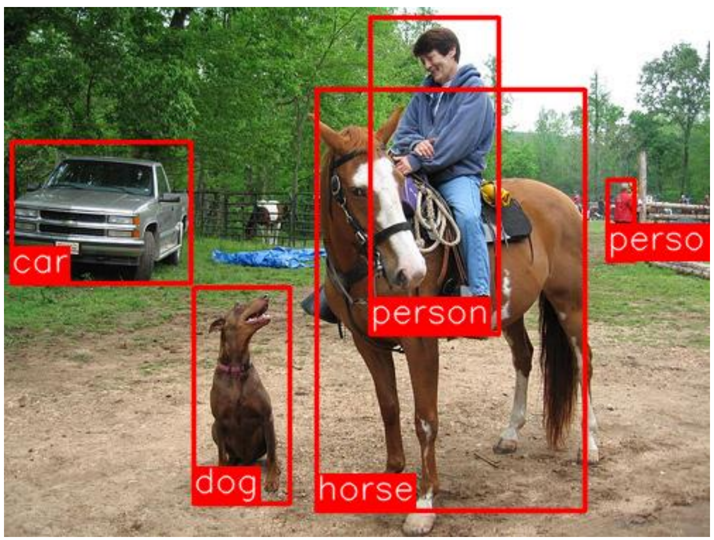
Object detection and segmentation

ISTD 50.035

Computer Vision

Object detection / segmentation

- Finding different objects in an image and classify them

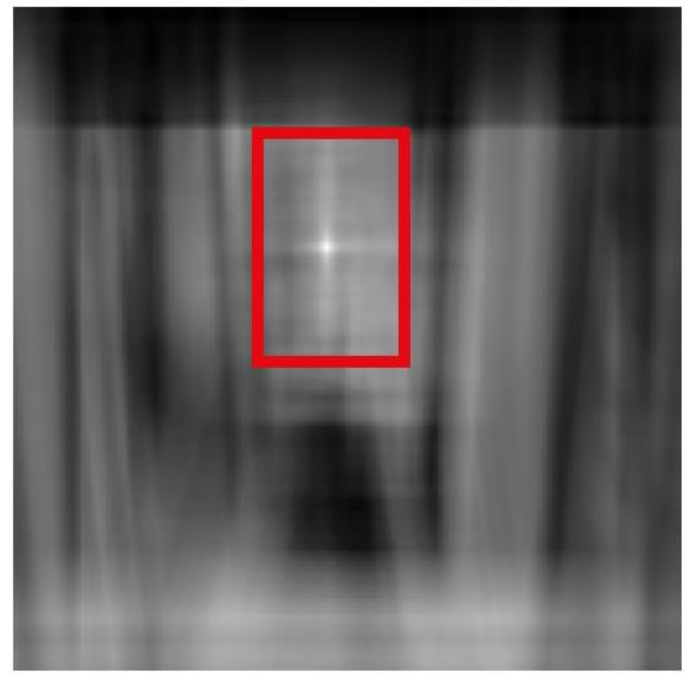


Object detection: challenge

This is a chair

Find the chair in this image

Output of normalized correlation



Template matching: correlation of template with the image

Object detection: challenge

Find the chair in this image

Object detection: challenge

view-
point

illumination

occlusion

clutter

intra-class variation

object
pose

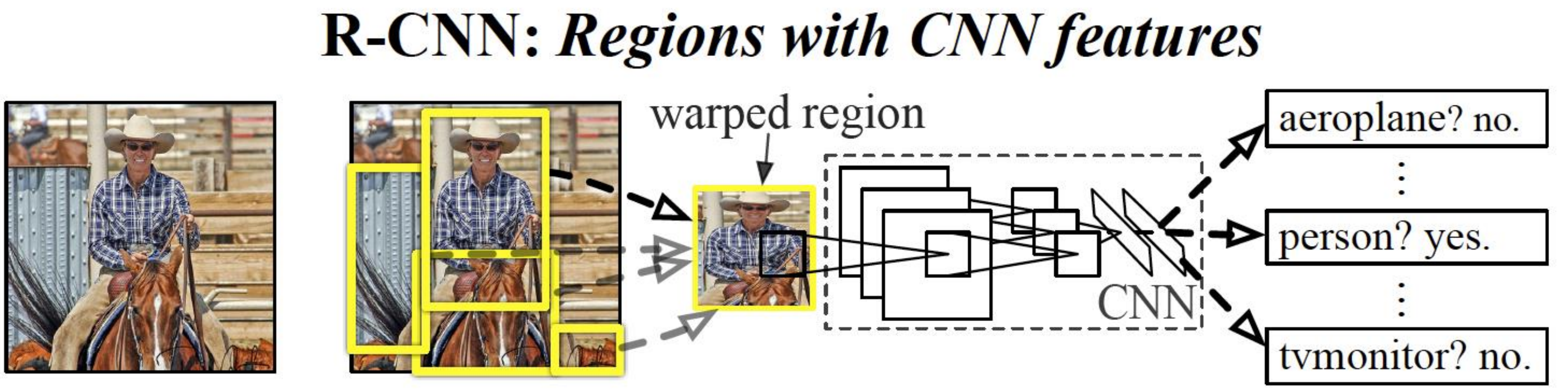
Approaches to be discussed

- Regional CNN (R-CNN)
- Fast R-CNN
- Faster R-CNN
- Mask R-CNN
- YOLO

Learning objectives

- Understand object detection problem
- Understand SOTA object detection approaches
- Understand the principles and ideas behind these approaches

R-CNN



Extract region proposal (~2K)

Compute CNN feature

Classify regions

[Ross Girshick et al.; 2014]

