Fast R-CNN

Object detection with Caffe

Ross Girshick

Microsoft Research

arXiv   code

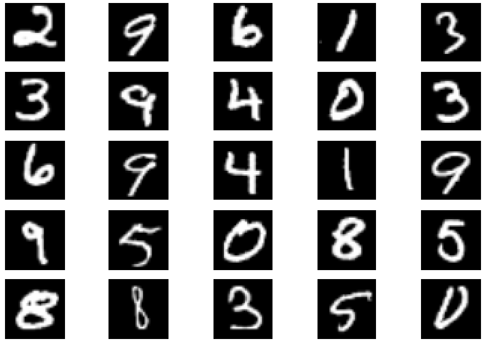
Latest roasts

Goals for this section

- Super quick intro to object detection
- Show one way to tackle obj. det. with ConvNets
- Highlight some more sophisticated uses of Caffe
  - Python layers
  - Multi-task training with multiple losses
  - Batch sizes that change dynamically during Net::Forward()
- Pointers to open source code so you can explore, try, and understand!

Image classification (mostly what you’ve seen)

- K classes
- Task: Assign the correct class label to the whole image

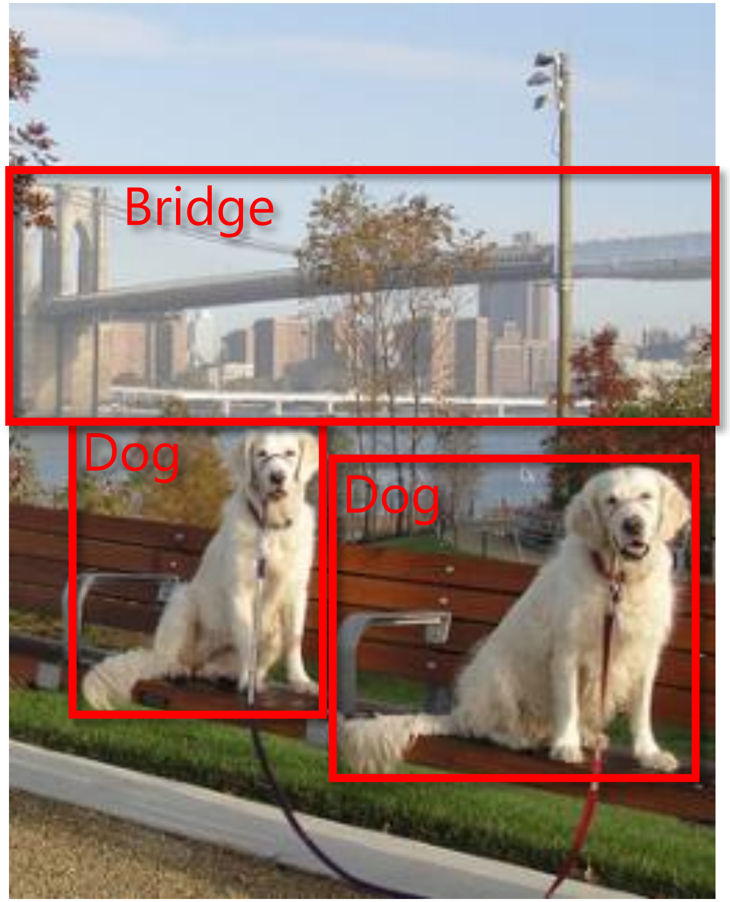


Digit classification (MNIST)

Object recognition (Caltech-101, ImageNet, etc.)

Classification vs. Detection

Easyish, these days



Still quite a lot harder

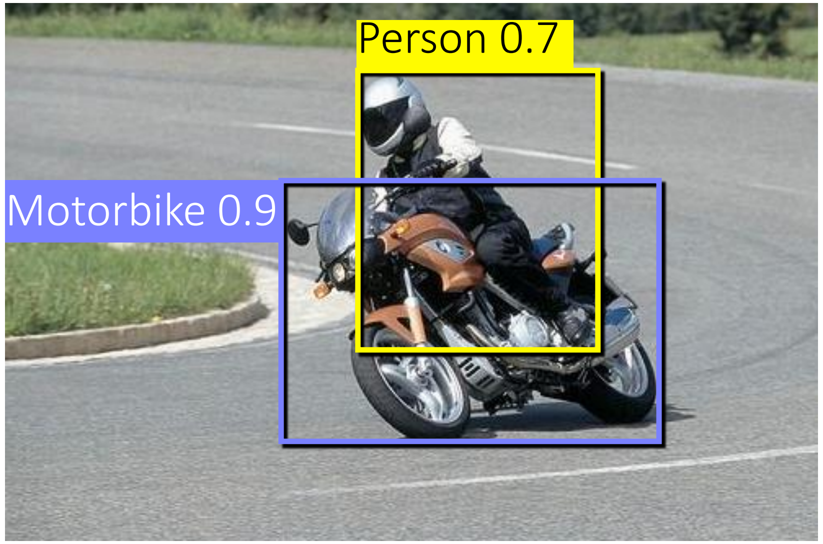
Problem formulation

The Visual World ≈ K object classes

{airplane, bird, motorbike, person, sofa, bg}

YODA:

*Y*et *a*nother *O*bject *D*etection *A*lgorithm

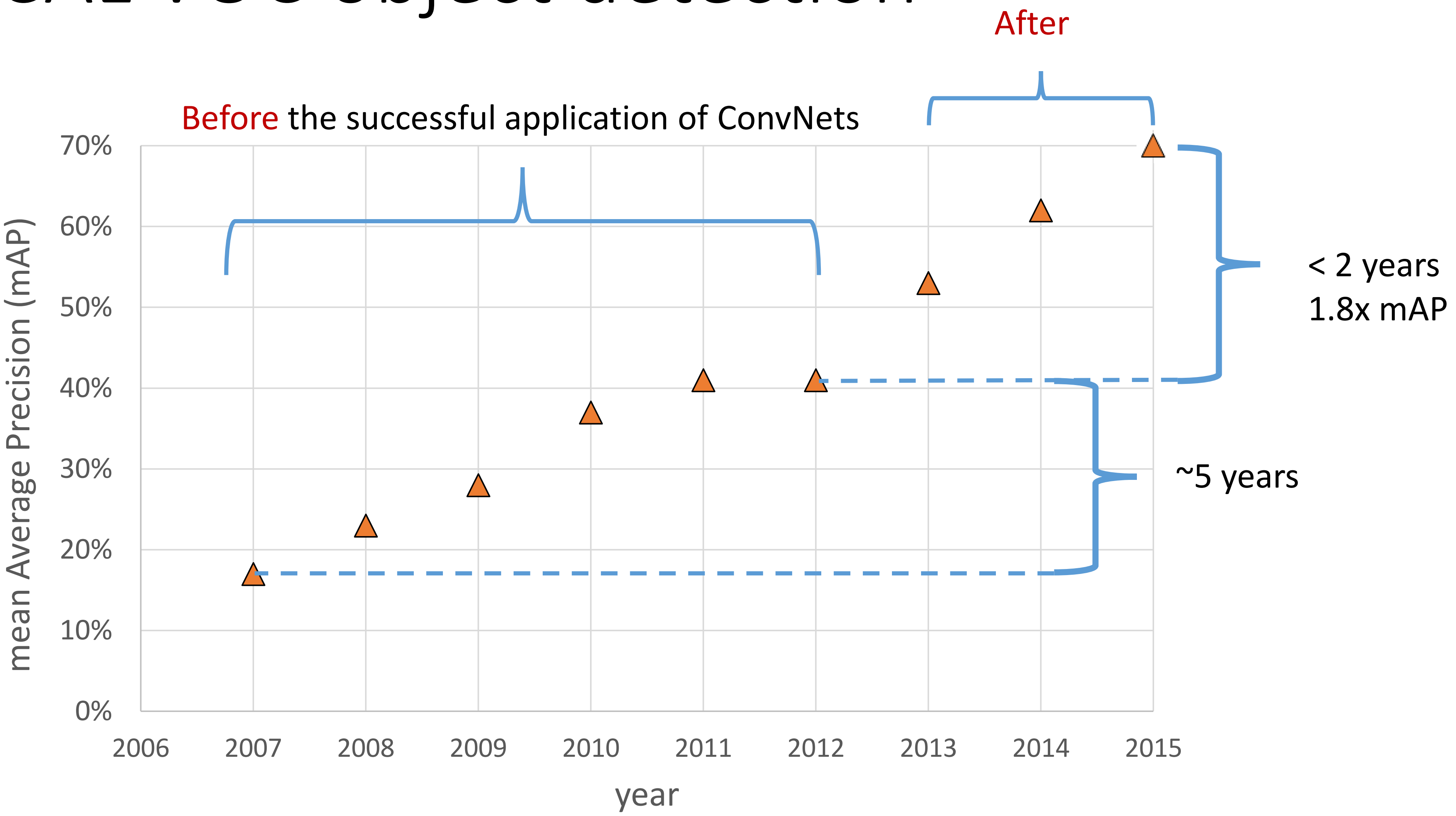


Input

Desired output

*Actual results may vary

PASCAL VOC object detection



Precision: higher is better

Fast R-CNN (Region-based Convolutional Networks)

