CMPT 733-G200 Practices for Visual Computing II

Ali Mahdavi Amiri

Object Detection

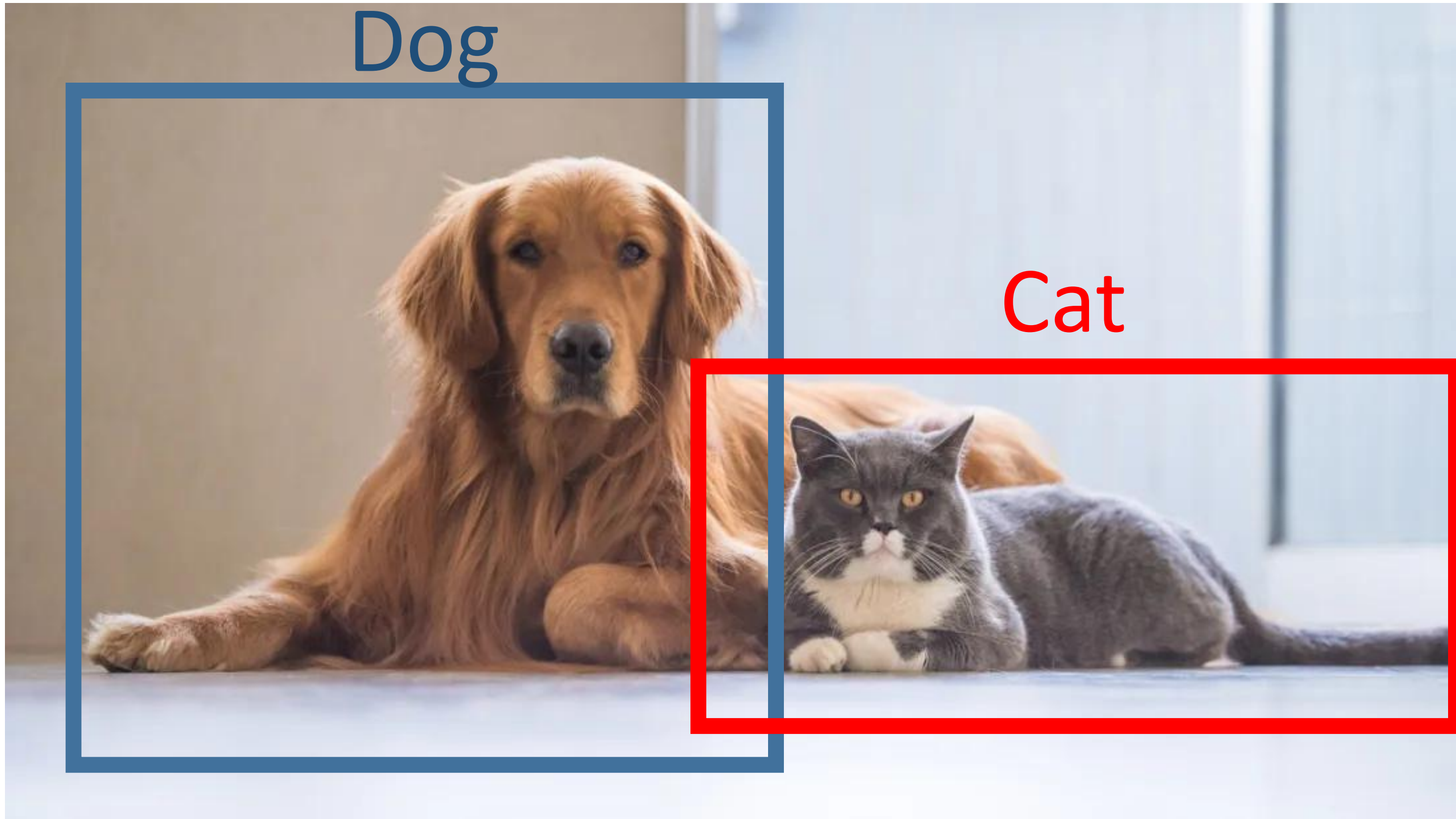
- What is Object Detection?

Object Detection

- Input: an image

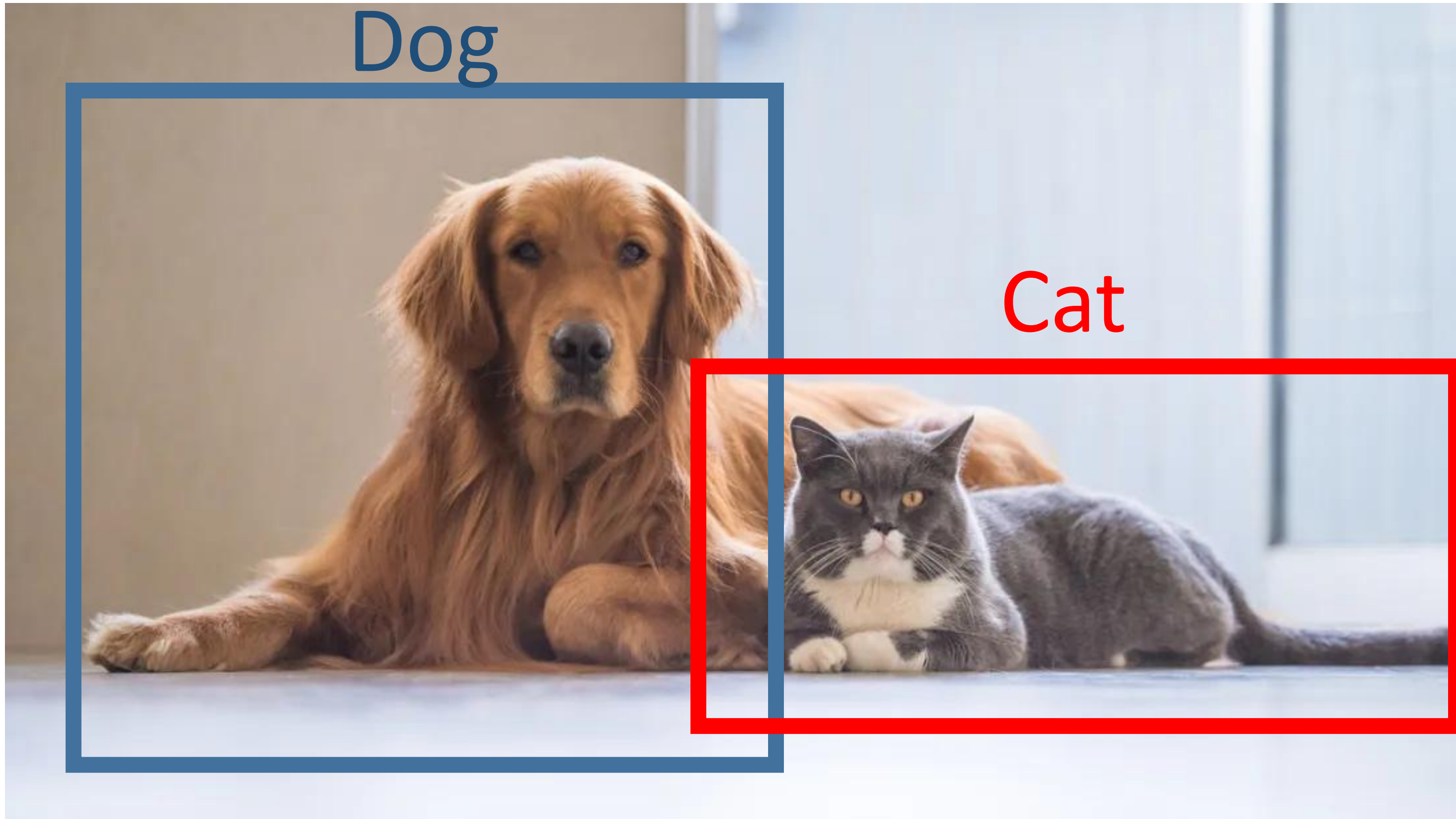
Object Detection

- Input: an image
- Output: bounding box with the right class



Object Detection

- Why do we need object detection?

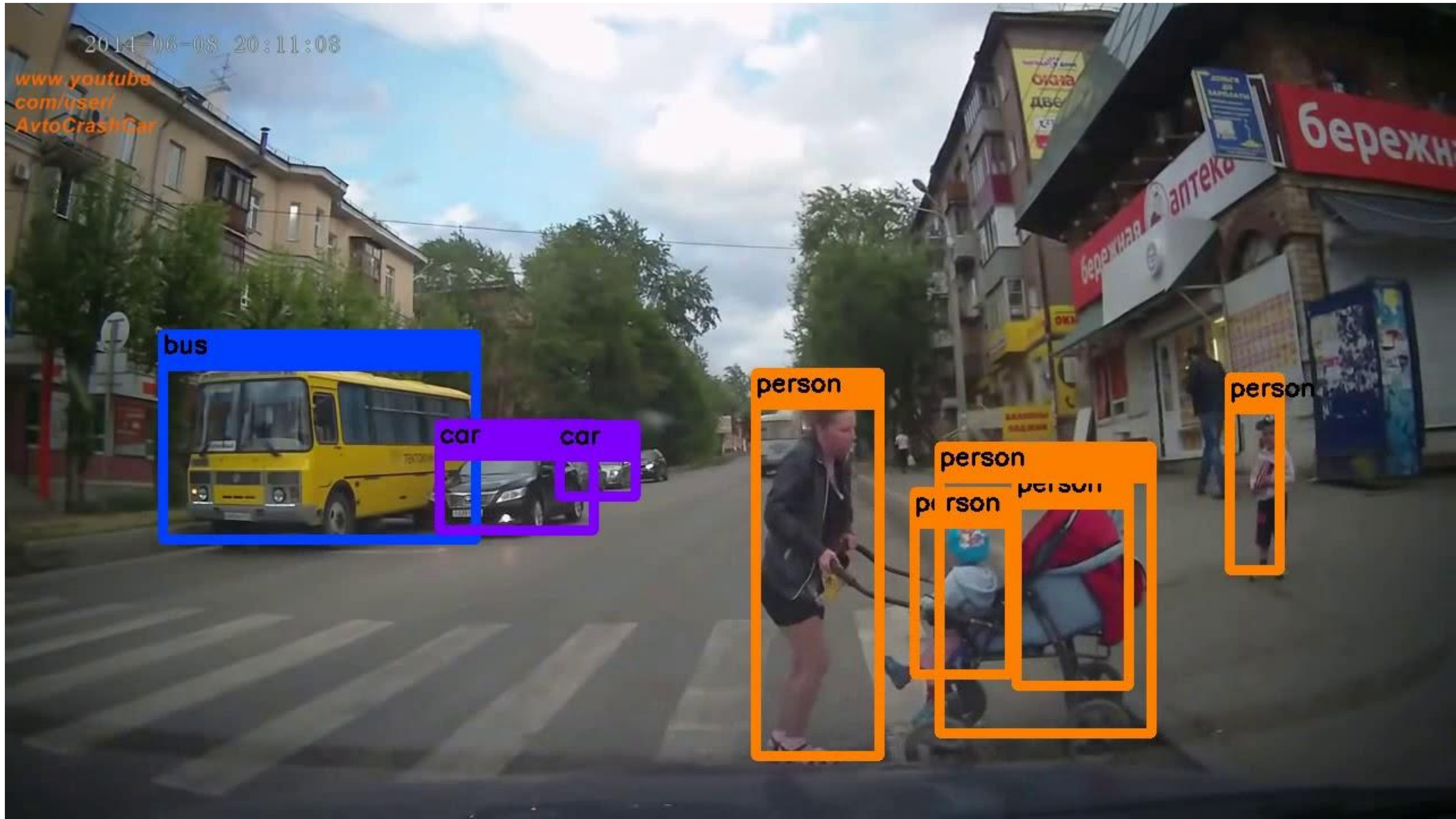


Object Detection Applications

- Self-driving Cars

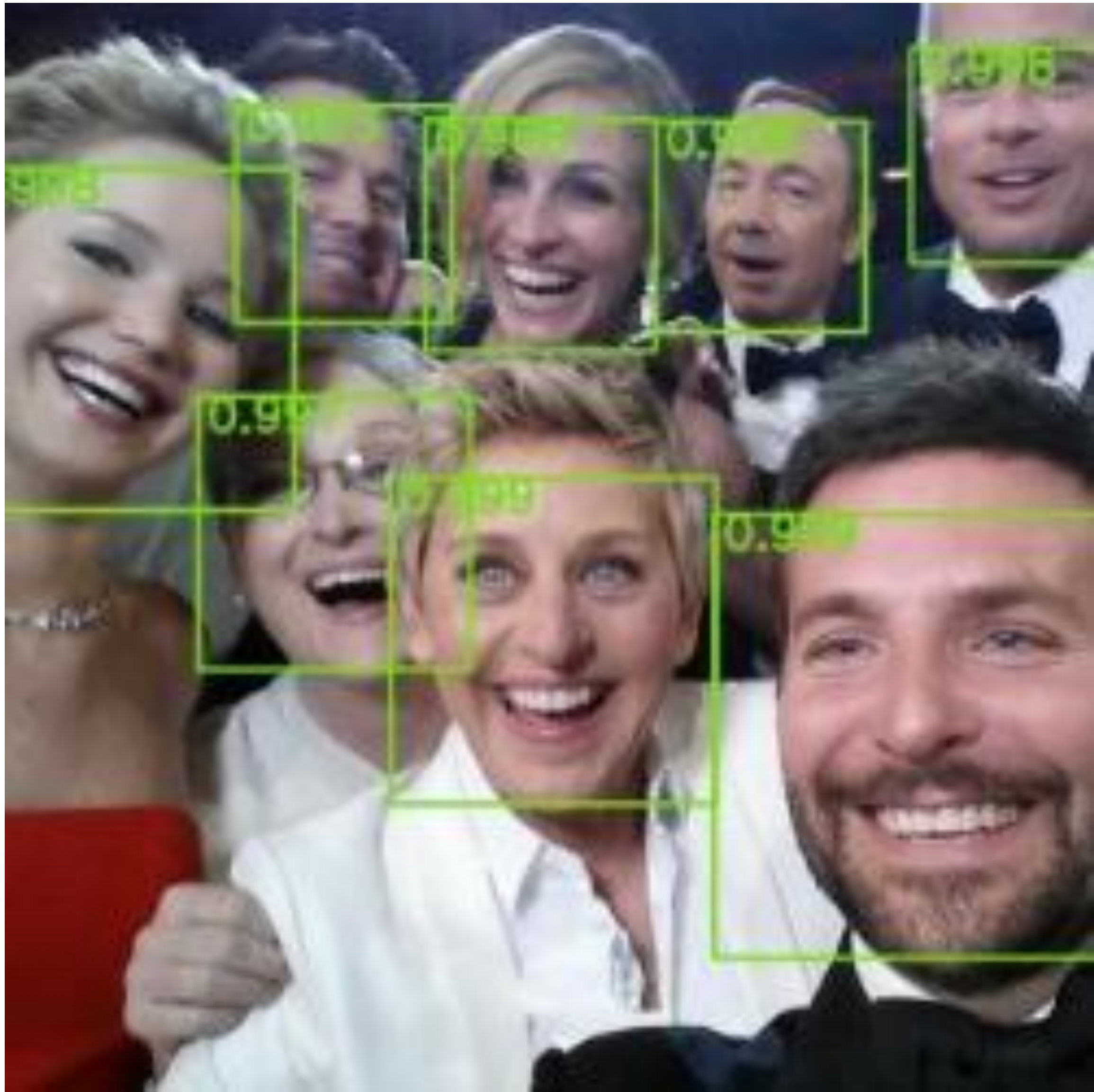
Object Detection Applications

- Self-driving Cars
  - It needs to detect obstacles to avoid them.



Object Detection Applications

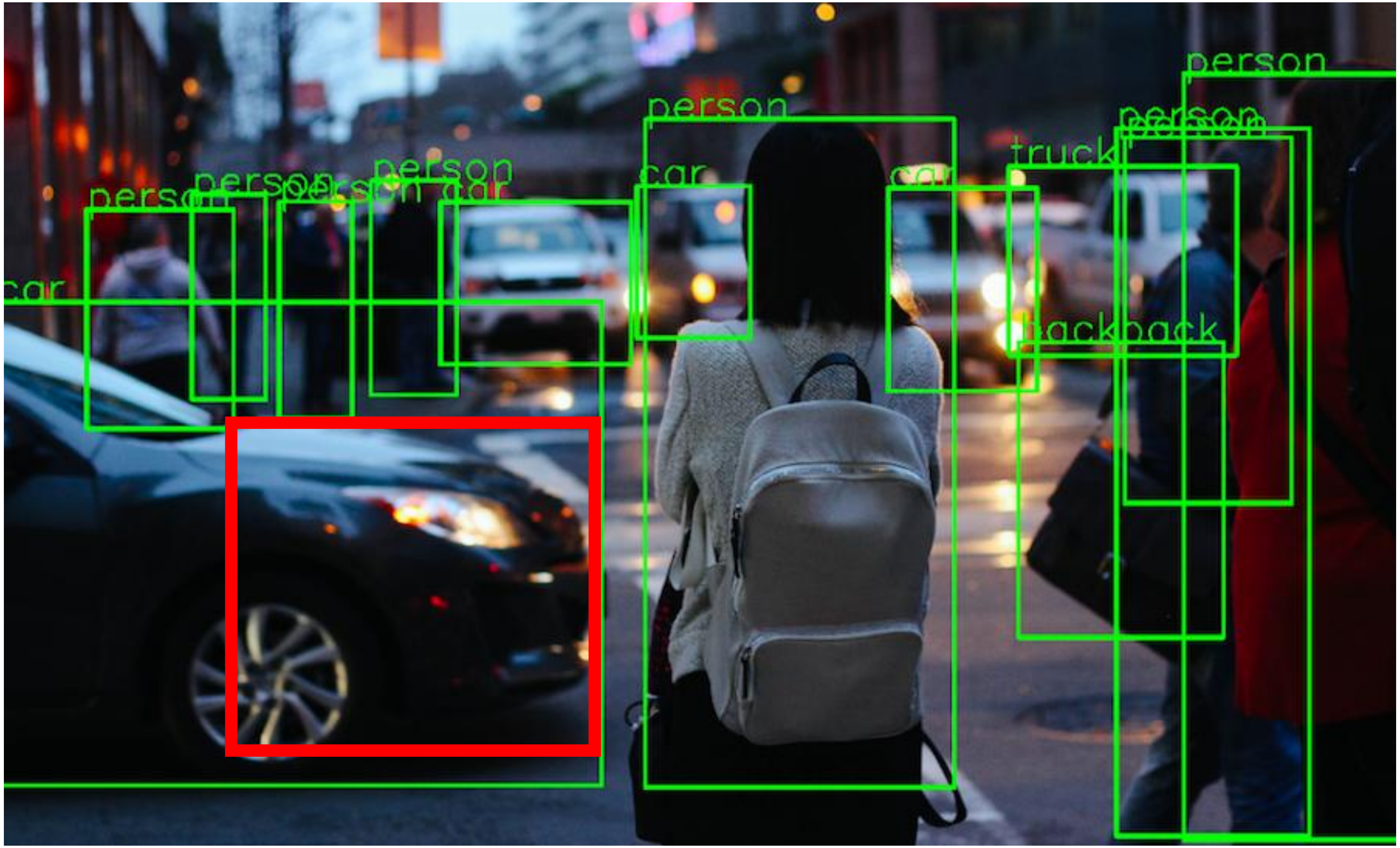
- Face Detection



Object Detection Applications

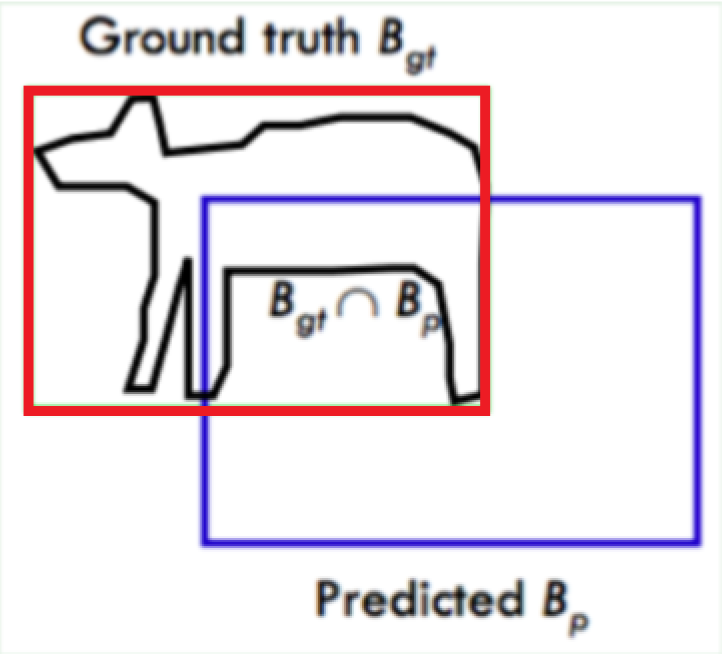
- People Counting

How to measure?



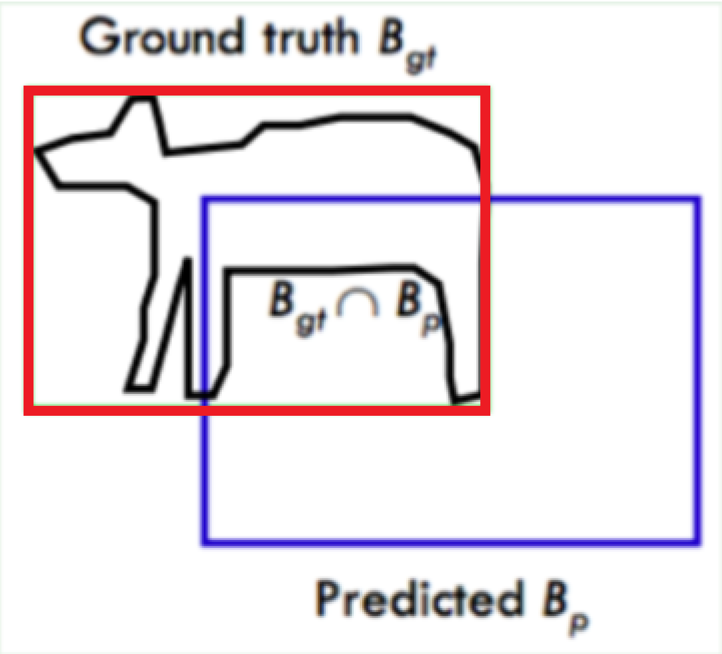
How to measure?

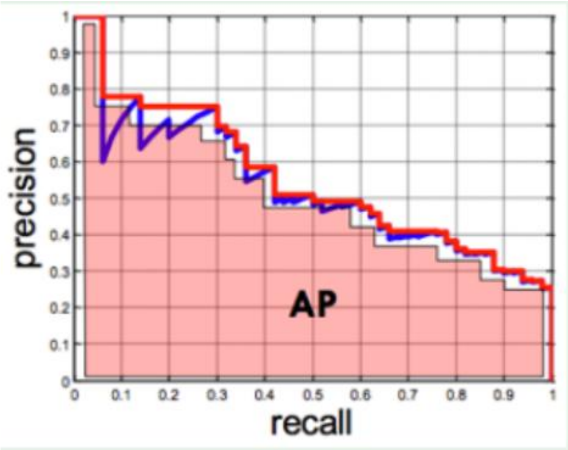
- We have two bounding boxes, how can we measure if it is a good detection?



How to measure?

