Object Detection

Slow Fast Faster RCNN

Vision@OUC

Wang Chao

Group of DL

Magic book

Overview

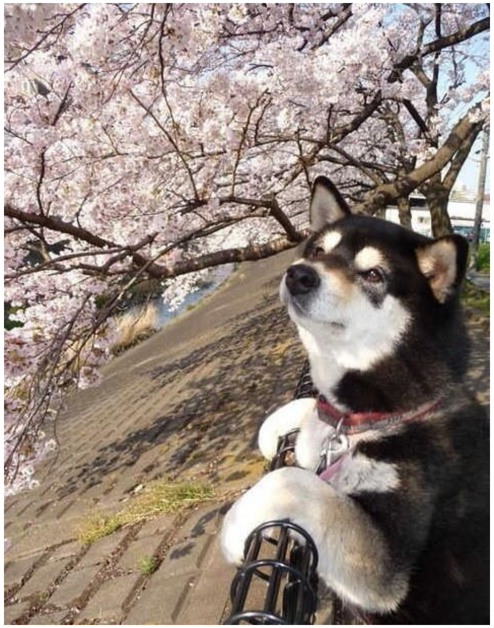
- Introduction
- Slow Fast Faster R-CNN
- Py-faster-rcnn
- Other works
- Q&A

Magic book

Introduction

To give computers visual intelligence

---Feifei Li

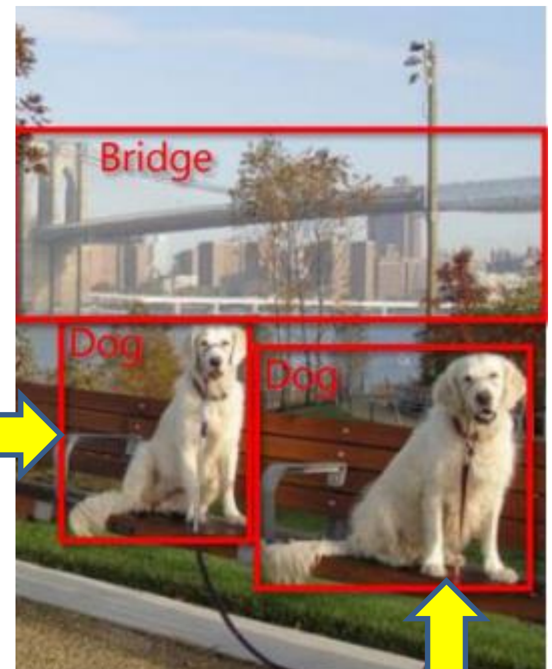


Magic book

Classification VS Detection

Classification: What

Detection: What and Where



Localization
Where ?

Recognition
What ?

Usually， We need a bounding box to tell us “What” and “Where”

Magic book

Efficient Object Detection

- Object detection is arguably a harder problem than image classification
- Usually a large number of image sub-window need to scanned in order to localize objects, leading to heavy computational processing
- Challenge: In many real-world applications, running a fast object detector is as critical as running an accurate object detector

Magic book

What is an elegent object detector

- Speed:
  - Per image
- Accuracy
  - MAP
- Others
  - Training time
  - Model
  - generalization

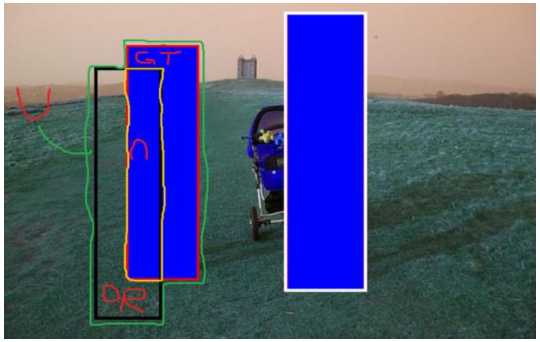
Magic book

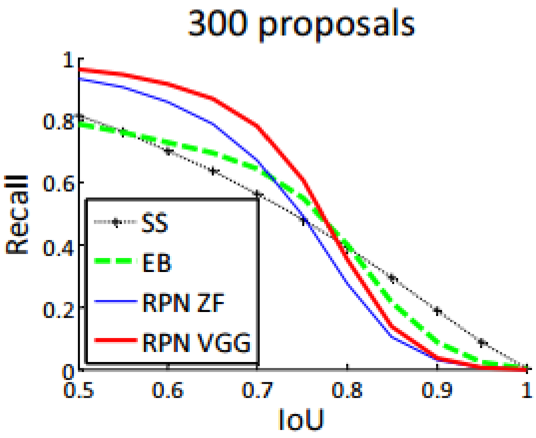
Evaluation

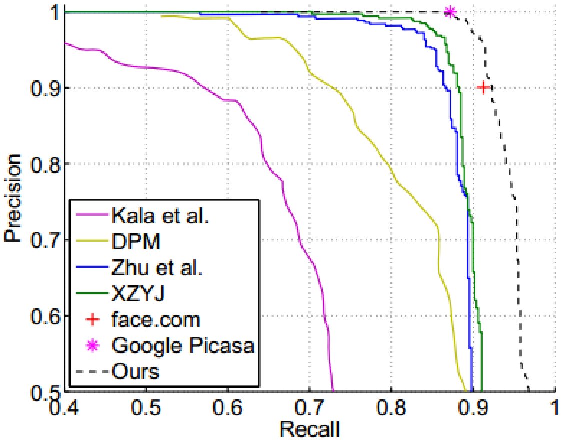
- Box match
  - IoU: Intersection-over-Union

$$IOU = \frac{DetectionResult \bigcap GroundTruth}{DetectionResult \bigcup GroundTruth}$$

- Accuacy
  - Recall
  - Miss rate
  - False Positive per Image
  - Precision
  - Average Precision







- Dataset
  - Pascal VOC
    - 2007， 2010， 2012
    - 20 categories
  - Ms COCO
    - 2015
    - 80 categories
  - ImageNet
    - ILSVRC 2013,2014,2015,2016
    - 200 categories
  - Others
    - Face： AFW， FDDB， MALF， IJB-A
    - Pedestrain ： INRIA， KITTI
    - Vehicle： KITTI