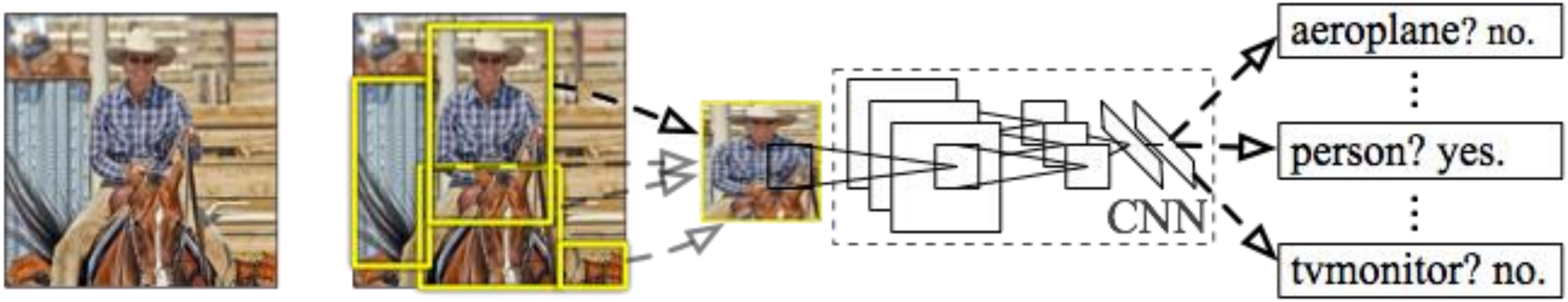
A fast object detector implemented with Caffe

- Caffe fork on GitHub that adds two new layers (ROIPoolingLayer and SmoothL1LossLayer)
- Python (using pycaffe) / more advanced Caffe usage
- A type of Region-based Convolutional Network (R-CNN)

Let's see how it works!

Quick background

Region-based Convolution Networks (R-CNNs)



Input image

Extract region proposals (~2k / image)
e.g., selective search
[van de Sande, Uijlings et al.]

Compute CNN features on regions

Classify and refine regions

[Girshick et al. CVPR'14]

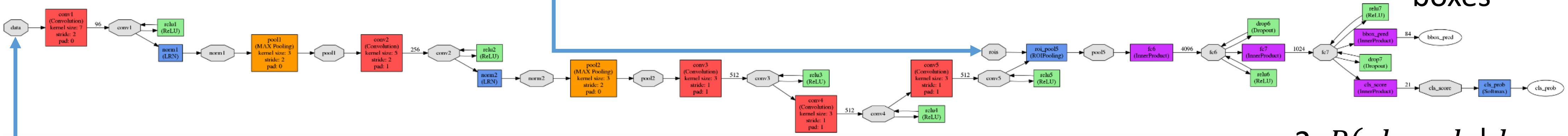
Fast R-CNN (test-time detection)

Object box proposals (N)
e.g., selective search

A Fast R-CNN network
(VGG_CNN_M_1024)

Two output types:

1. NK regressed object boxes



2. $P(cls = k \mid box = n, image)$ for each NK boxes

Net::Forward() takes 60 to 330ms

Image

Given an image and object proposals,
detection happens with a single call to the Net::Forward()

Fast R-CNN (test-time detection)

Object box proposals (*N*)

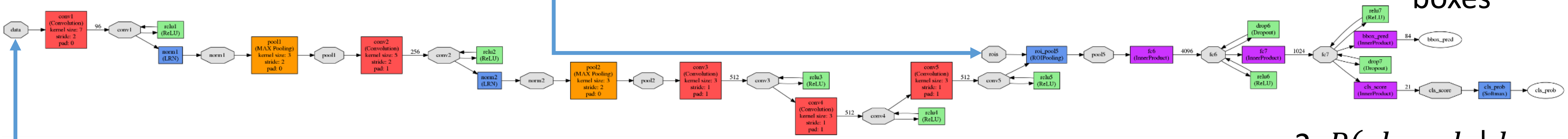
e.g., selective search

A Fast R-CNN network

(VGG_CNN_M_1024)

Two output types:

1. *NK* regressed object boxes



2. $P(cls = k \mid box = n, image)$ for each *NK* boxes

Minimal post-processing:

- Non-maximum suppression (NMS)

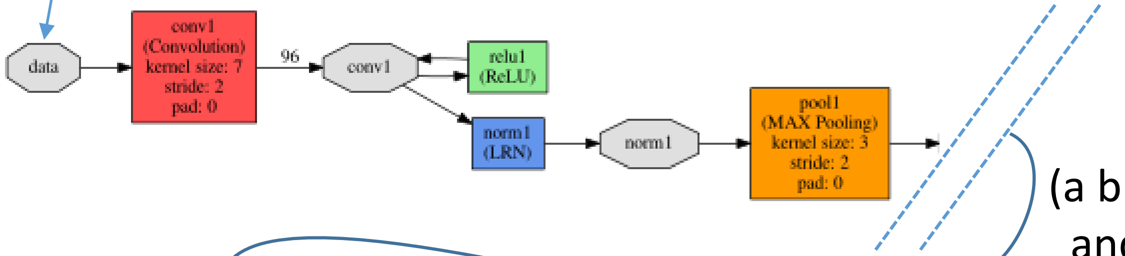
Image

Object proposals comes from:

- Selective Search (2s / image) [van de Sande/Uijlings et al.]
- EdgeBoxes (0.2s / image) [Zitnick & Dollar]
- MCG (30s / image) [Arbelaez et al.]
- Etc.

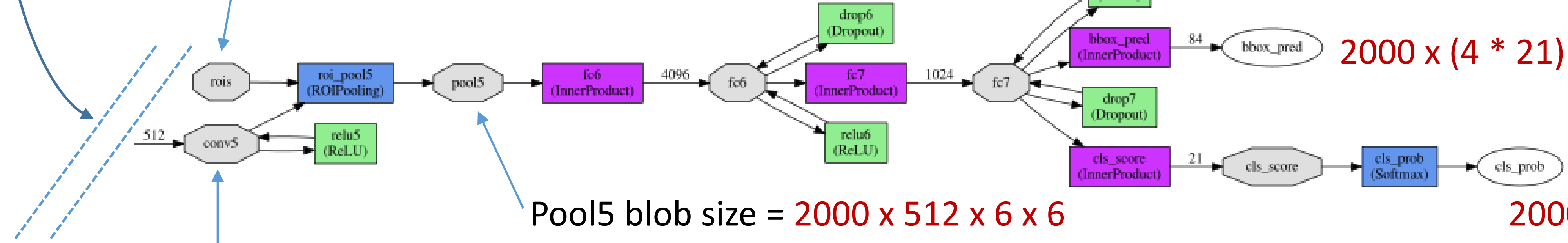
Zooming into the net

image comes in here, blob size = *S* x 3 x *H* x *W* (e.g., *S* = 1 or 5, *H* = 600, *W* = 1000)



(a bunch of conv layers
and whatnot)

2000 image regions come
in here, blob size = 2000 x 5



2000 x (4 * 21)

