

Computer Vision

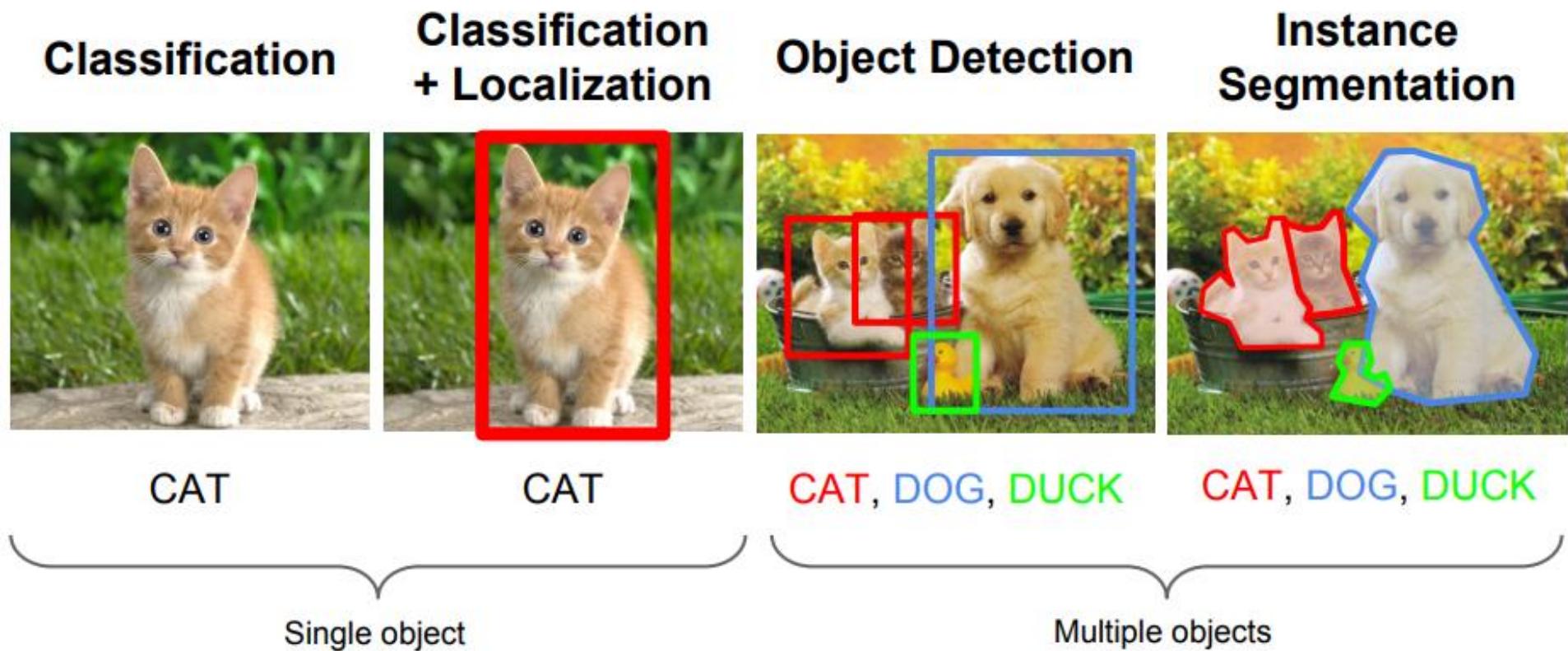
Object Detection

Dr. Mrinmoy Ghorai

**Indian Institute of Information Technology
Sri City, Chittoor**



Roadmap

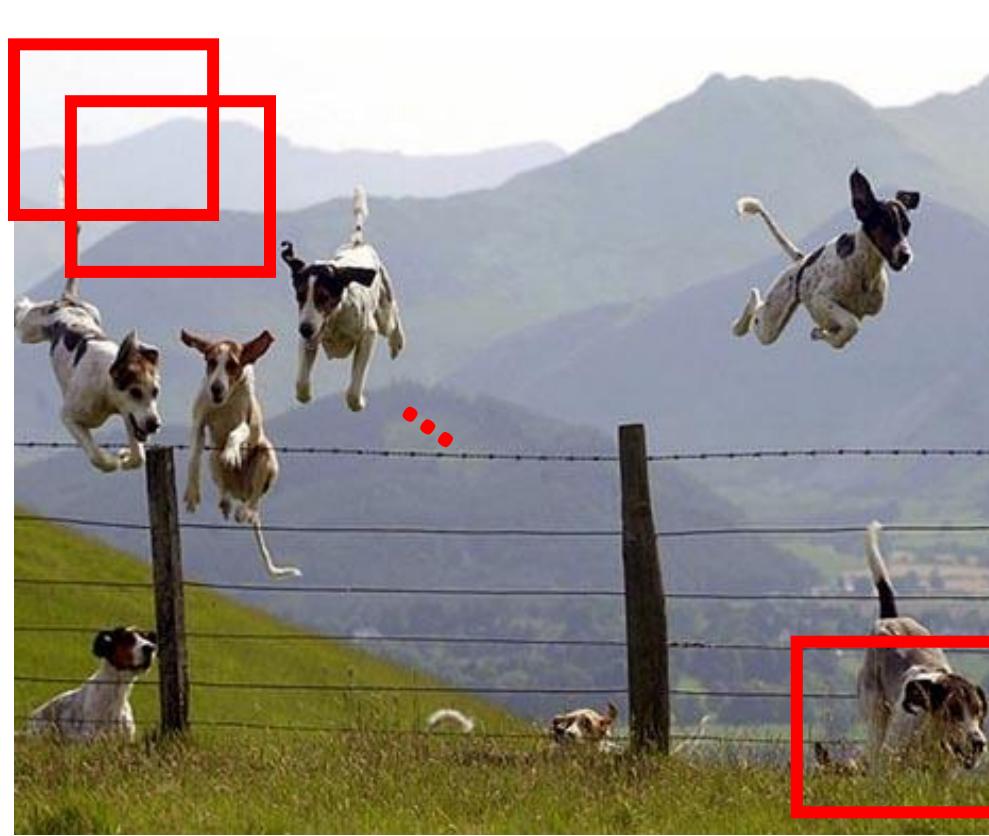


Today's class

- Overview of object category detection
- Traditional methods
 - Dalal-Triggs detector (basic concept)