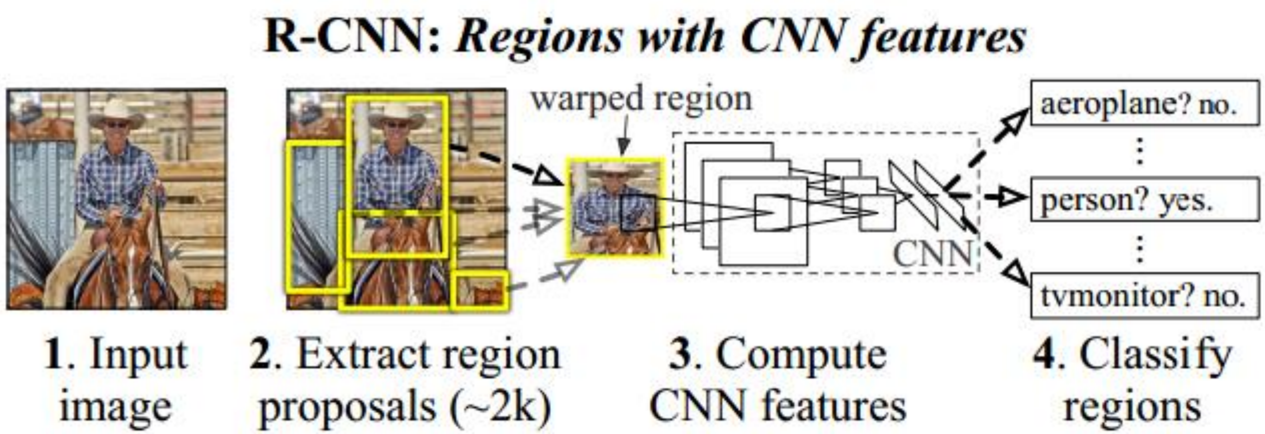
R-CNN

- Past methods： complex ensemble systems
  - Combine multiple low-level image features with high-level context
- RCNN： Regions with CNN features
  - CNNs for region proposal
  - Supervised pre-training for scarce training data

Features mater！

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In CVPR，2014

- R-CNN architecture



- Takes an input image
- Extracs around 2k bottom-up region proposals
- Compute features for each proposal using CNN
- Classifies each proposal region using linear SVMs

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In CVPR，2014

Training pipeline

Steps for training a slow R-CNN detector

1. [offline]M <<< Pre-train a ConvNet for ImageNet Classification
2. M’ <<< Fine-tune M for object detection(softmax + log loss)
3. F <<< Cache feature vectors to disk using M’
4. Train post hoc linear SVMs on F(hinge loss/L2 loss)
5. Train post hoc linear bounding-box regressors on F(squared loss)

* Ignoring pre-training, there are three separate training stages

Magic book

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In CVPR，2014

Why I call it "Slow" R-CNN

Example timing for R-CNN on VOC07(only 5k training images) using VGG16 and a K40 GPU

Fine-tune(BP, SGD): 18 hours

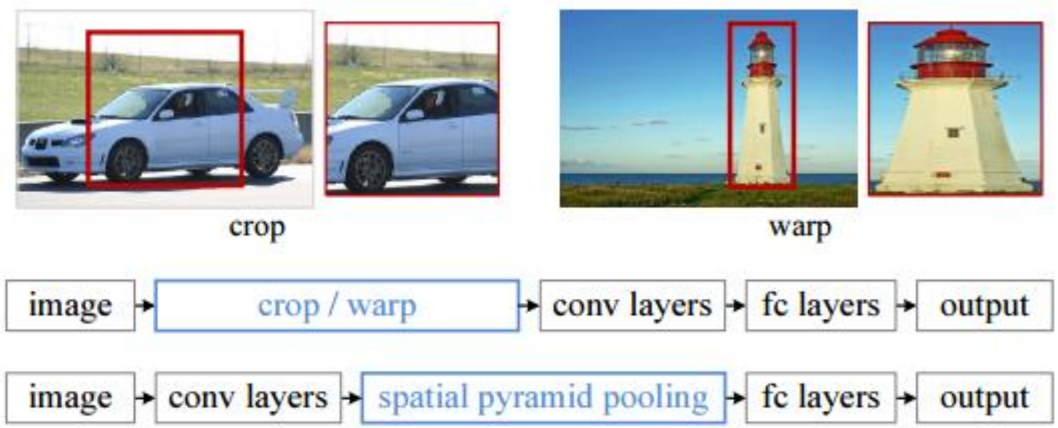
Feature extraction: 63 hours

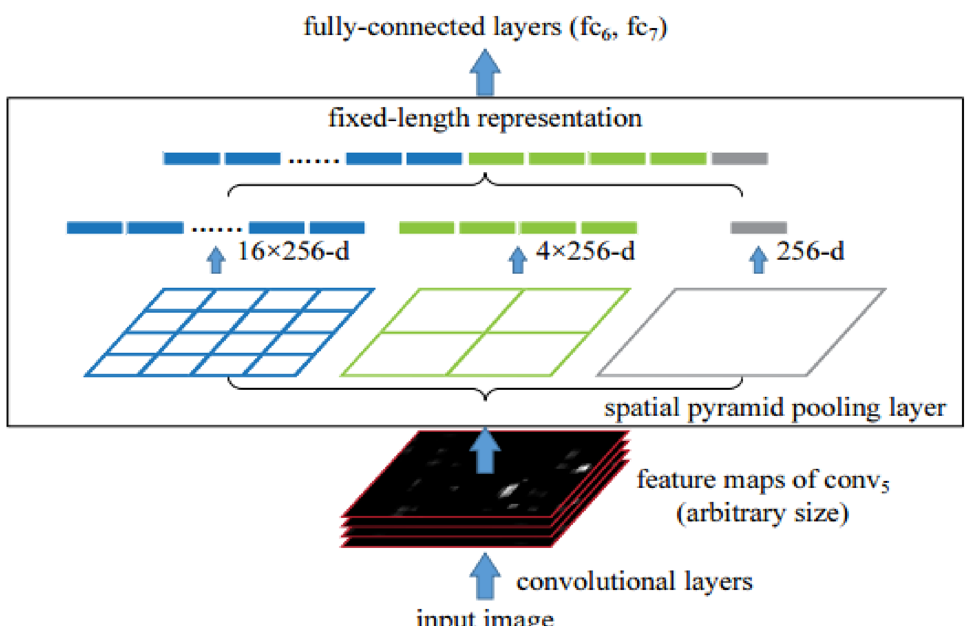
SVM and bounding-box regressor training: 3 hours

Total: 84 hours

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In CVPR，2014

SPP-net





K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014.

SPP-net helps

Example timing for R-CNN/SPP-net

- Fine-tuning(BP, SGD): 18 hours/16 hours
- Feature extraction: 63 hours/ 5.5 hours
  - Forward pass time(helps here)
  - Disk I/O is costly(dominates SPP-net extraction time)
- SVM and bounding-box regressor training 3 hours/4 hours
- Total: 84 hours/25.5 hours

K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014.