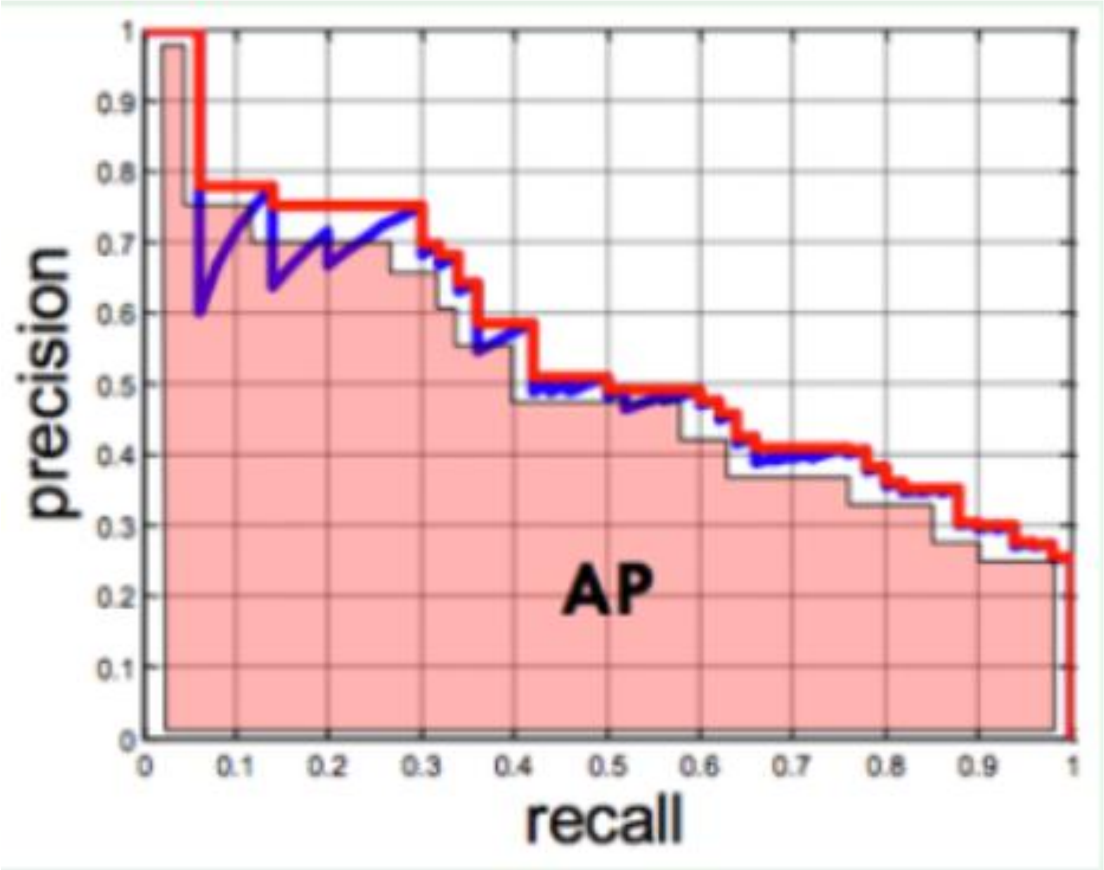
- We use a measurement metric called Average Precision.





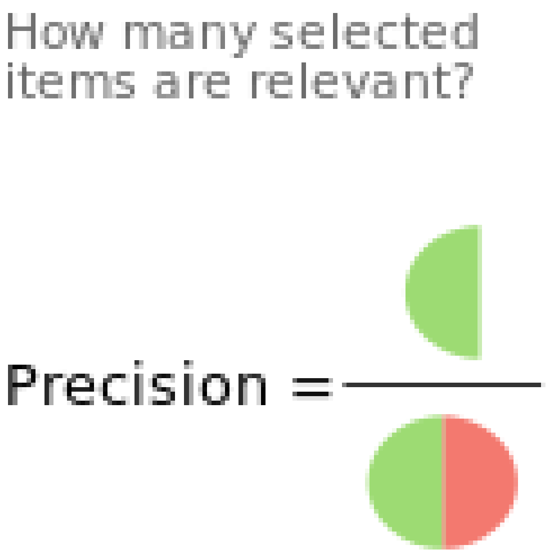
How to measure?

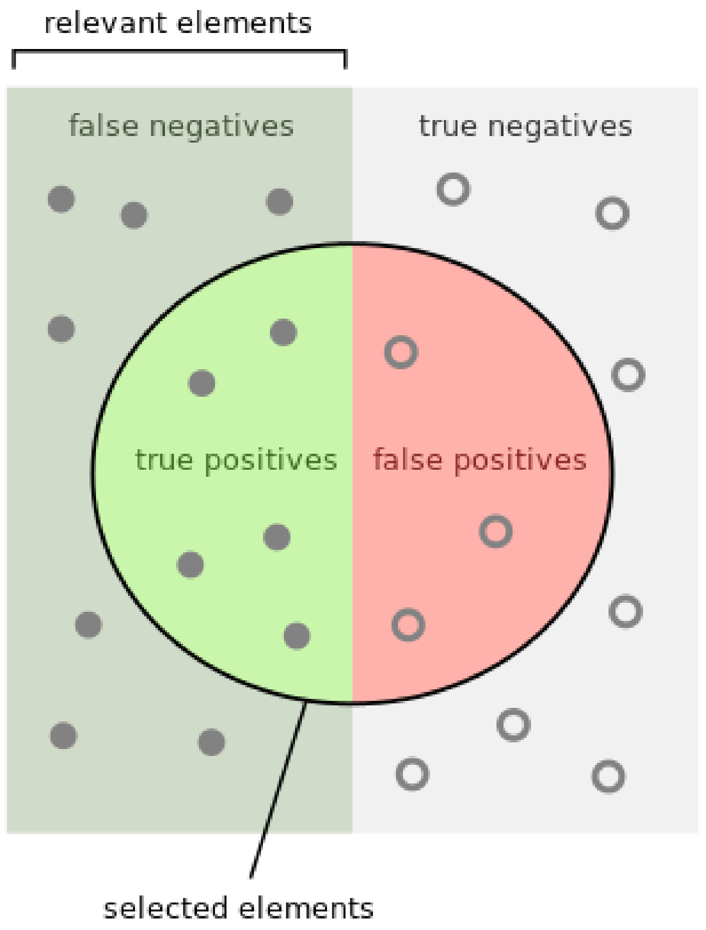
- Average precision is the average precision value for recall value from 0 to 1.



Precision

- Precision measures how accurate is your prediction. The percentage of your positive predictions that are correct.





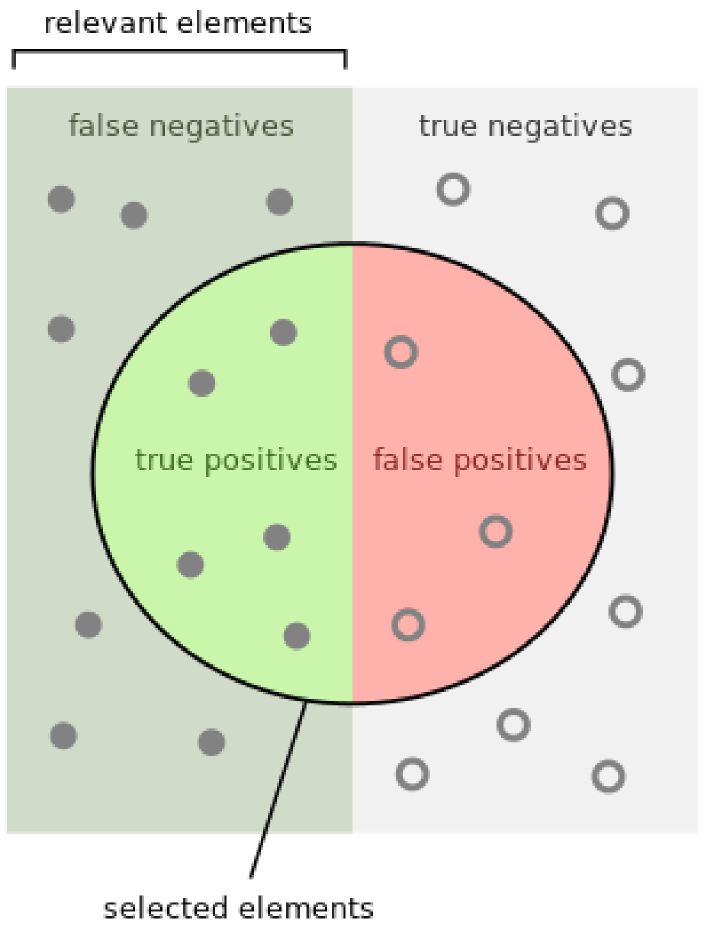
Precision

- Mathematical Formula

$$Precision = \frac{TP}{TP + FP}$$

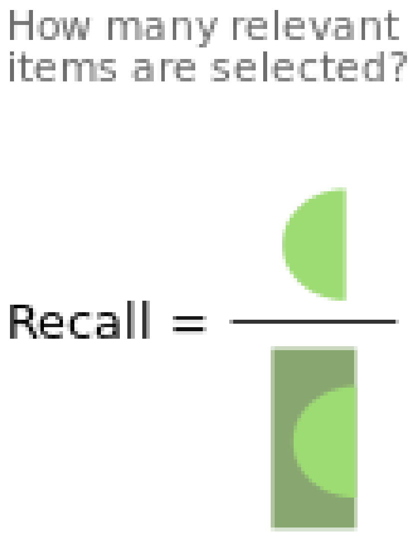
TP = True positive

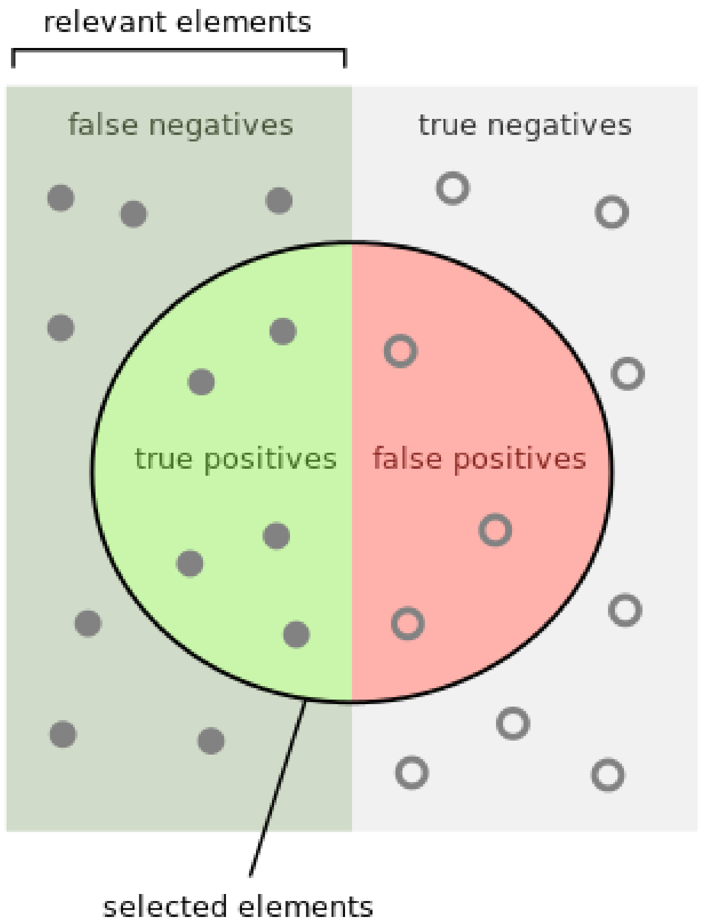
FP = False positive



Recall

- Recall measures how well you find all the positives in your data.





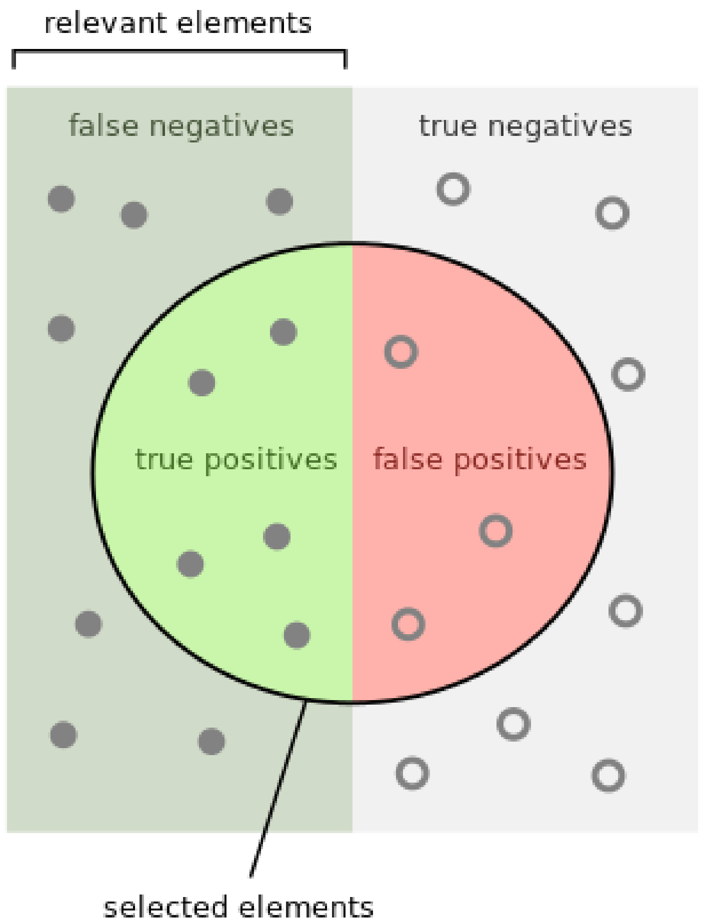
Recall

- Mathematical Formula

$$Recall = \frac{TP}{TP + FN}$$

$$TP = \text{True positive}$$

$$FN = \text{False negative}$$



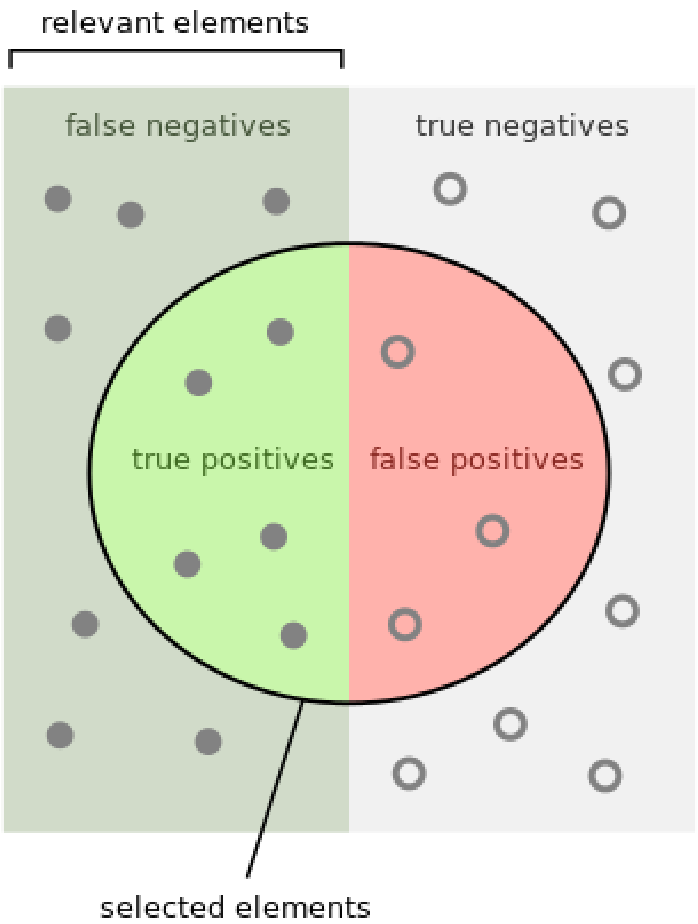
Recall

- In fact recall is TP over all the GT.

$$Recall = \frac{TP}{TP + FN}$$

$$TP = \text{True positive}$$

$$FN = \text{False negative}$$



How to measure?

- So what?? How do you want to relate it to Object Detection?

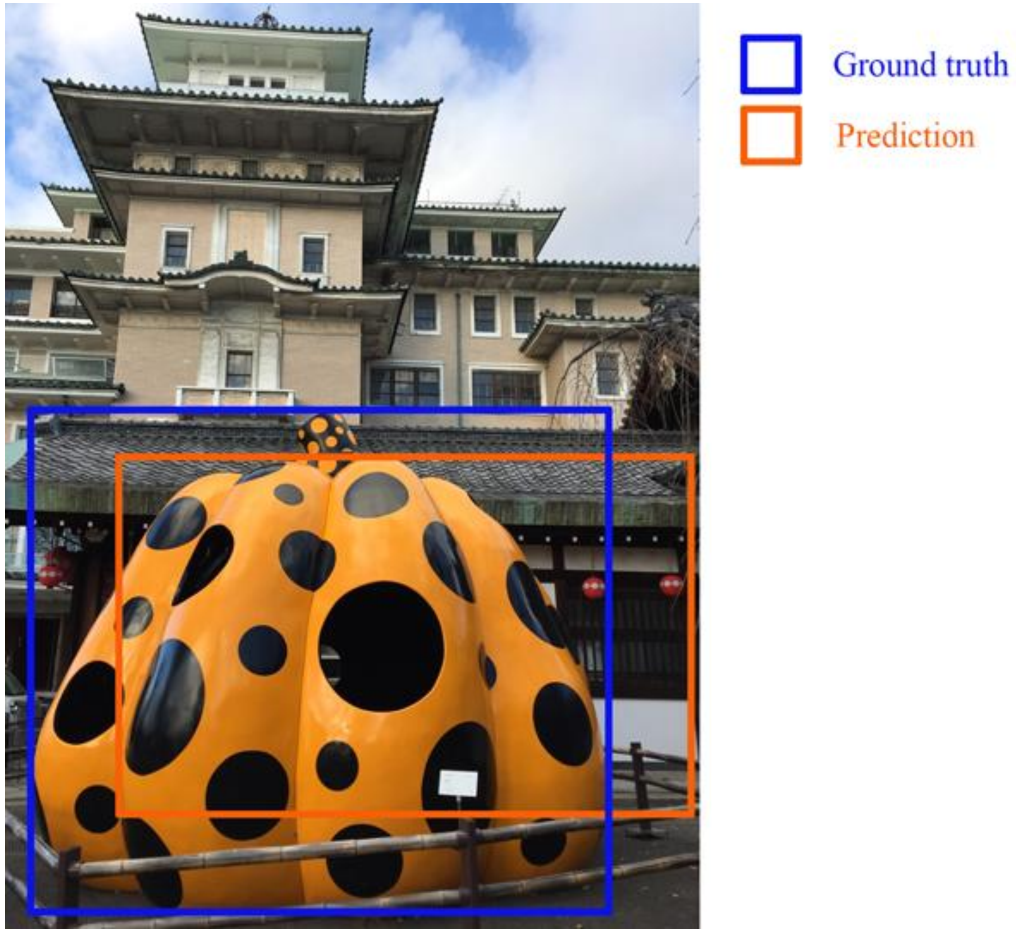
How to measure?

- So what?? How do you want to relate it to Object Detection?

- Wait, I still need to define a few things.

How to measure?

- In object detection, you have a ground truth box and a prediction box.



https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173

How to measure?

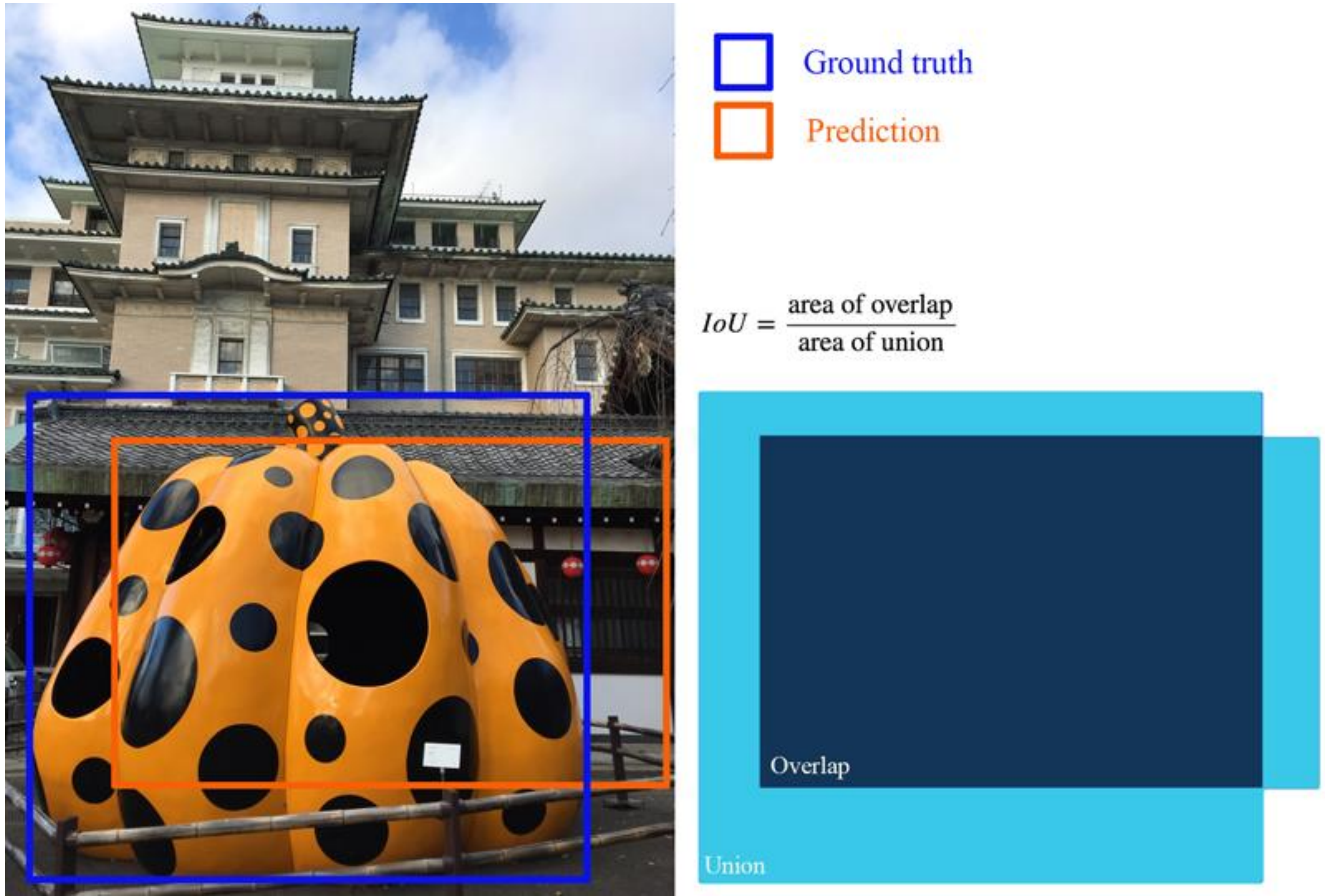
- In object detection, you have a ground truth box and a prediction box.

- We define **IoU** (**I**ntersection **o**ver **U**nion)

$$IoU = \frac{\text{area of overlap}}{\text{area of union}}$$

IoU

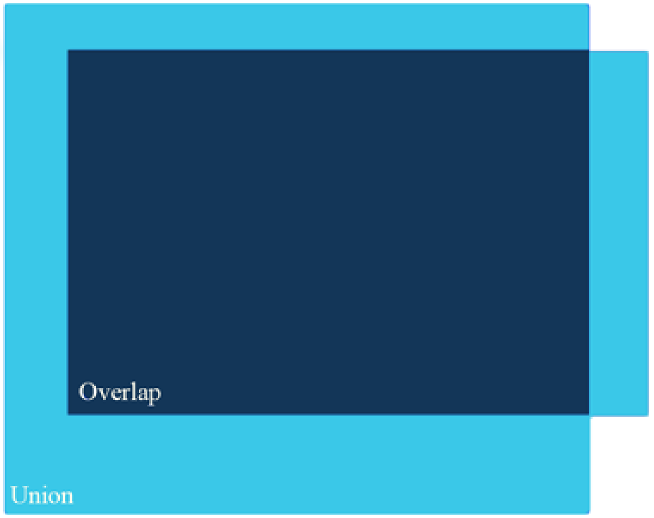
- We use it to measure how much our predicted boundary overlaps with the ground truth.