 - Viola-Jones detector (cascades, integral images)
- Deep learning methods
 - Two-stage: R-CNN
 - One-stage: YOLO, SSD, Retina Net

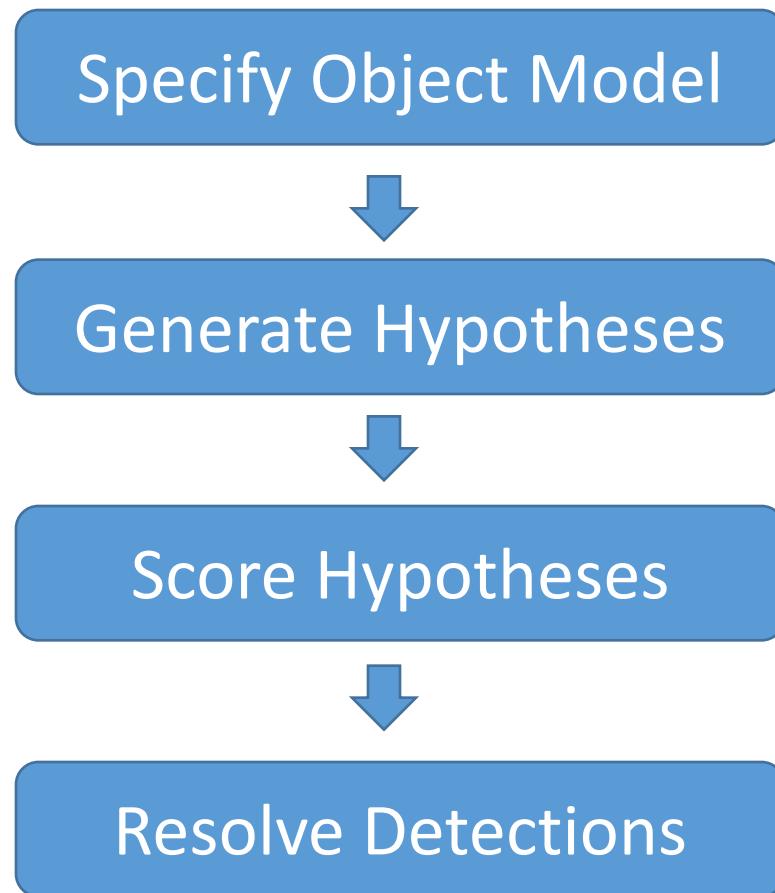
Object Category Detection

- Focus on object search: “Where is it?”
- Build templates that quickly differentiate object patch from background patch