Fast R-CNN

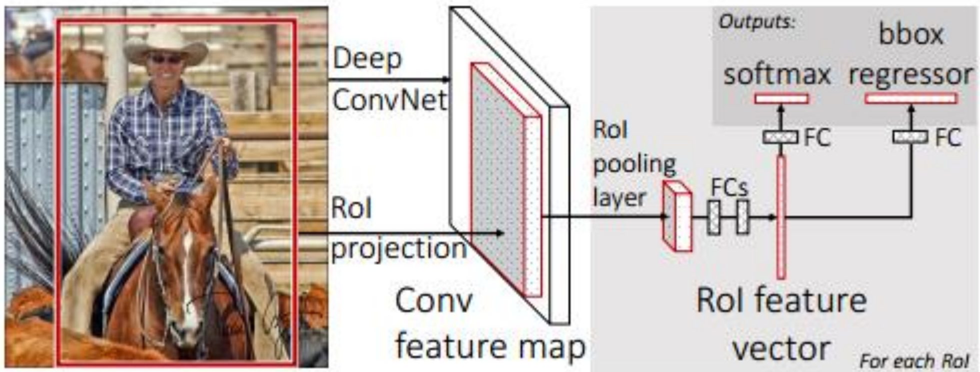
Train models in multi-stage pipelines that are slow and inelegant

R-CNN and SPPnet

1. Training is a multi-stage pipeline
   ConvNet; SVM; bounding-box regressor
2. Training is expensive in space and time
   features of each propos in each image
3. Object detection is slow
   features extract； VGG16 47s/image(GPU)

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation。In，ICCV，2015.

Fast R-CNN architecture

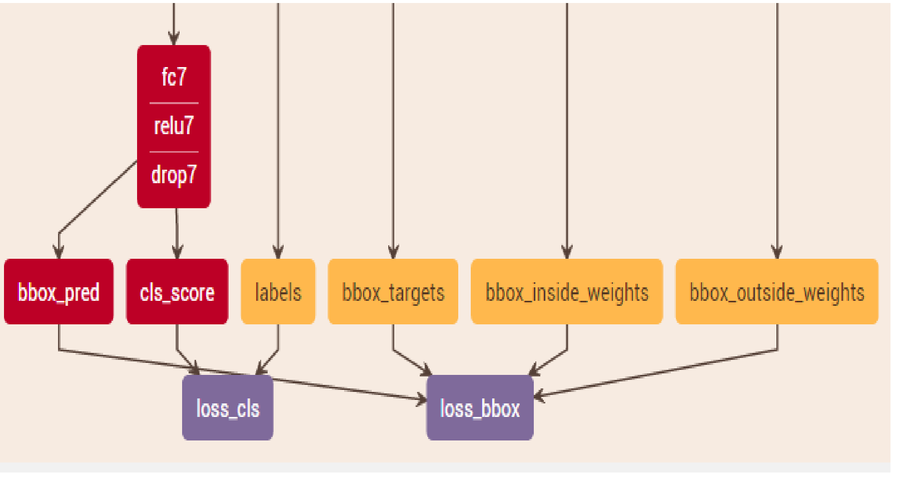


- An input image and multiple RoIs are input into fully convNet
- Each RoIs is pooled into a fixed-size feature map and then mapped to a feature vector by FC layes
- The network has two output per RoI： Softmax probability and bouding-box regression
- Trained end-to-end with multi-task loss

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation。In，ICCV，2015.

Training Fast R-CNN end-to-end

- Define one network with two loss branches
  - Branch 1： softmax classifer
  - Branch 2： linear bounding-box regressors
  - Overall loss is the sum of the two loss branches
- Fine-tune the network jointly with SGD
  - Optimizes features for both tasks
- BP errors all the way back to the conv layers



Magic book

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation。In，ICCV，2015.

Benefits of end-to-end training

- Simpler implementation
- Faster training
  - No reading/writing features from/to disk
  - No training post hoc SVMs and bounding-box regressors
- Optimizing a single multi-task objective work better than optimizing objectives independently

End to End training requires overcoming two technical obstacles

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation。In，ICCV，2015.