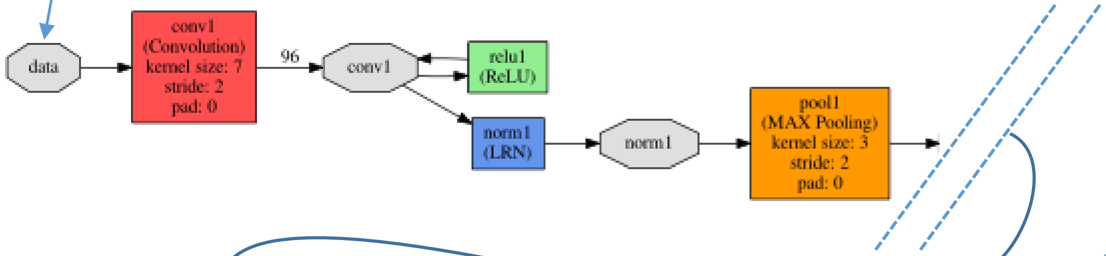
Pool5 blob size = 2000 x 512 x 6 x 6

2000 x 21

conv5 feature map blob size = *S* x 512 x H/16 x W/16

Zooming into the net

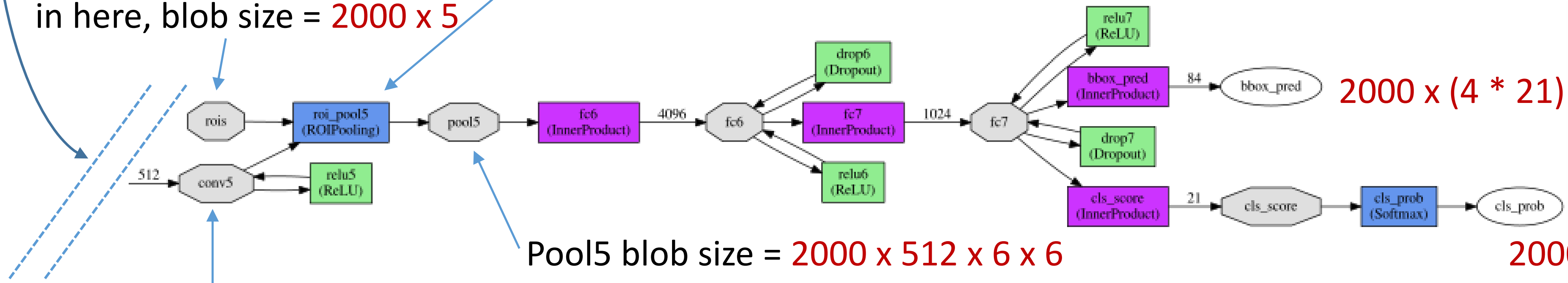
image comes in here, blob size = S x 3 x H x W (e.g., S = 1 or 5, H = 600, W = 1000)



RoI Pooling Layer:

- *adaptive max* pooling layer
- *dynamically* expands batch from S to R (e.g., 2000)

2000 image regions come in here, blob size = 2000 x 5



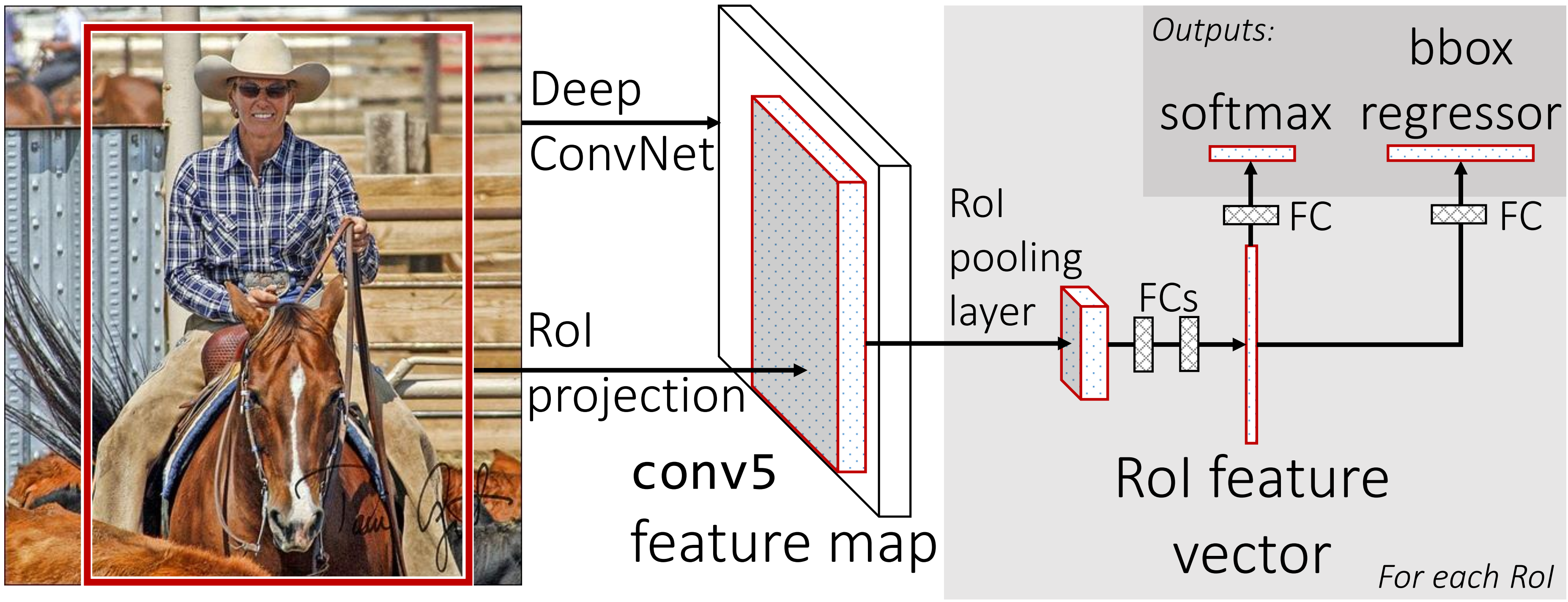
2000 x (4 * 21)

Pool5 blob size = 2000 x 512 x 6 x 6

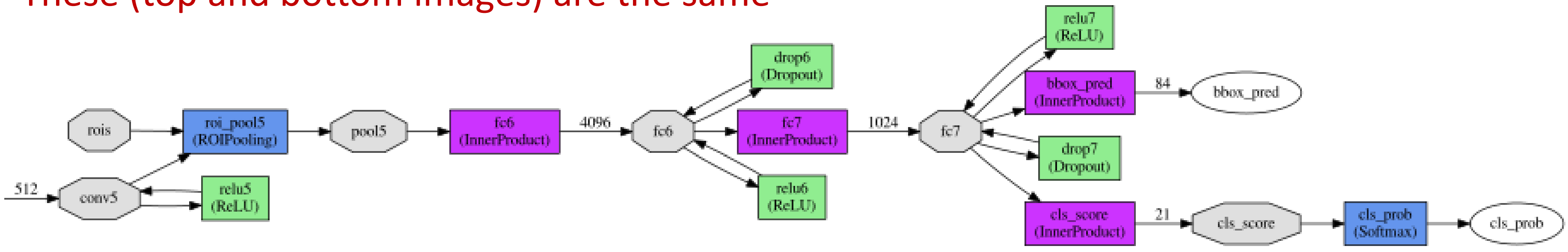
2000 x 21

conv5 feature map blob size = S x 512 x H/16 x W/16

Another view of the same thing



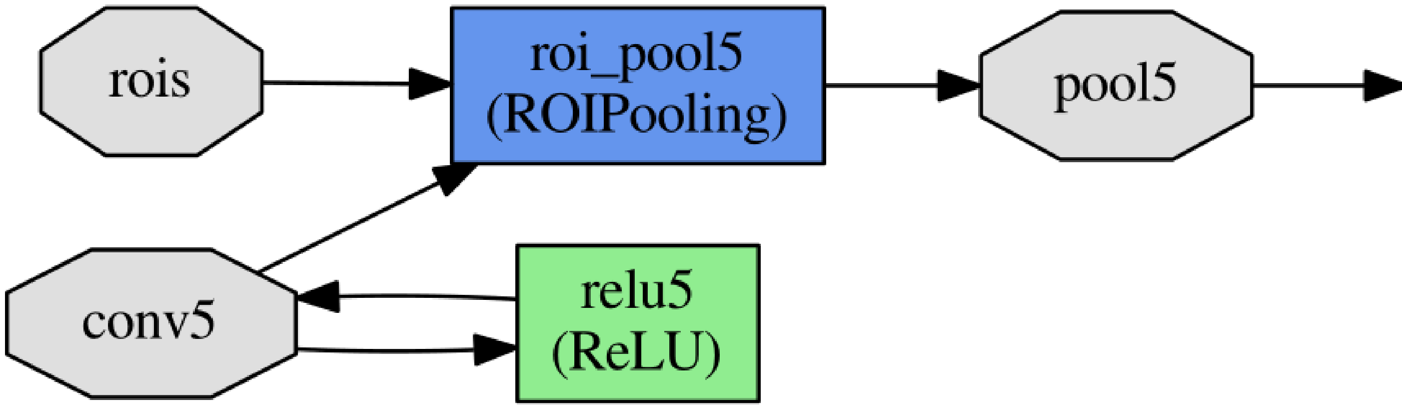
These (top and bottom images) are the same



