Talks on Deep Learning, Caffe, RCNN

Zhe Li

08/28/2015

Outline

- Introduction to Deep Learning
- Caffe: (Open Source Library --Implementation)
- RCNN: Object Detection Scheme Based on Caffe (Usage)

Deep Learning Library

- Cuda-convnet
  - Very fast on state-of-the-art GPUs with Multi-GPU parallelism
  - C++/CUDA Library
- Caffe
  - C++/CUDA Library
  - Seamless GPU/CPU switch
  - An active research and development community
- Torch7, Theano, Mocha

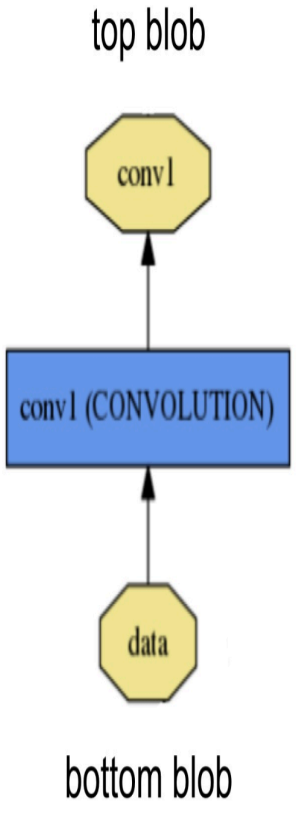
Caffe Library

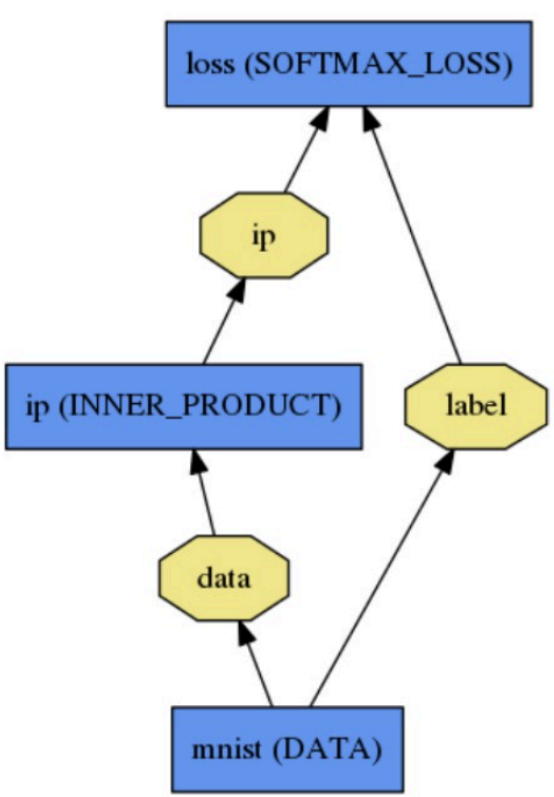
- Pure C++/CUDA architecture for deep learning
  - Command line, python, MATLAB interfaces
- Tools, reference models, demos and recipes
- Seamless switch between CPU and GPU
- Fast, Well tested code.

Caffe: look at deeper

- There are three components more important
  - Blob
    - A wrapper over actual data being processed and passed along Caffe
    - Provides synchronization capacity between CPU and GPU
    - Actually, it is an N-dimensional array
  - Layer
    - Essence of a model and fundamental unit of computation, convolution, pooling, inner product, and so on
    - Define Forward and backward function
  - Net
    - the structure of neuron network
    - Composition of all layers

Blob, Layer, Net





Net

- A network is a set of layers connected Directed Acyclic Graph(DAG)