Obstacle1: ROI pooling

- Spatial Pyramid Pooling layer

From Kaiming's slides

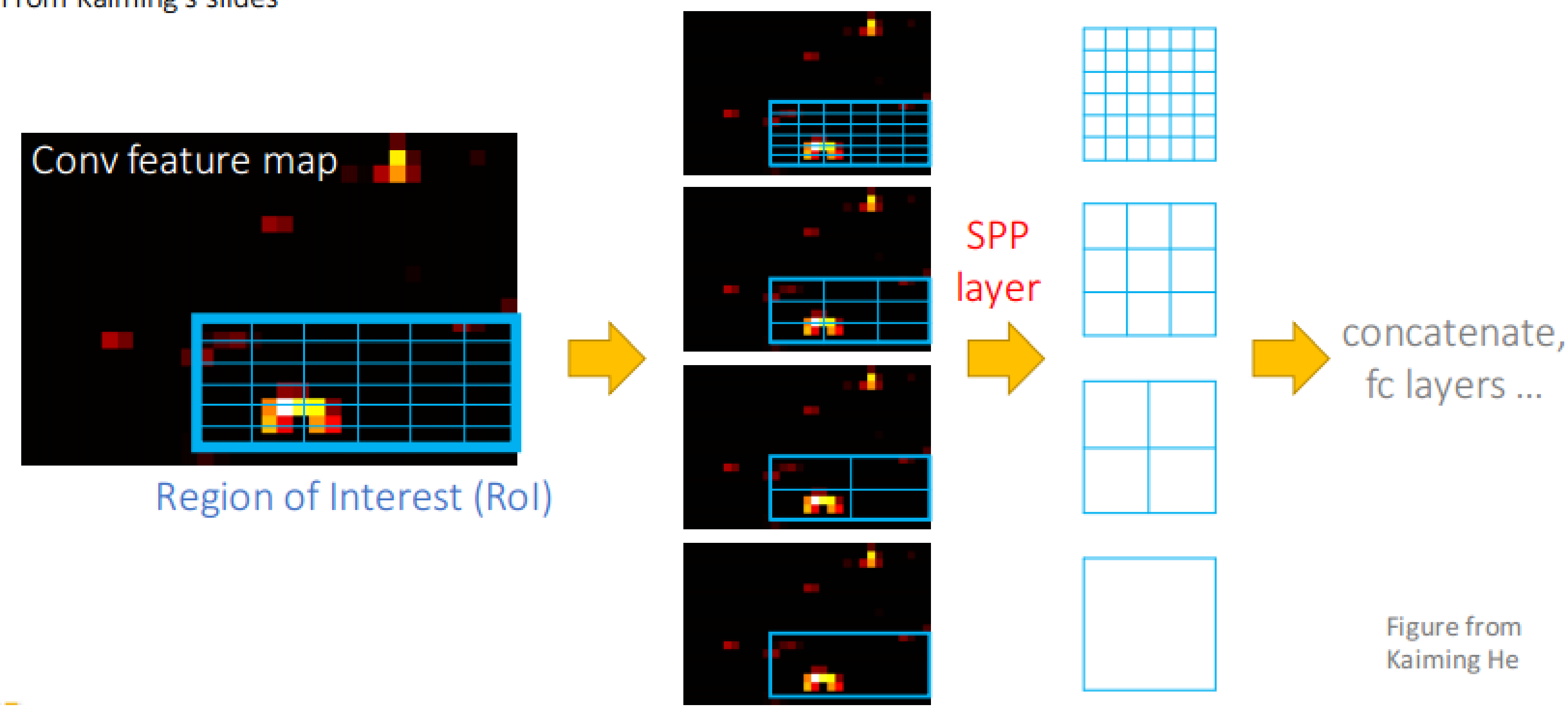
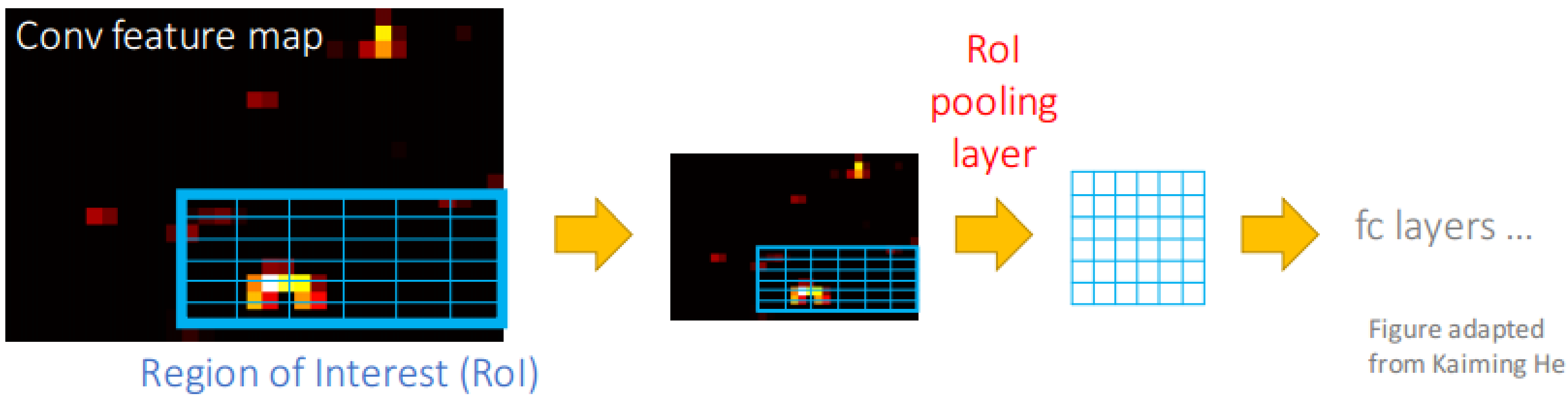


Figure from
Kaiming He

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation。In，ICCV，2015.

Region of Interest pooling layer

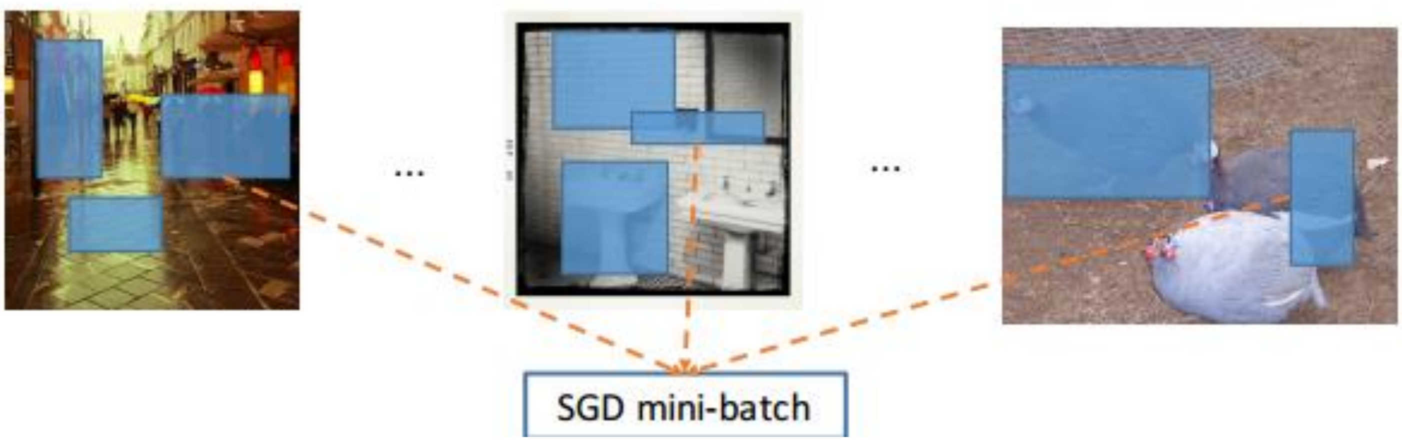


Just a special case of the SPP layer with one pyramid level

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation。In，ICCV，2015.

