连接层InnerProduct layers，示意图如下图：



其计算公式如下：

$$(x_1 \quad x_2 \quad x_3) \begin{pmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \\ w_{31} & w_{32} \end{pmatrix} + (b_1 \quad b_2) = (y_1 \quad y_2)$$

其中W和bias B都是预先训练好的，即大小是固定的，当然输入X和输出Y也就是固定大小。所以，也就印证了Roi Pooling的必要性。

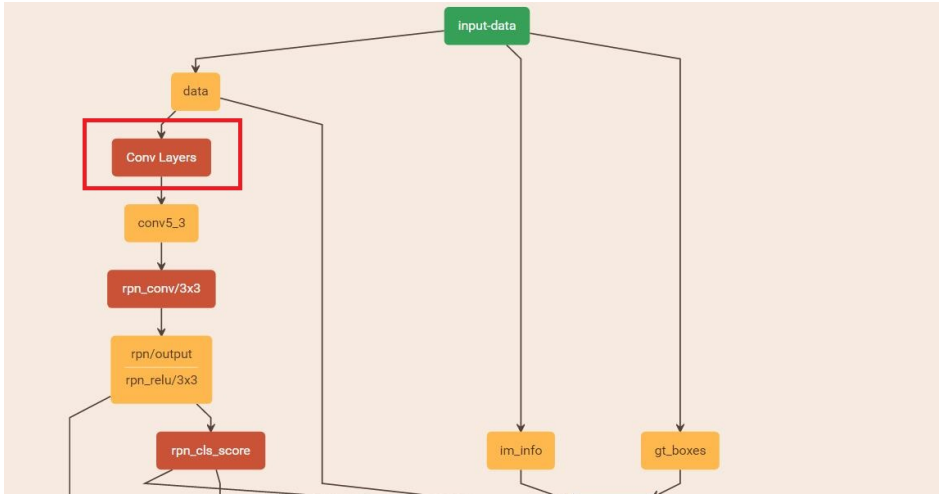
四.整个网络是如何训练的？

这里使用论文中提到的“Approximate joint training”，是一种end to end的训练方式。

将其称为Approximate joint training的原因是：将proposal层看作是固定的，而不对其计算loss。

下面，进行分解讲解：

1) RPN网络

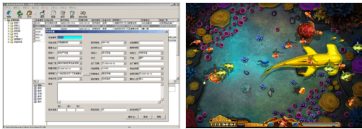


内容举报

返回顶部



0



设备管理软件

捕鱼游戏源码

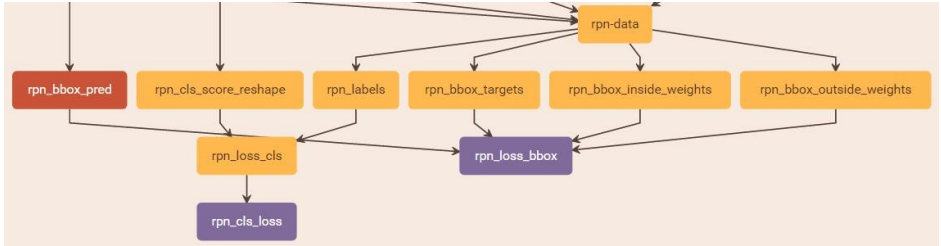
望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

workflow管理系统 数据可视化分析

仓库管理系统 开源网店系统





与检测网络类似的是，依然使用Conv Layers提取feature maps。整个网络使用的Loss如下：

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*).$$

在上述公式中，i表示anchors index，pi表示foreground softmax predict概率，pi*代表对应的GT predict概率（即当第i个anchor与GT间IoU>0.7，认为是该anchor是foreground，pi*=1；反之IoU<0.3时，认为是该anchor是background，pi*=0；至于那些0.3<IoU<0.7的anchor则不参与训练）；ti代表predict bounding box，ti*代表对应foreground anchor对应的GT box。可以看到，整个Loss分为2部分：

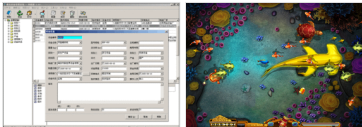
1. cls loss，即rpn_loss_cls层计算的softmax loss，用于分类anchors为foreground与background的网络训练
2. reg loss，即rpn_loss_bbox层计算的smooth L1 loss，用于bounding box regression网络训练。注意在该loss中乘了pi*，相当于只关心foreground anchors的回归。