**Object or
Non-Object?**

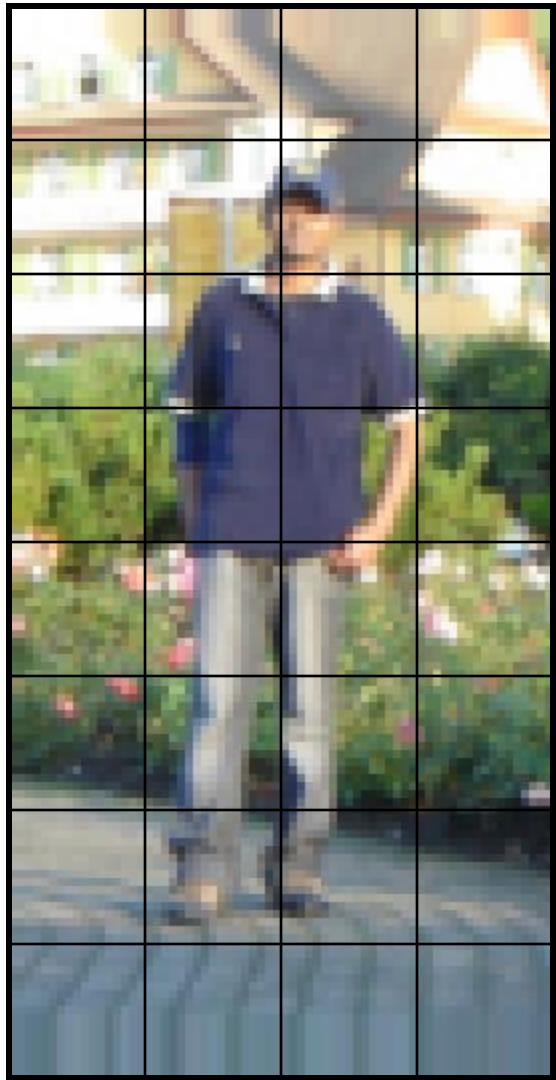
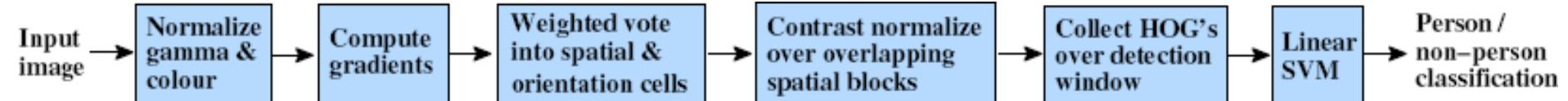
General Process of Object Recognition

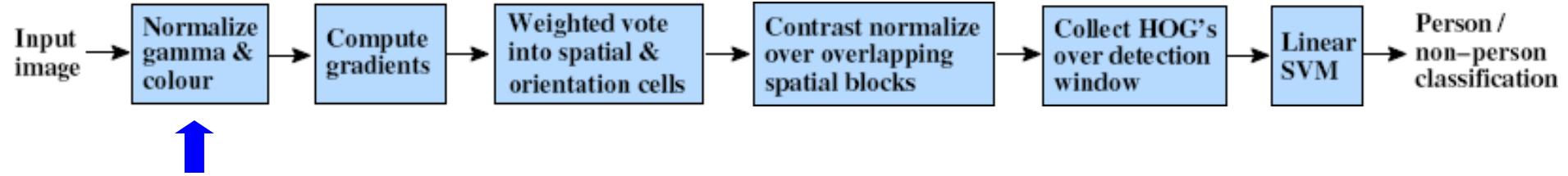


Dalal-Triggs detector: HOG

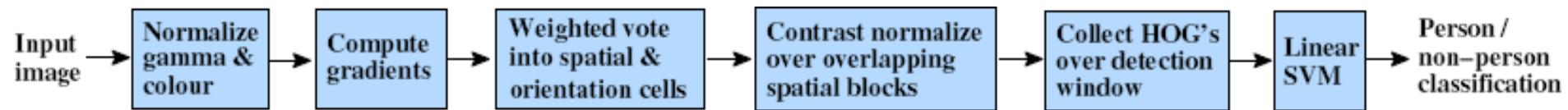


1. Extract fixed-sized (64x128 pixel) window at each position and scale
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

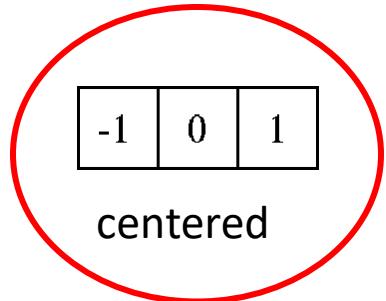




- Tested with
 - Grayscale
 - RGB
 - LAB
- Slightly better performance vs. grayscale
- Gamma Normalization and Compression
 - Square root
 - Log
- Superior performance vs. no adjustment