Obstacle2: Make SGD steps efficient

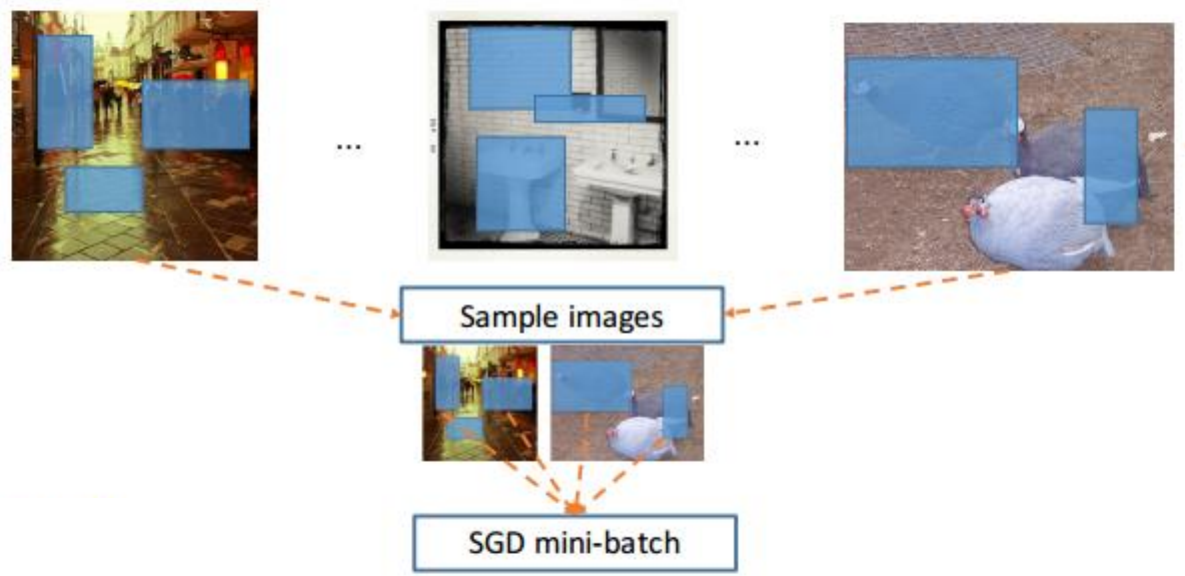
- R-CNN and SPP-net use region-wise sampling to make mini-batch
  - Sample 128 example ROIs uniformly at random
  - Examples will come from different images



Magic book

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation。In，ICCV，2015.

- Solution: use hierarchical sampling to build mini-batches
  - Sample a small number of images(2)
  - Sample many examples from each image



Cost per mini-batch compared to slow R-CNN

input size for Fast R-CNN input size for slow R-CNN

2*600*1000 / (128*224*224) = 0.19x < computation than slow R-CNN

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation。In，ICCV，2015.

Towards fast and end-to-end

- Multi-task loss for uinfied network

$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1] L_{\text{loc}}(t^u, v),$$

- Turuncated SVD for faster detection

  Compress FC layers

  $$W \approx U \Sigma_t V^{\mathrm{T}}$$

  Before: $u \times v$

  After: $t(u+v)$

  Especially: t <= min(u,v)

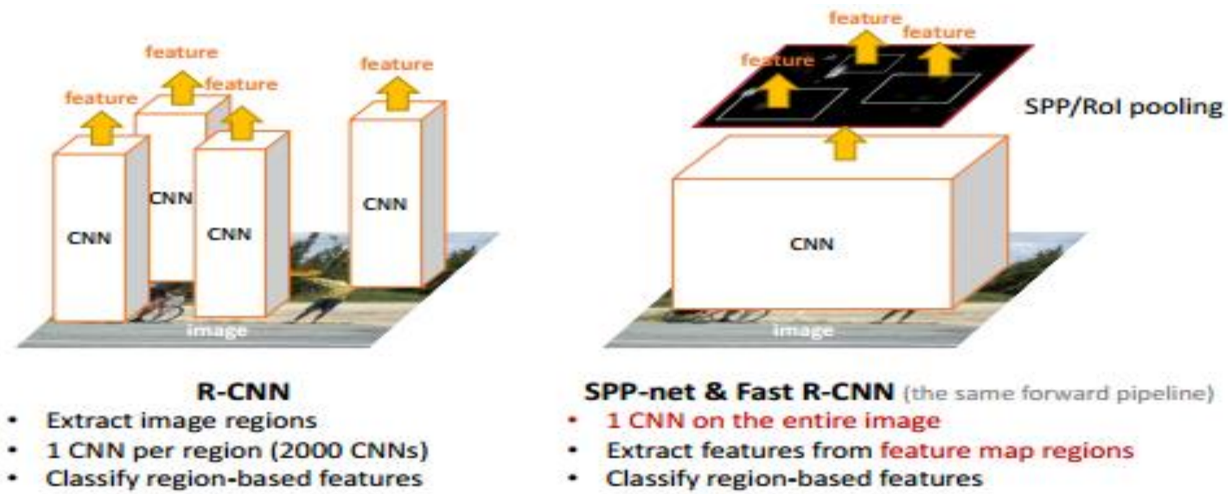
Magic book

Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation。In，ICCV，2015.

Fast R-CNN outcome

- Better training time and testing time with better accuracy than slow R-CNN / SPP-net

- Training time: 84 hours/25.5hours/8.75hours
- VOC07 test mAP: 66.0%/63.1%/68.1%
- Testing time per image: 47s/2.3s/0.32s
  - Plus 0.2 to > 2s per image depending on proposal method
  - With selective search: 49s/4.3s/2.32s



Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation。In，ICCV，2015.

Faster R-CNN

State-of-the-art object detection networks depend on region proposal algorithms to hypothesize object locations. Advances like SPP-net and Fast R-CNN have reduced the running time, exposing region proposal computation as a bottleneck.

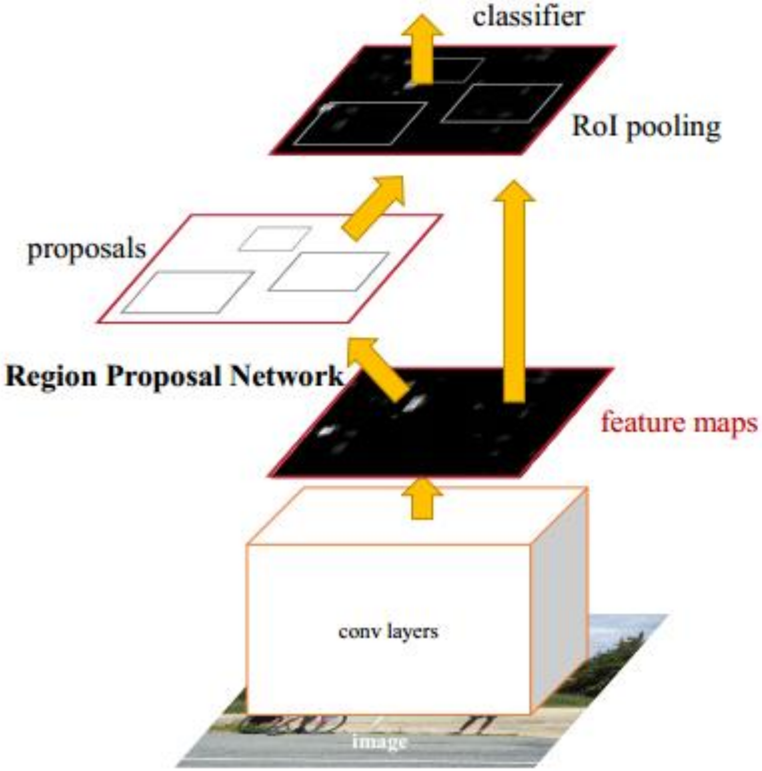
Magic book

Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2016:1-1.