Propose many bounding boxes (region proposals), check if any of them correspond to an object by classifying them

Step 1: Extract region proposals

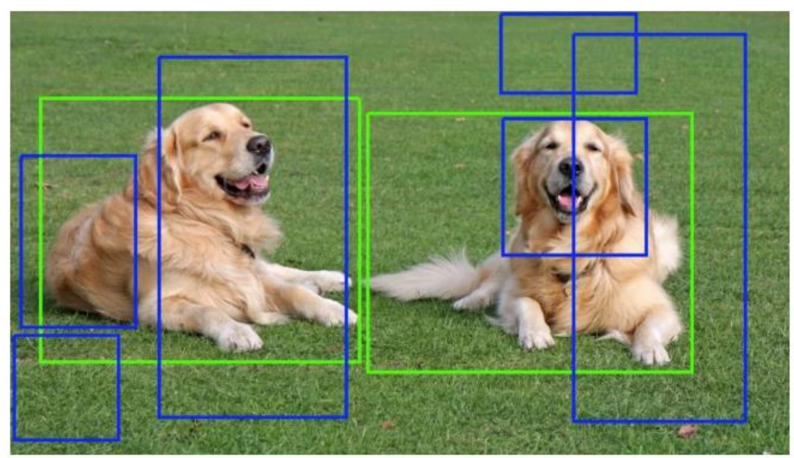
- Many techniques to generate category-independent region proposals
- Selective search [Uijlings et al.; 2013]

Step 1: Extract region proposals

- Sliding window approach
  - Slide a window over the image
  - Classify each image patch
  - Exhaustive search: search all possible locations, scale, aspect ratio
  - Computationally very expensive

Step 1: Extract region proposals

- Selective search [Uijlings et al.; 2013]
- Generate **region proposal**: Output bounding boxes corresponding to all patches that most likely contain objects
- Region proposals can be noisy, overlapped, and may not contain objects prefectly
- Some will be close to the actual objects -> use object classification to identify them
- Need high recall: RP includes regions with objects (true positives), even there are many false positives
- FP are rejected by classification