Outperforms

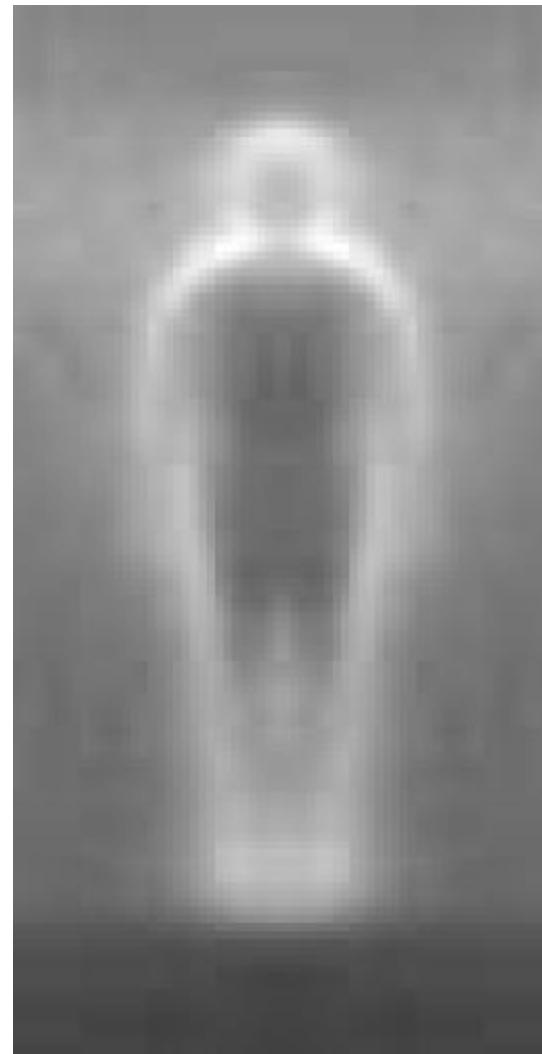


-1	1
----	---

uncentered

1	-8	0	8	-1
---	----	---	---	----

cubic-corrected

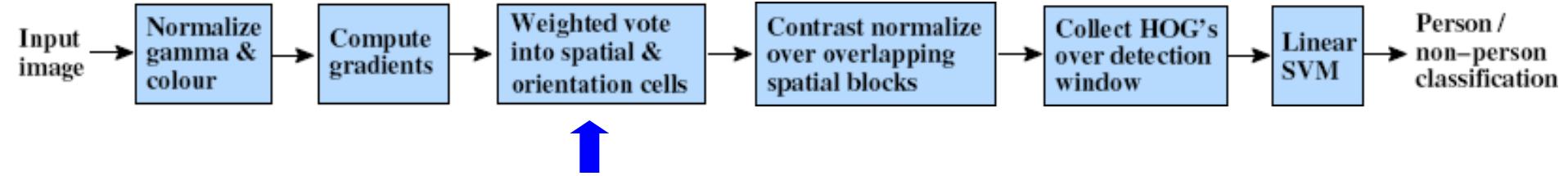


0	1
-1	0

diagonal

-1	0	1
-2	0	2
-1	0	1

Sobel

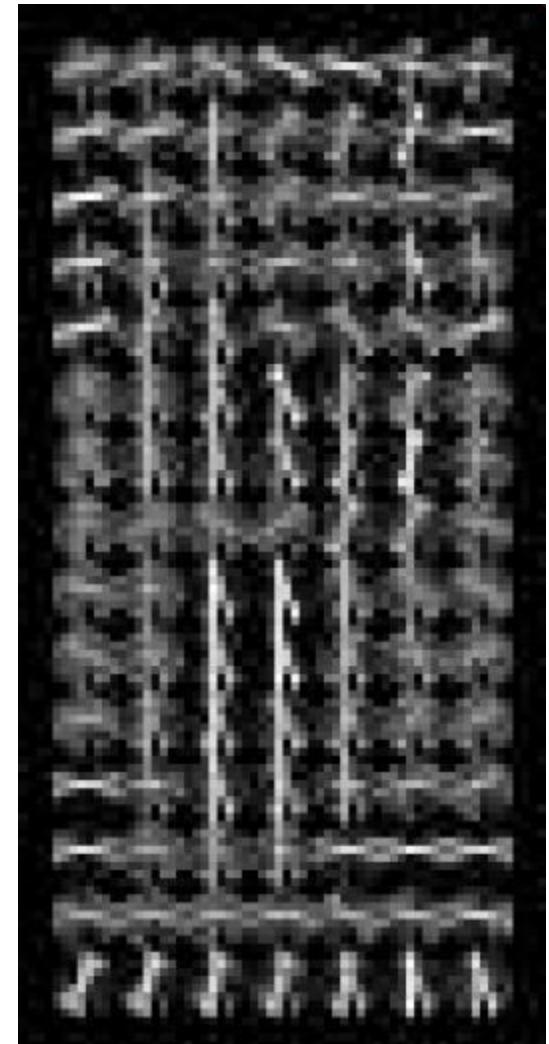


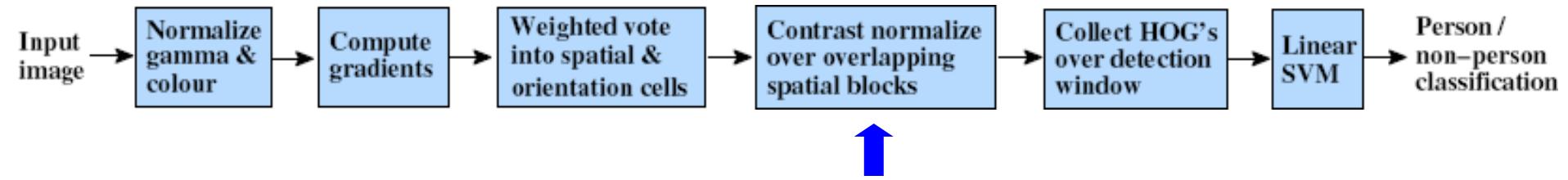
- Histogram of gradient orientations

Orientation: 9 bins
(in 0-180 degree)