RoI Pooling Layer

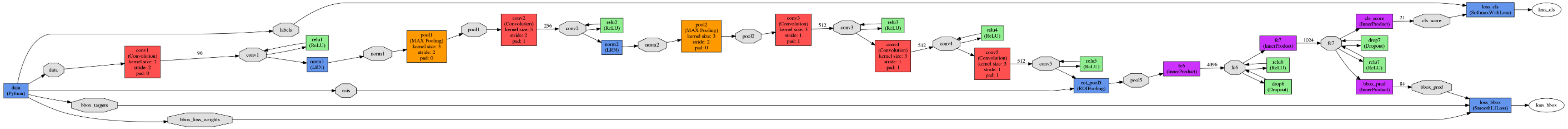
- Special case of SPPnet's SPP layer [He et al. ECCV'14]
- Two inputs ("bottoms")
  - Conv feature map: *S* x 512 x *H* x *W*
  - Regions of Interest: *R* x 5
    - 5 comes from [*r, x1, y1, x2, y2*], where *r* in [0, *R* – 1] specifies an image batch index

```
layer {
  name: "roi_pool5"
  type: "ROIPooling"
  bottom: "conv5"
  bottom: "rois"
  top: "pool5"
  roi_pooling_param {
    pooled_w: 6
    pooled_h: 6
    spatial_scale: 0.0625 # 1/16
  }
}
```



The train-time net

Single fine-tuning operation all in Caffe



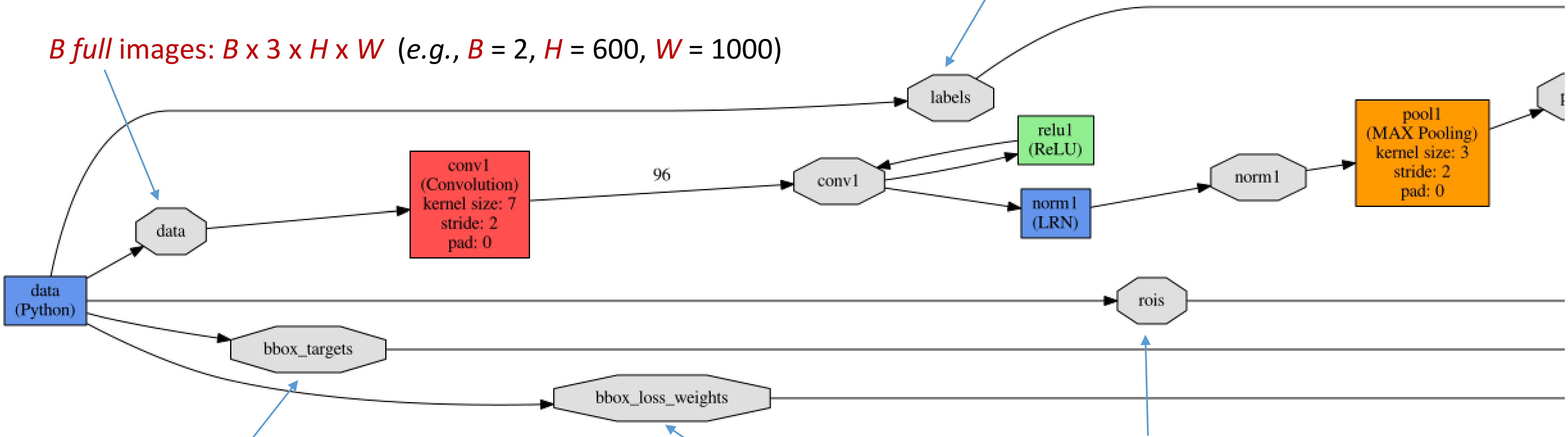
Even more boxes and arrows

Let's look at them

The train-time net (inputs)

Class labels: 128 x 21

B full images: *B* x 3 x *H* x *W* (*e.g.*, *B* = 2, *H* = 600, *W* = 1000)

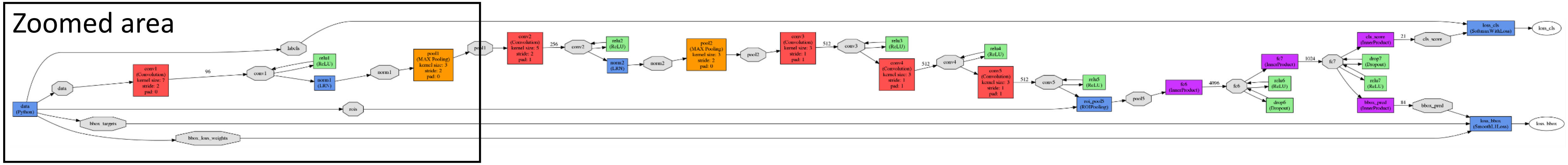


Bounding-box
regression targets: 128 x 84

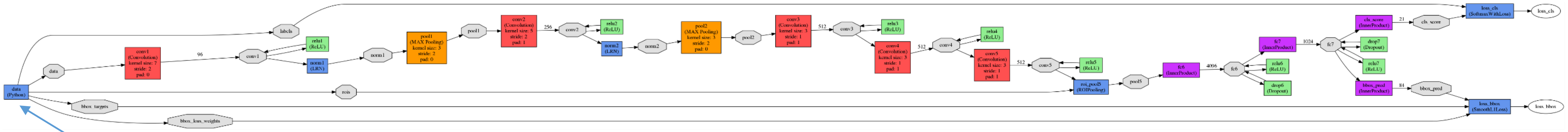
Bounding-box
regression loss weights: 128 x 84

RoIs: 128 x 5
(75% background)

Zoomed area



The train-time net (exotic data layers)



Custom Python data layer

- Samples 2 images
- From each sampled image, takes 64 RoIs
- Input batch is initially 2 elements
- Gets expanded by the RoI Pooling Layer to 128 elements
- Outputs 5 "tops"
  - `data [images]`
  - `rois [regions of interest]`
  - `labels [class labels for the rois]`
  - `bbox_targets [box regression targets]`
  - `bbox_loss_weights […details…]`

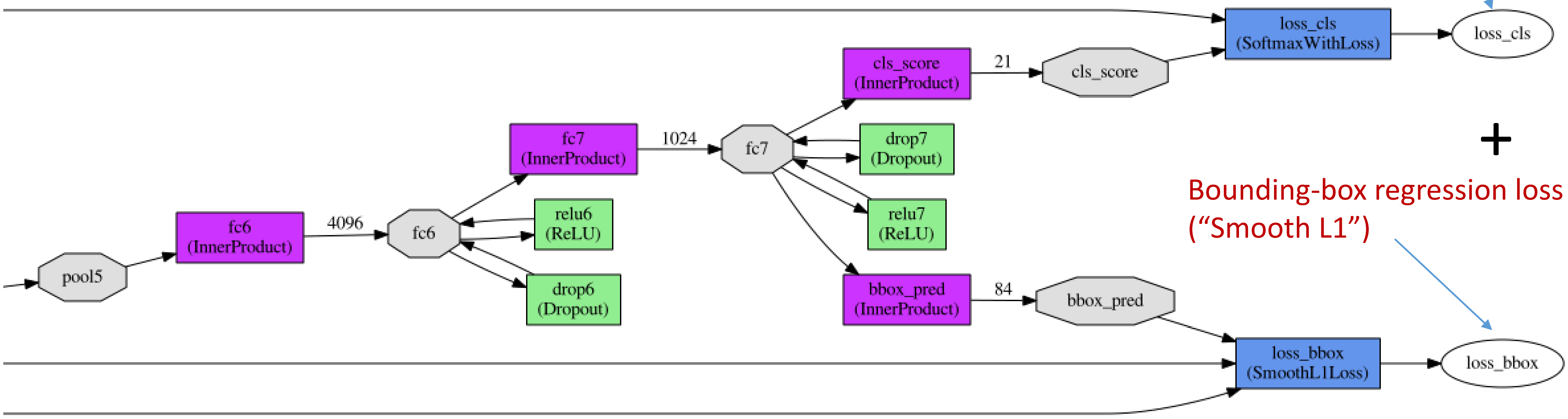
```
layer {
  name: 'data'
  type: 'Python'
  top: 'data'
  top: 'rois'
  top: 'labels'
  top: 'bbox_targets'
  top: 'bbox_loss_weights'
  python_param {
    module: 'roi_data_layer.layer'
    layer: 'RoIDataLayer'
    param_str: "'num_classes': 21"
  }
}
```