IoU

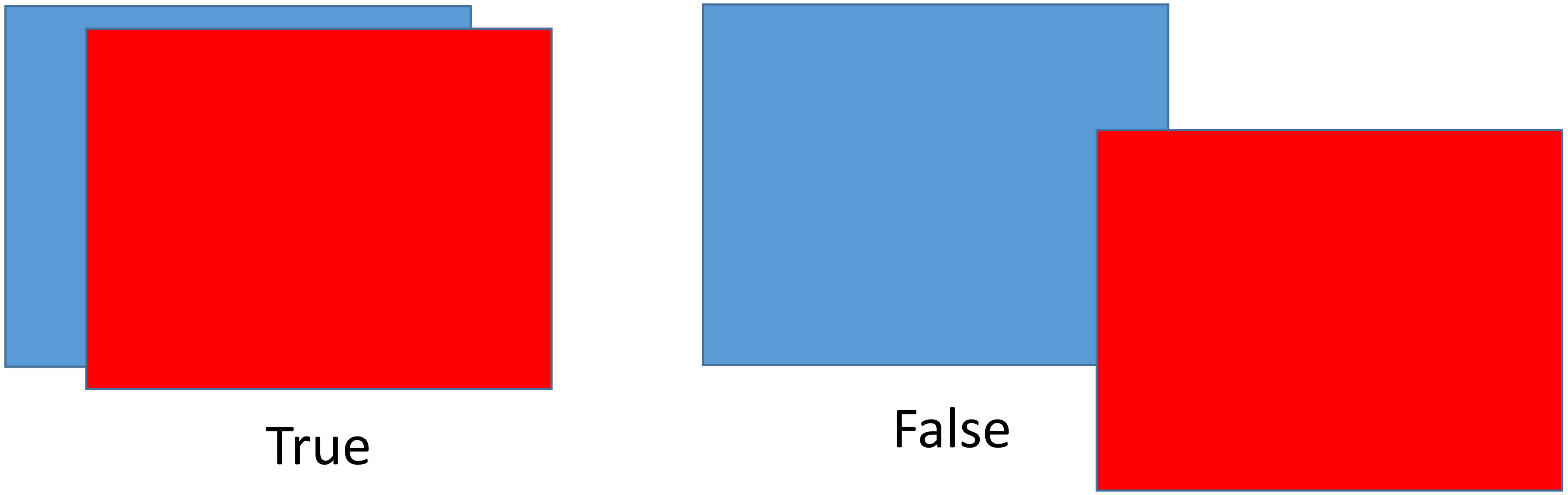
- Mathematical Formula

$$IOU = \frac{\text{area of overlap}}{\text{area of union}} =$$



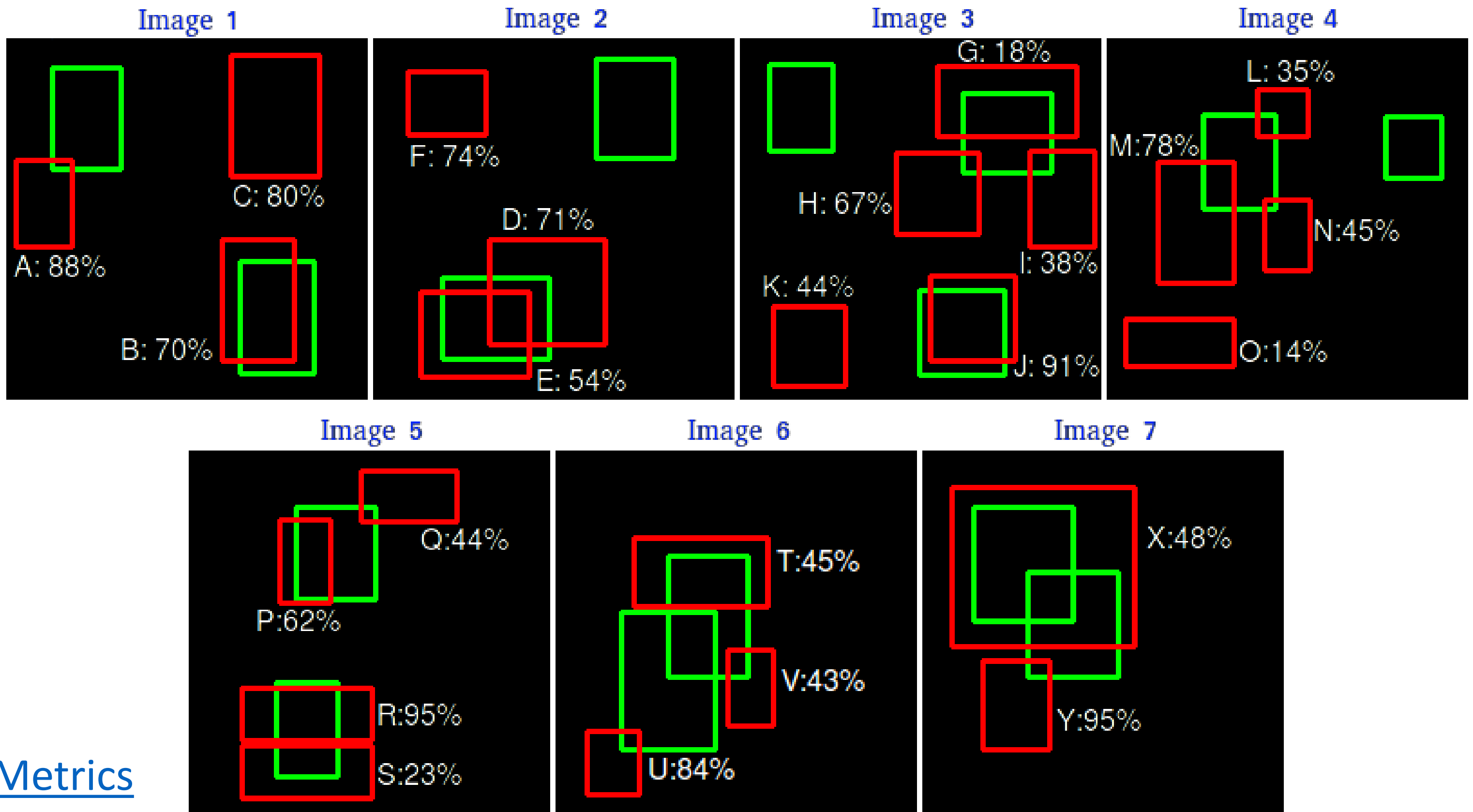
IoU

- We predefine an IoU threshold (say 0.5) in classifying whether the prediction is a true positive or a false positive.



Example

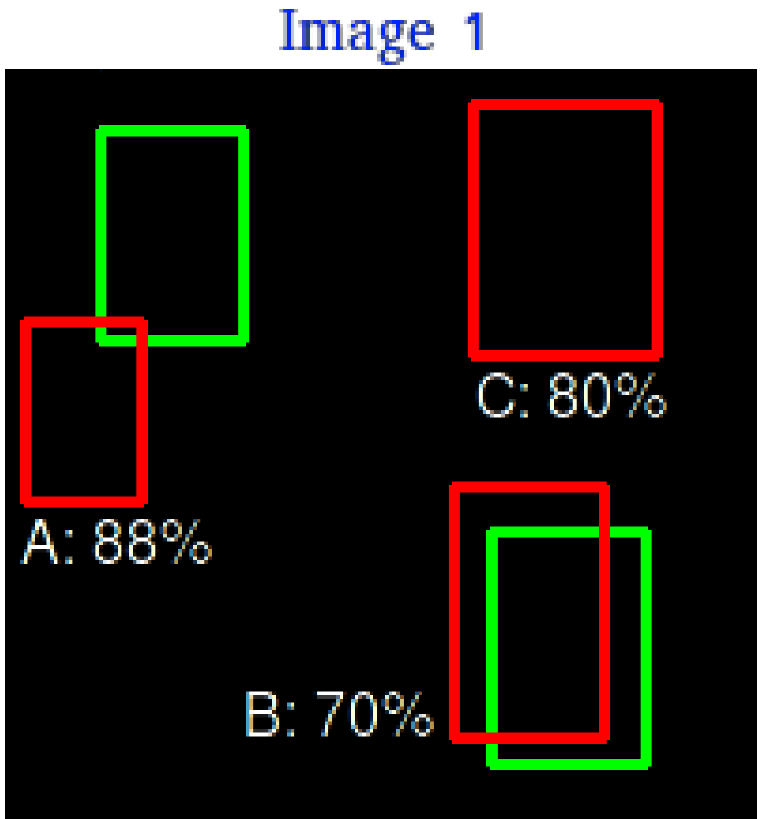
- 7 images with 15 ground truth and 24 detected objects.



https://github.com/rafaelpadilla/Object-Detection-Metrics

Example

- 7 images with 15 ground truth and 24 detected objects.
- Each detected object has a confidence level and is identified by a letter (A,B,...,Y).



https://github.com/rafaelpadilla/Object-Detection-Metrics

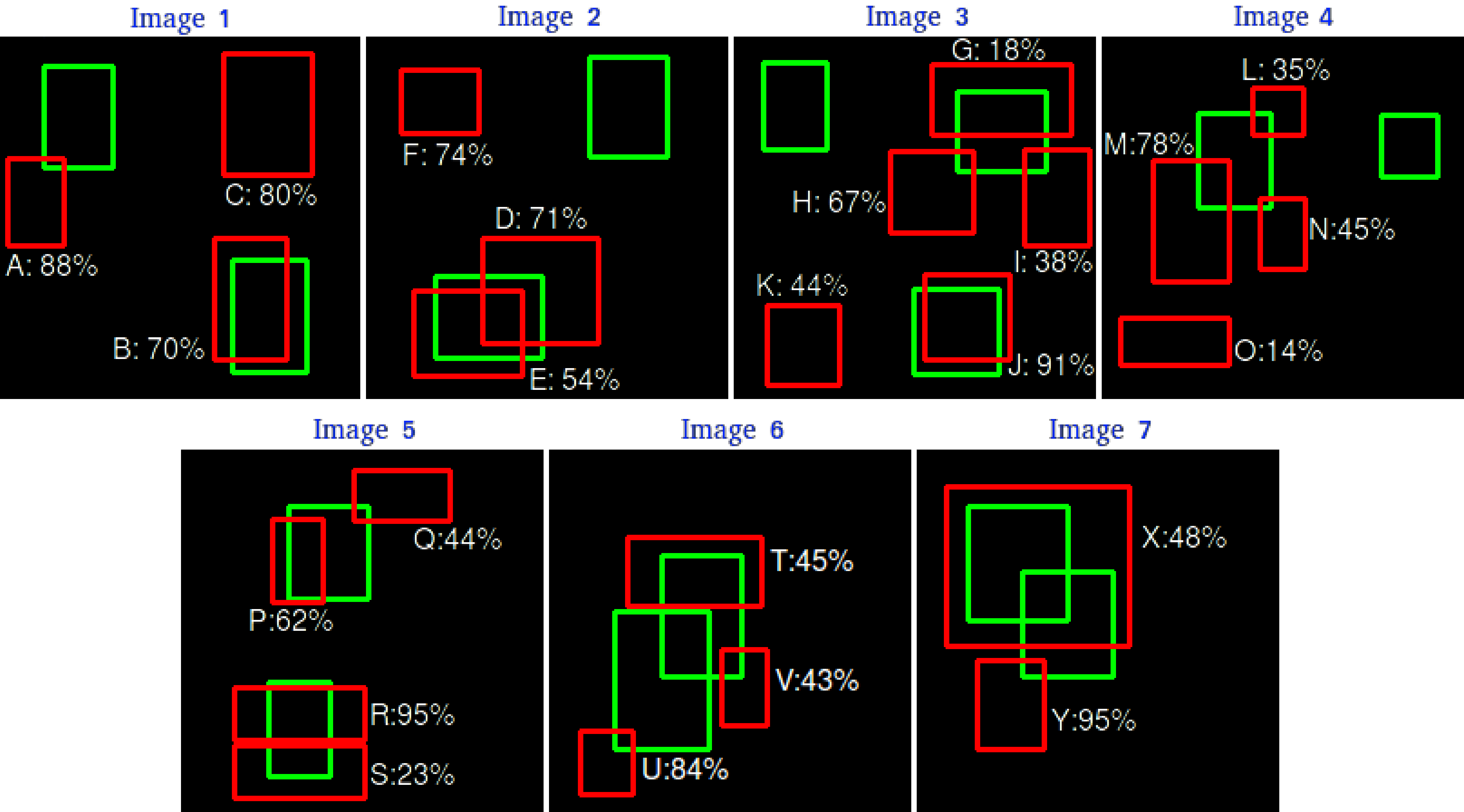


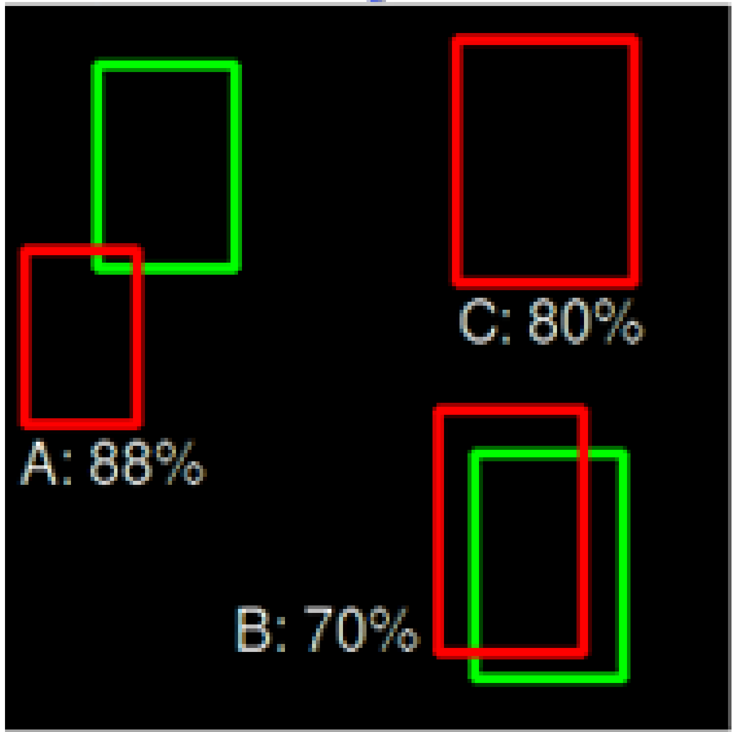
Image 1
A: 88%
B: 70%
C: 80%
Image 2
F: 74%
D: 71%
E: 54%
Image 3
G: 18%
H: 67%
I: 38%
K: 44%
J: 91%
Image 4
L: 35%
M:78%
N:45%
O:14%
Image 5
Q:44%
P:62%
R:95%
S:23%
Image 6
T:45%
V:43%
U:84%
Image 7
X:48%
Y:95%

Example

- We can make a table of this image along with their confidence scores.

Images	Detections	Confidences	TP or FP
Image 1	A	88%	FP
Image 1	B	70%	TP
Image 1	C	80%	FP
Image 2	D	71%	FP
Image 2	E	54%	TP
Image 2	F	74%	FP
Image 3	G	18%	TP
Image 3	H	67%	FP

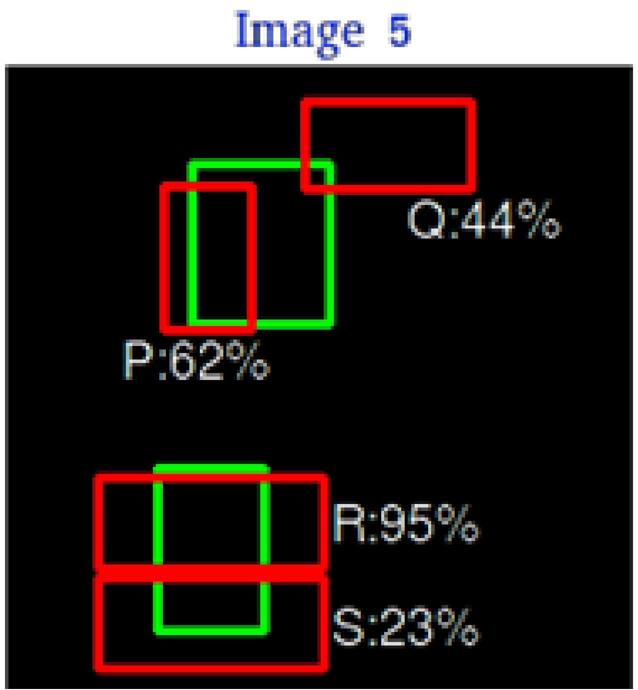
Image 1



Example

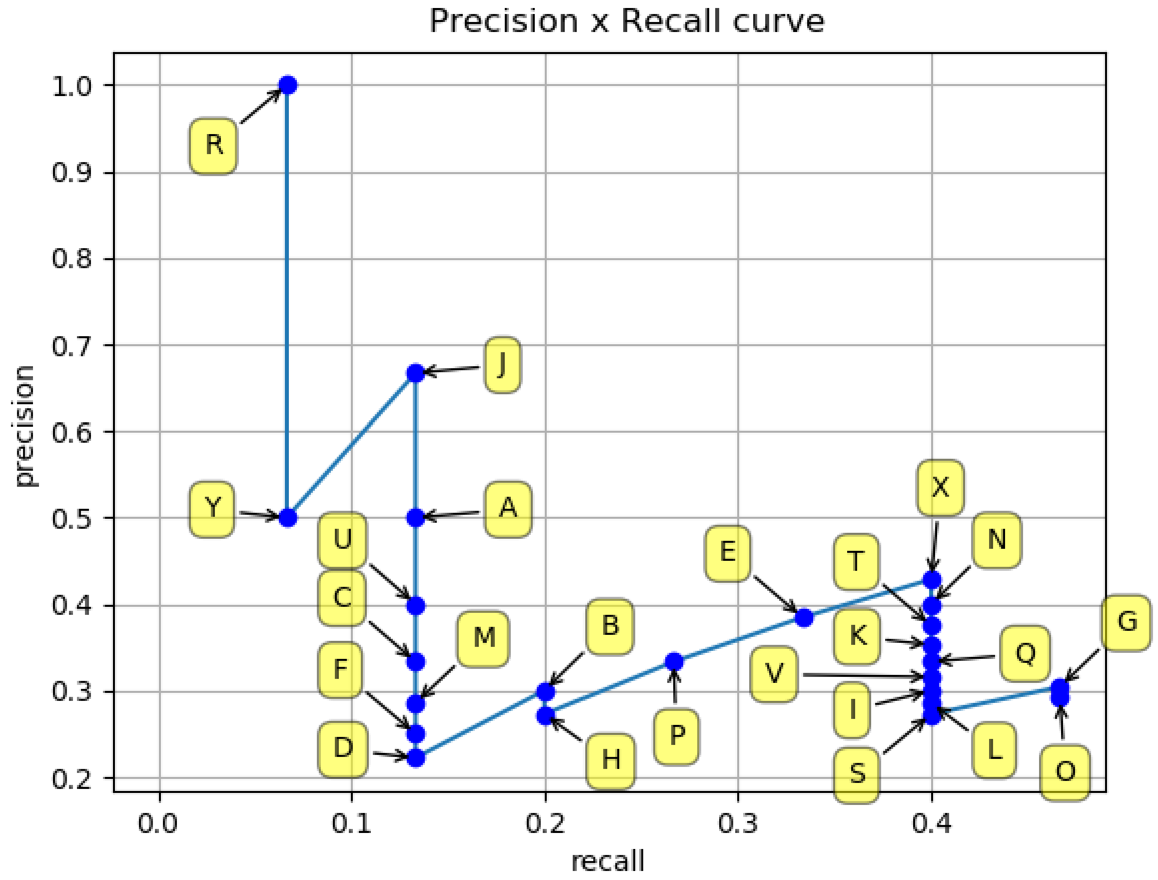
- We sort the table based on the confidence score.

Images	Detections	Confidences	TP	FP	Acc TP	Acc FP	Precision	Recall
Image 5	R	95%	1	0	1	0	1	0.0666
Image 7	Y	95%	0	1	1	1	0.5	0.0666
Image 3	J	91%	1	0	2	1	0.6666	0.1333
Image 1	A	88%	0	1	2	2	0.5	0.1333
Image 6	U	84%	0	1	2	3	0.4	0.1333
Image 1	C	80%	0	1	2	4	0.3333	0.1333
Image 4	M	78%	0	1	2	5	0.2857	0.1333
Image 2	F	74%	0	1	2	6	0.25	0.1333



Example

- We plot Recall-Precision values according to the sorted table.

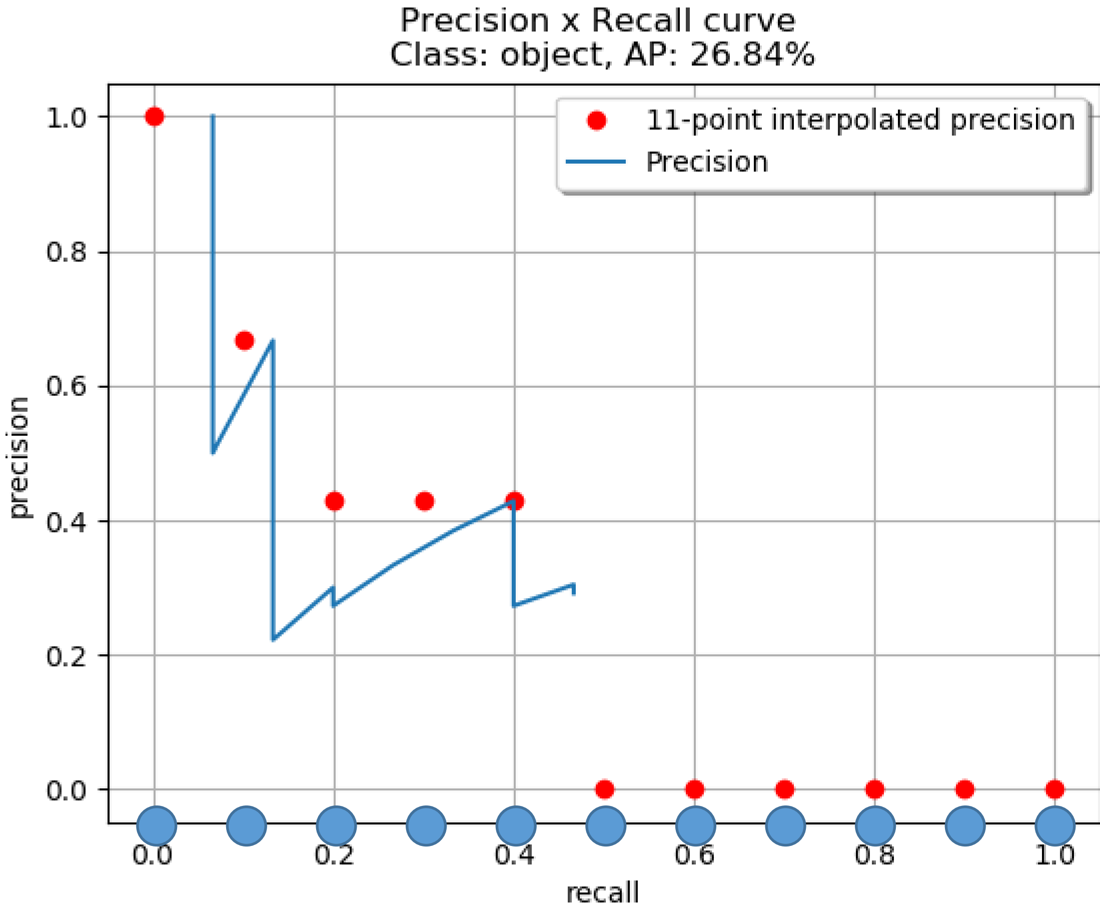


Example

- How to measure interpolated AP (Average Precision)?
  - **11 point interpolation**
  - **Interpolation over in all points**

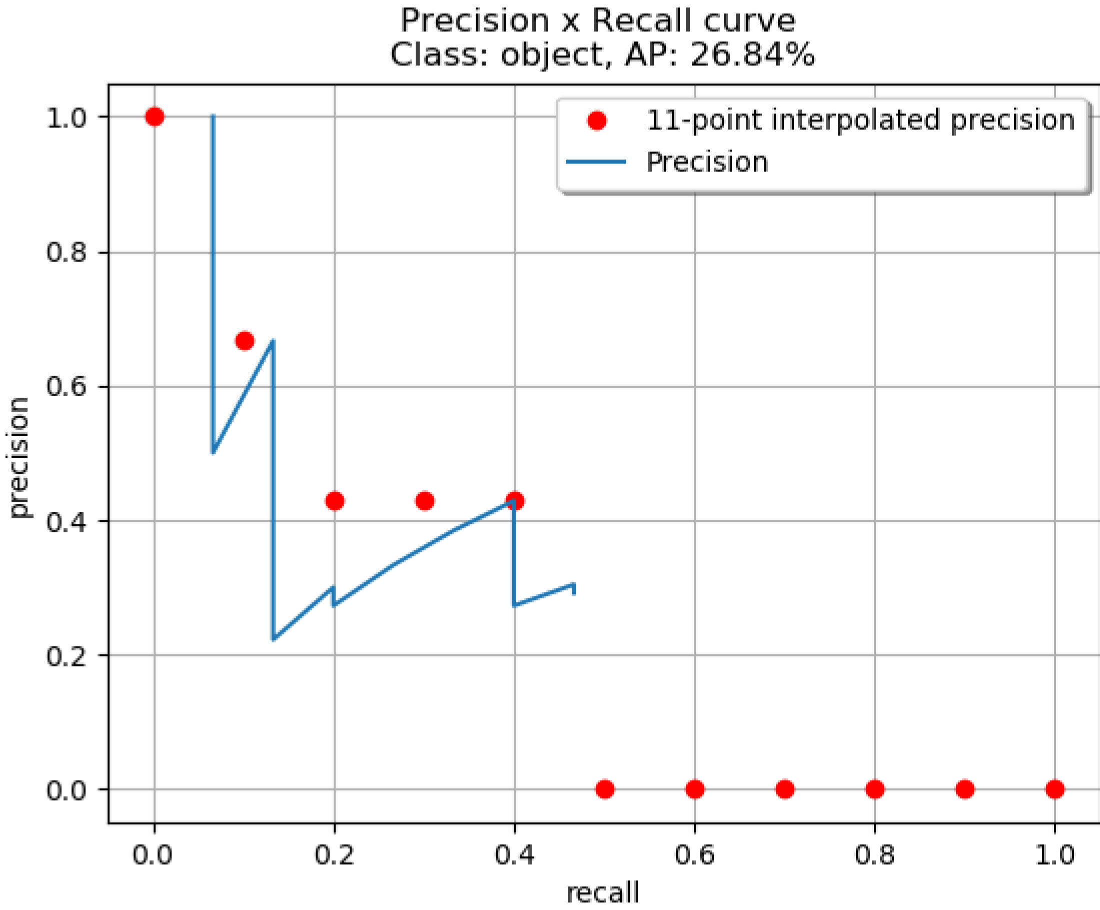
Example

- 11 Point Interpolation:
  - Discretize the recall values by 11 samples (0,0.1,...,1).