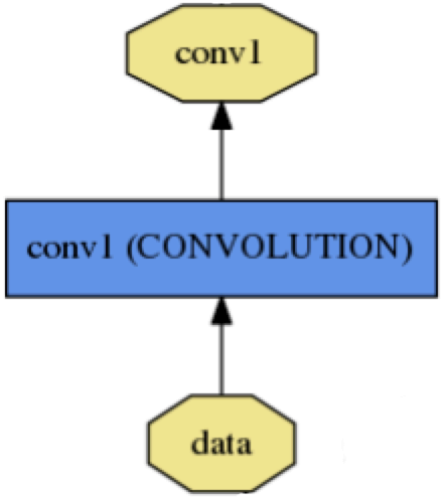
```
name: "dummy-net"
layers { name: "data" …}
layers { name: "conv" …}
layers { name: "pool" …}
    … more layers …
layers { name: "loss" …}
```

Layers

```
name: "conv1"
type: CONVOLUTION
bottom: "data"
top: "conv1"
convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
        type: "xavier"
    }
}
```

name, type, and the connection structure (input blobs and output blobs)

layer-specific parameters



Each layer defines Forward_cpu(), Backward_cpu() in C++ and Forward_gpu() and Backward_gpu() in CUDA.

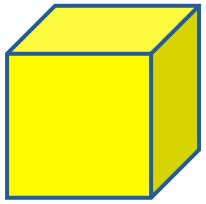
Data layers (image_layer, window_layers); convolution, pooling layer

…

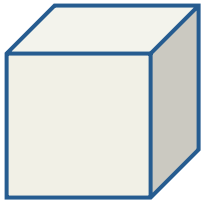
Blob

Blobs are 4-D arrays for storing and communicating information.

- hold data, derivatives, and parameters
- lazily allocate memory
- shuttle between CPU and GPU



Data
*N*umber x *K* Channel x *H*eight x *W*idth
256 x 3 x 227 x 227 for ImageNet train input

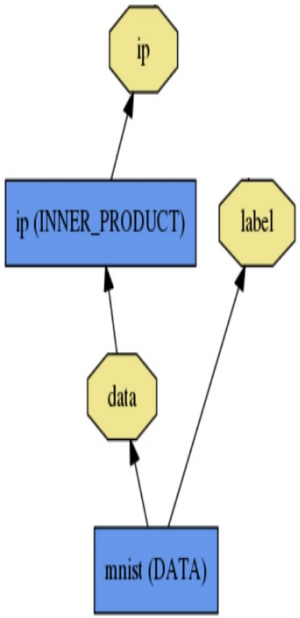


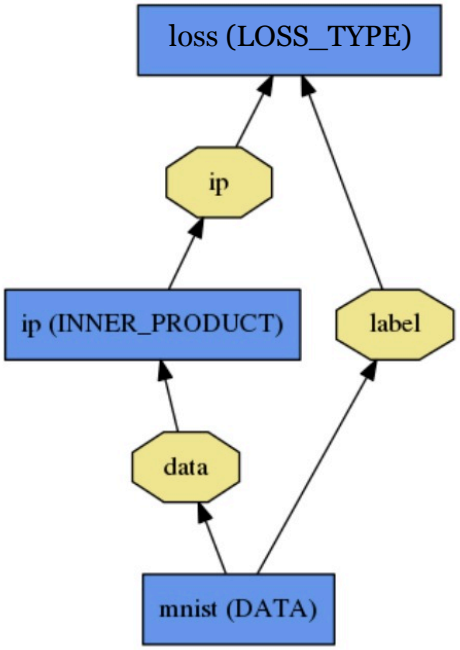
Parameter: Convolution Weight
N Output x *K* Input x *H*eight x *W*idth
96 x 3 x 11 x 11 for CaffeNet conv1

Parameter: Convolution Bias
96 x 1 x 1 x 1 for CaffeNet conv1

Loss layer

What kind of model is this?