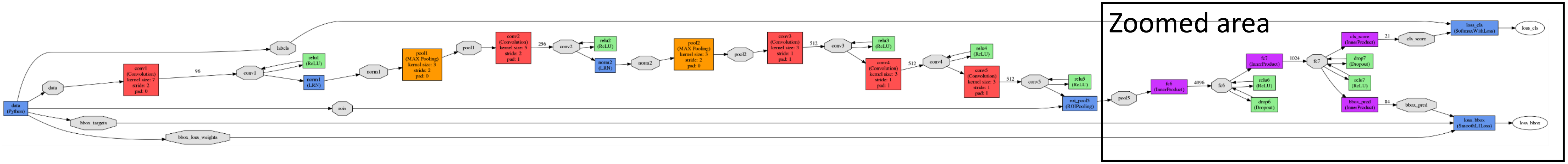
The train-time net (multi-task losses)

Classification loss
(Cross-entropy)

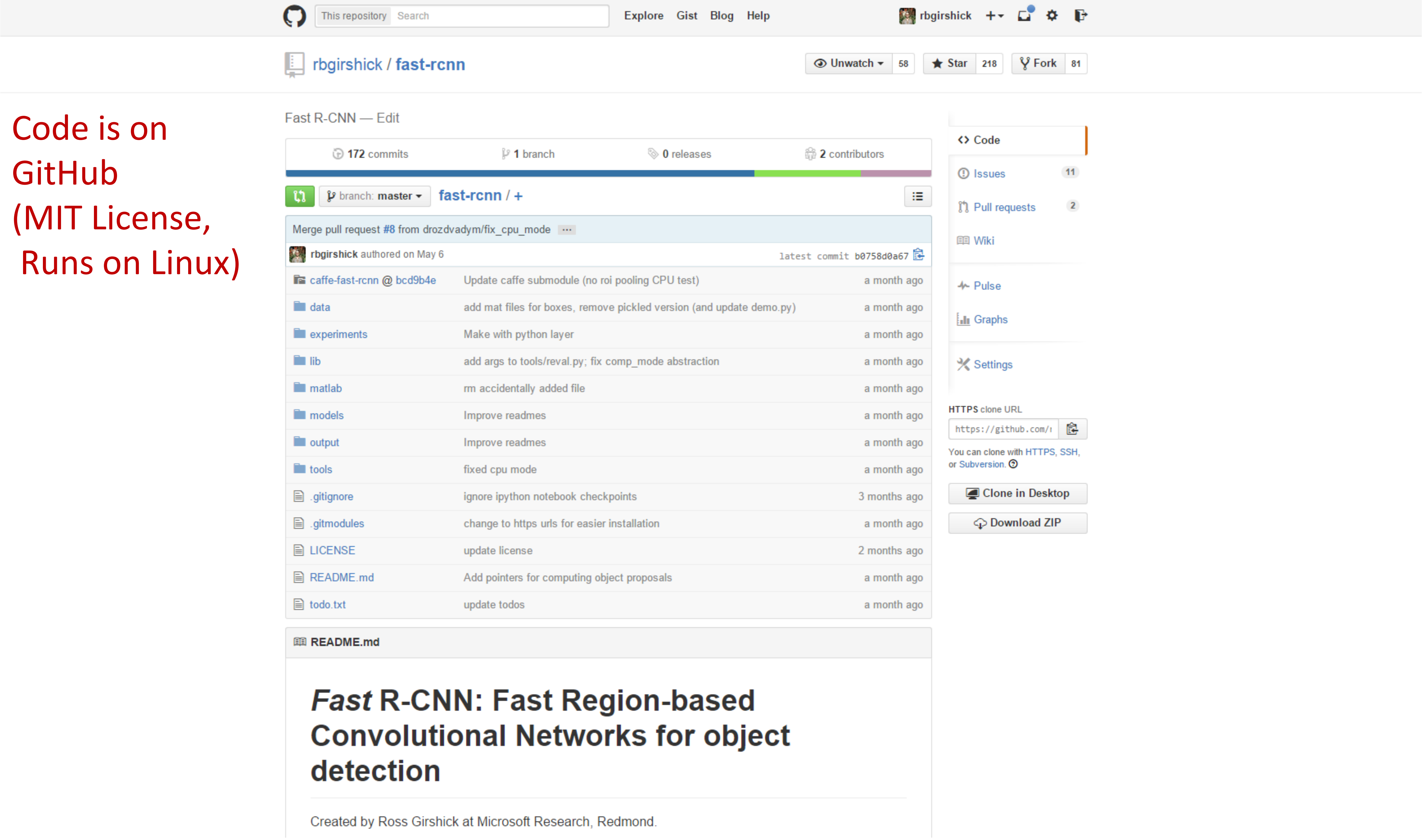
+

Bounding-box regression loss
("Smooth L1")





Code is on GitHub (MIT License, Runs on Linux)



A brief tour of some of the code

Caffe fork → caffe-fast-rcnn

Python modules → lib

Train, test → tools

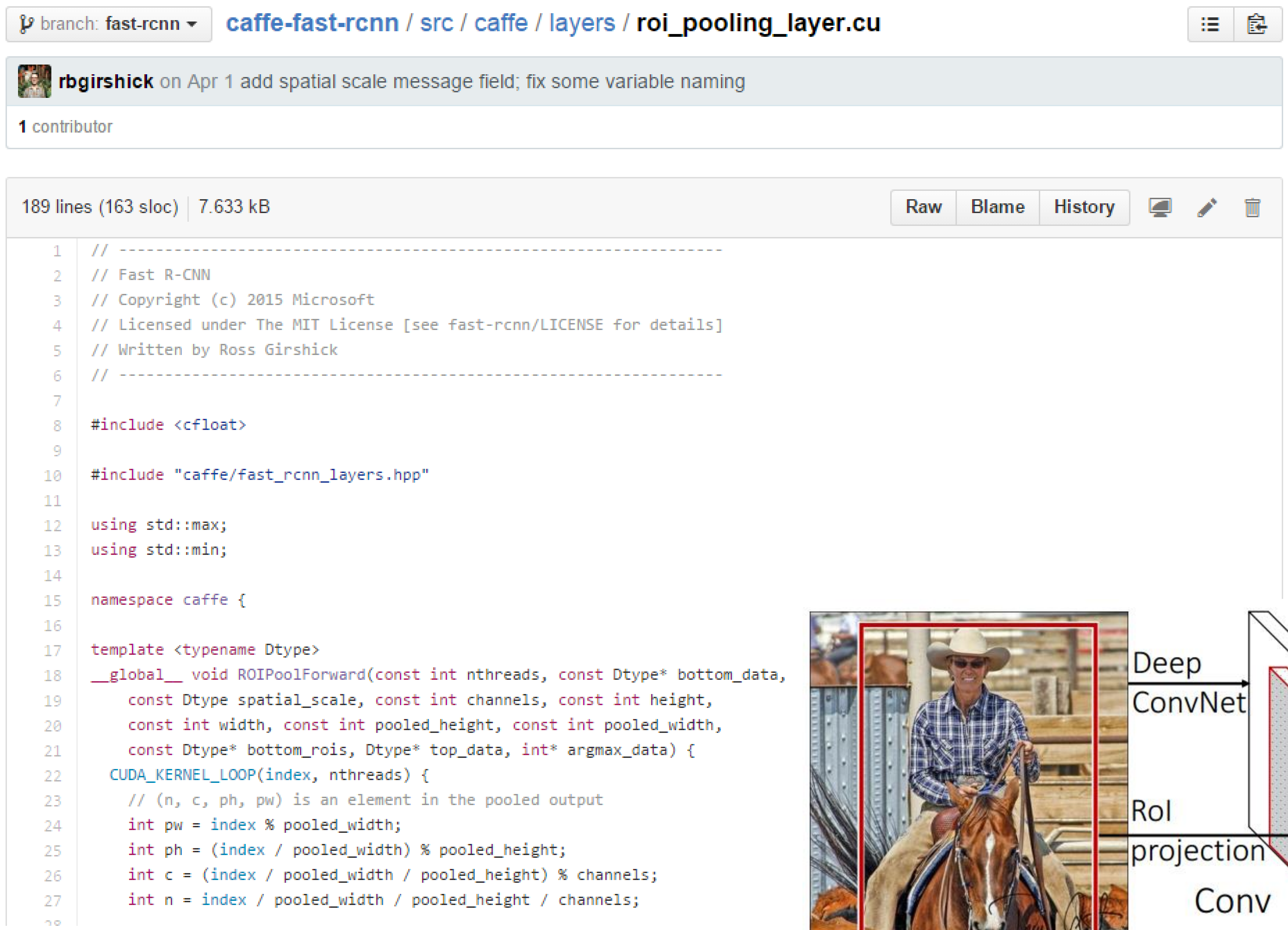
caffe-fast-rcnn @ bcd9b4e	Update caffe submodule (no roi pooling CPU test)	a month ago
data	add mat files for boxes, remove pickled version (and update demo.py)	a month ago
experiments	Make with python layer	a month ago
lib	add args to tools/reval.py; fix comp_mode abstraction	a month ago
matlab	rm accidentally added file	a month ago
models	Improve readmes	a month ago
output	Improve readmes	a month ago
tools	fixed cpu mode	a month ago
.gitignore	ignore ipython notebook checkpoints	3 months ago
.gitmodules	change to https urls for easier installation	a month ago
LICENSE	update license	2 months ago
README.md	Add pointers for computing object proposals	a month ago
todo.txt	update todos	a month ago