Example

- 11 Point Interpolation:
  - Discretize the recall values by 11 samples (0,0.1,…,1).
  - Obtain interpolated precision values by taking the maximum precision whose recall value is greater than its current recall value (red dots).



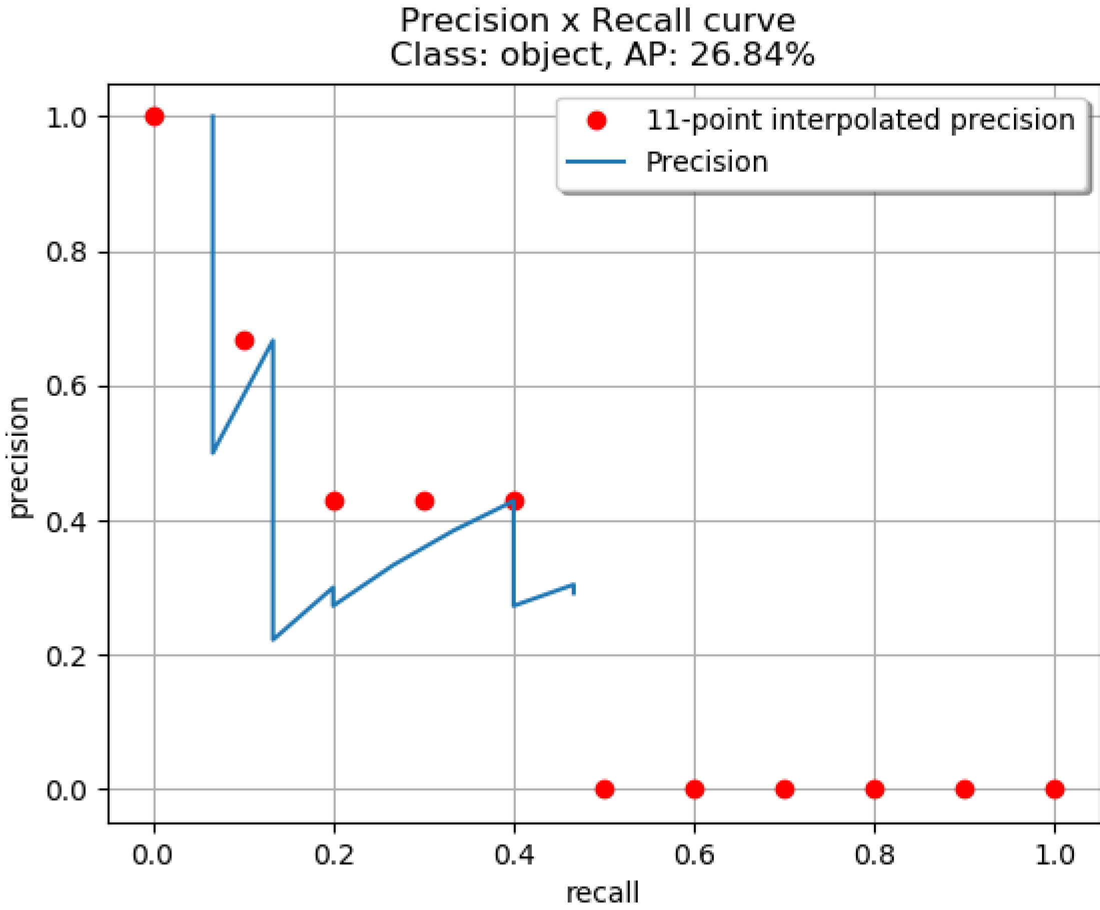
Example

- Now, calculate AP as an integration (discrete)

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} \rho_{\text{interp}(r)}$$

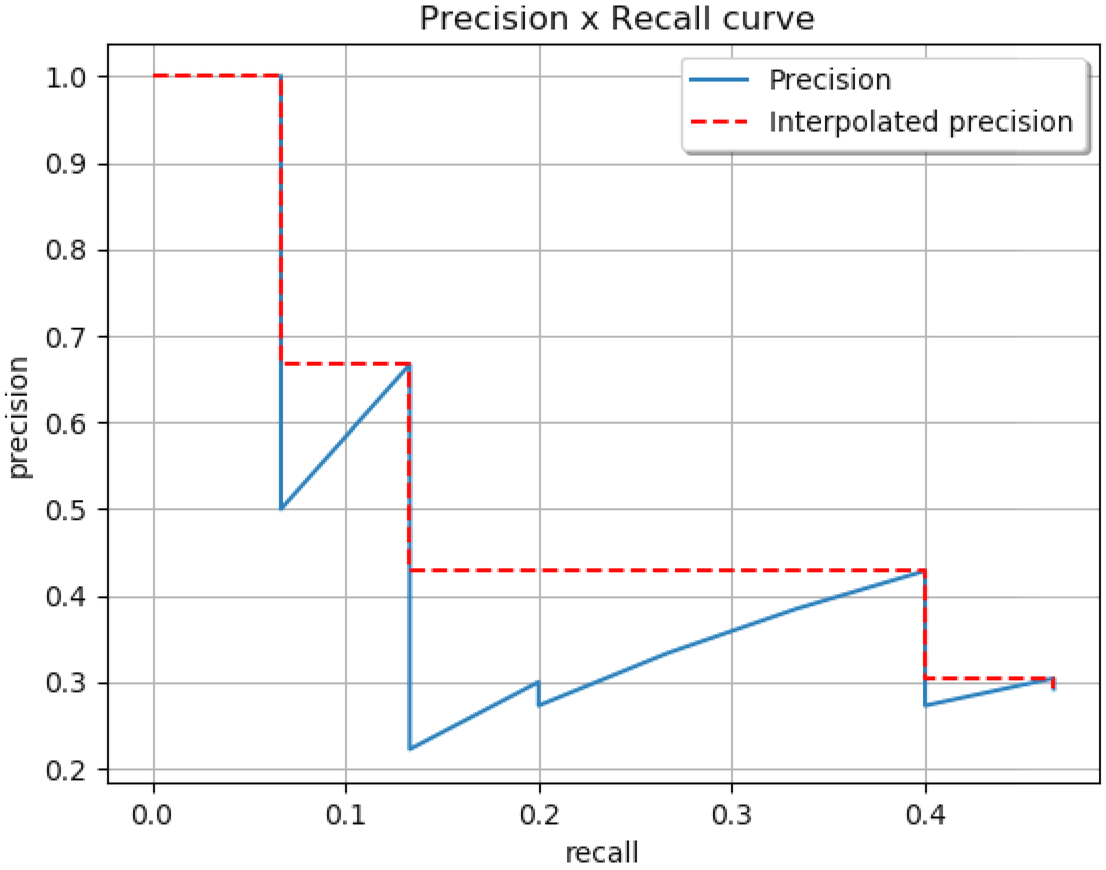
$$AP = \frac{1}{11}(1 + 0.6666 + 0.4285 + 0.4285 + 0.4285 + 0 + 0 + 0 + 0 + 0 + 0)$$

$$AP = 26.84\%$$



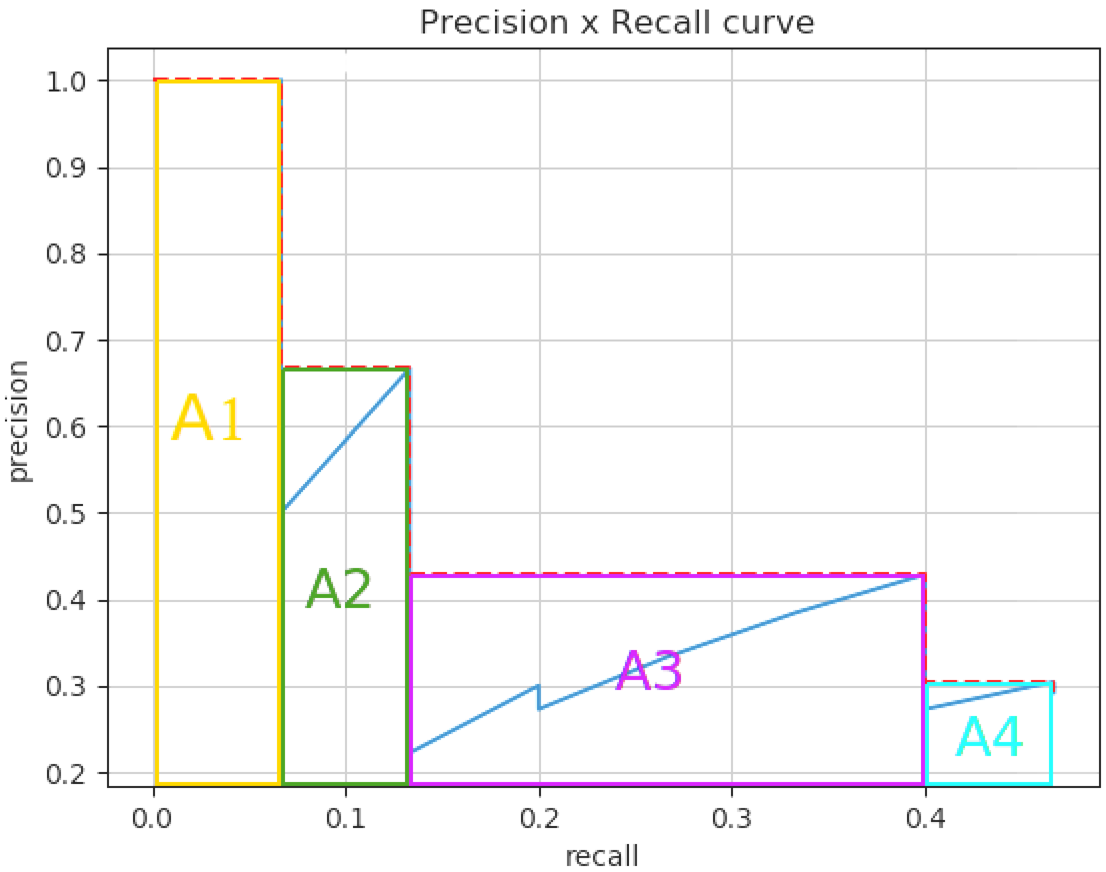
Example

- Interpolation over all points.
  - Approximate the area under the curve by finding maximum precisions.



Example

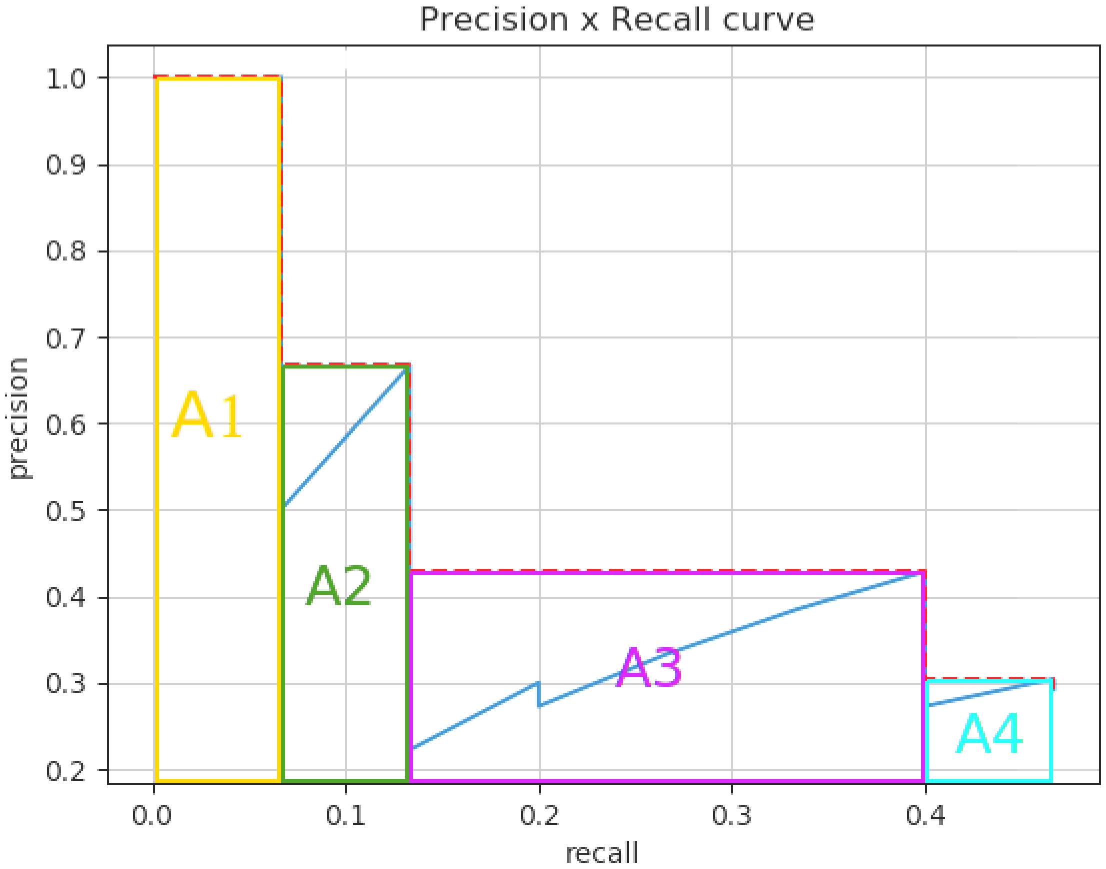
- Interpolation over all points.
  - Approximate the area under the curve by finding maximum precisions.
  - You will get a set of rectangles.



Example

- Interpolation over all points.
  - Approximate the area under the curve by finding maximum precisions.
  - You will get a set of rectangles.

$$AP = A1 + A2 + A3 + A4$$



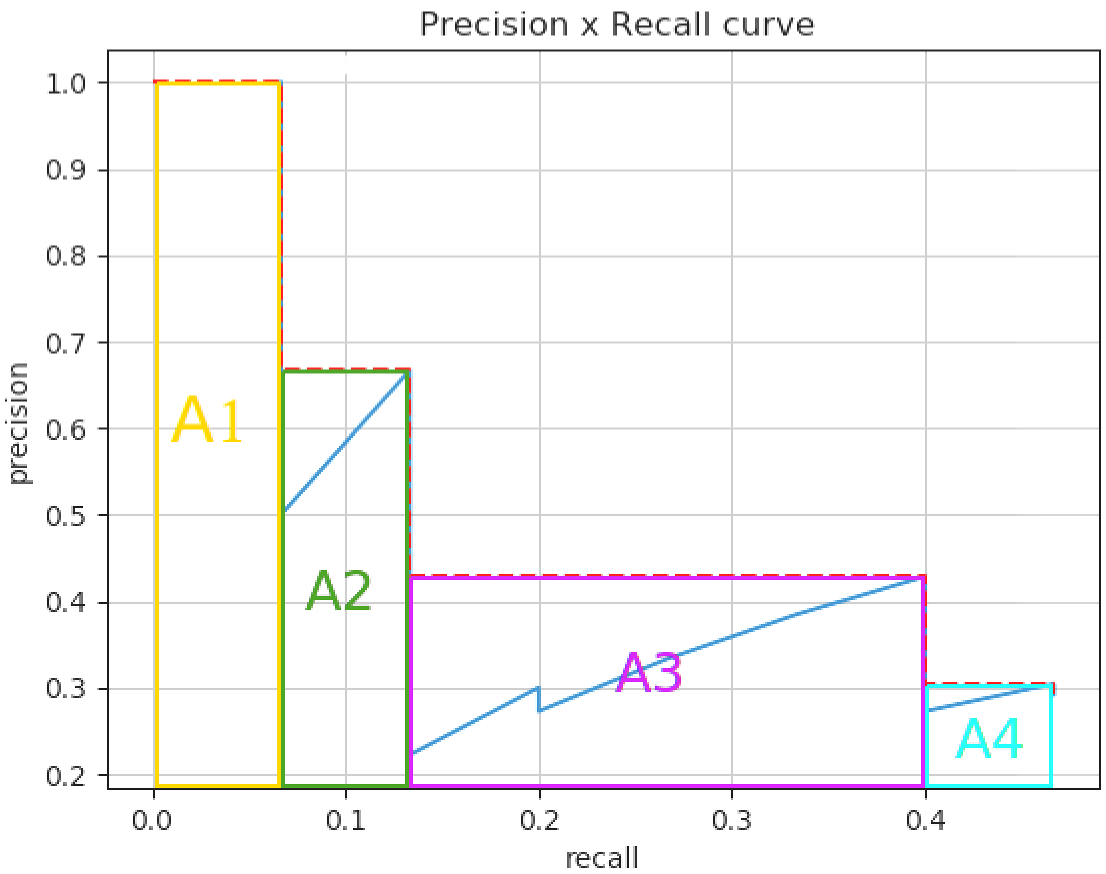
Example

- Interpolation over all points.
  - Approximate the area under the curve by finding maximum precisions.
  - You will get a set of rectangles.

$$AP = A1 + A2 + A3 + A4$$

$$A1 = (0.0666 - 0) \times 1 = \mathbf{0.0666}$$
$$A2 = (0.1333 - 0.0666) \times 0.6666 = \mathbf{0.04446222}$$
$$A3 = (0.4 - 0.1333) \times 0.4285 = \mathbf{0.11428095}$$
$$A4 = (0.4666 - 0.4) \times 0.3043 = \mathbf{0.02026638}$$

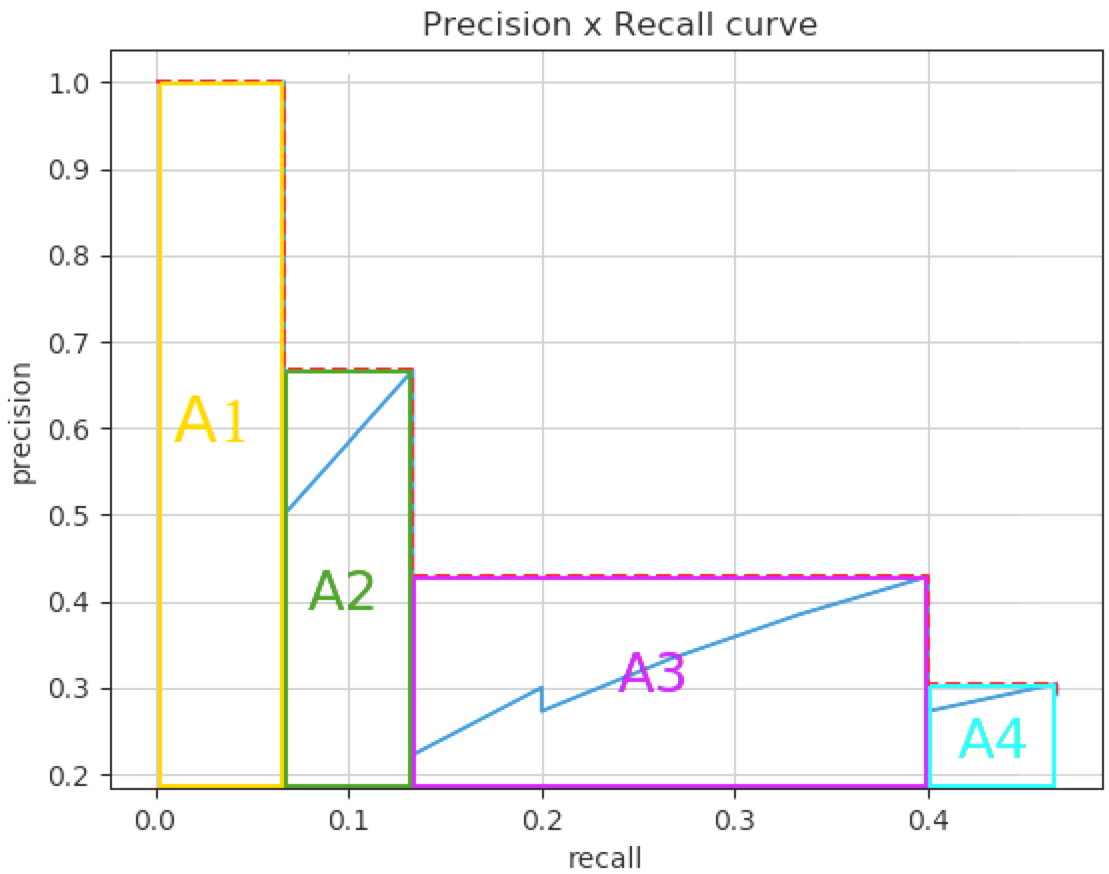
$$AP = 0.0666 + 0.04446222 + 0.11428095 + 0.02026638$$
$$AP = 0.24560955$$
$$AP = \mathbf{24.56\%}$$



Example

- 0 is worst, 100 is perfect.

$$AP = \mathbf{24.56\%}$$



Object Detection

- Now we know if have a set of predictions and ground truth boxes, we can evaluate how good the object detection algorithm is performing.

Object Detection

- Now we know if have a set of predictions and ground truth boxes, we can evaluate how good the object detection algorithm is performing.

- How to predict those boxes?

Object Detection

- Before CNN, how could we detect objects?

Object Detection

- Basic idea: slide a filter over the entire image and find where it responds the most.

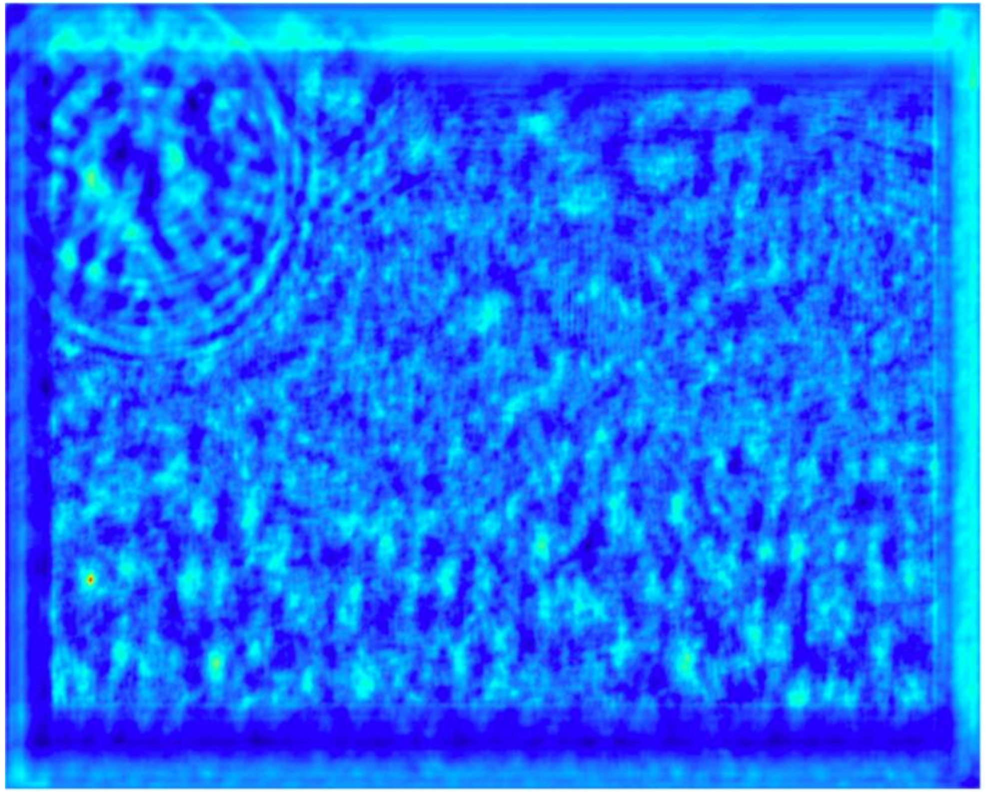
Object Detection

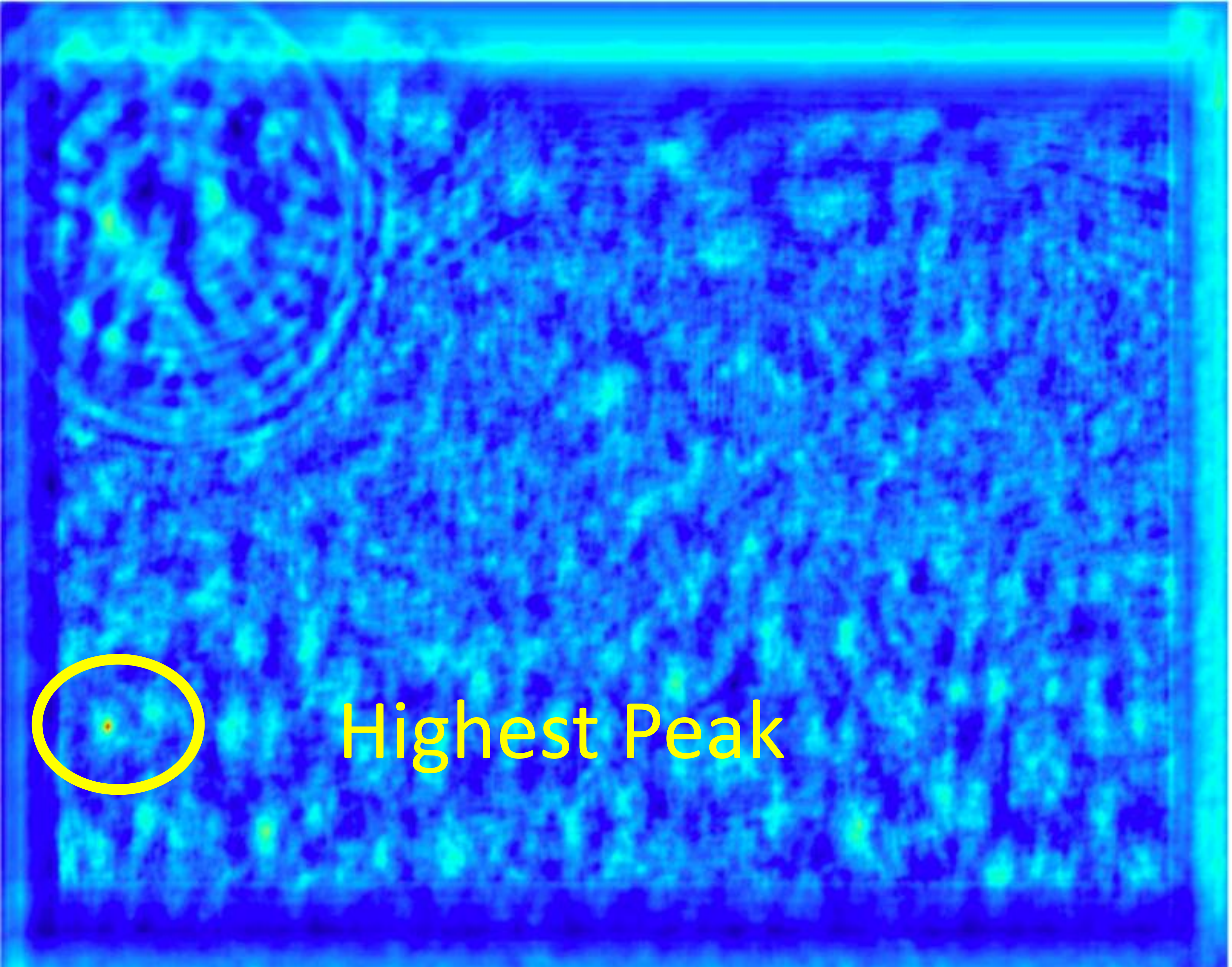
Object Detection

Filter

Object Detection

- Cross-correlation result




Highest Peak


WHERE'S WALDO?
TM

Object Detection

- Find all people?