Classification
`SOFTMAX_LOSS`
`HINGE_LOSS`

Linear Regression
`EUCLIDEAN_LOSS`

Attributes / Multiclassification
`SIGMOID_CROSS_ENTROPY_LOSS`

Others…

New Task
`NEW_LOSS`

Loss Layer

- Loss function determines the learning task.
- Given data D, a Neuron Network typically minimizes

$$L(W) = \frac{1}{|D|} \sum_{i}^{|D|} f_W \left(X^{(i)} \right) + \lambda r(W)$$

Solver

- **Solver** optimizes the network weights W to minimize the loss L(w) over the data D

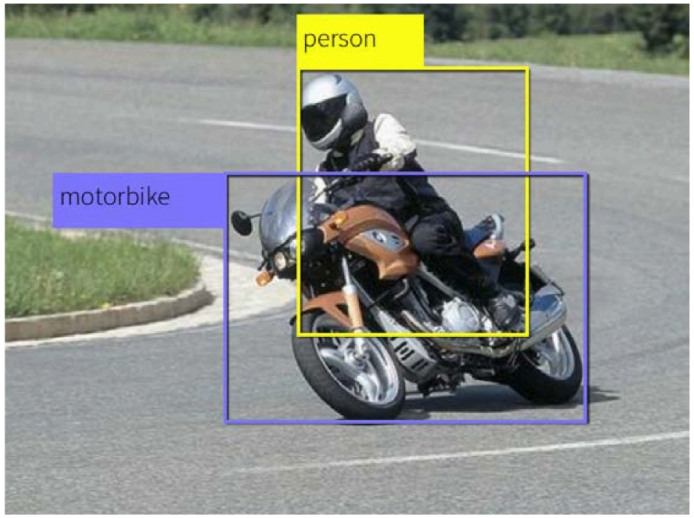
$$L(W) = \frac{1}{|D|} \sum_{i}^{|D|} f_W \left(X^{(i)} \right) + \lambda r(W)$$

- Caffe provides three different methods
  - Stochastic gradient descent
  - Nesterov's accelerated gradient
  - Adaptive gradient