Faster R-CNN architecture

Faster R-CNN =

RPN + Fast RCNN



Does not depend on an external region proposal algrorithm

Does object detection in a single forward pass

Magic book

Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2016:1-1.

Training method

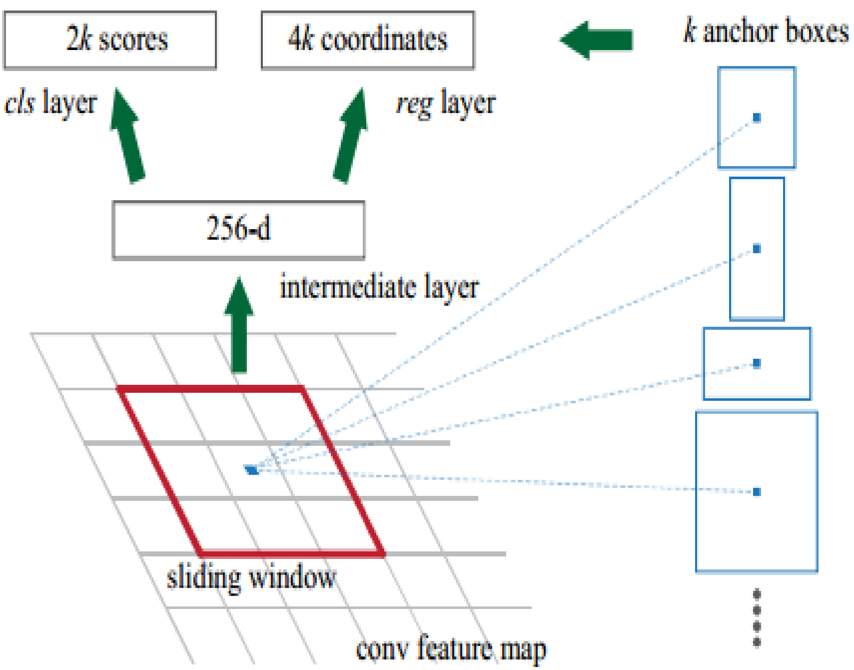
1. Train RPN: RPN is initialized with imagenet pre-trained model and fine-tuned end to end for region proposal task
2. Use region proposal by RPN, train Fast-RCNN : initialized by ImageNet pre-trained model
3. Train RPN and fix the shared convolutional layers
4. Train Fast R-CNN and fix the convolutional layers and form a unified network.

Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2016:1-1.

Magic book

Region Proposal Network

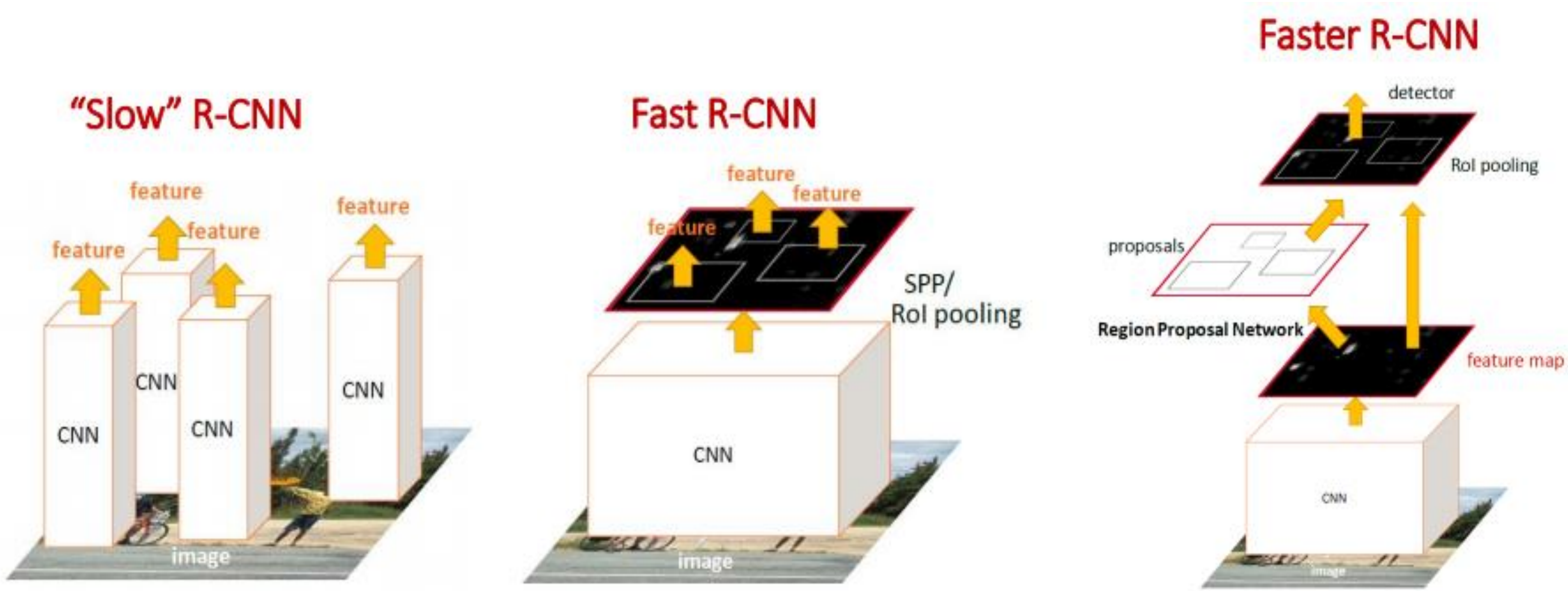
To generate region proposals：
Slide window in last shared ConvNet： 3*3
Lower-dimensional feature： 256D(FZ)\512D(VGG16)
Generate: cls loss and reg loss
1*1 convolutional layer
Anchors:
Multibox uses k-means: 800 anchors
Multibox:(4 + 1)* 800 dimensional fc output layer
Output layer：
6.1 * 10 ^6, (1536 * (4 + 1)* 800) for googlenet
RPN:(4 +2) * 9 dimensional conv output layer
output layer： 2.8 * 10^4,
512 * （4 +2） * 9 for VGG16
Multi-scale design:
From single-scale image; 3 scale and 3 aspect ratios
Scale:128,256,512; ratios: 1:1;1:2;2:1
Translation-Invariant Anchors
Without extra cost for adressing scales



Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2016:1-1.