Object Detection

- Find all people?

Object Detection

- Before CNN, how could we detect objects?

The HOG Detector

N. Dalal and B. Triggs

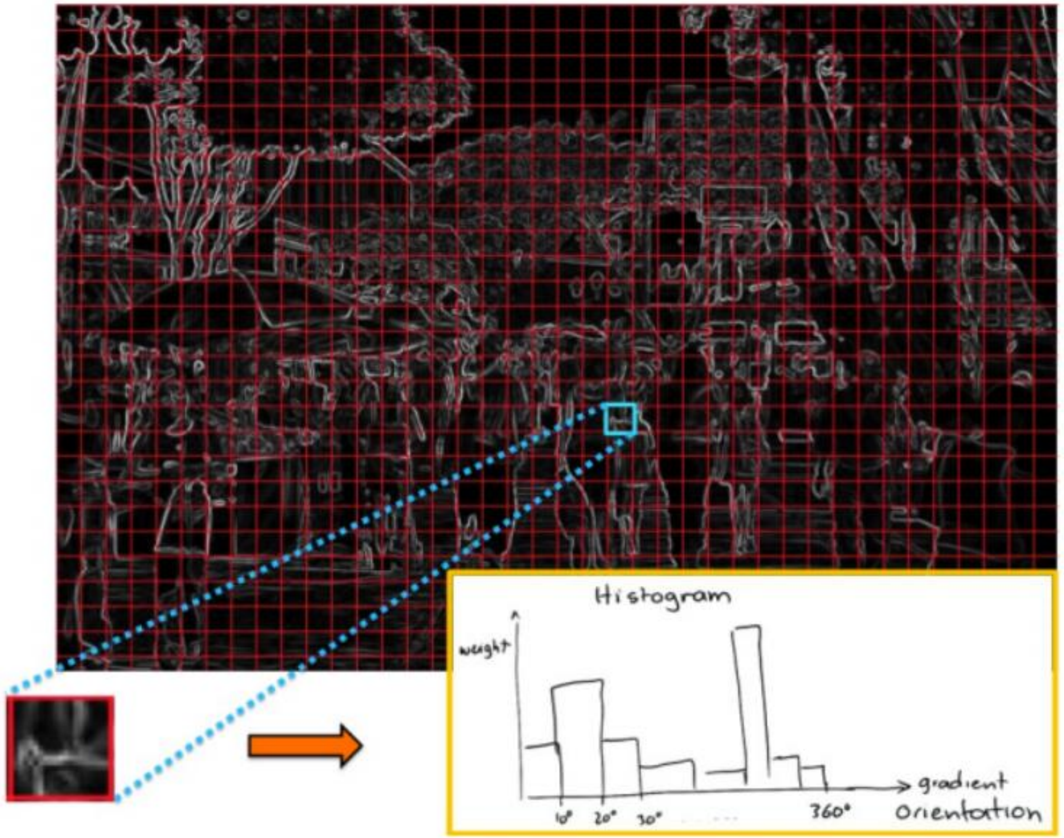
Histograms of oriented gradients for human detection

CVPR, 2005

Paper: http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf

HOG

- The method is based on evaluating well-normalized local histograms of image gradient orientations in a dense grid.



HOG

- The method is based on evaluating well-normalized local histograms of image gradient orientations in a dense grid.

- Basic Idea:
  - Divide the image into small spatial regions: cells

HOG

- The method is based on evaluating well-normalized local histograms of image gradient orientations in a dense grid.

- Basic Idea:
  - Divide the image into small spatial regions: cells
  - For each cell, accumulate a local 1-D histogram of gradient directions or edge orientations over the pixels of the cell.

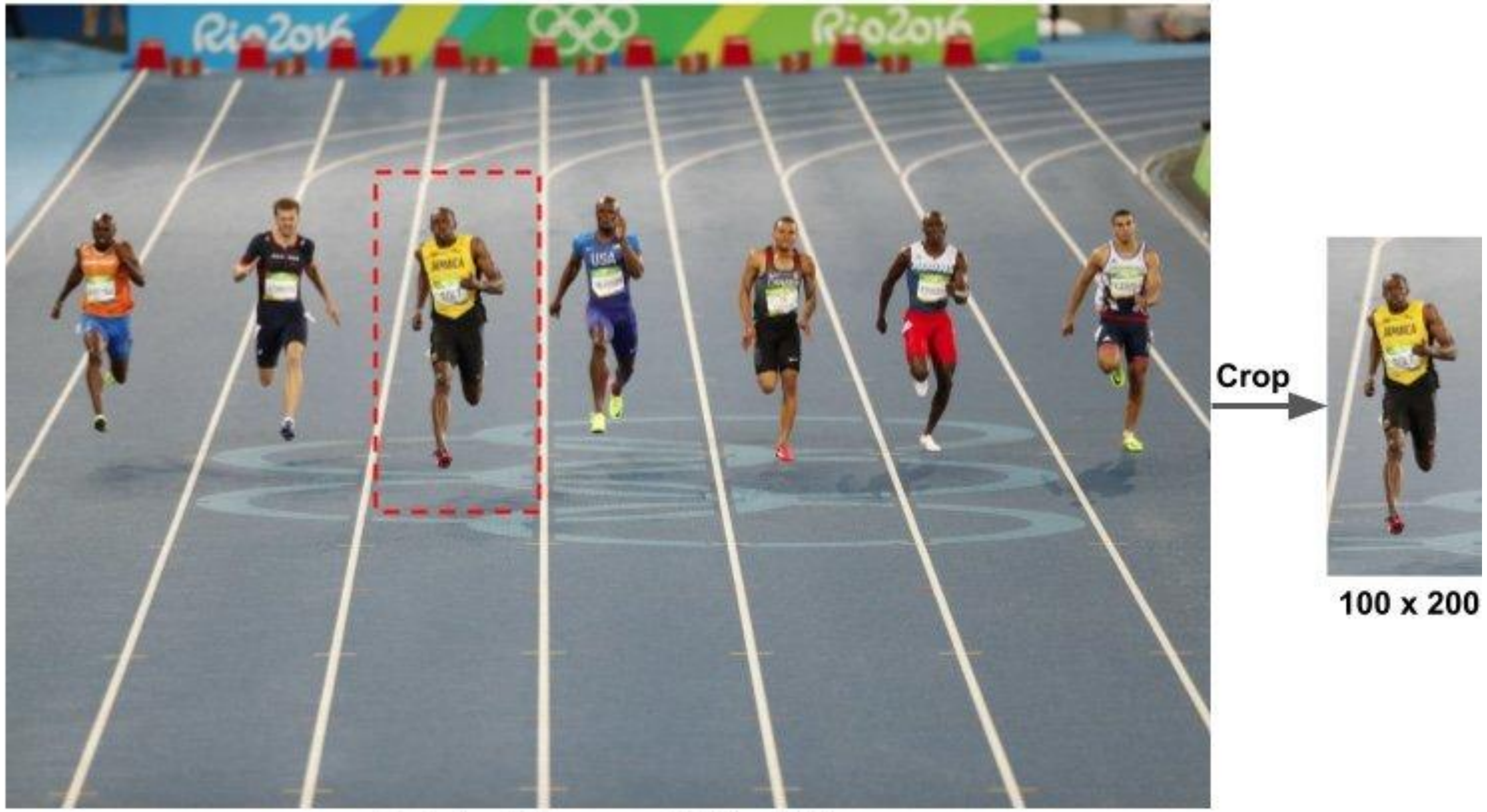
HOG

- The method is based on evaluating well-normalized local histograms of image gradient orientations in a dense grid.

- Basic Idea:
  - Divide the image into small spatial regions: cells
  - For each cell, accumulate a local 1-D histogram of gradient directions or edge orientations over the pixels of the cell.
  - The combined histogram entries form the representation.

Orientation Cells

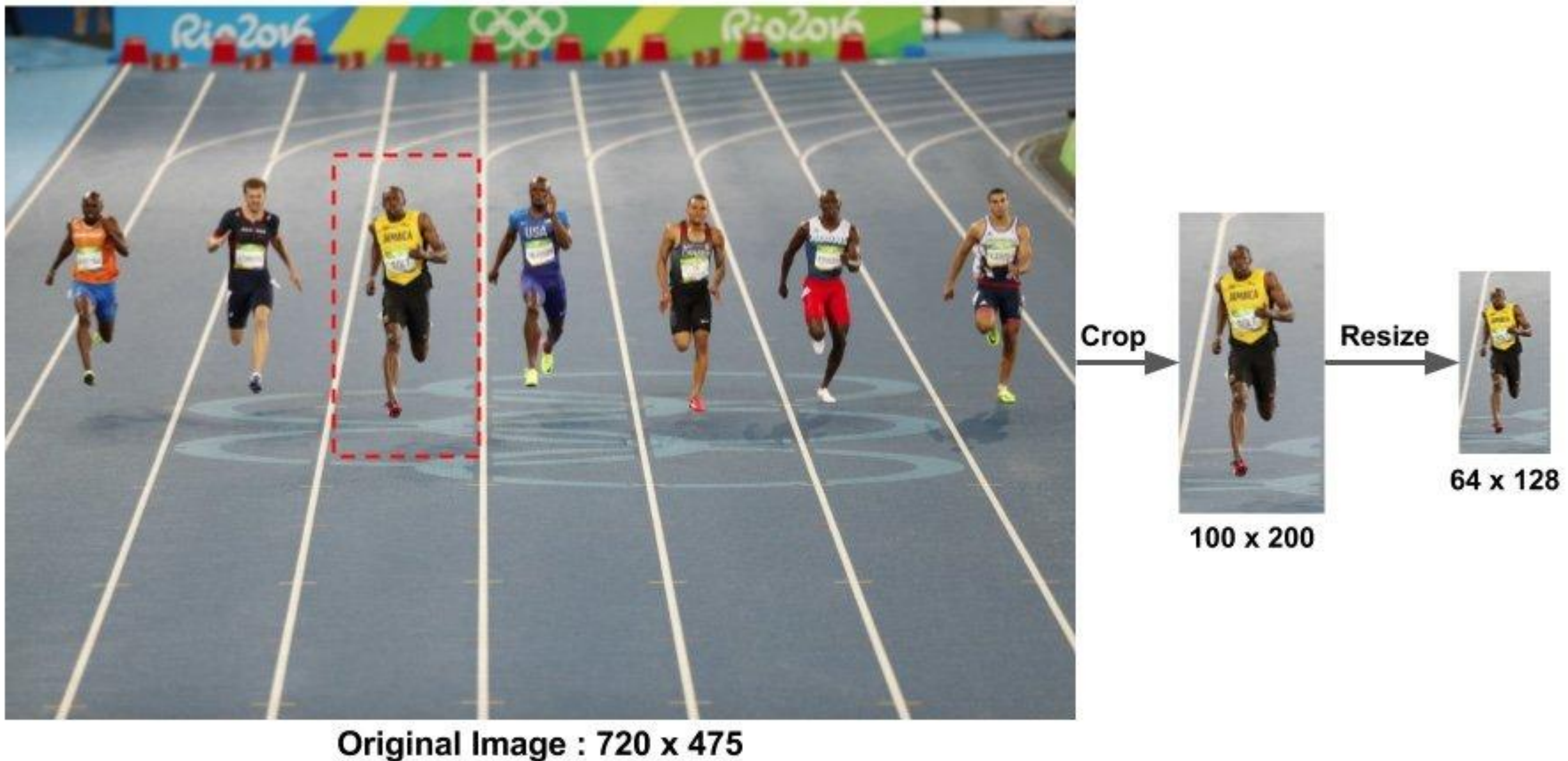
- Patches at multiple scales are analyzed at many image locations. The only constraint is that the patches being analyzed have a fixed aspect ratio.



Original Image : 720 x 475

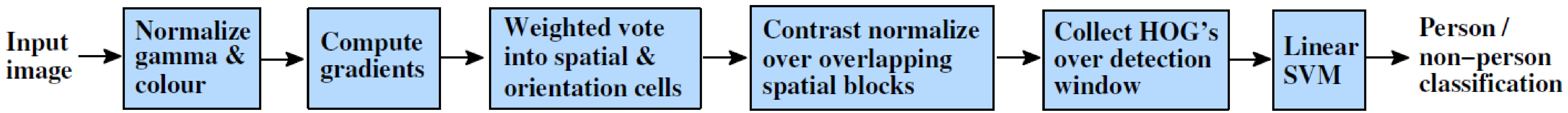
Orientation Cells

- We need to resize them into the desired size.



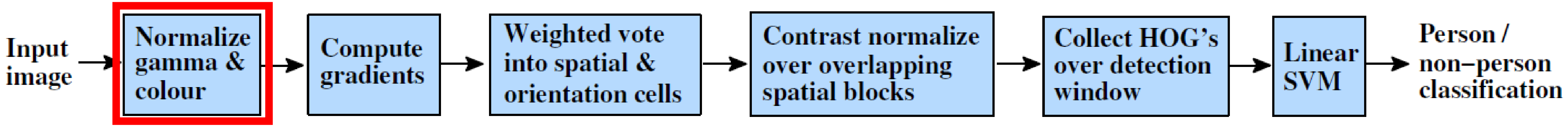
HOG

- Pipe-line:



HOG

- Pipe-line:

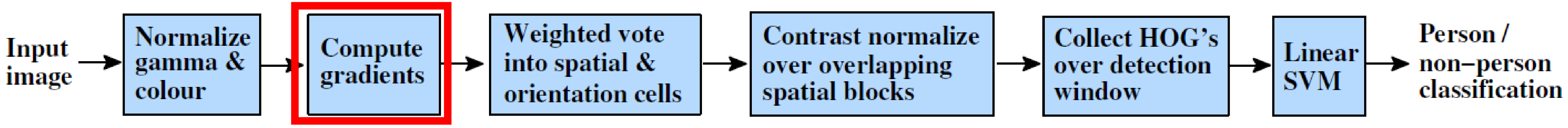


Color Normalization

- To avoid being affected by illumination

HOG

- Pipe-line:



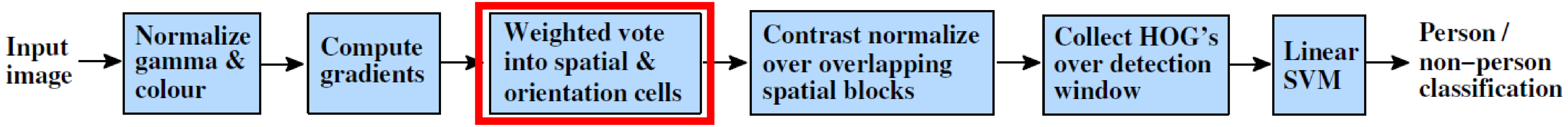
Gradient Computation

- Simple derivative formula



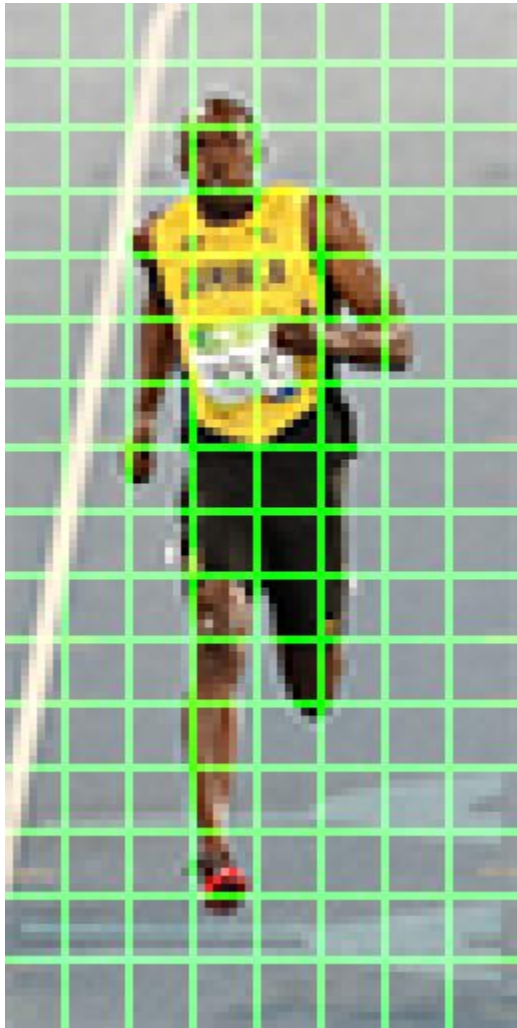
HOG

- Pipe-line:



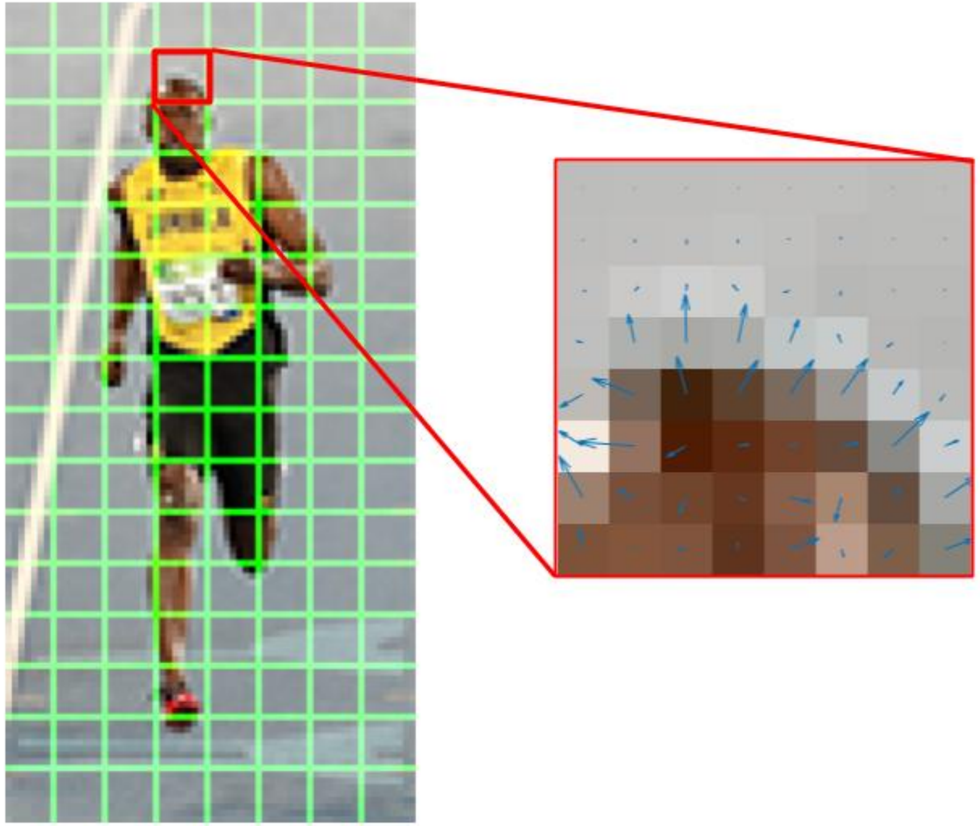
Orientation Cells

- Make an 8 by 8 grid (8 is arbitrary)



Orientation Cells

- Find gradient for each patch



2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

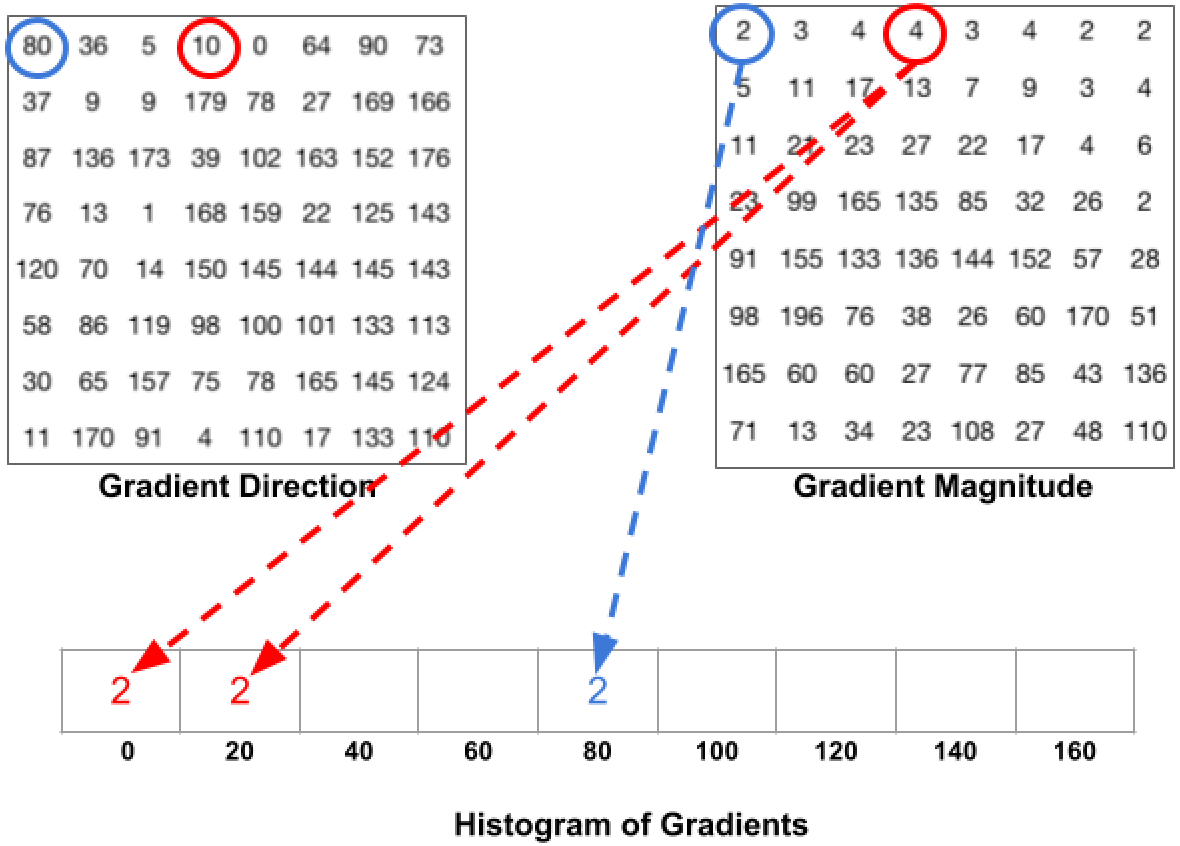
Gradient Magnitude

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

Orientation Cells

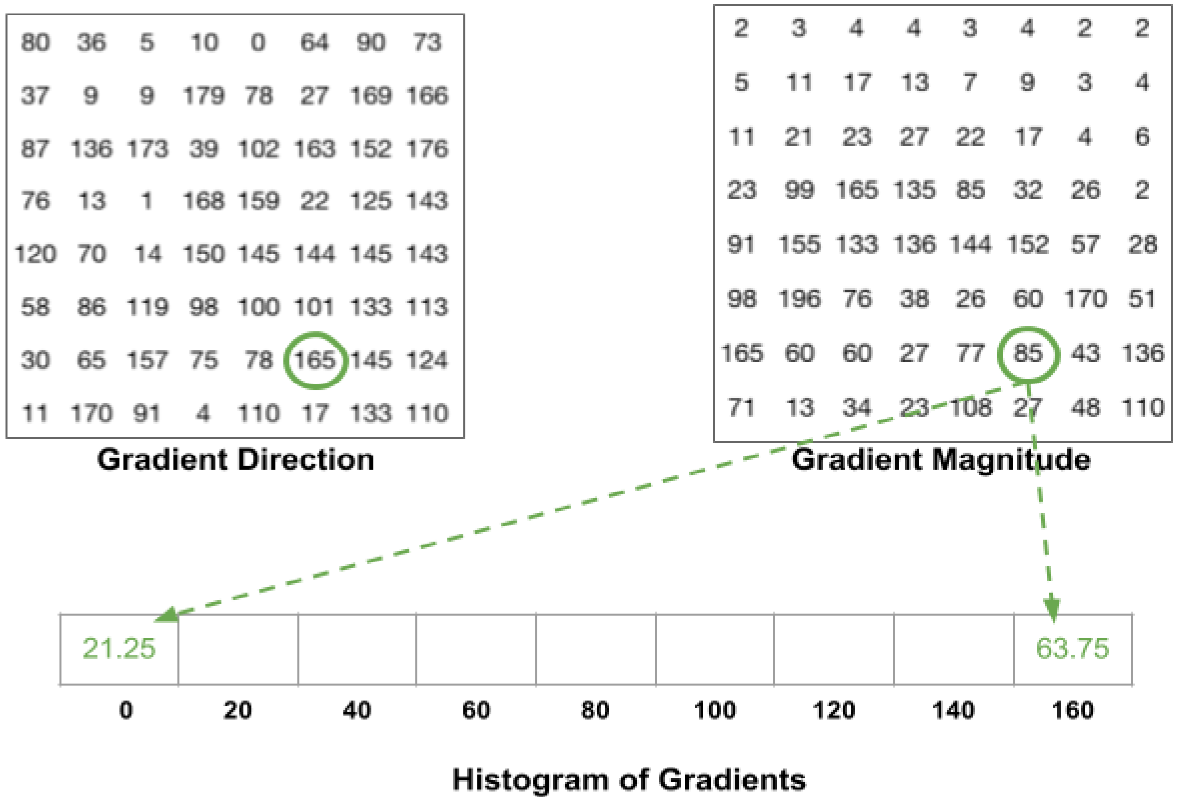
- Bin the angles from 0-180 into 9 buckets (proportionally).