Object Detection -RCNN

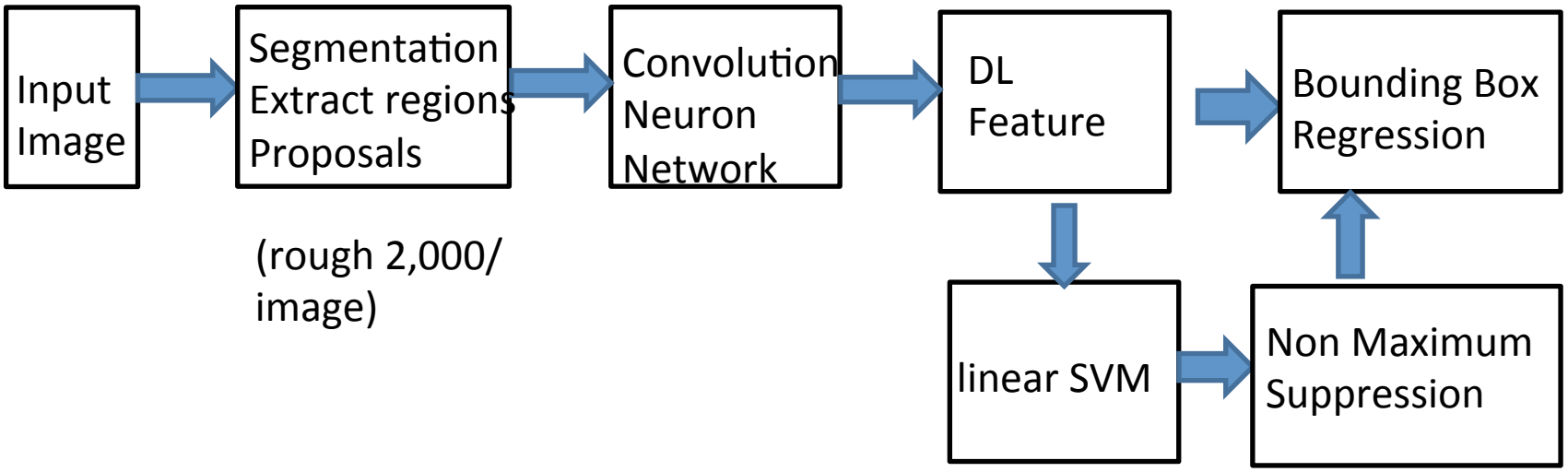
Input Image

Output Image

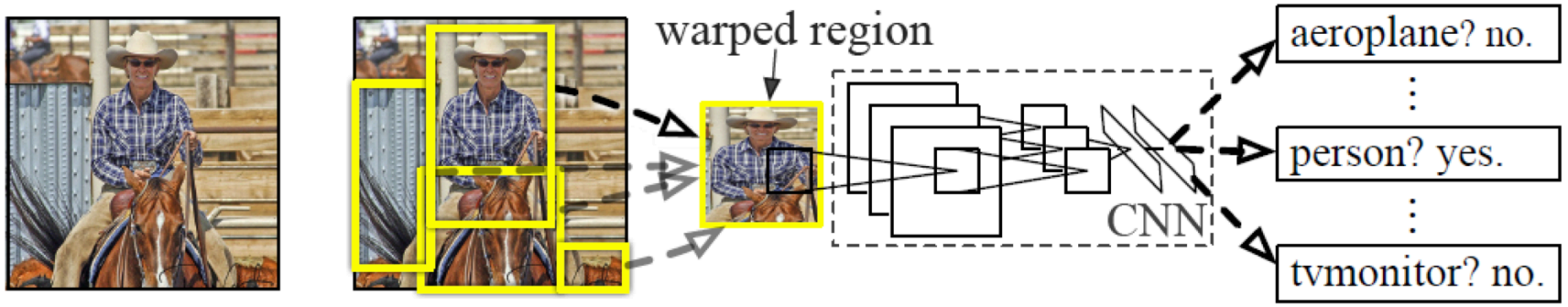


Object Detection -RCNN

- RCNN(University of California, Berkeley)