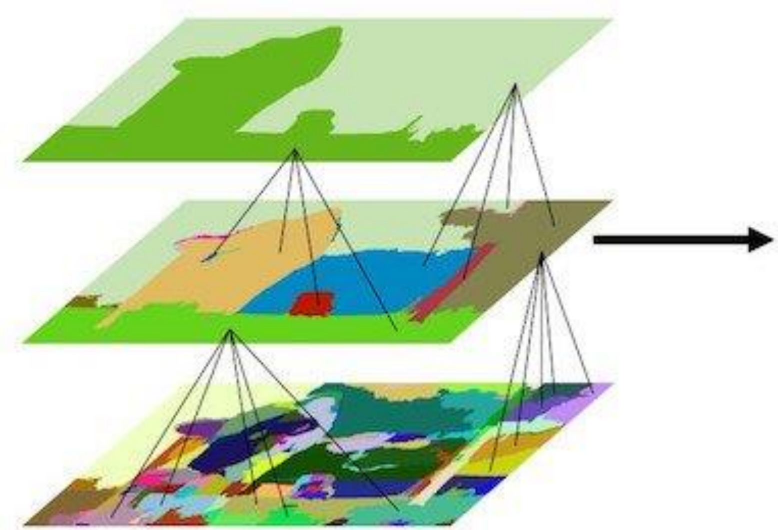
Blue Boxes: False Positives; Green Boxes: True Positives

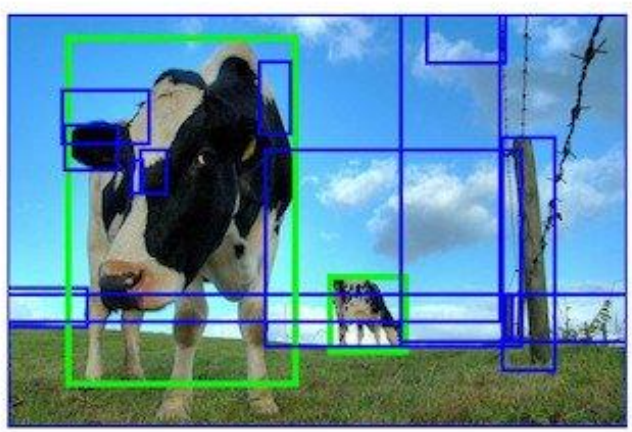
Step 1: Extract region proposals

- Selective search [Uijlings et al.; 2013]
- Hierarchical grouping of similar regions based on color, texture, size and shape
- SS starts by **over-segmentation** based on pixel intensity (graph based segmentation)

Step 1: Extract region proposals

- Selective search [Uijlings et al.; 2013]
  - A) add all bounding boxes corresponding to segmented parts to the list of region proposals
  - B) Merge adjacent segments based on similarity
  - C) Go to (A)
- Bottom up approach: create RP from smaller segments to larger segments





Step 1: Extract region proposals

- Similarity on color, texture, size and shape

- Color similarity based on histogram intersection
  - Color histogram of 25 bins => 25x3 = 75-dim color descriptor
  - Color similarity of two regions:

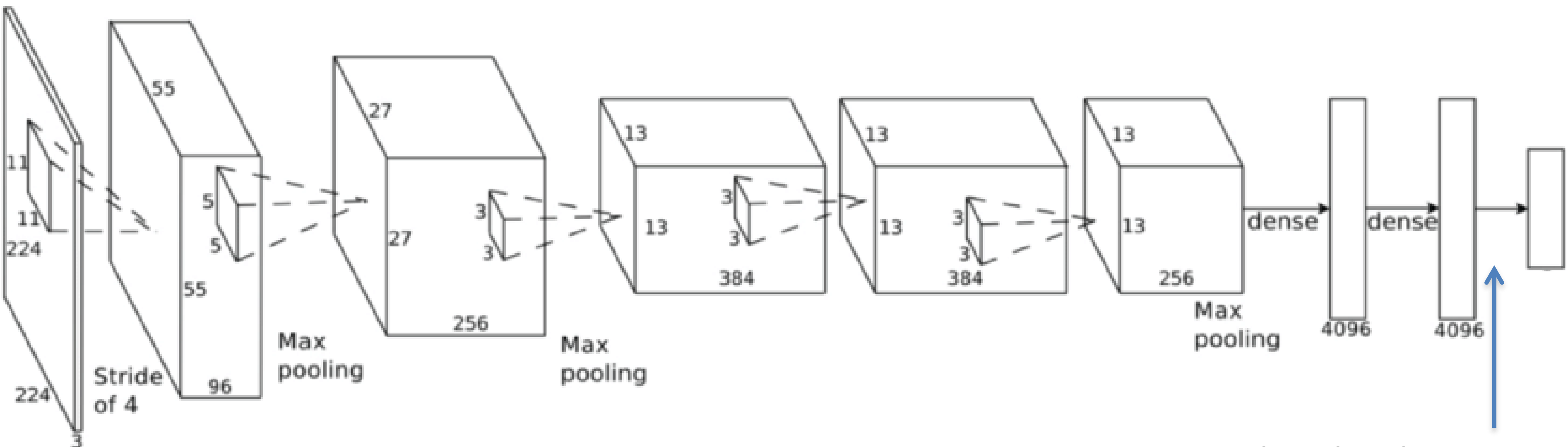
$$s_{color}(r_i, r_j) = \sum_{k=1}^{n} min(c_i^k, c_j^k)$$

Histogram value for the k-th bin of region i

Step 2: Compute CNN features

- Challenge: Labeled data in detection is not sufficient to train a large CNN
- Discriminative pre-training on a large auxiliary dataset
  - Imagenet
  - Image level annotation
- Domain-specific fine-tuning
  - Warped proposal windows

Step 2: Compute CNN features

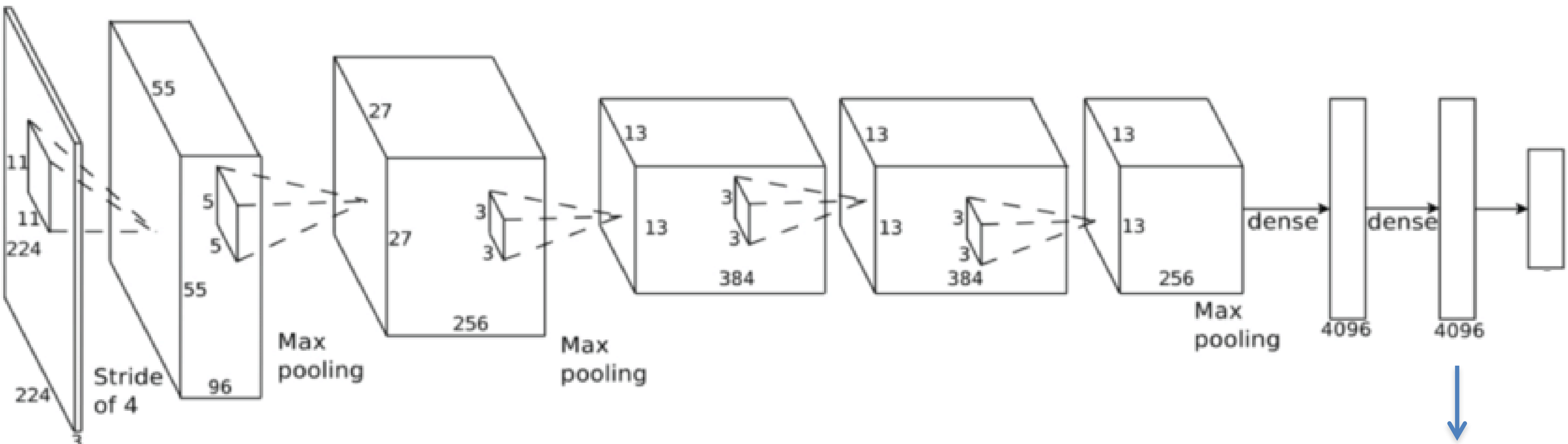


Input: a region proposal

Replace last layer to produce N+1 outputs

- Domain specific fine-tuning:
  - Replace specific 1000-way classification layer with a randomly initialized (N+1)-way classification layer
    - N is the number of object classes, plus 1 for background
  - Smaller learning rate in stochastic gradient descent so that not to significantly deviate from initialization