A brief tour of some of the code (Caffe bits)

Caffe fork	caffe-fast-rcnn @ bcd9b4e	Update caffe submodule (no roi pooling CPU test)	a month ago
	data	add mat files for boxes, remove pickled version (and update demo.py)	a month ago
	experiments	Make with python layer	a month ago
Python modules	lib	add args to tools/reval.py; fix comp_mode abstraction	a month ago
	matlab	rm accidentally added file	a month ago
	models	Improve readmes	a month ago
	output	Improve readmes	a month ago
Train, test	tools	fixed cpu mode	a month ago
	.gitignore	ignore ipython notebook checkpoints	3 months ago
	.gitmodules	change to https urls for easier installation	a month ago
	LICENSE	update license	2 months ago
	README.md	Add pointers for computing object proposals	a month ago
	todo.txt	update todos	a month ago

Region of Interest (RoI) Pooling Layer

Expands a small batch into a big batch


branch: fast-rcnn caffe-fast-rcnn / src / caffe / layers / roi_pooling_layer.cu

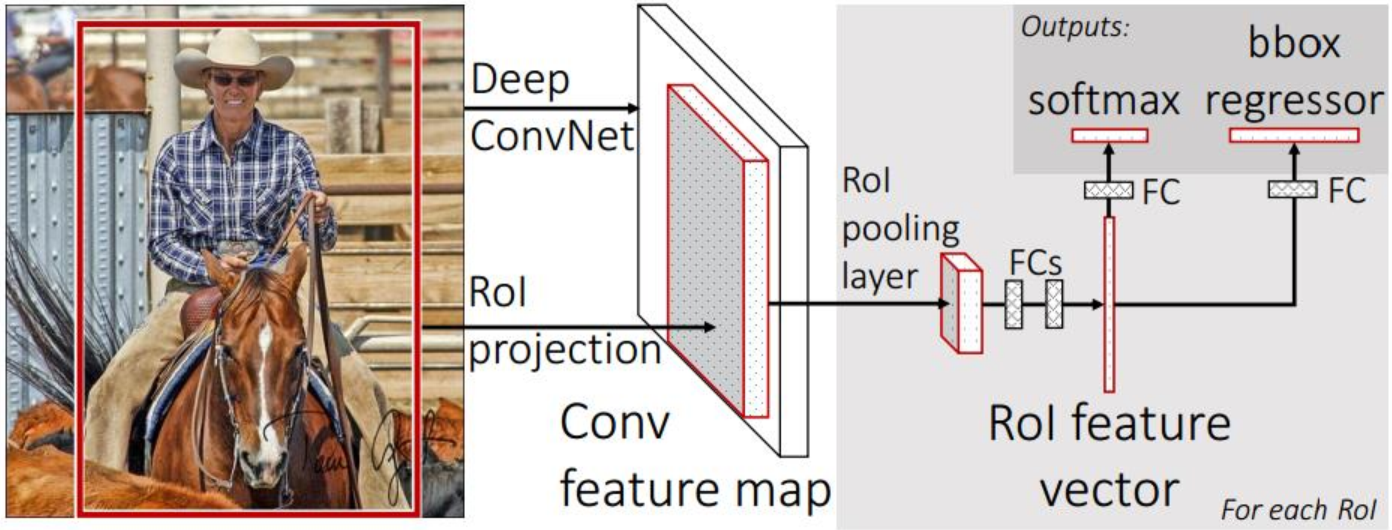
rbgirshick on Apr 1 add spatial scale message field; fix some variable naming

1 contributor

189 lines (163 sloc) | 7.633 kB

Raw Blame History

```
// ------------------------------------------------------------------
// Fast R-CNN
// Copyright (c) 2015 Microsoft
// Licensed under The MIT License [see fast-rcnn/LICENSE for details]
// Written by Ross Girshick
// ------------------------------------------------------------------

#include <cfloat>

#include "caffe/fast_rcnn_layers.hpp"

using std::max;
using std::min;

namespace caffe {

template <typename Dtype>
__global__ void ROIPoolForward(const int nthreads, const Dtype* bottom_data,
    const Dtype spatial_scale, const int channels, const int height,
    const int width, const int pooled_height, const int pooled_width,
    const Dtype* bottom_rois, Dtype* top_data, int* argmax_data) {
  CUDA_KERNEL_LOOP(index, nthreads) {
    // (n, c, ph, pw) is an element in the pooled output
    int pw = index % pooled_width;
    int ph = (index / pooled_width) % pooled_height;
    int c = (index / pooled_width / pooled_height) % channels;
    int n = index / pooled_width / pooled_height / channels;
```



Smooth L1 Loss Layer

Robust to outliers
Optimizer friendly

Per-dimension loss weights

branch: fast-rcnn ▾ caffe-fast-rcnn / src / caffe / layers / **smooth_L1_loss_layer.cu**

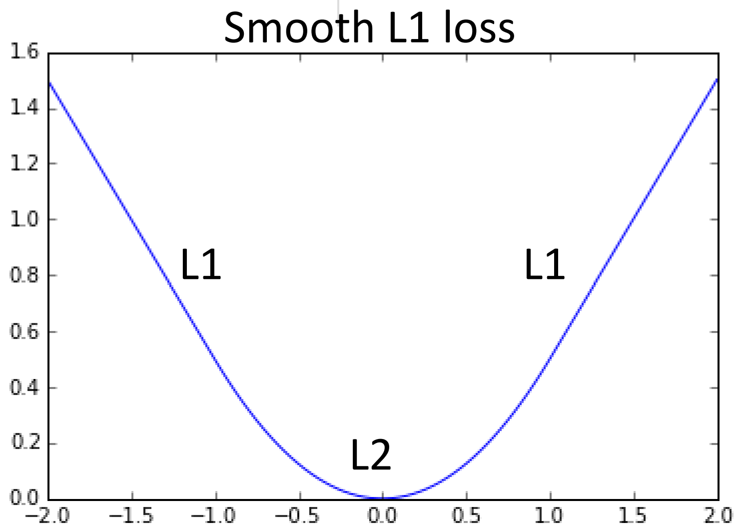
rbgirshick on Mar 30 add license blurb; split out fast r-cnn layers into separate header file

1 contributor

91 lines (81 sloc) | 2.809 kB

Raw | Blame | History

```
// ------------------------------------------------------------------
// Fast R-CNN
// Copyright (c) 2015 Microsoft
// Licensed under The MIT License [see fast-rcnn/LICENSE for details]
// Written by Ross Girshick
// ------------------------------------------------------------------

#include "caffe/fast_rcnn_layers.hpp"

namespace caffe {

template <typename Dtype>
__global__ void SmoothL1Forward(const int n, const Dtype* in, Dtype* out) {
  // f(x) = 0.5 * x^2    if |x| < 1
  //        |x| - 0.5    otherwise
  CUDA_KERNEL_LOOP(index, n) {
    Dtype val = in[index];
    Dtype abs_val = abs(val);
    if (abs_val < 1) {
      out[index] = 0.5 * val * val;
    } else {
      out[index] = abs_val - 0.5;
    }
  }
}

```



A brief tour of some of the code (Python bits)

Caffe fork	caffe-fast-rcnn @ bcd9b4e	Update caffe submodule (no roi pooling CPU test)	a month ago
	data	add mat files for boxes, remove pickled version (and update demo.py)	a month ago
	experiments	Make with python layer	a month ago
Python modules	lib	add args to tools/reval.py; fix comp_mode abstraction	a month ago
	matlab	rm accidentally added file	a month ago
	models	Improve readmes	a month ago
	output	Improve readmes	a month ago
Train, test	tools	fixed cpu mode	a month ago
	.gitignore	ignore ipython notebook checkpoints	3 months ago
	.gitmodules	change to https urls for easier installation	a month ago
	LICENSE	update license	2 months ago
	README.md	Add pointers for computing object proposals	a month ago
	todo.txt	update todos	a month ago