由于在实际过程中，Ncls和Nreg差距过大，用参数λ平衡二者（如Ncls=256，Nreg=2400时设置λ=10），使总的网络Loss计算过程中能够均匀考虑2种Loss。这里比较重要是Lreg使用的smooth L1 loss，计算公式如下：

$$L_{reg}(t_i, t_i^*) = \sum_{i \in \{x,y,w,h\}} \text{smooth}_{L1}(t_i - t_i^*)$$
$$\text{smooth}_{L1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$

1. 在RPN训练阶段，rpn-data（python AnchorTargetLayer）层会按照和test阶段Proposal层完全一样的方式生成Anchors用于训练
 2. 对于rpn_loss_cls，输入的rpn_cls_scores_reshape和rpn_labels分别对应p与p*，Ncls参数隐含在p与p*的caffe blob的大小中
 3. 对于rpn_loss_bbox，输入的rpn_bbox_pred和rpn_bbox_targets分别对应ti于ti*，rpn_bbox_inside_weights对应p*，rpn_bbox_outside_weights对应1/Nreg。
- 特别需要注意的是，在训练和检测阶段生成和存储anchors的顺序完全一样，这样训练结果才能被用于检测！

内容举报
返回顶部



设备管理软件 捕鱼游戏源码

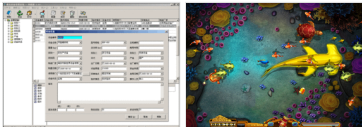
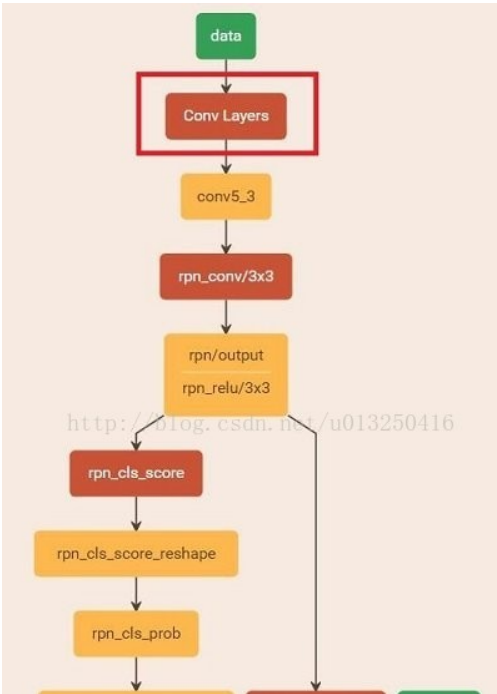
望京soho 穷人贷款 希腊房价
开源商城系统 商城系统源码
 workflow管理系统 数据可视化分析
仓库管理系统 开源网店系统

广告

2) 通过训练好的RPN网络收集proposals

在该步骤中，利用之前的RPN网络，获取proposal rois，同时获取foreground softmax probability，如下图。注意：在前向传播中，将该部分看作
是固定的，不对其计算loss。而实际上，本应该对proposal rois的坐标进行回归。所以，这种端到端的训练方式称为Approximate joint training。
如果是分步计算，此处应该产生loss。

```
[python]
1. layer {
2.   name: 'proposal'
3.   type: 'Python'
4.   bottom: 'rpn_cls_prob_reshape'
5.   bottom: 'rpn_bbox_pred'
6.   bottom: 'im_info'
7.   top: 'rpn_rois'
8.   # top: 'rpn_scores'
9.   python_param {
10.    module: 'rpn.proposal_layer'
11.    layer: 'ProposalLayer'
12.    param_str: "'feat_stride': 16"
13.  }
14. }
```



设备管理软件 捕鱼游戏源码

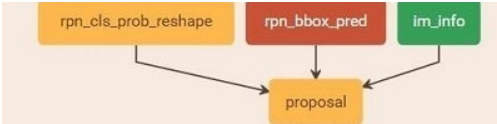
望京soho 穷人贷款 希腊房价
开源商城系统 商城系统源码
工作流管理系统 数据可视化分析
仓库管理系统 开源网店系统