Magic book

Slow vs Fast vs Faster



Magic book

Py-Faster-RCNN 环境搭建

Requirements: hardware

Installation

1. Clone the Faster R-CNN

   git clone –recursive https://github.com/rbgirshick/py-faster-rcnn.git

2. Build Cython modules

   cd $FRCN_ROOT/lib

   Make

3. Build Caffe and pycaffe(see: Caffe installation instructions)

   Note: In Makefile.config

   WITH_PYTHON_LAYER : = 1

4. Download pre-computed Faster R-CNN detectors

   cd $FRCN_ROOT

   ./data/scripts/fetch_faster_rcnn_models.sh

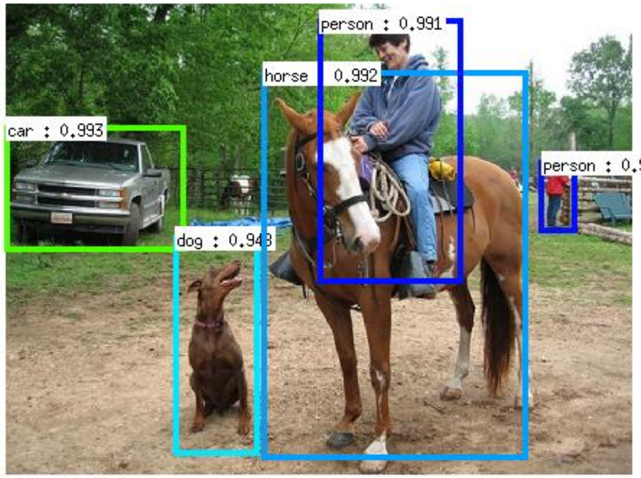
5. Demo

   After successfully completing basic installation

   To run the demo

   cd $FRCN_ROOT

   ./tools/demo.py



Questions occurred

- cudnn version problems

For cudnn V5:

1. Replace these document with new caffe

include/caffe/layers/cudnn_relu_layer.hpp,

src/caffe/layers/cudnn_relu_layer.cpp, src/caffe/layers/cudnn_relu_layer.cu

include/caffe/layers/cudnn_sigmoid_layer.hpp,

src/caffe/layers/cudnn_sigmoid_layer.cpp, src/caffe/layers/cudnn_sigmoid_layer.cu

include/caffe/layers/cudnn_tanh_layer.hpp,

src/caffe/layers/cudnn_tanh_layer.cpp, src/caffe/layers/cudnn_tanh_layer.cu

2. Replace the name in include/caffe/util/cudnn.hpp

cudnnConvolutionBackwardData_v3 函数名替换为 cudnnConvolutionBackwardData

cudnnConvolutionBackwardFilter_v3函数名替换为 cudnnConvolutionBackwardFilter

- Demo in CPU version

http://blog.sina.com.cn/s/blog_679f93560102wpyf.html

More than Demo

- Datasets: MS coco/ ImageNet
  - factory.py
  - new dataset class：　my_dataset.py(pascal_voc.py)
  - _load_image_set_index, image_path_from_index
  - _load_pascal_annotation
- Sources:
  - https://github.com/deboc/py-faster-rcnn/tree/master/help
  - https://github.com/rbgirshick/py-faster-rcnn/issues/243

Magic book

New network archticture

- ResNet-152
  - Download the prototxt:
    https://github.com/KaimingHe/deep-residual-networks
  - train.prototxt
  - test.prototxt
- Train your model
  - class number: input-data layer
  - Add Fast R-CNN layer
  - RPN layer

```
layer {
  name: "rpn_conv/3x3"
  type: "Convolution"
  bottom: "res4f"
  top: "rpn/output"
  param { lr_mult: 1.0 }
  param { lr_mult: 2.0 }
  convolution_param {
    num_output: 512
    kernel_size: 3 pad: 1 stride: 1
    weight_filler { type: "gaussian" std: 0.01 }
    bias_filler { type: "constant" value: 0 }
  }
}
```

```
...
name: "rpn_relu/3x3"
...
name: "rpn_cls_score"
...
name: "rpn_bbox_pred"
...
name: "rpn_cls_score_reshape"
...
name:  rpn-data
...
name: "rpn_loss_cls"
...
name: "rpn_loss_bbox"
...
```

Magic book

Other works recently

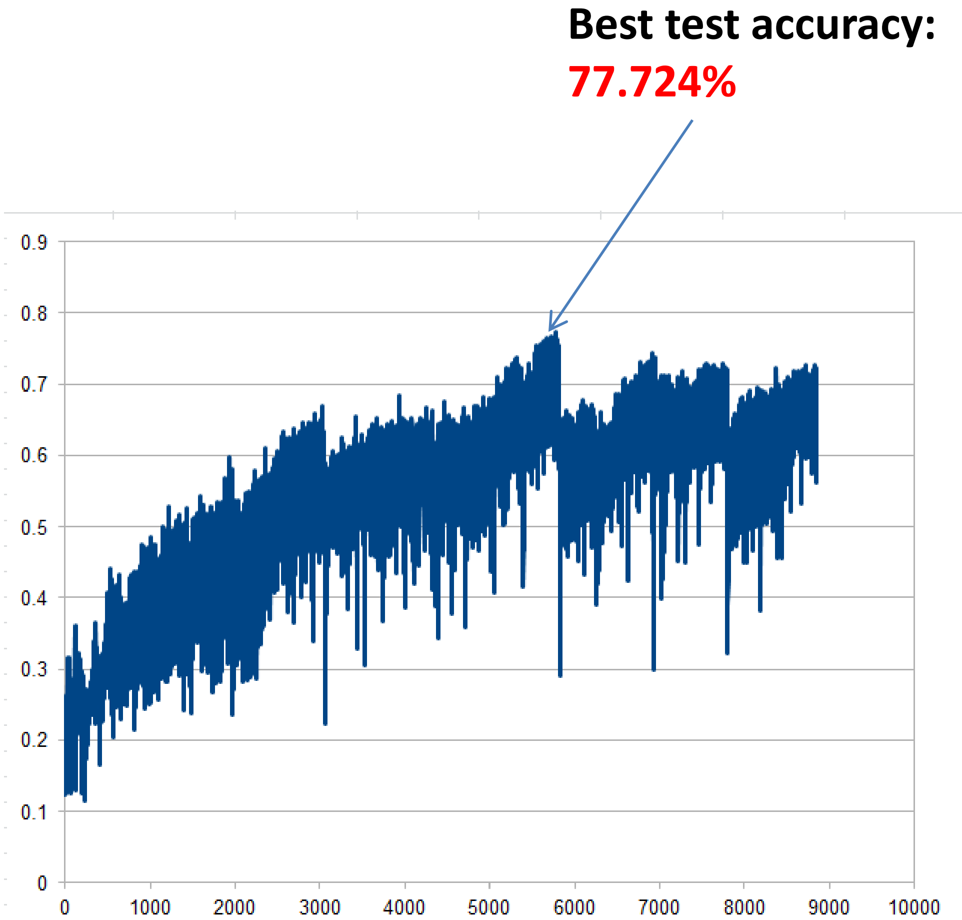
- LSTM converge & bidirectional LSTM
- Deploy sever with python Flask