Histograms in
8x8 pixel cells

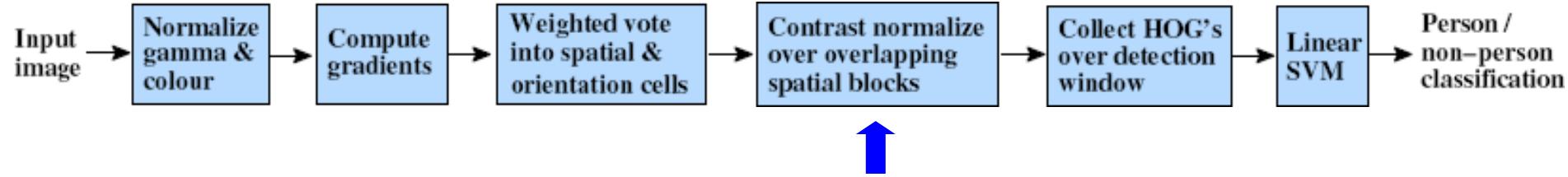
- Votes weighted by magnitude





Consider **16×16 pixel blocks** (i.e. four **8×8 pixel cells**)
with block spacing stride of 8 pixels

Total: 7×15 blocks (as Window size: 64×128 pixel)



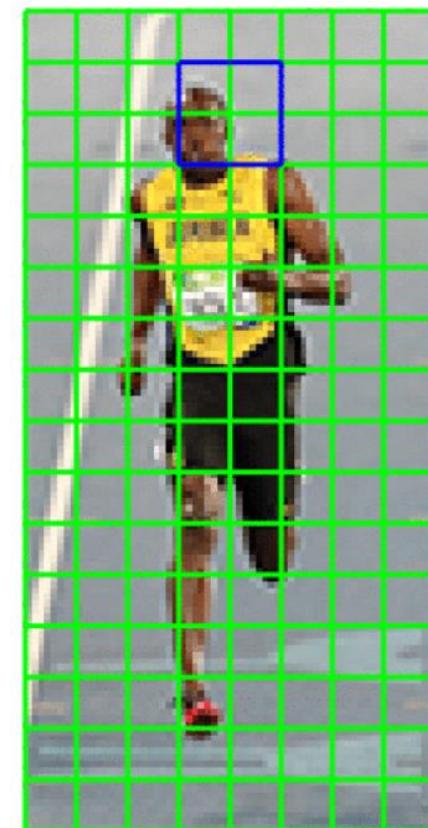
Consider **16×16 pixel blocks** (i.e. four 8×8 pixel cells)
with block spacing stride of 8 pixels

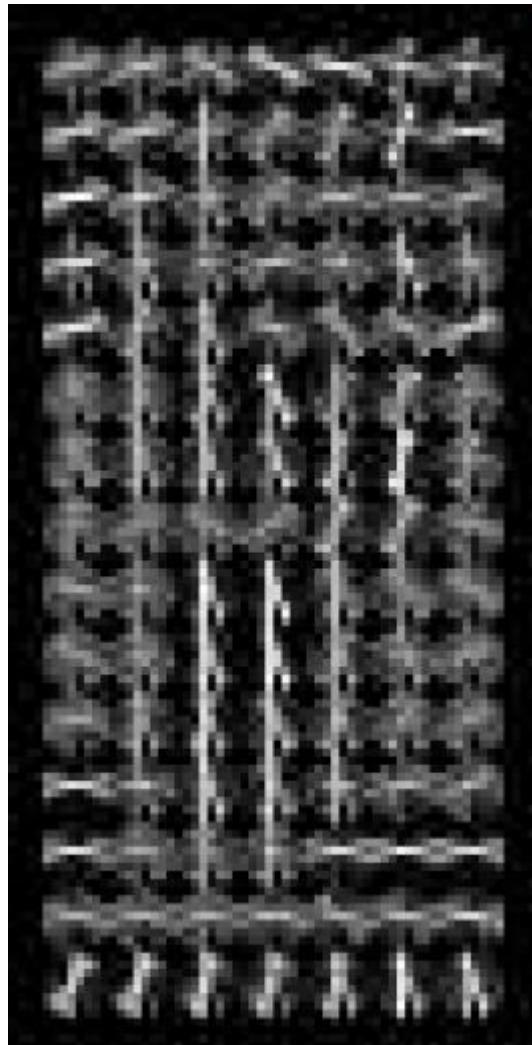
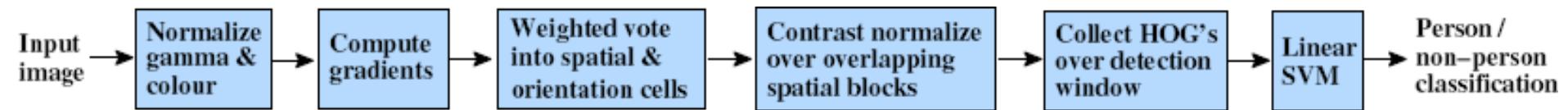
Total: 7×15 blocks (as Window size: 64×128 pixel)