Step 2: Compute CNN features



Input: a region proposal

4096-dim activation as feature

Feature extraction

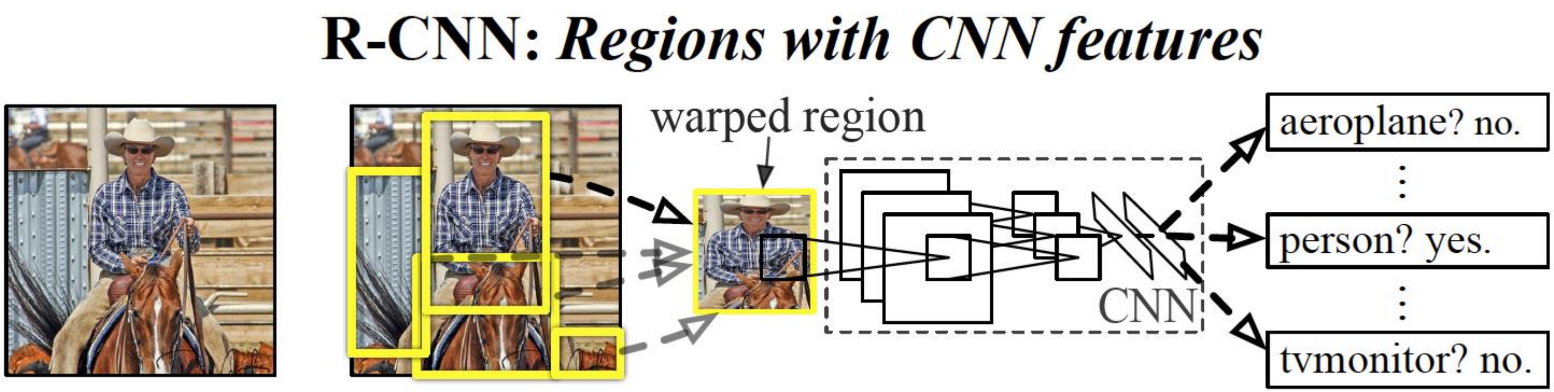
Softmax classifier

- Extract 4096-dim feature vector for each region proposal
- Robust, discriminative and low-dimensional representation of the input region proposal
- View the network as feature extraction + classification

Step 3: Classify regions

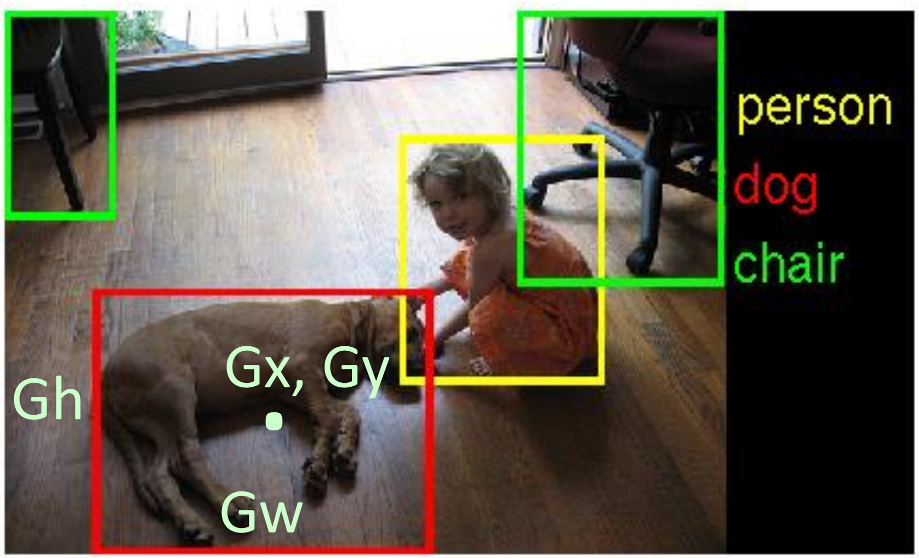
- One linear binary SVM per class

- Why not use the 21-way softmax classifier?