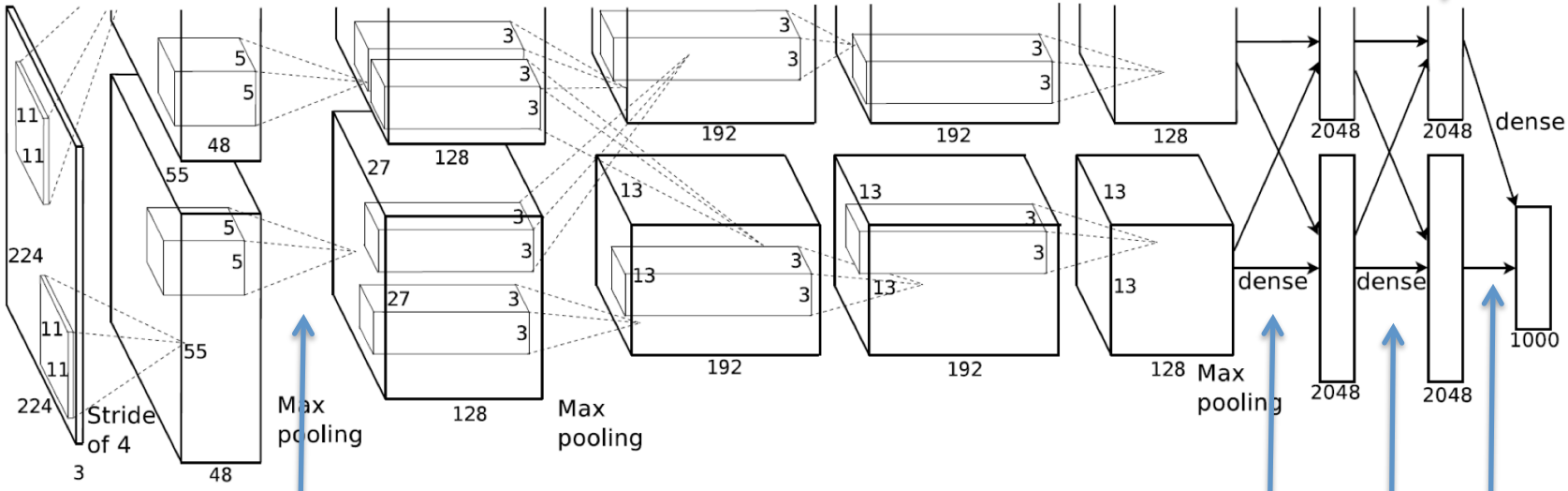


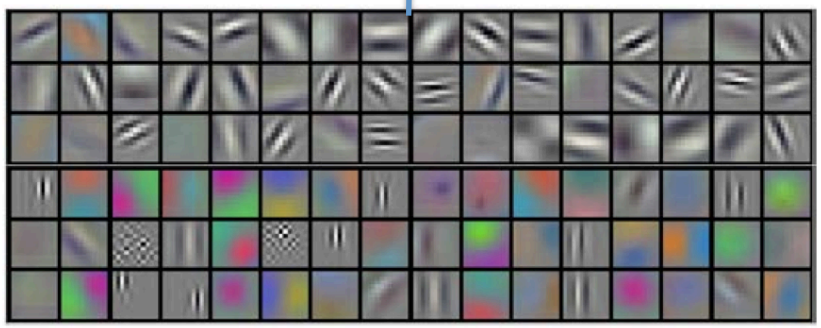
Object Detection -RCNN



	localization	feature extraction	classification
this paper:	selective search	deep learning CNN	binary linear SVM
alternatives:	objectness, constrained parametric min-cuts, sliding window …	HOG, SIFT, LBP, BoW, DPM …	SVM, Neural networks, Logistic regression …