Compute the normalized histograms in a block.

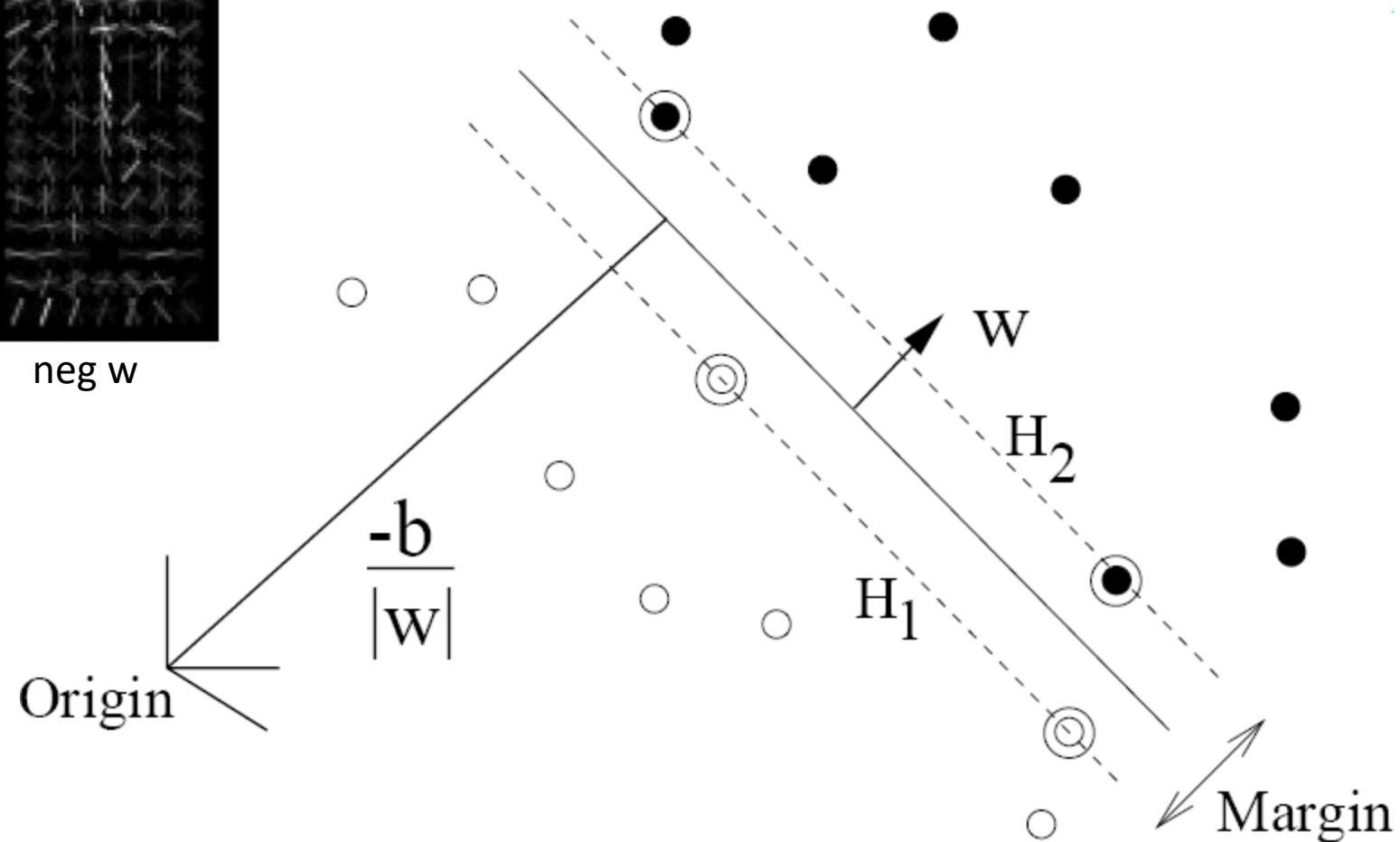
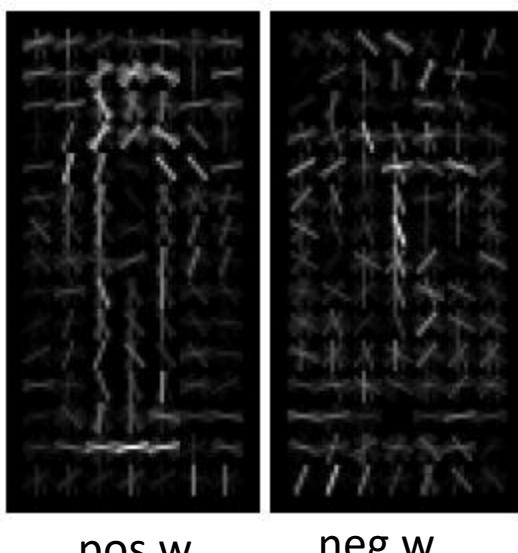
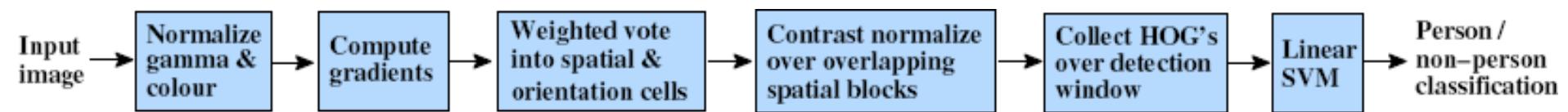
Normalization (different approaches such as