Orientation Cells

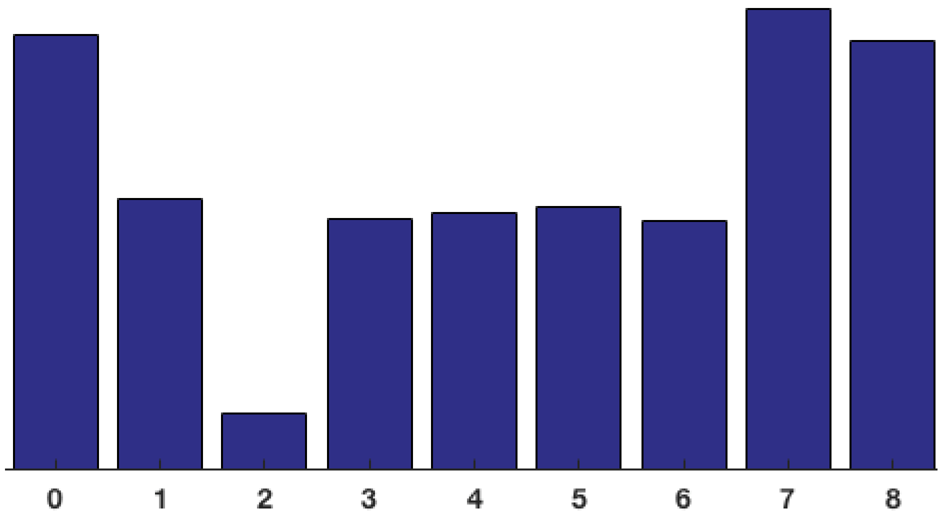
- Bin the angles from 0-180 into 9 buckets (proportionally).

Why?



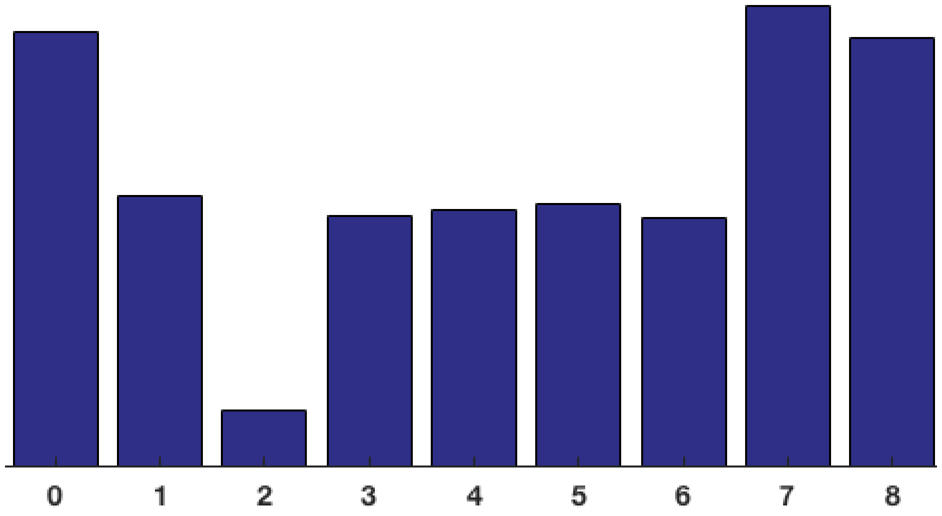
Orientation Cells

- Bin the angles from 0-180 into 9 buckets.
- Make a histogram.

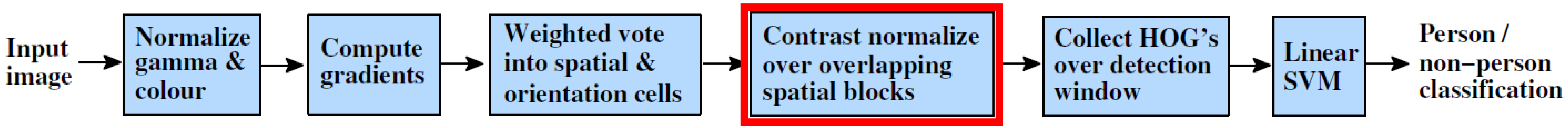


Orientation Cells

- Bin the angles from 0-180 into 9 buckets.
- Make a histogram.
- This is your 9 dimensional feature.

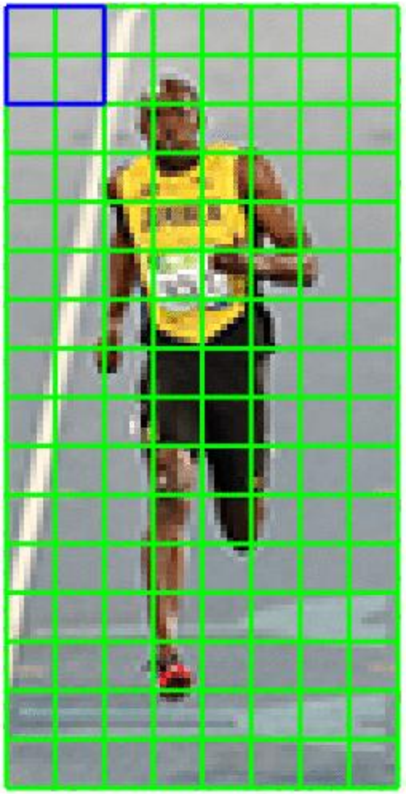


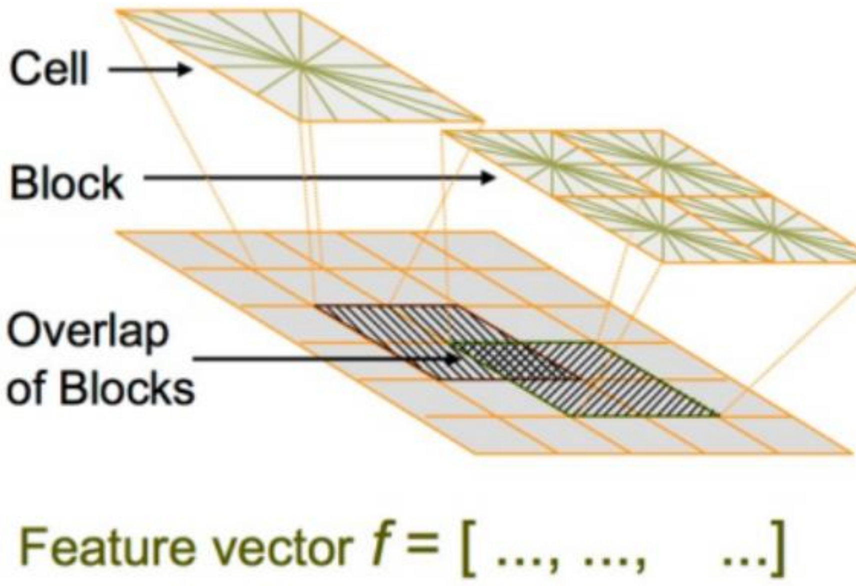
CNN-Based Detector



Contrast Normalization

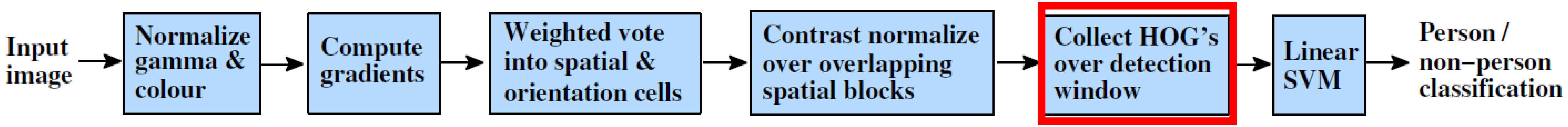
- Normalize over each block (16*16 pixels).
    - Gives smoother transitions for each cell between blocks.





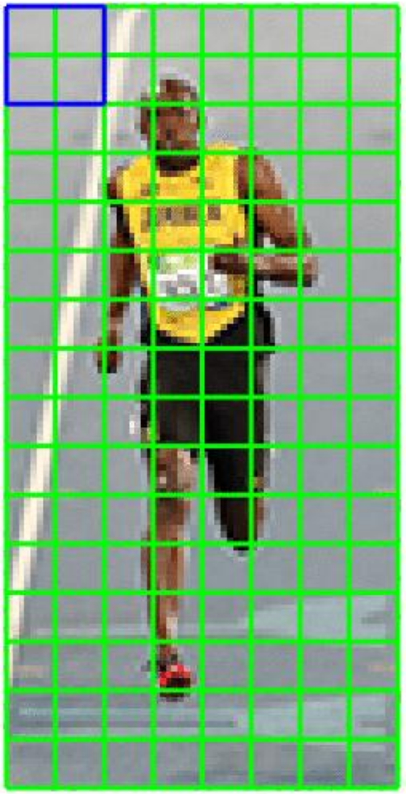
L2 normalization in each block: $$\mathbf{f} = \frac{\mathbf{f}}{\sqrt{||\mathbf{f}||_2^2 + \epsilon^2}}$$

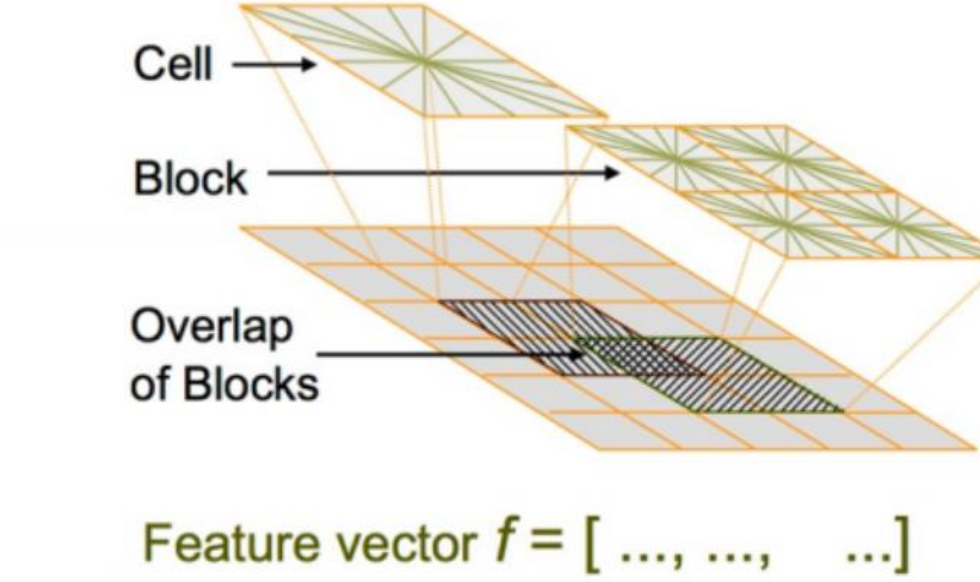
CNN-Based Detector



Contrast Normalization

- Normalize over each block (16*16 pixels).
  - Gives smoother transitions for each cell between blocks.



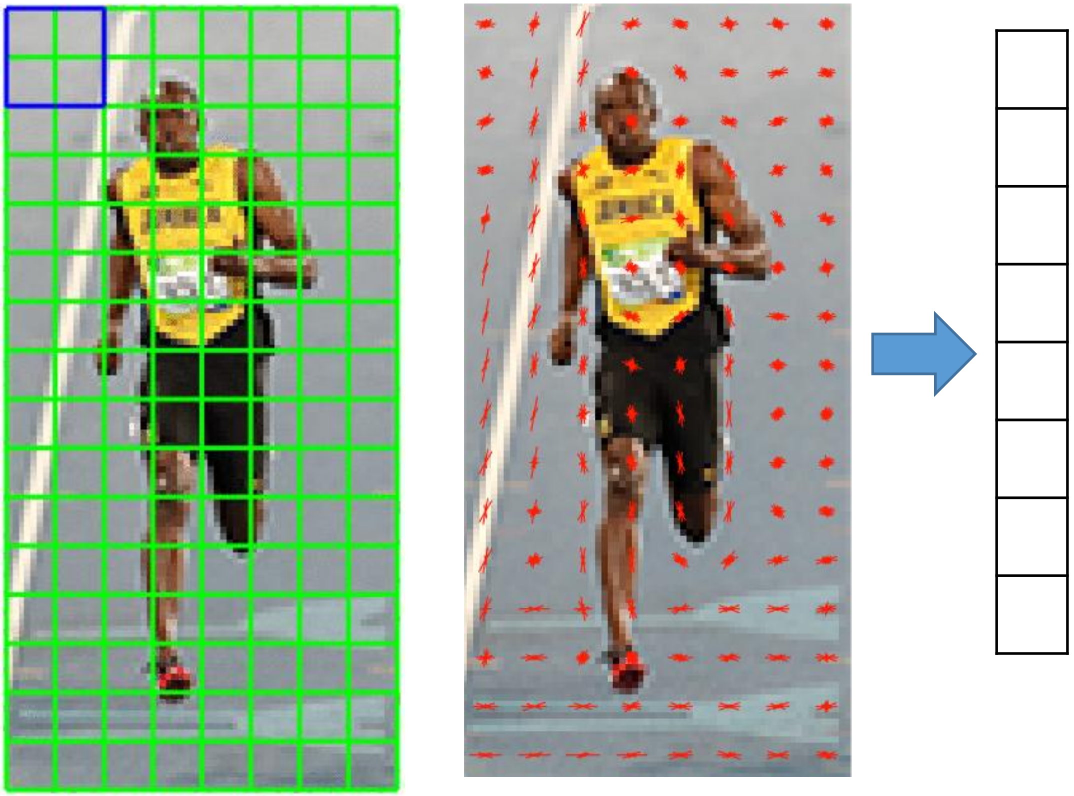


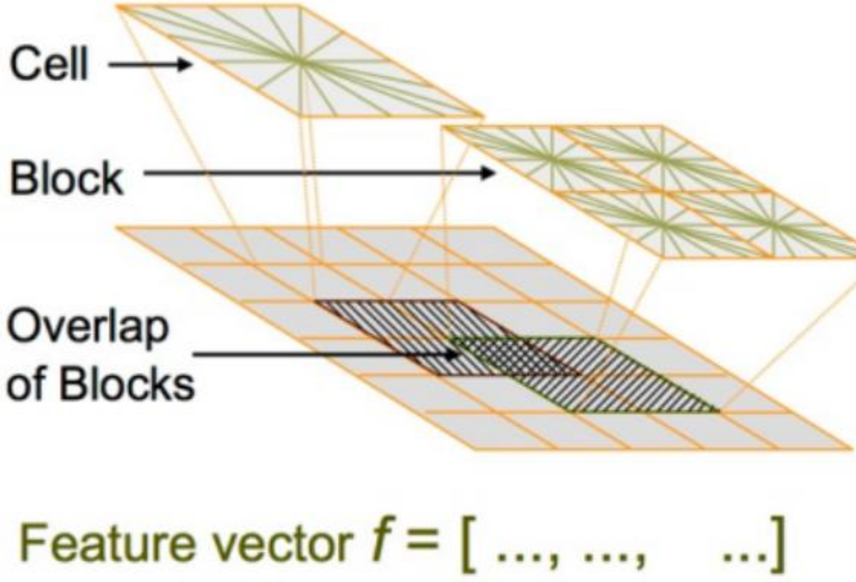
L2 normalization in each block:

$$\mathbf{f} = \frac{\mathbf{f}}{\sqrt{||\mathbf{f}||_2^2 + \epsilon^2}}$$

Contrast Normalization

- Normalize over each block (16*16 pixels).
  - Gives smoother transitions for each cell between blocks.

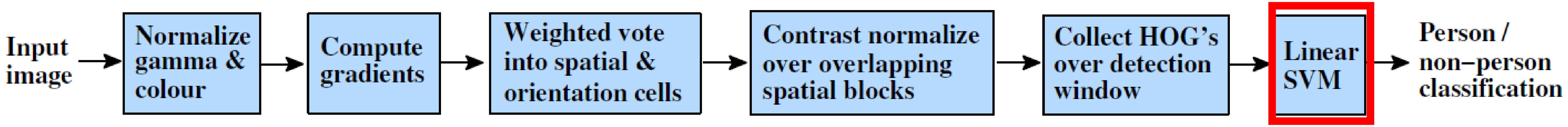




L2 normalization in each block:

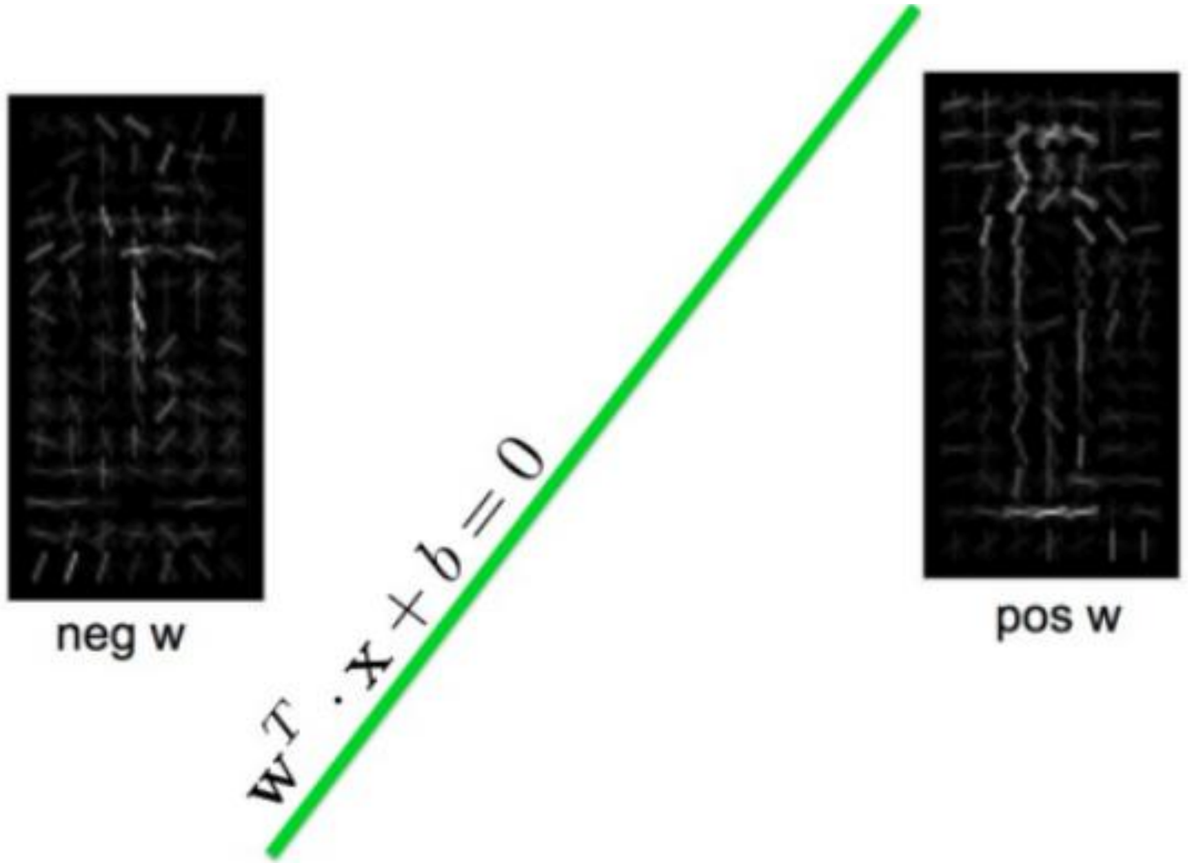
$$\mathbf{f} = \frac{\mathbf{f}}{\sqrt{||\mathbf{f}||_2^2 + \epsilon^2}}$$

CNN-Based Detector



Linear SVM

- Linear classifier



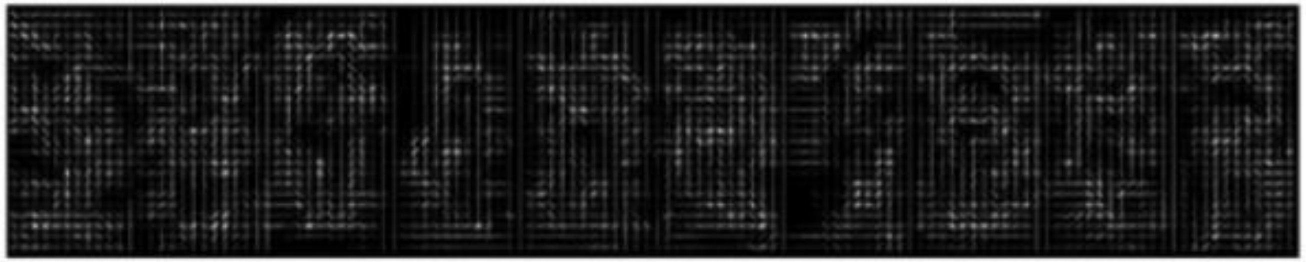
Linear SVM

- Training

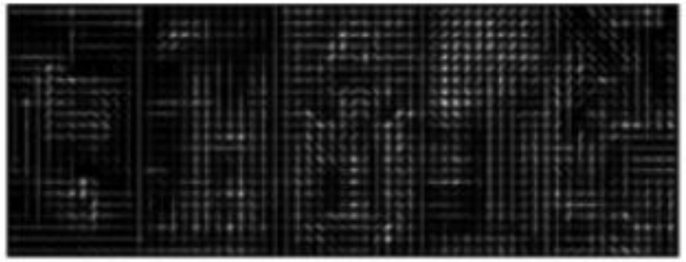
Linear SVM

- Training

positive training examples



negative training examples



Pros and Cons

- Is there any non-linearity?

- What is the limitation?