⚠
内容举报

⬆
TOP
返回顶部

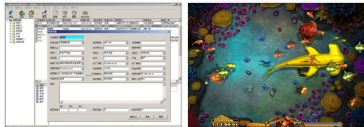




3) 训练Faster RCNN网络

将上面得到的rpn_rois和gt_boxes输入网络，进行变换，得到rois, labels, bbox_targets, bbox_inside_weights和bbox_outside_weights。

```
[python]
1. layer {
2.   name: 'roi-data'
3.   type: 'Python'
4.   bottom: 'rpn_rois'
5.   bottom: 'gt_boxes'
6.   top: 'rois'
7.   top: 'labels'
8.   top: 'bbox_targets'
9.   top: 'bbox_inside_weights'
10.  top: 'bbox_outside_weights'
11.  python_param {
12.    module: 'rpn.proposal_target_layer'
13.    layer: 'ProposalTargetLayer'
14.    param_str: "'num_classes': 2"
15.  }
16. }
```



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价

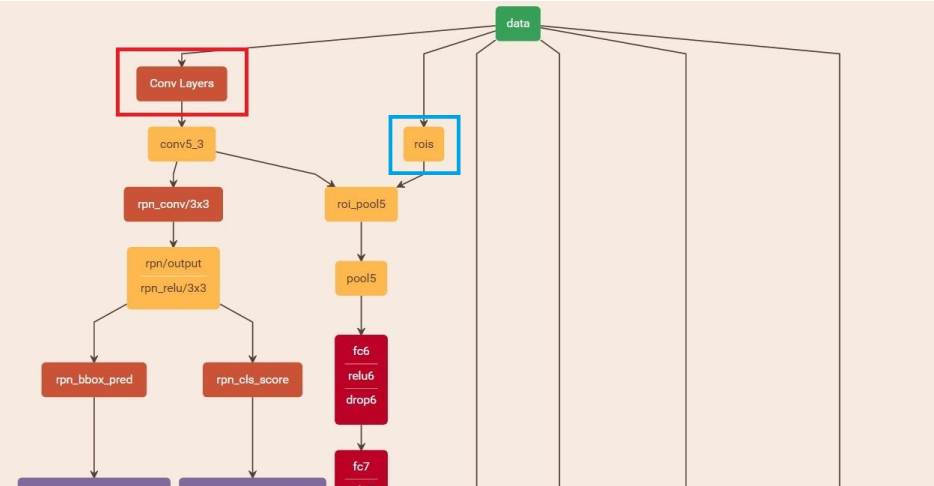
开源商城系统 商城系统源码

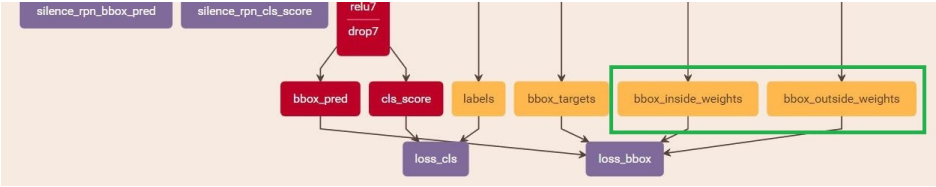
工作流管理系统 数据可视化分析

仓库管理系统 开源网店系统

广告

现在，这样就可以训练最后的识别softmax与最终的bounding regression了，如下图。






这就是对整个代码的理解。

参考：
数据准备部分，参考了faster-rcnn 之训练数据是如何准备的：imdb和roidb的产生 (<http://blog.csdn.net/sloanqin/article/details/51537713>)；
faster-rcnn网络部分，参考了CNN目标检测（一） (<http://blog.csdn.net/zy1034092330/article/details/62044941>)。

在CNN目标检测（一） (<http://blog.csdn.net/zy1034092330/article/details/62044941>)这篇博文中，根据我自己的理解，可能的问题在于：
bbox_outside_weights指的并不是 λ ，而是 $1/N_{reg}$ 。
这篇博文讲的是Alternative training的方法，经过对比，与Approximate joint training的区别在于，proposal产生层不再单独产生loss，只对训练anchors产生的loss进行反向传播。

版权声明：本文为博主原创文章，未经博主允许不得转载。




http://my.csdn.net/weixin_35068028


相关文章推荐

[训练测试过程记录]Text-Detection-with-FRCN (<http://blog.csdn.net/u013250416/article/det...>)

写在前面：github上面的Text-Detection-with-FRCN项目是基于py-faster-rcnn项目在场景文字识别领域的扩展。和py-faster-rcnn相比，该项目的主要改动为...