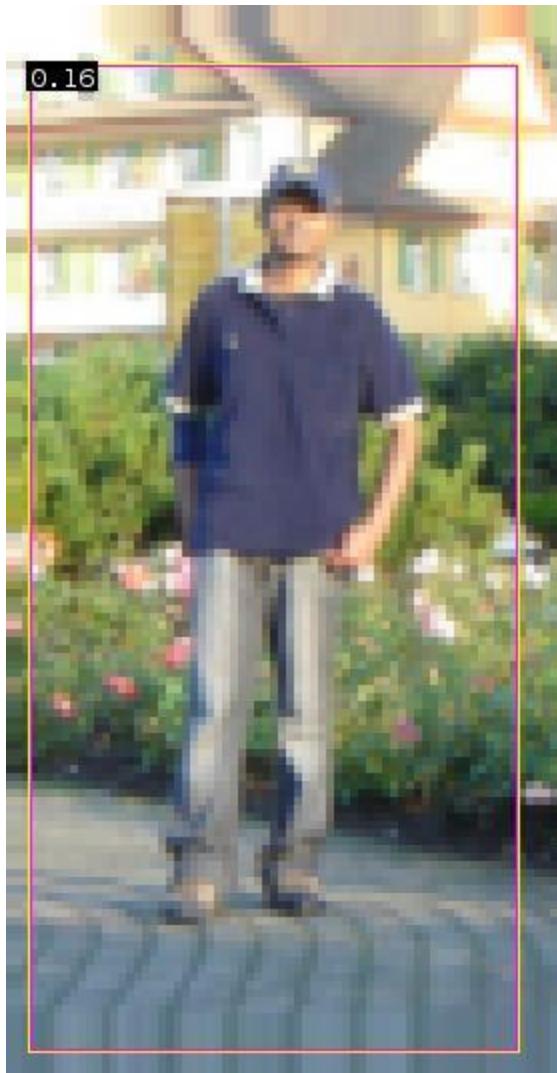
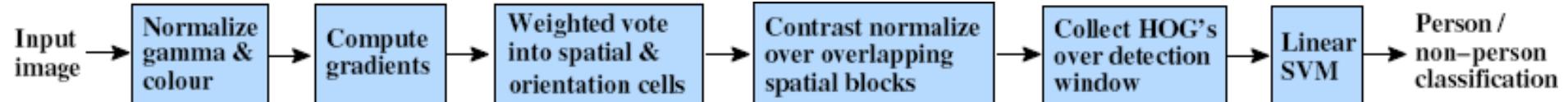
- (a) L2-norm,
- (b) L2-Hys, L2-norm followed by clipping,
- (c) L1-norm,
- (d) L1-sqrt, L1-norm followed by square root)





orientations
 $\# \text{ features} = 15 \times 7 \times 9 \times 4 = 3780$
blocks # cells