Recent Methods

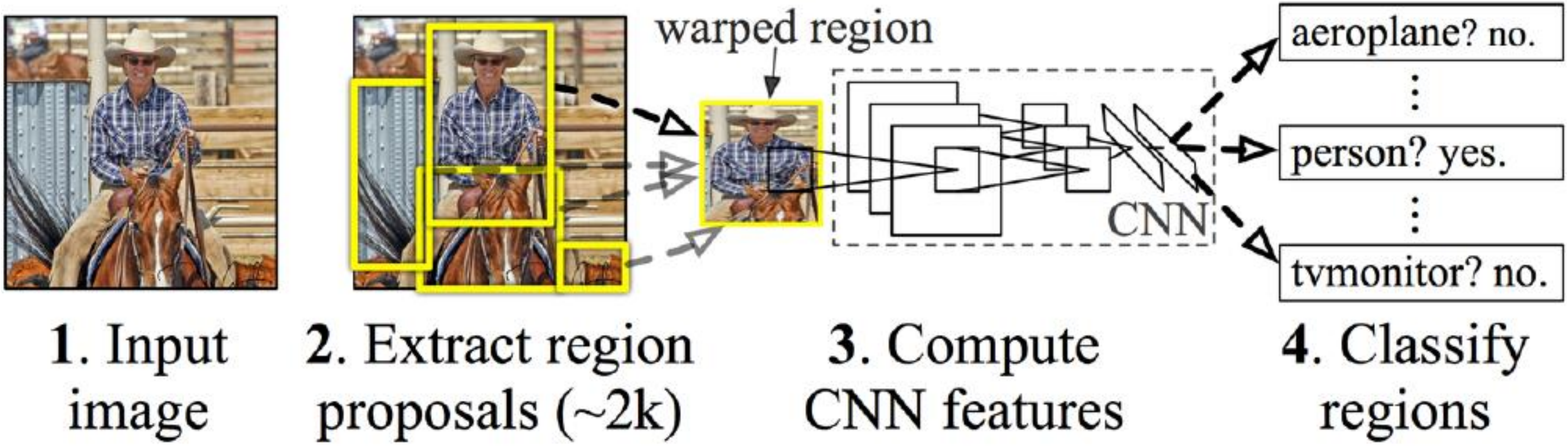
- What about now?

Recent Methods

- Think that you are a researcher in Computer Vision. You now have deep learning networks (let's say VGG). You are familiar with HOG. You want to make Object Detection better. What is your suggestion?

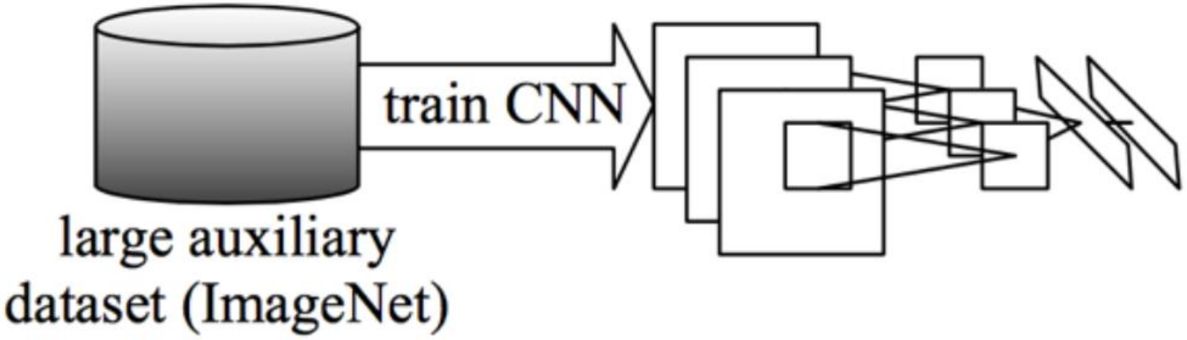
R-CNN

- Region CNN



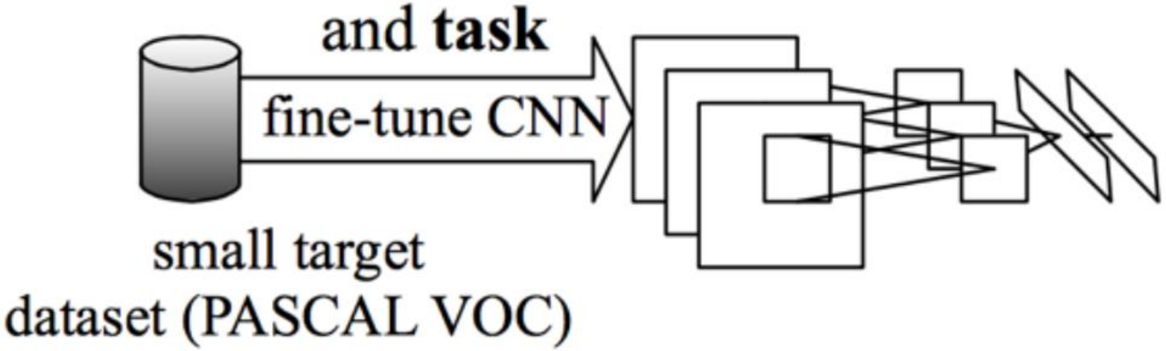
R-CNN

- Training
  - Pre-train a CNN for image classification (e.g., VGG).



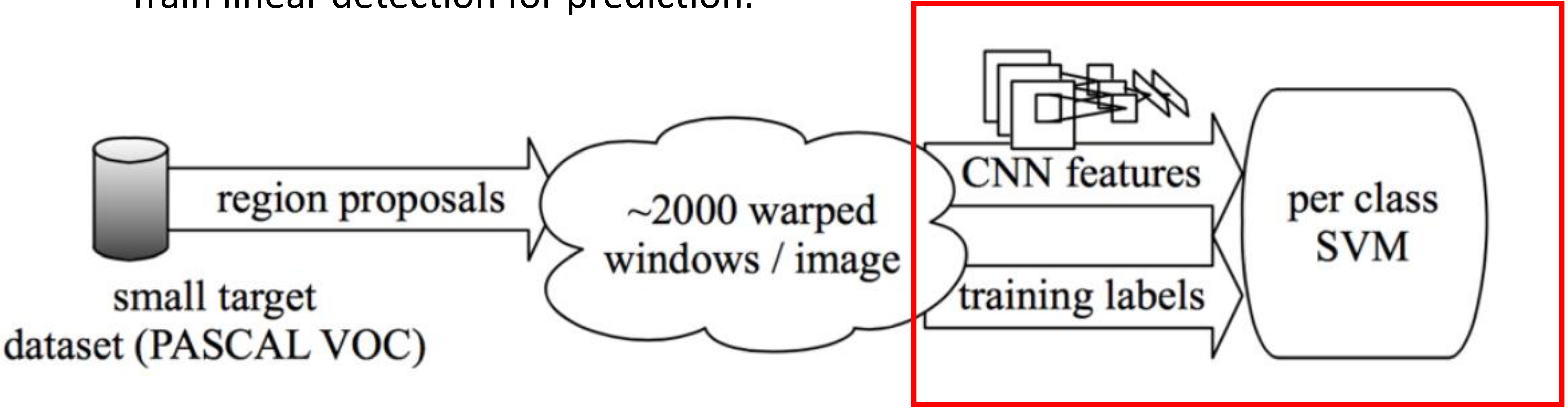
R-CNN

- Training
  - Pre-train a CNN for image classification (e.g., VGG).
  - Fine tune the network for a specific data set (optional).



R-CNN

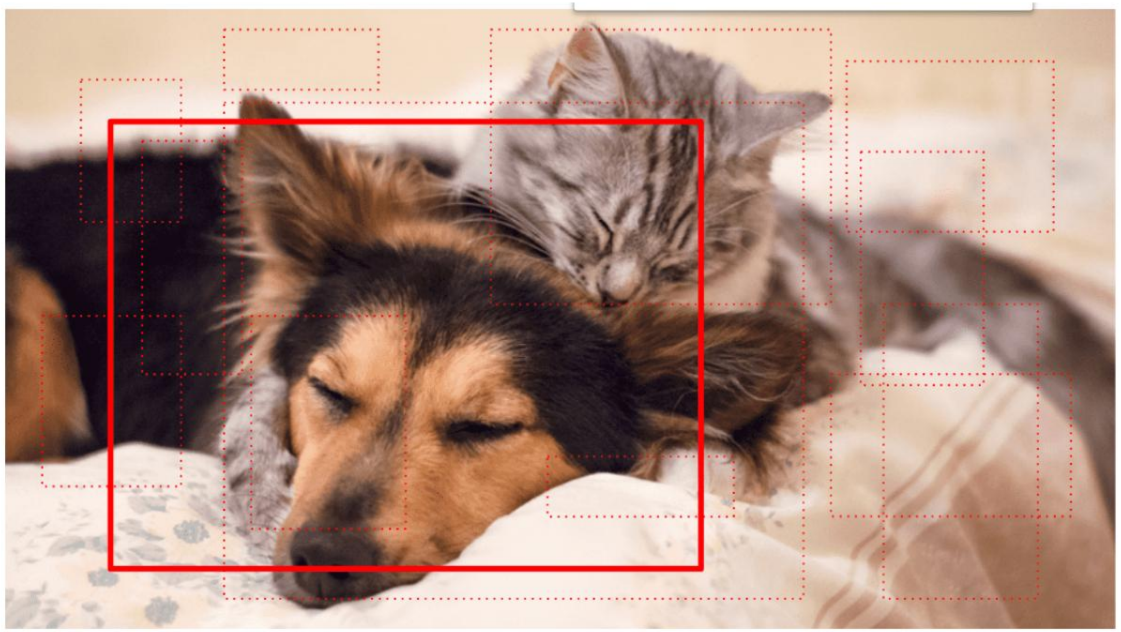
- Training
  - Pre-train a CNN for image classification (e.g., VGG).
  - Fine tune the network for a specific data set (optional).
  - Train linear detection for prediction.



Let's dig into it

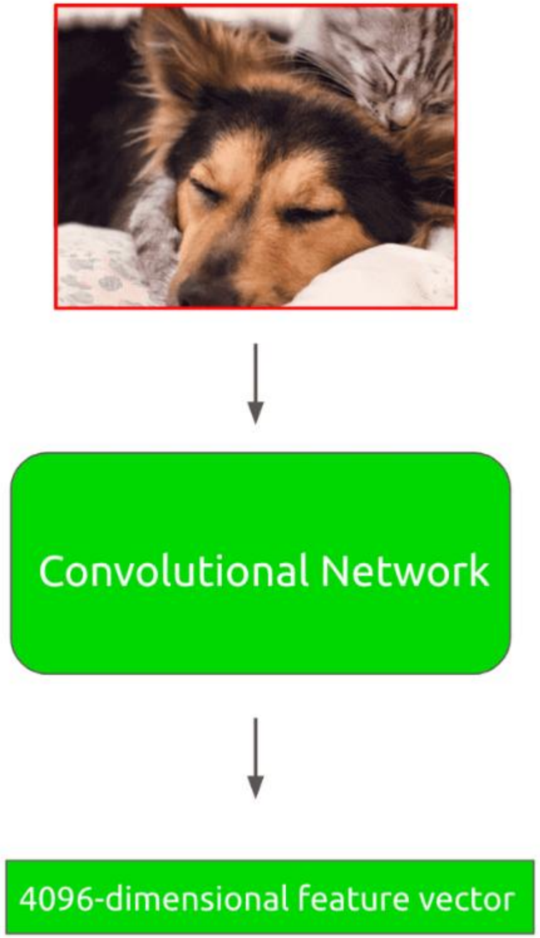
R-CNN

- Input image with a candidate bounding box



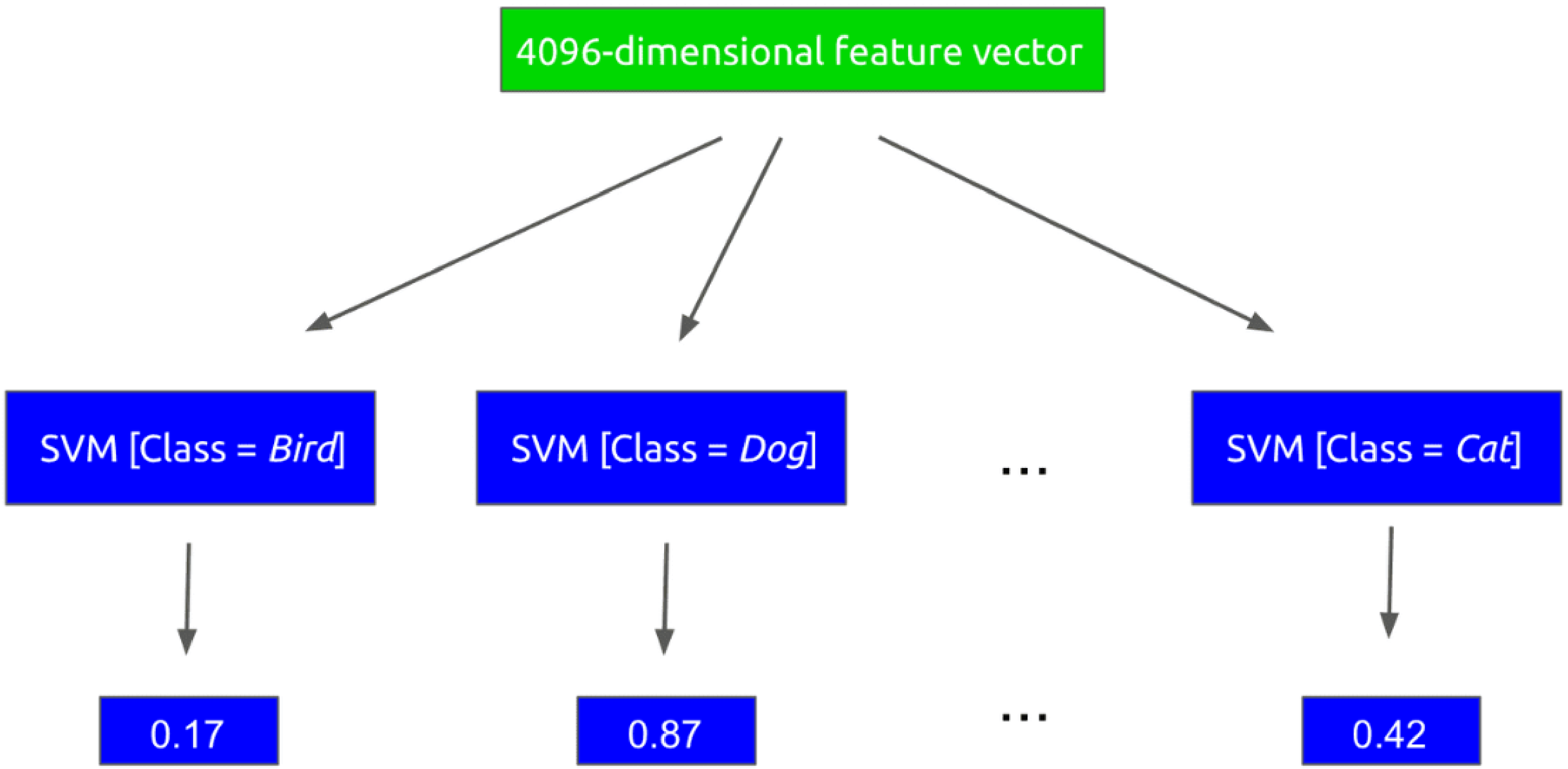
R-CNN

- Warp it and give it to a CNN and obtain a feature vector



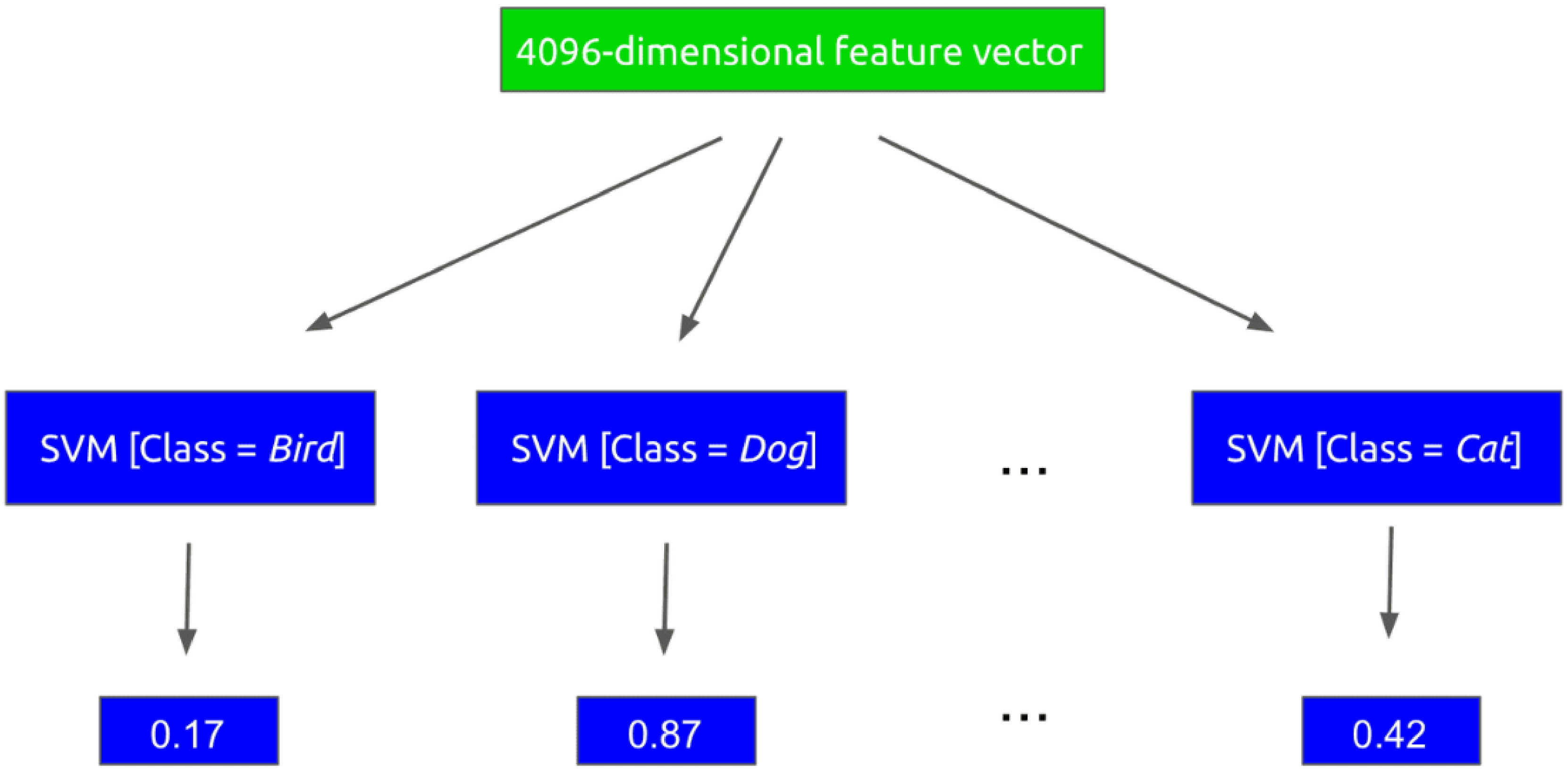
R-CNN

- Give the feature to a collection of linear **S**upport **V**ector **M**achines (**SVM**).



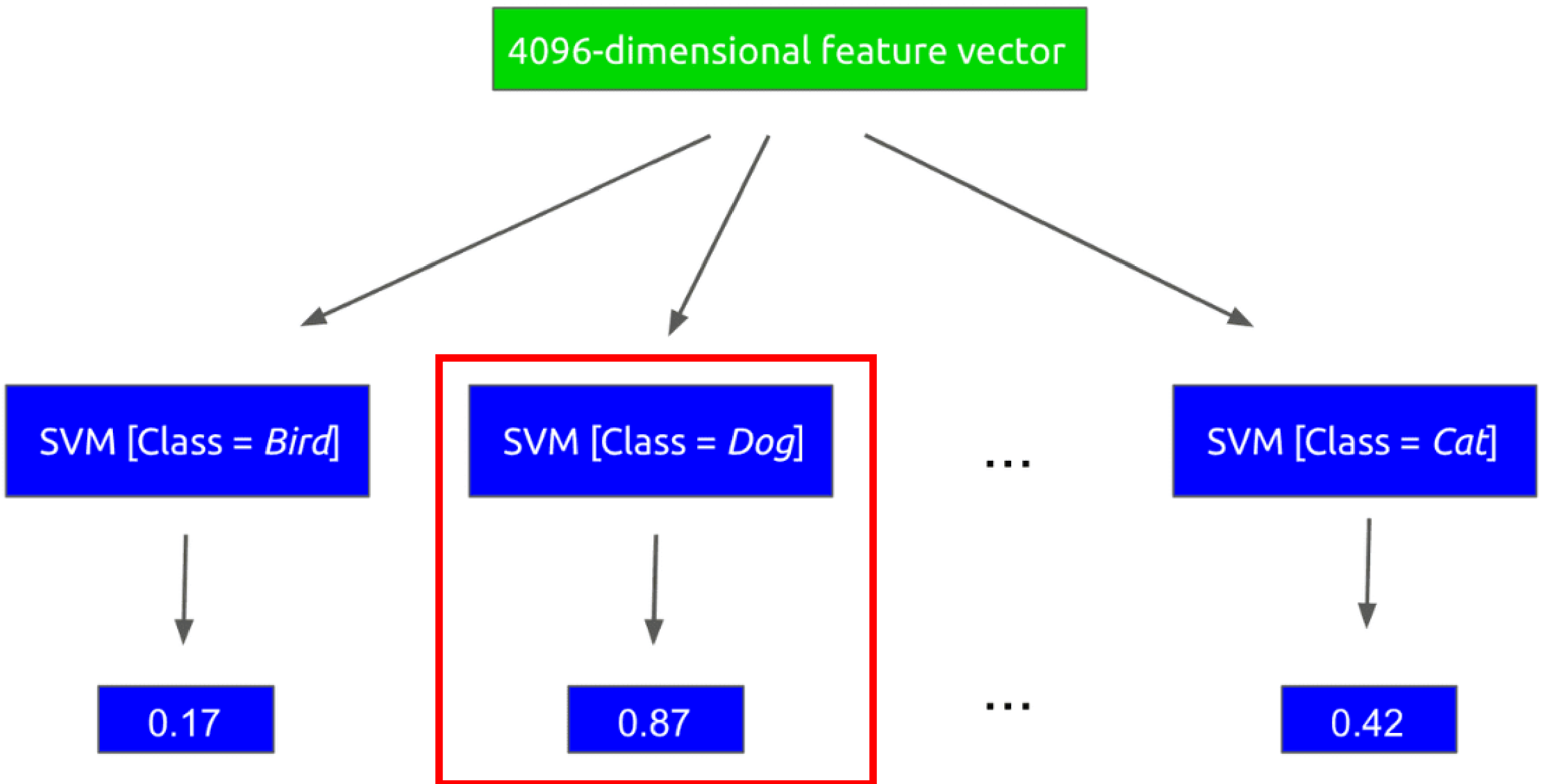
R-CNN

- Each SVM is designed to classify for a single object class. In other words, there is an SVM trained to detect *cat*, another one for *bird*, etc.



R-CNN

- Pick the class with the highest value.



VOC

- The PASCAL **V**isual **O**bject **C**lasses Challenge 2007

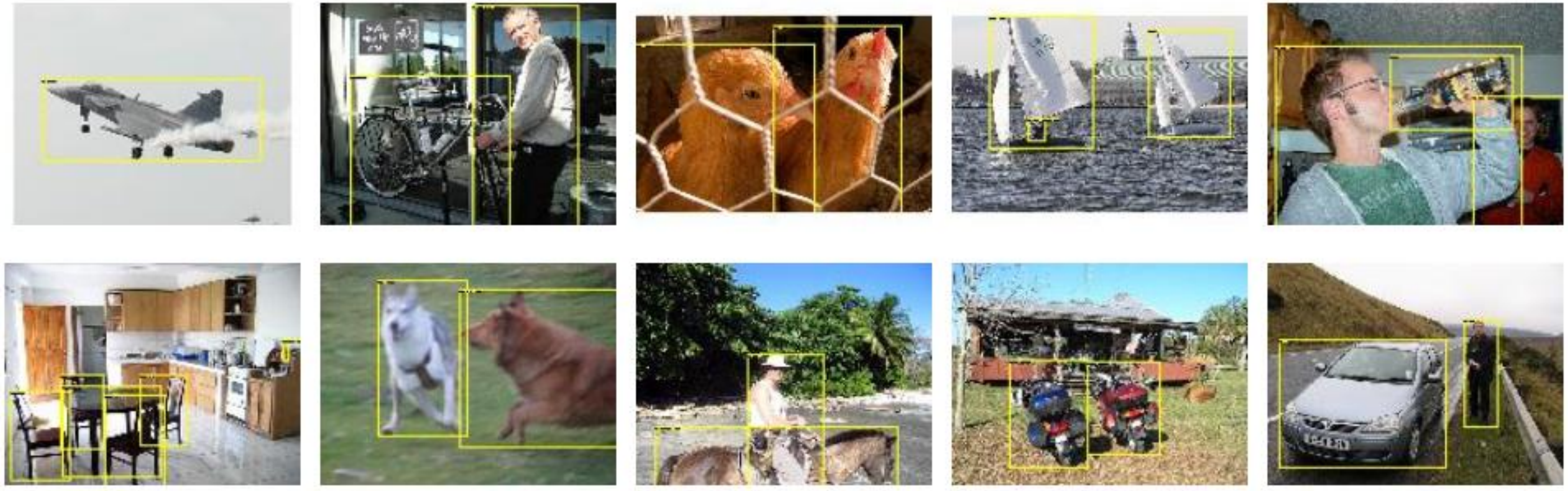
http://host.robots.ox.ac.uk/pascal/VOC/voc2007/

VOC

- The PASCAL **V**isual **O**bject **C**lasses Challenge 2007

http://host.robots.ox.ac.uk/pascal/VOC/voc2007/

- The goal of this challenge is to recognize objects from a number of visual object classes in realistic scenes.