LSTM

1.Error fixed

2.Momentum

3.L2 Norm(weight decay)



Flask for a web sever

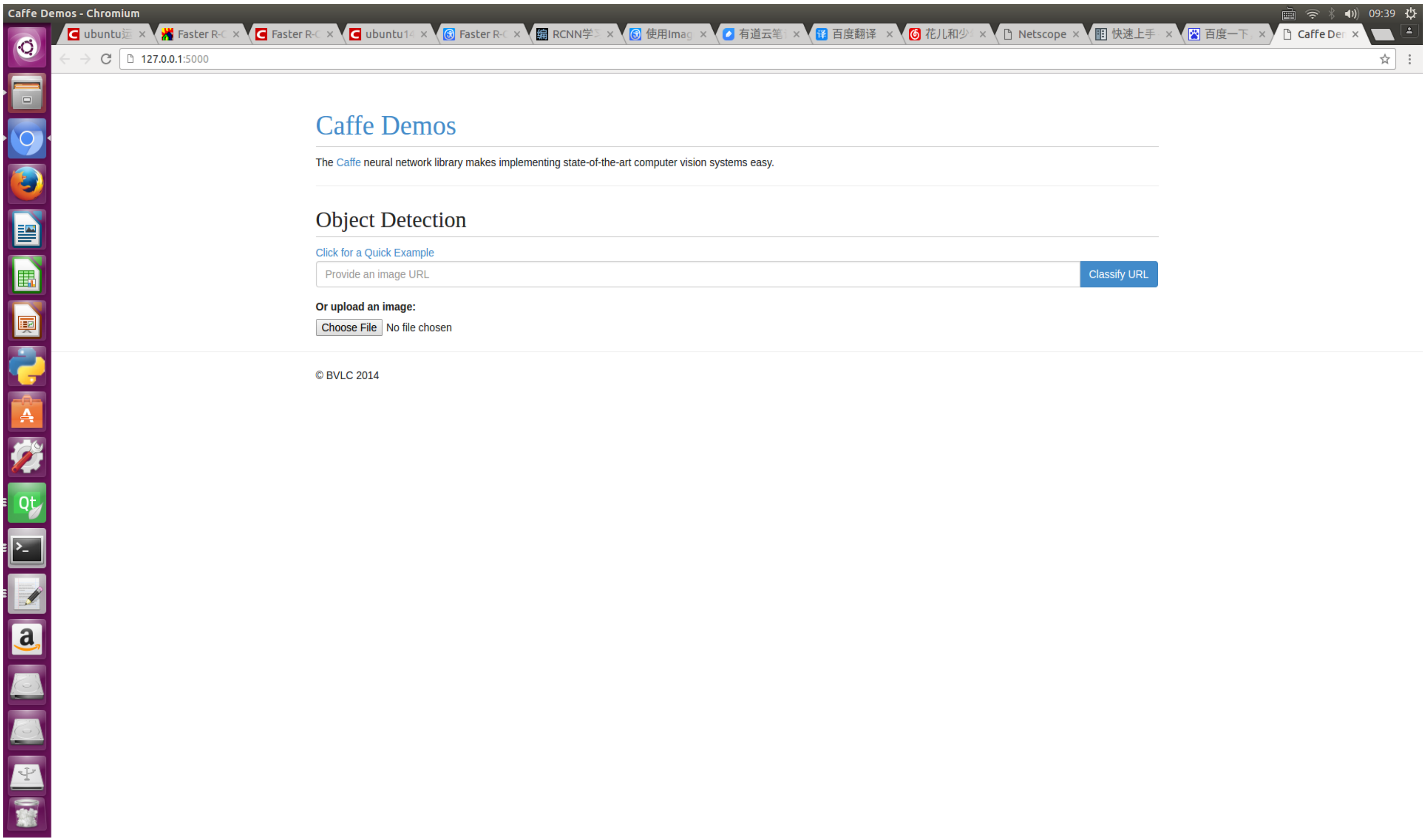
A small application:

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello World!'

if __name__ == '__main__':
    app.run()
```

```
import datetime
import logging
import flask
import werkzeug
import optparse
import tornado.wsgi
import tornado.httpserver
import numpy as np
import pandas as pd
from PIL import Image
import cStringIO as StringIO
import urllib
import exifutil


REPO_DIRNAME = os.path.abspath(os.path.dirname(os.path.abspath(__fi
UPLOAD_FOLDER = '/tmp/caffe_demos_uploads'
ALLOWED_IMAGE_EXTENSIONS = set(['png', 'bmp', 'jpg', 'jpe', 'jpeg',

# Obtain the flask app object
app = flask.Flask(__name__)


@app.route('/')
def index():
    return flask.render_template('index.html', has_result=False)


@app.route('/detection_url', methods=['GET'])
def dectection_url():
    imageurl = flask.request.args.get('imageurl', '')
    try:
        string_buffer = StringIO.StringIO(
            urllib.urlopen(imageurl).read())
        image = caffe.io.load_image(string_buffer)
```

Source: http://docs.jinkan.org/docs/flask/quickstart.html#a-minimal-application

Web Demos


Caffe Demos - Chromium
127.0.0.1:5000
Caffe Demos
The Caffe neural network library makes implementing state-of-the-art computer vision systems easy.
Object Detection
Click for a Quick Example
Provide an image URL
Classify URL
Or upload an image:
Choose File
No file chosen
© BVLC 2014

Q & A