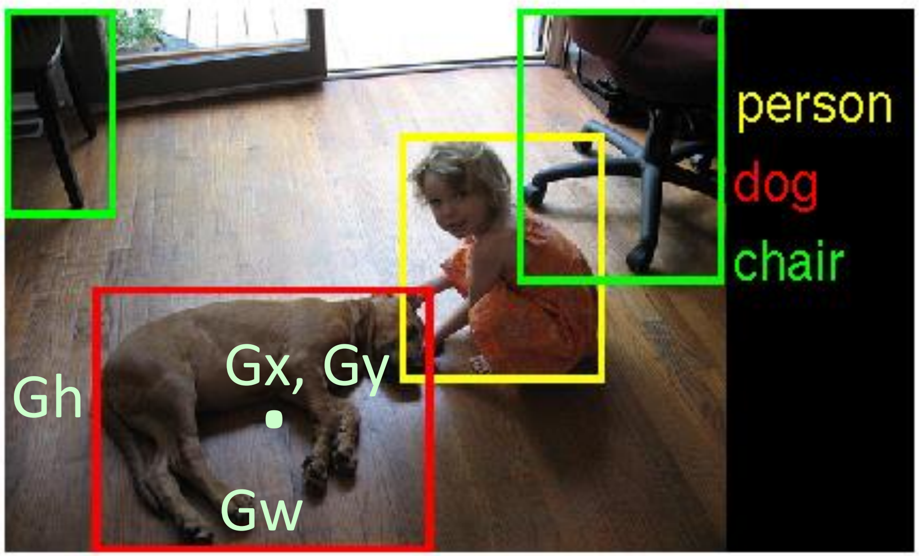
Bounding-box regression

- The original bounding-box from selective search may not be very accurate
- Predict a new bounding box: 4 parameters



Bounding-box regression

- Predict a new bounding box: 4 parameters
- Given Px, Py, Pw, Ph, learn to predict Gx, Gy, Gw, Gh



Scale invariant translation

$$\hat{G}_x = P_w d_x(P) + P_x$$
$$\hat{G}_y = P_h d_y(P) + P_y$$
$$\hat{G}_w = P_w \exp(d_w(P))$$
$$\hat{G}_h = P_h \exp(d_h(P)).$$

scaling

Bounding-box regression

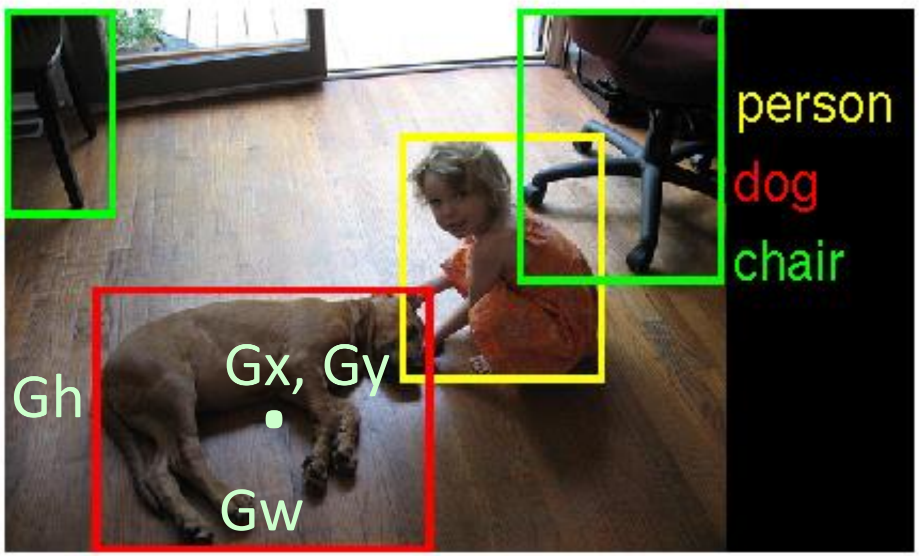
- Assume each function is a linear function of the the pool5 features of the proposal P: $\Phi_5(P)$