Alex Network



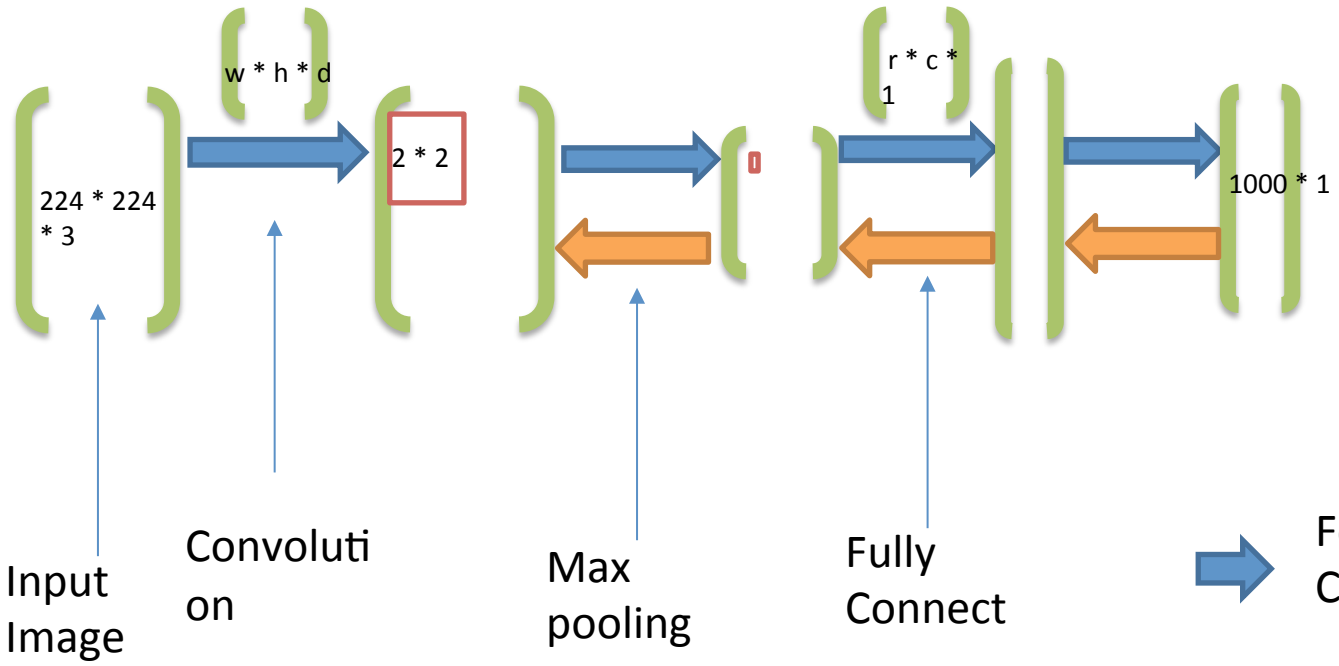


Pool5 9126
Dim

Fc6 4096-Dim

Fc7 4096-Dim

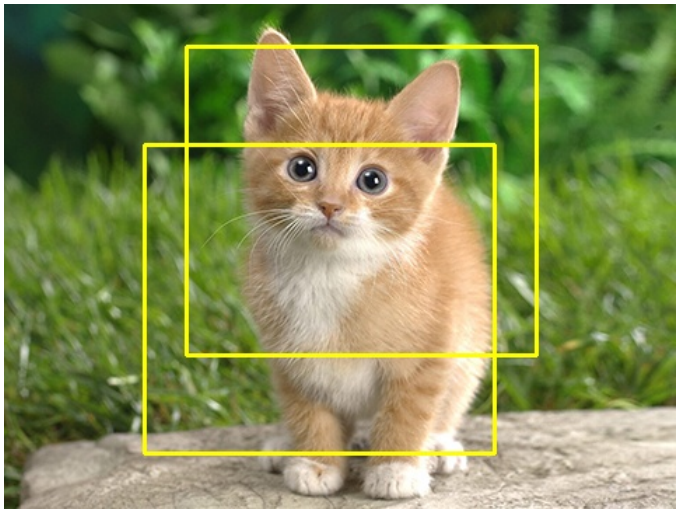
Implementation View

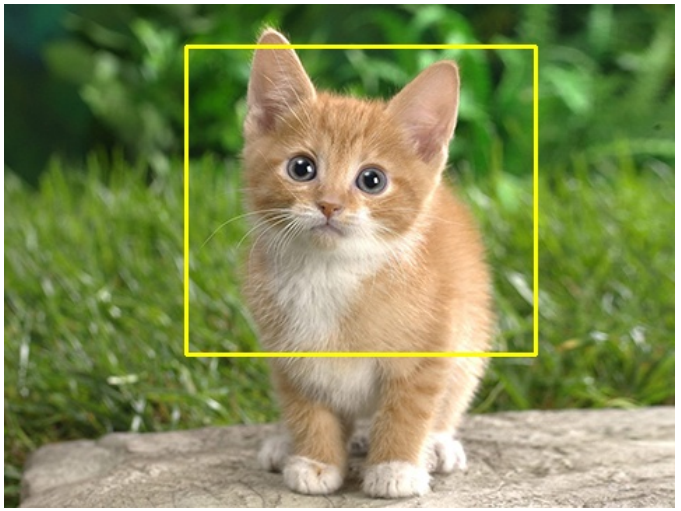


w, h, d, r, c parameter are pre-defined.

Non Maximum-Suppression

Functionality: If several boxes cover one same objects, they should be chosen or merged properly.