Python data layer for Fast R-CNN

Reshapes blobs on-the-fly

branch: master | fast-rcnn / lib / roi_data_layer / layer.py

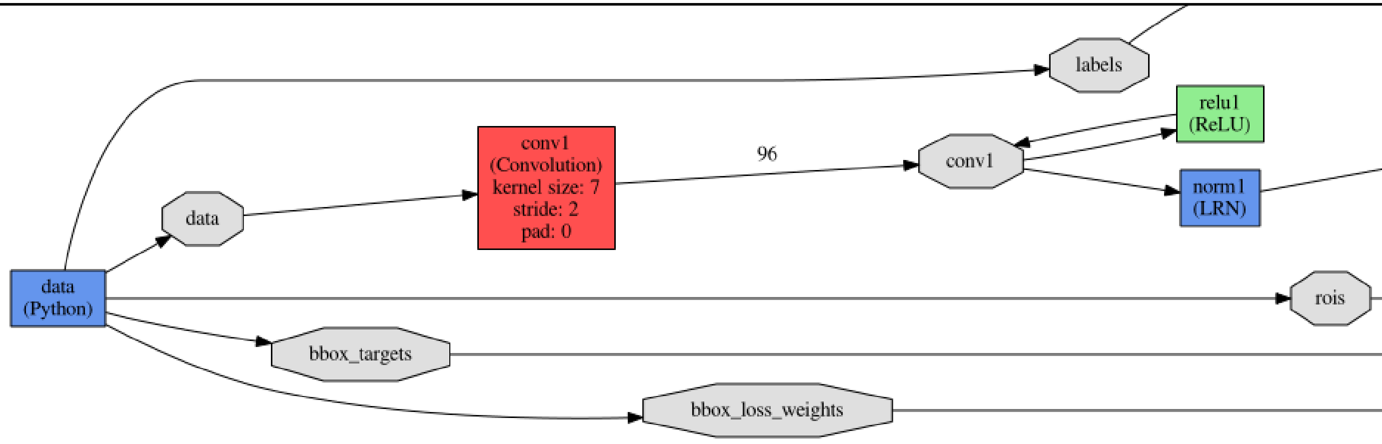
rbgirshick on Apr 29 cleanup script files

1 contributor

161 lines (131 sloc) | 5.93 kB | Raw | Blame | History

```python
# --------------------------------------------------------
# Fast R-CNN
# Copyright (c) 2015 Microsoft
# Licensed under The MIT License [see LICENSE for detail
# Written by Ross Girshick
# --------------------------------------------------------

"""The data layer used during training to train a Fast R

RoIDataLayer implements a Caffe Python layer.
"""

import caffe
from fast_rcnn.config import cfg
from roi_data_layer.minibatch import get_minibatch
import numpy as np
import yaml
from multiprocessing import Process, Queue

class RoIDataLayer(caffe.Layer):
    """Fast R-CNN data layer used for training."""

    def _shuffle_roidb_inds(self):
        """Randomly permute the training roidb."""
        self._perm = np.random.permutation(np.arange(len(self._roidb)))
        self._cur = 0

    def _get_next_minibatch_inds(self):
        """Return the roidb indices for the next minibatch."""
        if self._cur + cfg.TRAIN.IMS_PER_BATCH >= len(self._roidb):
```

```python
    def forward(self, bottom, top):
        """Get blobs and copy them into this layer's top blob vector."""
        blobs = self._get_next_minibatch()

        for blob_name, blob in blobs.iteritems():
            top_ind = self._name_to_top_map[blob_name]
            # Reshape net's input blobs
            top[top_ind].reshape(*(blob.shape))
            # Copy data into net's input blobs
            top[top_ind].data[...] = blob.astype(np.float32, copy=False)
```



Python training code

Custom solver loop with custom snapshot method

branch: master fast-rcnn / lib / fast_rcnn / train.py

rbgirshick on Apr 27 improve docstrings

1 contributor

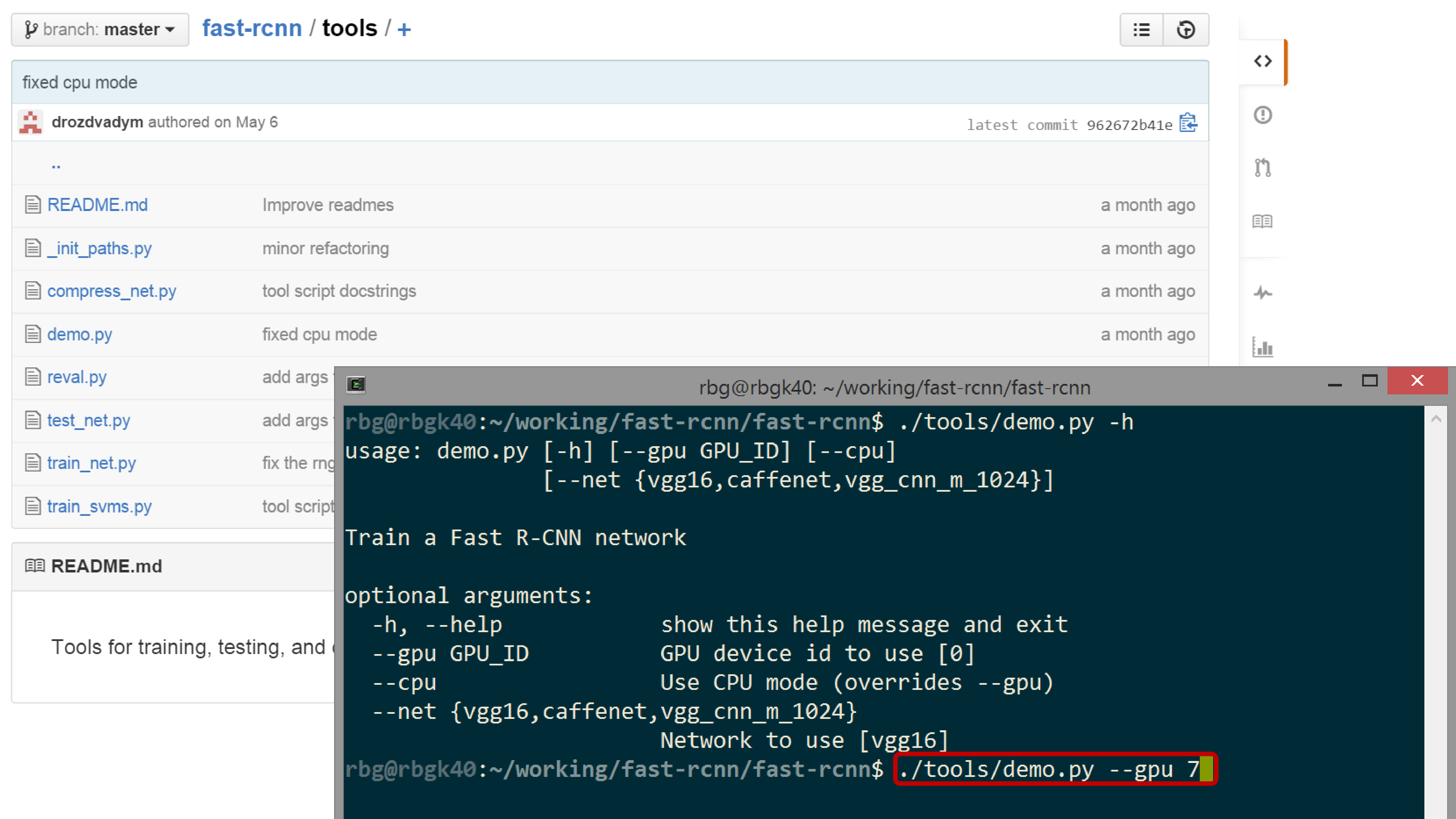
125 lines (100 sloc) | 4.449 kB Raw Blame History

```python
# --------------------------------------------------------
# Fast R-CNN
# Copyright (c) 2015 Microsoft
# Licensed under The MIT License [see LICENSE for details]
# Written by Ross Girshick
# --------------------------------------------------------

"""Train a Fast R-CNN network."""

import caffe
from fast_rcnn.config import cfg
import roi_data_layer.roidb as rdl_roidb
from utils.timer import Timer
import numpy as np
import os

from caffe.proto import caffe_pb2
import google.protobuf as pb2

class SolverWrapper(object):
    """A simple wrapper around Caffe's solver.
    This wrapper gives us control over he snapshotting process, wh
    use to unnormalize the learned bounding-box regression weight
    """

    def __init__(self, solver_prototxt, roidb, output_dir,
                 pretrained_model=None):
        """Initialize the SolverWrapper."""
```

```python
    def train_model(self, max_iters):
        """Network training loop."""
        last_snapshot_iter = -1
        timer = Timer()
        while self.solver.iter < max_iters:
            # Make one SGD update
            timer.tic()
            self.solver.step(1)
            timer.toc()
            if self.solver.iter % (10 * self.solver_param.display) == 0:
                print 'speed: {:.3f}s / iter'.format(timer.average_time)

            if self.solver.iter % cfg.TRAIN.SNAPSHOT_ITERS == 0:
                last_snapshot_iter = self.solver.iter
                self.snapshot()

        if last_snapshot_iter != self.solver.iter:
            self.snapshot()
```