 [u013250416](http://blog.csdn.net/u013250416) (<http://blog.csdn.net/u013250416>)

2017年11月06日 20:09

 59



Saliency Detection with Multi-Scale Superpixels对应的源码 (<http://downl...>)

<http://download.csdn.net/detail/u013250416/10611111>

2017年08月23日 21:07

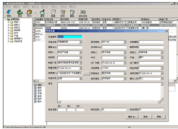
88KB


[下载](#)

内容举报

 TOP

返回顶部





设备管理软件

捕鱼游戏源码

- 望京soho 穷人贷款 希腊房价
- 开源商城系统 商城系统源码
- workflows管理系统 数据可视化分析
- 仓库管理系统 开源网店系统

广告



内容举报

 TOP

返回顶部

【程序员之路】我是前端工程师，怎么了？



今天我30岁了，在此之际，回想我的程序生涯之路，十分感慨，谈谈我作为程序员的选择之路..

(http://www.baidu.com/cb.php?c=lgF_pyfqHmknj0dP1f0lZ0qnfK9ujYzP1nYPH0k0Aw-5Hc3rHnYnHb0TAq15HfLPWRznjb0T1Ylm1whrj7hny7bP1NhPymz0AwY5HDdnHcznjcsnHb0lgF_5y9YIZ0lQzq-uZR8mLPbUB48ugfElAqspynElvNBnHqdlAdxTvqdThP-5yF_UvTkn0KzujYk0AFV5H00TZcq0KdpyfqHRLPjnvnfKEpyfqHc4rj6kP0KWpyfqP1cvrHnz0AqLUWYs0ZK45HcsP6K1

Java 源码 ——顺序存取文件的创建及写入 (Writing data to a sequential text file with class...

代码如下： // Fig. 15.3: CreateTextFile.java // Writing data to a sequential text file with class Format...

hpdizu80100 (http://blog.csdn.net/hpdizu80100) 2017年04月30日 01:57 179

TLD(Tracking-Learning-Detection)算法学习与源码解析（四）之LKTracker源码分析 (http://b..

本序列文章的目的是总结一下这段时间所学到的，主要分为以下几部分，本章是第四部分。 1 算法概述 2 runtd.cpp源码解析 3 tld.cpp源码解析 4 LKTracker(重点) ...

linger2012liu (http://blog.csdn.net/linger2012liu) 2014年03月05日 17:59 2589

Hadoop源码分析-Text (http://blog.csdn.net/judyge/article/details/45694615)

Text是Hadoop中的一个Writable类，定义了Hadoop中的其中的数据类型以及操作。 ...

judyge (http://blog.csdn.net/judyge) 2015年05月13日 16:16 312

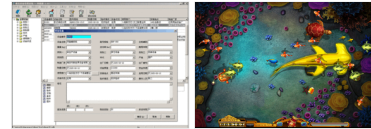


程序员跨越式成长指南

完成第一次跨越，你会成为具有一技之长的开发者，月薪可能翻上几番；完成第二次跨越，你将成为拥有局部优势或行业优势的专业人士，获得个人内在价值的有效提升和外在收入的大幅跃迁.....

(http://www.baidu.com/cb.php?c=lgF_pyfqHmknjfrjD0lZ0qnfK9ujYzP1f4PjnY0Aw-5Hc4nj6vPjm0TAq15Hf4rjn1n1b0T1YvnWmvny79nWbdPWK9nWTd0AwY5HDdnHcznjcsnHb0lgF_5y9YIZ0lQzqMpgwBUvqoQhP8QvIGIAPCmgfEmvq_lyd8Q1R4uWc4uHf3uAckPWN9PhcsmW9huWqdIAdxTv5HDknWFBmhkEusKzujYk0AFV5H00TZcq0KdpyfqHRLPjnvnfKEpyfqHnsnj0YnsKWpyfqP1cvrHnz0AqLUWYs0ZK45HcsP6KWThnqPW0Yrjb)

TLD (Tracking-Learning-Detection) 学习与源码理解之（六） (http://blog.csdn.net/zouxy...