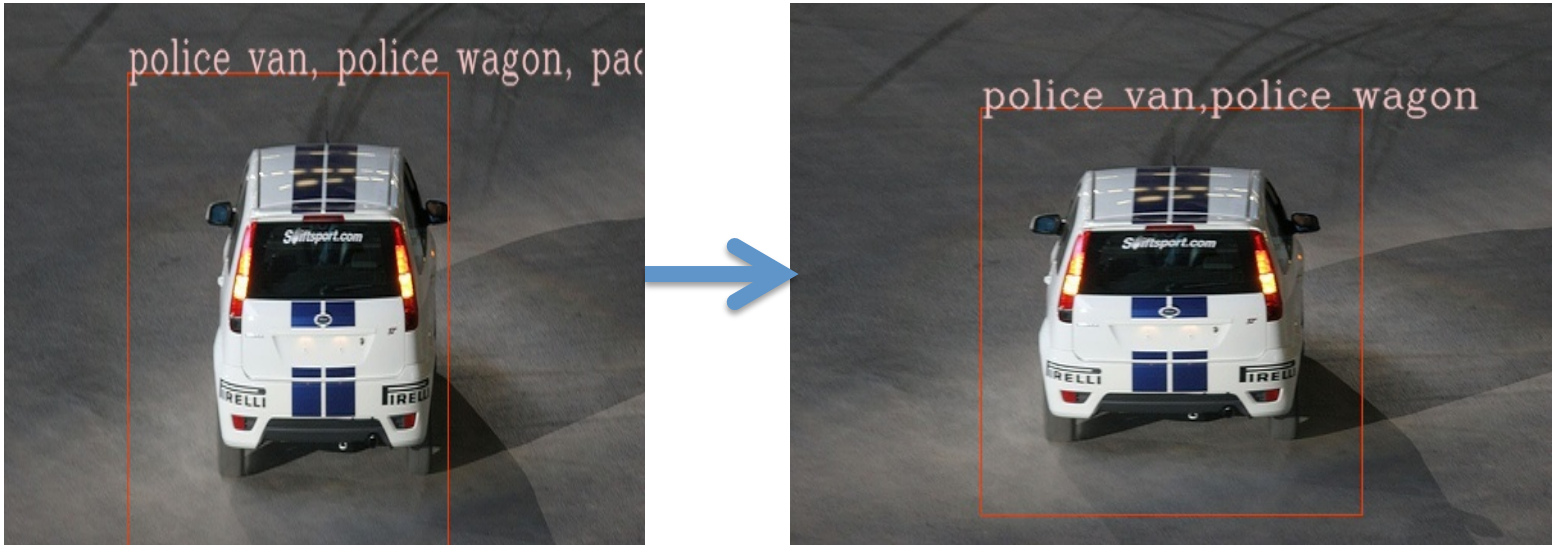




Basic idea: using overlap area and confidence score to determine which boxes that should be kept.

Bounding Box Regression

Functionality: Adjust bounding box that contains the object.



Bounding Box Regression

- Cast as learning problem.
- Input (Px,Py,Pw,Ph) four coordinates and features X on this batches
- Output: (Gx,Gy,Gw,Gh) adjusted four coordinates for final bounding box
- Structured learning problem.

Bounding Box Regression

- Constructed models (w1,w2,w3,w4) for adjusting four coordinates.
- Training:
  - proposal bounding boxes (Px,Py,Pw,Ph)
  - feature X
  - Ground truth bounding boxes(Gx,Gy,Gw,Gh)
- Need to do some invariant transformation

Bounding Box Regression

- Need to do some invariant transformation

$$Tx = (Gx - Px)/Px$$

$$Ty = (Gy - Py)/Py$$

$$Tw = \log(Gw/Pw)$$

$$Th = \log(Gh/Ph)$$

- Feature: deep learning feature X
- Target variable: Tx, Ty, Tw, Th

Bounding Box Regression

- Solving ridge regression

$$w_* = \underset{}{argmin} \sum_{i=1}^{n} (T_x^i - w^T X^i)^2 + \lambda ||w||^2$$

- Same for Ty, Tw,Th

- Once models for (x,y,w,h) are constructed, models can be used to adjust predictive bounding boxes.

Take home message

- Deep learning can be used for feature engineer
- Caffe is well designed deep learning library and you can write your own layer!