Regression target

$$d_\star(P) = \mathbf{w}_\star^{\mathrm{T}} \boldsymbol{\phi}_5(P)$$

$$\mathbf{w}_\star = \underset{\hat{\mathbf{w}}_\star}{\operatorname{argmin}} \sum_i^N (t_\star^i - \hat{\mathbf{w}}_\star^{\mathrm{T}} \boldsymbol{\phi}_5(P^i))^2 + \lambda \left\| \hat{\mathbf{w}}_\star \right\|^2$$

Prediction model



$$\hat{G}_x = P_w d_x(P) + P_x$$

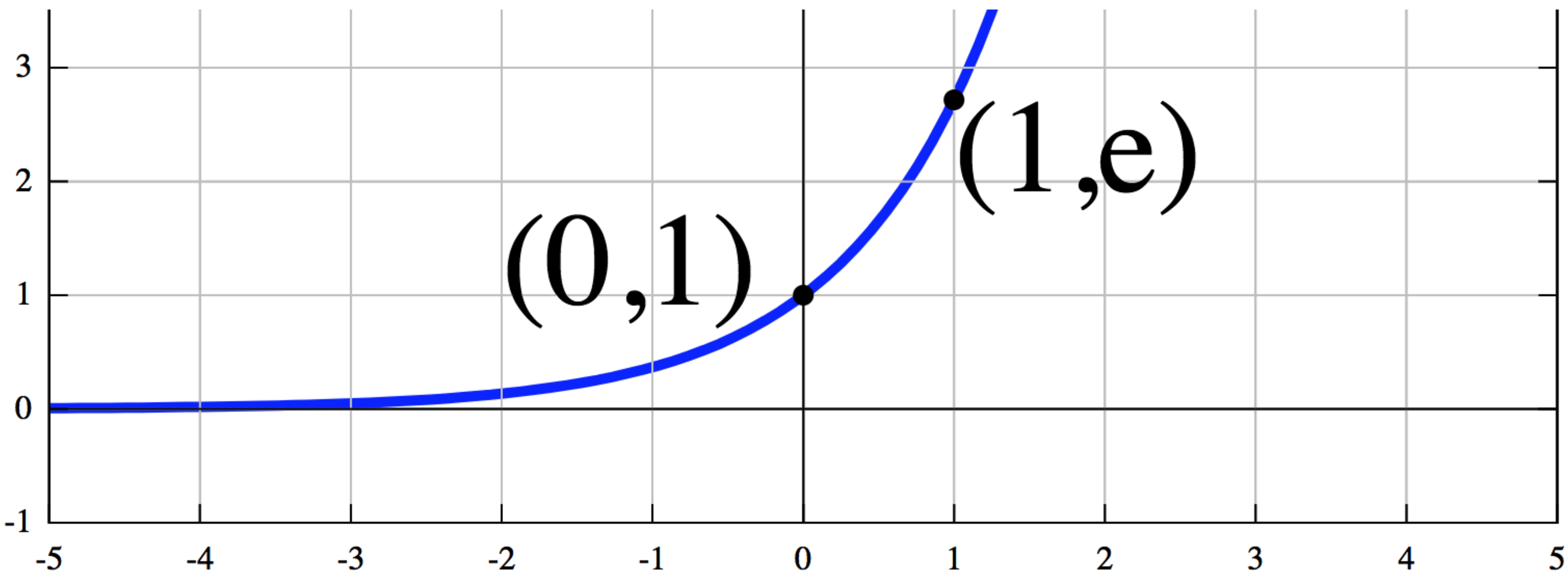
$$\hat{G}_y = P_h d_y(P) + P_y$$

$$\hat{G}_w = P_w \exp(d_w(P))$$

$$\hat{G}_h = P_h \exp(d_h(P)).$$

Bounding-box regression

- Why exp(.)?



Region to regress