设备管理软件 捕鱼游戏源码

望京soho 穷人贷款 希腊房价
开源商城系统 商城系统源码
工作流管理系统 数据可视化分析
仓库管理系统 开源网店系统




内容举报




TOP

下面是自己在看论文和这些大牛的分析过程中，对代码进行了一些理解，但是由于自己接触图像处理和机器视觉没多久，另外由于自己编程能力比较弱，所以分析过程可能会有不少的错误，希望各位不吝指正。而且，因为编程很...

 zouxy09 (<http://blog.csdn.net/zouxy09>) 2012年08月21日 20:35 26375


TLD (Tracking-Learning-Detection) 学习与源码理解之 (七) (<http://blog.csdn.net/zouxy09/article/details/7893011>)

下面是自己在看论文和这些大牛的分析过程中，对代码进行了一些理解，但是由于自己接触图像处理和机器视觉没多久，另外由于自己编程能力比较弱，所以分析过程可能会有不少的错误，希望各位不吝指正。而且，因为编程很...

 zouxy09 (<http://blog.csdn.net/zouxy09>) 2012年08月21日 20:37 16862


TLD (Tracking-Learning-Detection) 学习与源码理解之 (<http://blog.csdn.net/wotawomen/>)

文章来自：<http://blog.csdn.net/zouxy09/article/details/7893011> TLD(Tracking-Learning-Detection)

 wotawomen (<http://blog.csdn.net/wotawomen>) 2014年08月04日 14:06 407


TLD (Tracking-Learning-Detection) 学习与源码理解之 (四) (<http://blog.csdn.net/zouxy09/article/details/7893081>)

下面是自己在看论文和这些大牛的分析过程中，对代码进行了一些理解，但是由于自己接触图像处理和机器视觉没多久，另外由于自己编程能力比较弱，所以分析过程可能会有不少的错误，希望各位不吝指正。而且，因为编程很...

 zouxy09 (<http://blog.csdn.net/zouxy09>) 2012年08月21日 20:25 30509


TLD (Tracking-Learning-Detection) 学习与源码理解之 (四) . (<http://blog.csdn.net/weixiaomm/>)

下面是自己在看论文和这些大牛的分析过程中，对代码进行了一些理解，但是由于自己接触图像处理和机器视觉没多久，另外由于自己编程能力比较弱，所以分析过程可能会有不少的错误，希望各位不吝指正。而且，因为编程很...

 weixiaomm (<http://blog.csdn.net/weixiaomm>) 2013年01月09日 21:43 512

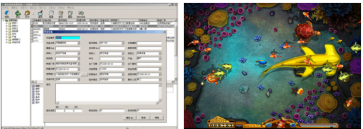
Sparselet Models for Efficient Multiclass Object Detection对应源码配置 (<http://blog.csdn.net/quincintial/>)

论文Sparselet Models for Efficient Multiclass Object Detection的源码环境配置 声明：论文对应源码的运行环境为Linux，本文的配置是在Lin...

 Quincintial (<http://blog.csdn.net/Quincintial>) 2015年04月16日 13:24 1083

TLD (Tracking-Learning-Detection) 学习与源码理解之 (六) (<http://blog.csdn.net/gxiaobai/>)

本文转自<http://blog.csdn.net/zouxy09/article/details/7893081> 下面是自己在看论文和这些大牛的分析过程中，对代码进行了一些理解，但是由于自己接触图...



设备管理软件

捕鱼游戏源码

望京soho 穷人贷款 希腊房价

开源商城系统 商城系统源码

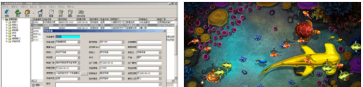
workflow管理系统 数据可视化分析

仓库管理系统 开源网店系统



广告

 内容举报

 TOP
返回顶部







 gxiaob (<http://blog.csdn.net/gxiaob>) 2012年11月22日 17:31  2395

TLD (Tracking-Learning-Detection) 学习与源码理解之 (一) (<http://blog.csdn.net/zouxy09>...