A brief tour of some of the code (CLI tools)

Caffe fork	caffe-fast-rcnn @ bcd9b4e	Update caffe submodule (no roi pooling CPU test)	a month ago
	data	add mat files for boxes, remove pickled version (and update demo.py)	a month ago
	experiments	Make with python layer	a month ago
Python modules	lib	add args to tools/reval.py; fix comp_mode abstraction	a month ago
	matlab	rm accidentally added file	a month ago
	models	Improve readmes	a month ago
	output	Improve readmes	a month ago
Train, test	tools	fixed cpu mode	a month ago
	.gitignore	ignore ipython notebook checkpoints	3 months ago
	.gitmodules	change to https urls for easier installation	a month ago
	LICENSE	update license	2 months ago
	README.md	Add pointers for computing object proposals	a month ago
	todo.txt	update todos	a month ago


branch: master
fast-rcnn / tools / +
fixed cpu mode
drozdvadym authored on May 6
latest commit 962672b41e
..
README.md
Improve readmes
a month ago
_init_paths.py
minor refactoring
a month ago
compress_net.py
tool script docstrings
a month ago
demo.py
fixed cpu mode
a month ago
reval.py
add args
test_net.py
add args
train_net.py
fix the rng
train_svms.py
tool script
README.md
Tools for training, testing, and
rbg@rbgk40: ~/working/fast-rcnn/fast-rcnn
rbg@rbgk40:~/working/fast-rcnn/fast-rcnn$ ./tools/demo.py -h
usage: demo.py [-h] [--gpu GPU_ID] [--cpu]
[--net {vgg16,caffenet,vgg_cnn_m_1024}]
Train a Fast R-CNN network
optional arguments:
-h, --help show this help message and exit
--gpu GPU_ID GPU device id to use [0]
--cpu Use CPU mode (overrides --gpu)
--net {vgg16,caffenet,vgg_cnn_m_1024}
Network to use [vgg16]
rbg@rbgk40:~/working/fast-rcnn/fast-rcnn$ ./tools/demo.py --gpu 7

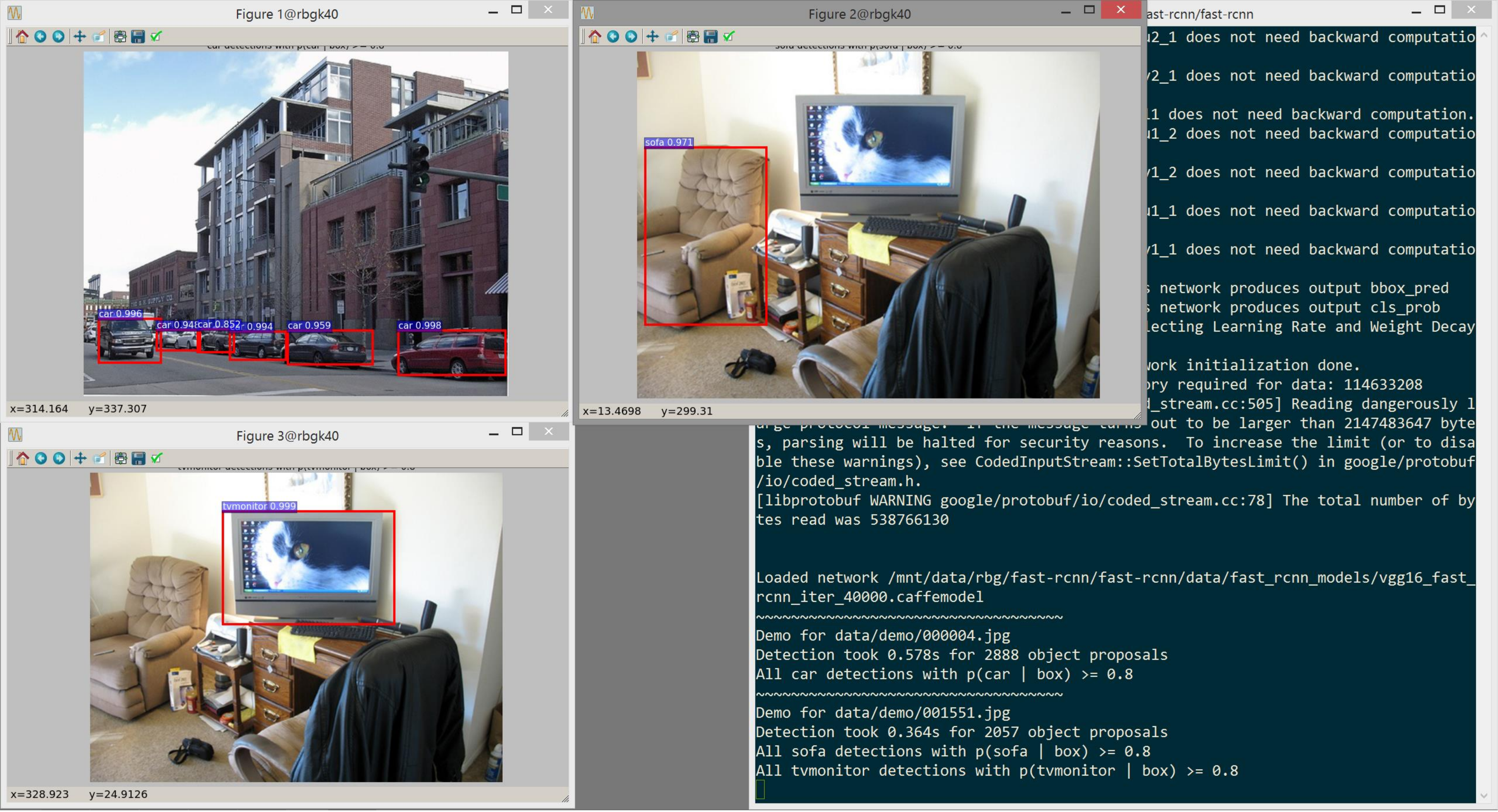


Figure 1@rbgk40
car 0.996
car 0.94
car 0.852
0.994
car 0.959
car 0.998
x=314.164 y=337.307
Figure 2@rbgk40
sofa 0.971
x=13.4698 y=299.31
Figure 3@rbgk40
tvmonitor 0.999
x=328.923 y=24.9126
ast-rcnn/fast-rcnn
u2_1 does not need backward computatio
v2_1 does not need backward computatio
l1 does not need backward computation.
u1_2 does not need backward computatio
v1_2 does not need backward computatio
u1_1 does not need backward computatio
v1_1 does not need backward computatio
s network produces output bbox_pred
s network produces output cls_prob
lecting Learning Rate and Weight Decay
work initialization done.
ory required for data: 114633208
d_stream.cc:505] Reading dangerously l
out to be larger than 2147483647 byte
s, parsing will be halted for security reasons. To increase the limit (or to disa
ble these warnings), see CodedInputStream::SetTotalBytesLimit() in google/protobuf
/io/coded_stream.h.
[libprotobuf WARNING google/protobuf/io/coded_stream.cc:78] The total number of by
tes read was 538766130
Loaded network /mnt/data/rbg/fast-rcnn/fast-rcnn/data/fast_rcnn_models/vgg16_fast_
rcnn_iter_40000.caffemodel
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Demo for data/demo/000004.jpg
Detection took 0.578s for 2888 object proposals
All car detections with p(car | box) >= 0.8
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Demo for data/demo/001551.jpg
Detection took 0.364s for 2057 object proposals
All sofa detections with p(sofa | box) >= 0.8
All tvmonitor detections with p(tvmonitor | box) >= 0.8

Teaser: *Faster R-CNN*

Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun. Microsoft Research

- The detection network *also proposes objects*
- Marginal cost of proposals: 10ms
- VGG16 runtime ~200ms *including all steps*
- Higher mAP, faster
- Open-source Caffe code coming later this summer

Region Proposal Network shares conv layers with Fast R-CNN object detection network