VOC

- The PASCAL **V**isual **O**bject **C**lasses Challenge 2007

  http://host.robots.ox.ac.uk/pascal/VOC/voc2007/

- The goal of this challenge is to recognize objects from a number of visual object classes in realistic scenes.

- 20 classes: person, animal, vehicle, etc.

RCNN Results

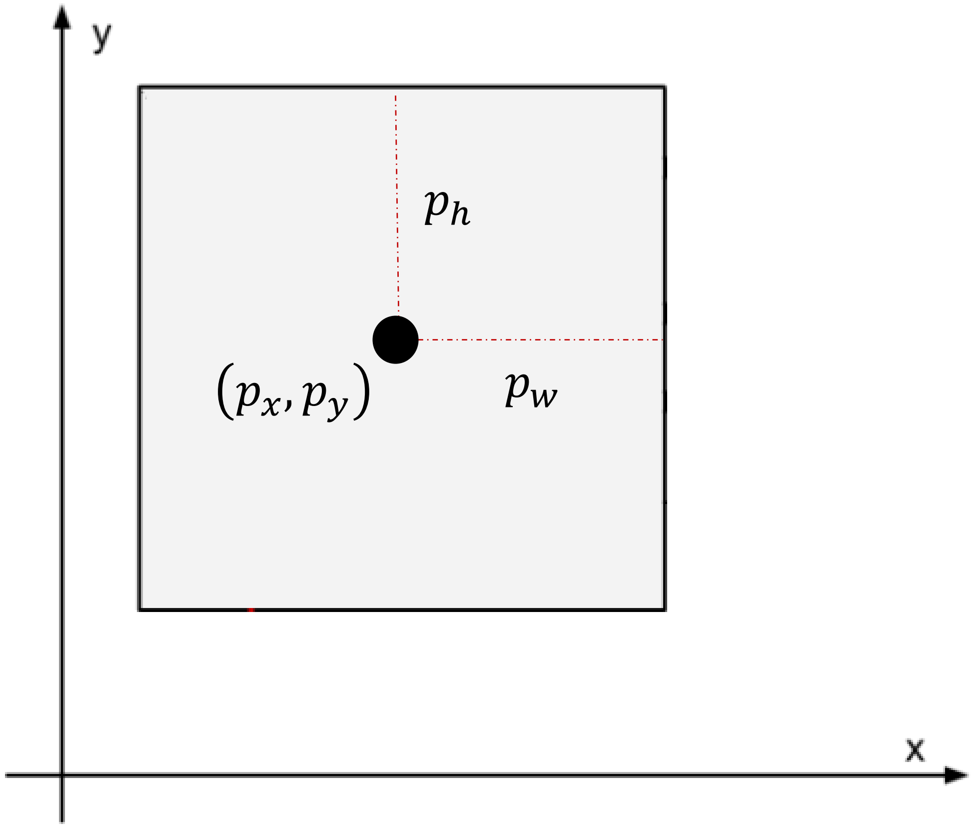
Method	VOC 2007/ AP
DPM v5 (Girshick et al. 2011)	33.7%
Regionlets (Wang et al. 2013)	41.7%
RCNN (AlexNet)	54.2%
R-CNN (AlexNet)+BB	58.5%
R-CNN (VGGNet)	62.2%
R-CNN (VGGNet)+BB	66.0%

RCNN Results

Method	VOC 2007/ AP
DPM v5 (Girshick et al. 2011)	33.7%
Regionlets (Wang et al. 2013)	41.7%
RCNN (AlexNet)	54.2%
R-CNN (AlexNet)+BB	58.5%
R-CNN (VGGNet)	62.2%
R-CNN (VGGNet)+BB	66.0%

Bounding Box Regression

- Predicted bounding box coordinates: $\boldsymbol{p} = (p_x, p_y, p_w, p_h)$



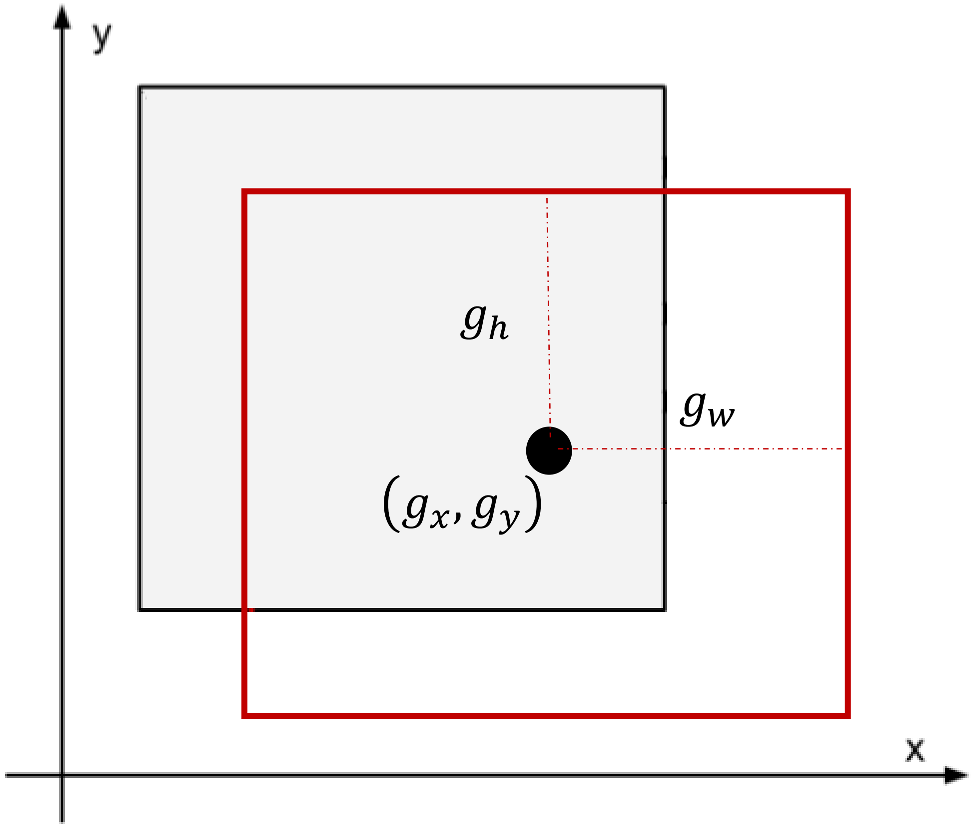
(p_x, p_y): center coordinates

p_w: half of width

p_h: half of height

Bounding Box Regression

- Ground truth bounding box coordinates: $\mathbf{g} = (g_x, g_y, g_w, g_h)$



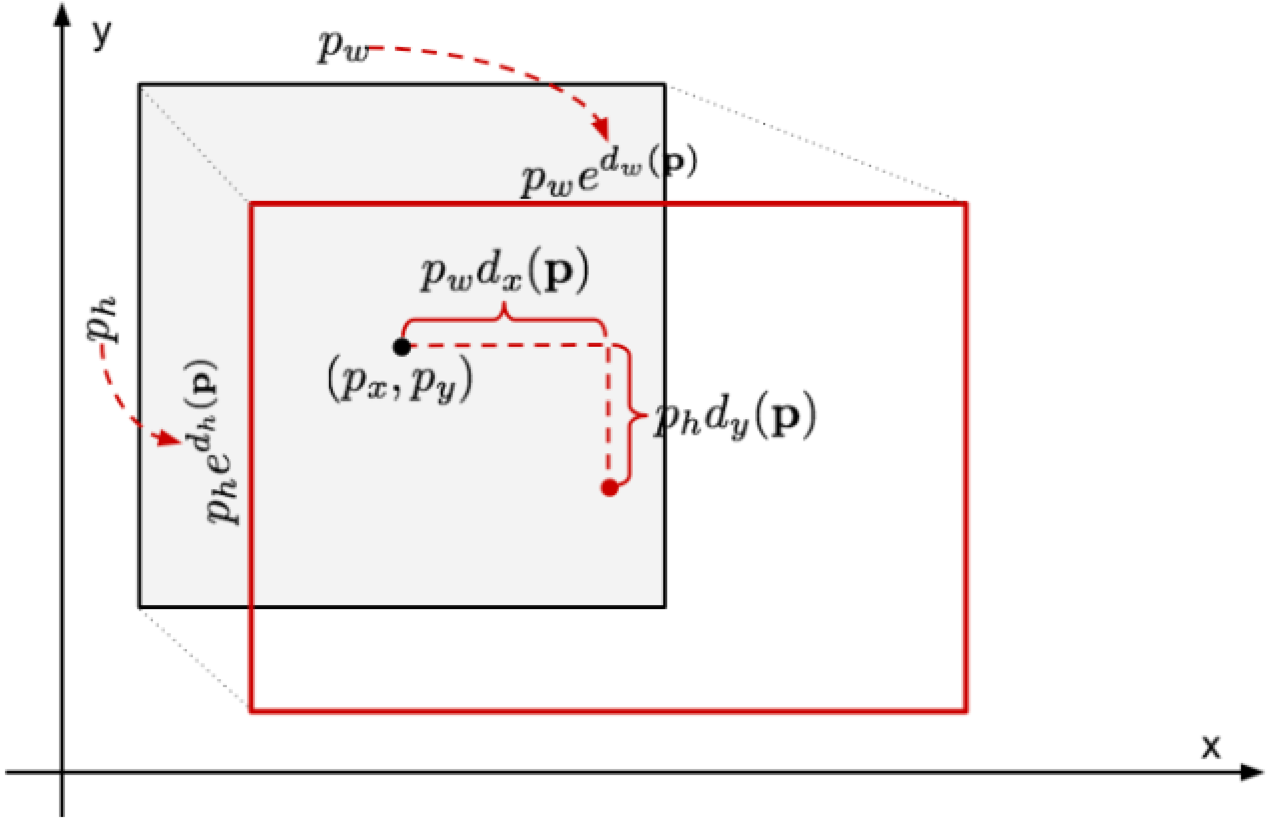
(g_x, g_y): center coordinates

g_w: half of width

g_h: half of height

Bounding Box Regression

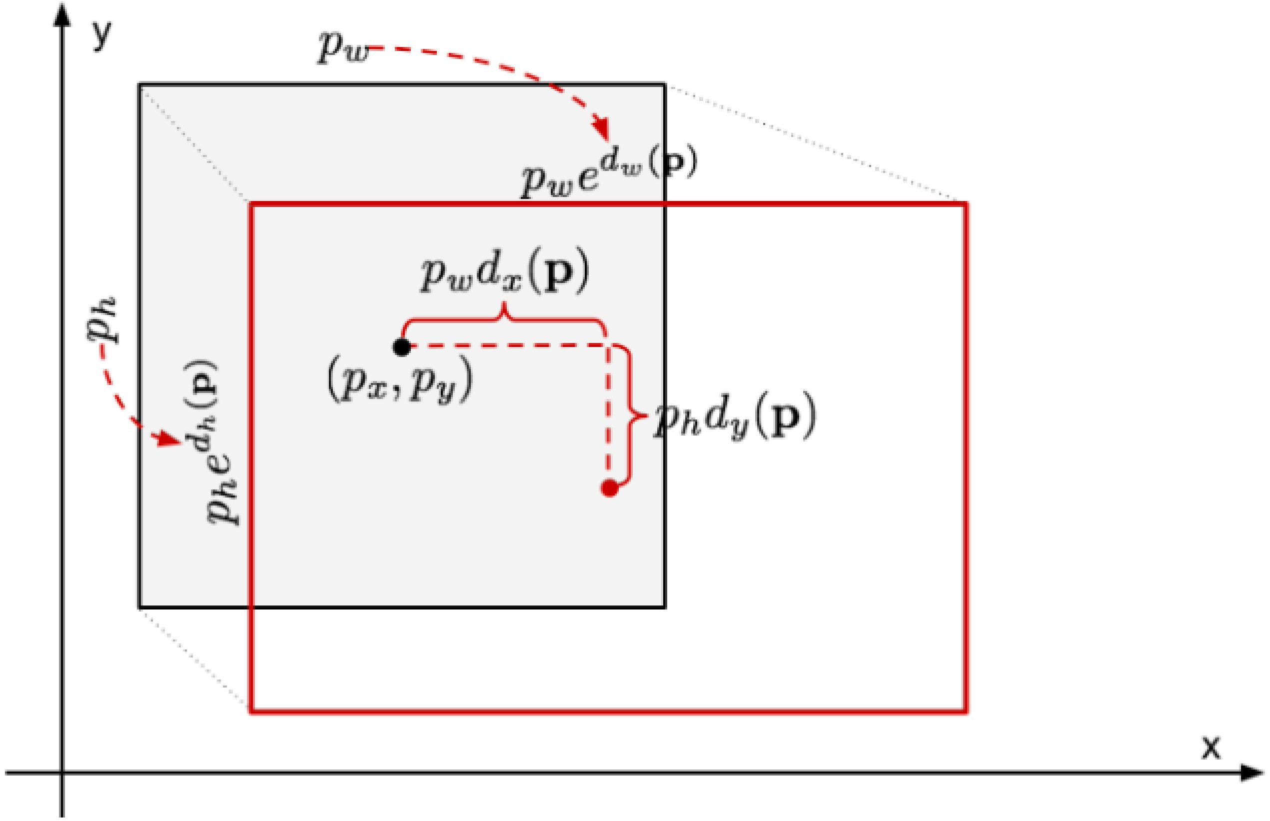
- Regressor learns a scale-invariant transformation between two centers and log-scale transformation between widths and



$$
\begin{aligned}
\hat{g}_x &= p_w d_x(\mathbf{p}) + p_x \\
\hat{g}_y &= p_h d_y(\mathbf{p}) + p_y \\
\hat{g}_w &= p_w \exp(d_w(\mathbf{p})) \\
\hat{g}_h &= p_h \exp(d_h(\mathbf{p}))
\end{aligned}
$$

Bounding Box Regression

- Regressor learns a scale-invariant transformation between two centers and log-scale transformation between widths and

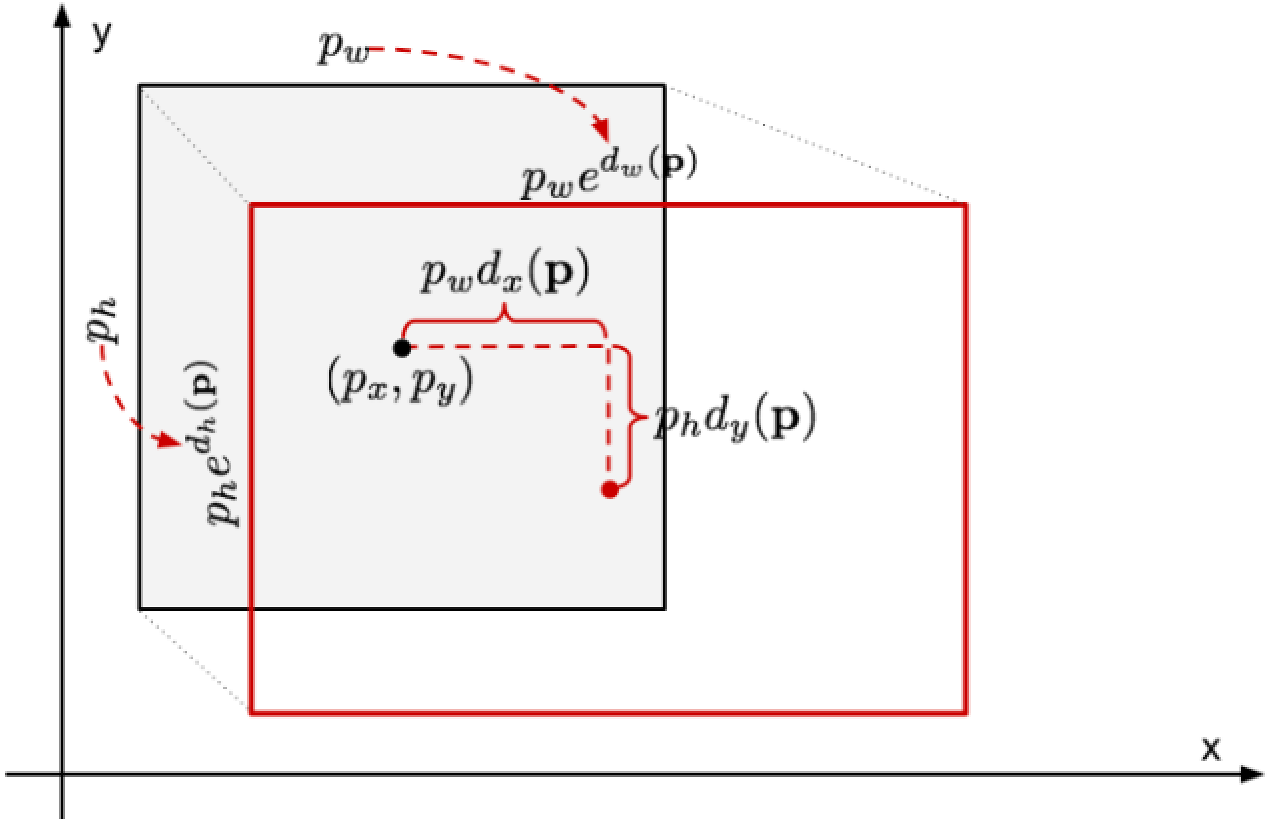


$$
\begin{aligned}
\hat{g}_x &= p_w \boxed{d_x(\mathbf{p})} + p_x \\
\hat{g}_y &= p_h d_y(\mathbf{p}) + p_y \\
\hat{g}_w &= p_w \exp(d_w(\mathbf{p})) \\
\hat{g}_h &= p_h \exp(d_h(\mathbf{p}))
\end{aligned}
$$

Scale function of p

Bounding Box Regression

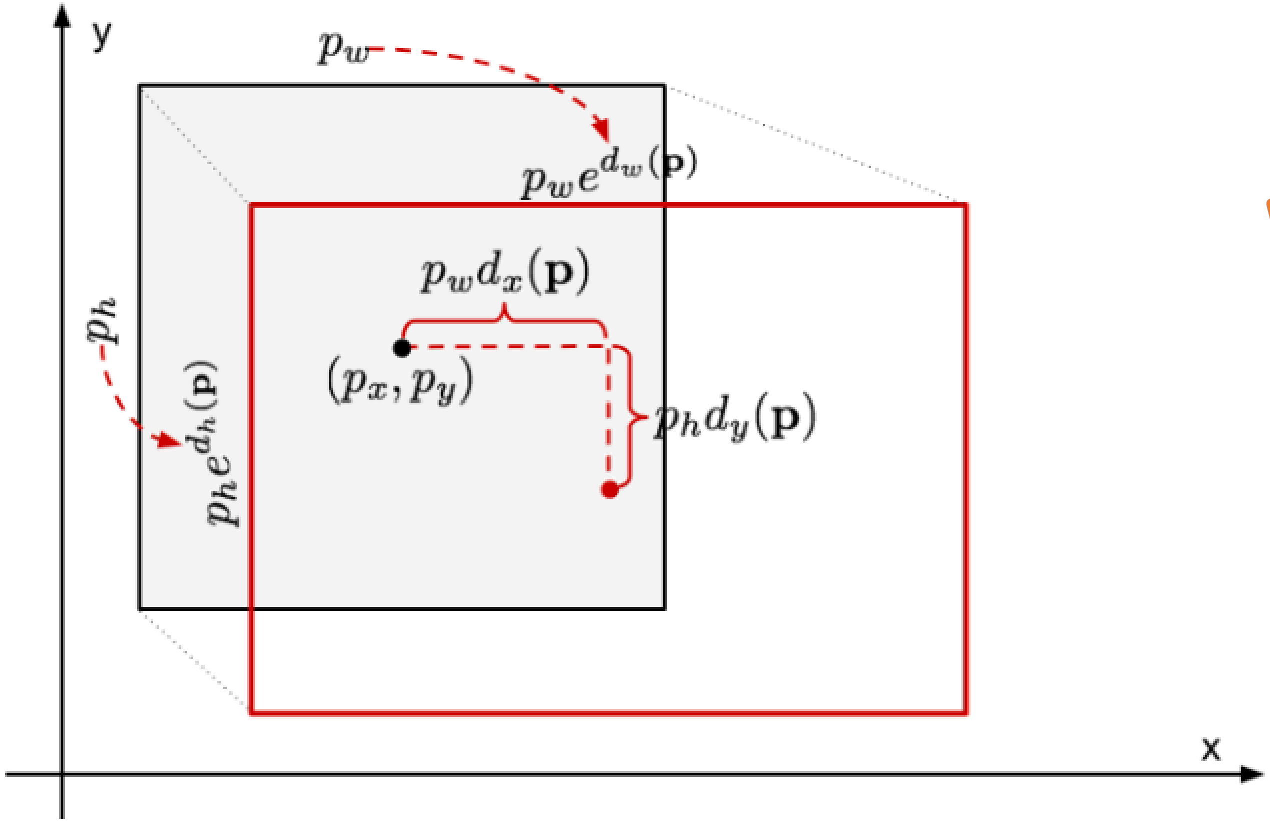
- Benefit is that all $d_i(p)$ where $i \in \{x, y, w, h\}$ attain values between $[-\infty, +\infty]$. The targets for them to learn are:



$$
\begin{aligned}
t_x &= (g_x - p_x)/p_w \\
t_y &= (g_y - p_y)/p_h \\
t_w &= \log(g_w/p_w) \\
t_h &= \log(g_h/p_h)
\end{aligned}
$$

Bounding Box Regression

- Benefit is that all $d_i(p)$ where $i \in \{x, y, w, h\}$ attain values between $[-\infty, +\infty]$. The targets for them to learn are:



What else?

Bounding Box Regression

- A standard regression model can solve the problem by minimizing the **S**um of **S**quared **E**rrors (**SSE**) loss with regularization.

$$\mathcal{L}_{\text{reg}} = \sum_{i \in \{x,y,w,h\}} (t_i - d_i(\mathbf{p}))^2 + \lambda \|\mathbf{w}\|^2$$

Bounding Box Regression

- A standard regression model can solve the problem by minimizing the **S**um of **S**quared **E**rrors (**SSE**) loss with regularization.

$$\mathcal{L}_{\text{reg}} = \sum_{i \in \{x,y,w,h\}} (t_i - d_i(\mathbf{p}))^2 + \boxed{\lambda \|\mathbf{w}\|^2}$$

Regularize to avoid large weights

Bounding Box Regression

- A standard regression model can solve the problem by minimizing the **S**um of **S**quared **E**rrors (**SSE**) loss with regularization.

$$\mathcal{L}_{\text{reg}} = \sum_{i \in \{x,y,w,h\}} (t_i - d_i(\mathbf{p}))^2 + \lambda \|\mathbf{w}\|^2$$

What is the problem?

$$\boldsymbol{p} = (p_x, p_y, p_w, p_h)$$

Bounding Box Regression

- Regressor should receive some information about the image so that it can correct the bounding box prediction.

$$\mathcal{L}_{\text{reg}} = \sum_{i \in \{x,y,w,h\}} (t_i - d_i(\mathbf{p}))^2 + \lambda \|\mathbf{w}\|^2$$

$$\mathbf{w}_\star = \underset{\hat{\mathbf{w}}_\star}{\operatorname{argmin}} \sum_i^N (t_\star^i - \hat{\mathbf{w}}_\star^{\mathrm{T}} \phi_5(P^i))^2 + \lambda \left\|\hat{\mathbf{w}}_\star\right\|^2$$

Bounding Box Regression

- For each predicted bounding box P^i, we retrieve pool5 features of the network and we apply a linear model using learnable parameters in vector $w_\star$.

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

RCNN Timing

RCNN (VGG)	Time
Train	84 hours
Test	47 Second/Image

Summary

- We learned a traditional object detection technique (HOG).

Summary

- We learned a traditional object detection technique (HOG).
  - Basic idea: slide bounding boxes at different locations with different sizes, define a feature (handcrafted histogram), apply SVM to categorize features in different boxes.

Summary

- We learned a traditional object detection technique (HOG).
  - Basic idea: slide bounding boxes at different locations with different sizes, define a feature (handcrafted histogram), apply SVM to categorize features in different boxes.

- We learned a deep learning technique.

Summary

- We learned a traditional object detection technique (HOG).
  - Basic idea: slide bounding boxes at different locations with different sizes, define a feature (handcrafted histogram), apply SVM to categorize features in different boxes.

- We learned a deep learning technique.
  - Pre-train an image classification network, use its features for bounding boxes with different locations and different sizes warped to a specific size, use SVM to classify.

Next…

- Other types of object detection techniques and segmentations in the next class.