TLD (Tracking-Learning-Detection) 学习与源码理解之 (一) zouxy09@qq.com TLD(Tracking-Learning-Detection)是英国萨里大学的一...

 zouxy09 (<http://blog.csdn.net/zouxy09>) 2012年08月21日 20:15  99659

Contour Detection and Hierarchical Image Segmentation 源码编译运行 (<http://blog.csdn...>

找了很久，终于找到了这一篇好的图像分割方法，还有一篇2013年CVPR 的Sketch Tokens，根据它提供的召回率曲线图的对比，能看出它们的分割效果基本一样，但没有具体对这两篇的比较。下面是我对...

 BlitzSkies (<http://blog.csdn.net/BlitzSkies>) 2014年02月22日 16:06  4674


TLD (Tracking-Learning-Detection) 学习与源码理解之 (六) . ([http://blog.csdn.net/weixi...](http://blog.csdn.net/weixiaomm)

转自：<http://blog.csdn.net/zouxy09/article/details/7893081> 下面是自己在看论文和这些大牛的分析过程中，对代码进行了一些理解，但是由于自己接触图像处...

 weixiaomm (<http://blog.csdn.net/weixiaomm>) 2013年01月09日 21:54  685



TLD (Tracking-Learning-Detection) 学习与源码理解之 (五) ([http://blog.csdn.net/huang...](http://blog.csdn.net/huangli19870217)

下面是自己在看论文和这些大牛的分析过程中，对代码进行了一些理解，但是由于自己接触图像处理和机器视觉没多久，另外由于自己编程能力比较弱，所以分析过程可能会有不少的错误，希望各位不吝指正。而且，因为编程很...

 huangli19870217 (<http://blog.csdn.net/huangli19870217>) 2013年04月25日 15:58  604

TLD (Tracking-Learning-Detection) 学习与源码理解之 (跟踪器) (<http://blog.csdn.net/m...>

下面是自己在看论文和这些大牛的分析过程中，对代码进行了一些理解，但是由于自己接触图像处理和机器视觉没多久，另外由于自己编程能力比较弱，所以分析过程可能会有不少的错误，希望各位不吝指正。而且，因为编程很...

 mydear_11000 (http://blog.csdn.net/mydear_11000) 2015年08月24日 10:02  779



TLD (Tracking-Learning-Detection) 学习与源码理解之 (分类器) (<http://blog.csdn.net/m...>

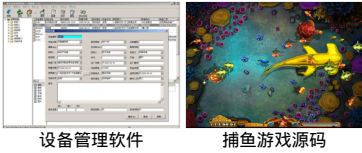
下面是自己在看论文和这些大牛的分析过程中，对代码进行了一些理解，但是由于自己接触图像处理和机器视觉没多久，另外



设备管理软件 捕鱼游戏源码
望京soho 穷人贷款 希腊房价
开源商城系统 商城系统源码
工作流管理系统 数据可视化分析
仓库管理系统 开源网店系统



 内容举报
 返回顶部



设备管理软件 捕鱼游戏源码
望京soho 穷人贷款 希腊房价

0



由于自己编程能力比较弱，所以分析过程可能会有不少的错误，希望各位不吝指正。而且，因为编程很...



mydear_11000 (http://blog.csdn.net/mydear_11000) 2015年08月24日 10:04 833

OpenStack源码中的with分析 (<http://blog.csdn.net/epugv/article/details/28293881>)

作为个人学习笔记分享，有任何问题欢迎交流! 在OpenStack的源码中经常会看到一个语法：with，如下面的函数实现的是释放已经分配给实例的网络端口，其中就用到了with。 def ...



epugv (<http://blog.csdn.net/epugv>) 2014年06月03日 22:00 2163

TLD (Tracking-Learning-Detection) 学习与源码理解之 (三) (<http://blog.csdn.net/zouxy09>...

TLD (Tracking-Learning-Detection) 学习与源码理解之 (三) zouxy09@qq.com 下面是自己在看论文和这些大牛的分析过程中，对代码进行了一些理解，但是由于自己接触图...



zouxy09 (<http://blog.csdn.net/zouxy09>) 2012年08月21日 20:18 43499

开源商城系统 商城系统源码
 workflows管理系统 数据可视化分析
 仓库管理系统 开源网店系统



内容举报



返回顶部