$$0.16 = w^T x - b$$

$$\text{sign}(0.16) = 1$$

\Rightarrow pedestrian

Detection examples

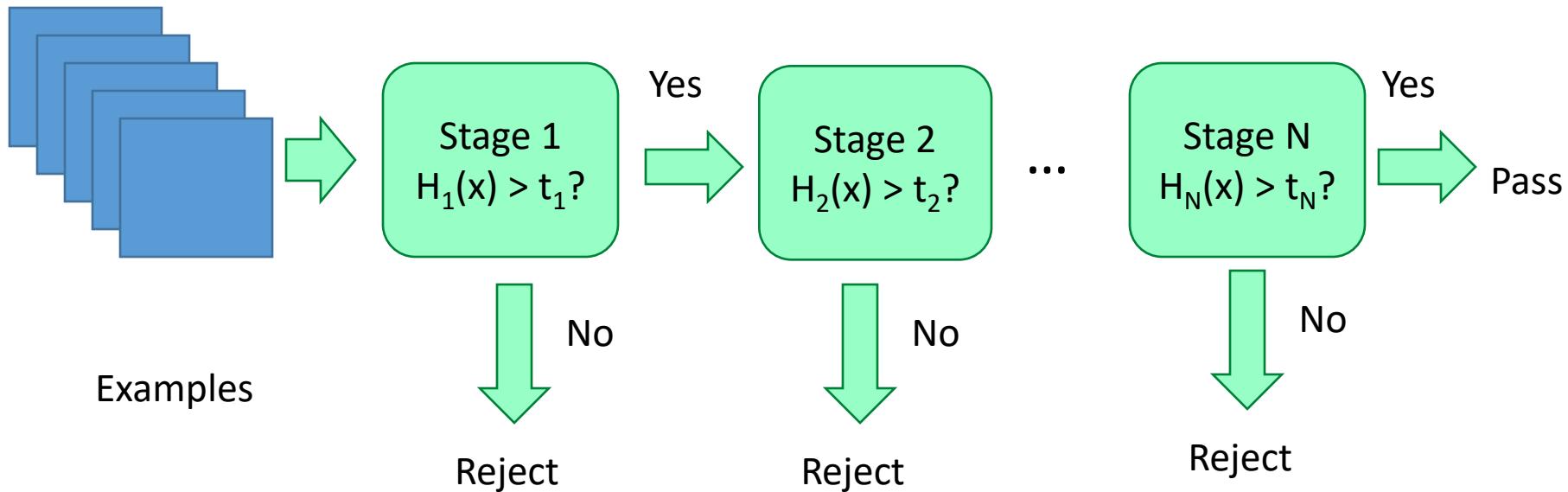


Viola-Jones sliding window detector

Fast detection through two mechanisms

- Quickly eliminate unlikely windows
 - A megapixel image has $\sim 10^6$ pixels and a comparable number of candidate object locations
 - Positive instances are rare: 0–10 per image
- Use features that are fast to compute
 - Utilization of integral image

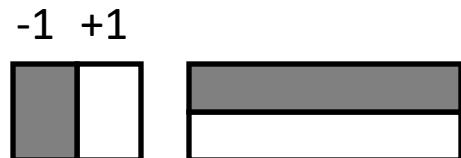
Cascade for Fast Detection



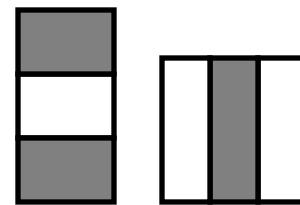
- Choose threshold for low false negative rate
- Fast classifiers early in cascade
- Slow classifiers later, but most examples don't get there

Features that are fast to compute

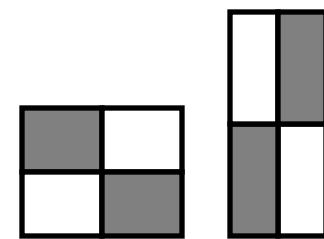
- “Haar-like features”
 - Differences of sums of intensity
 - Thousands, computed at various positions and scales within detection window



Two-rectangle features



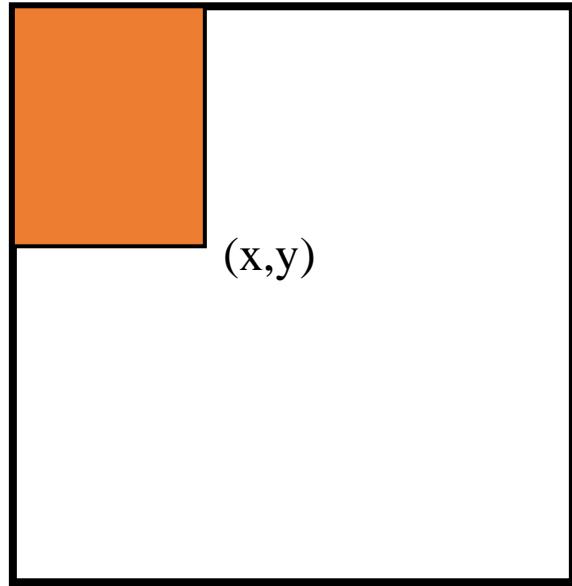
Three-rectangle features



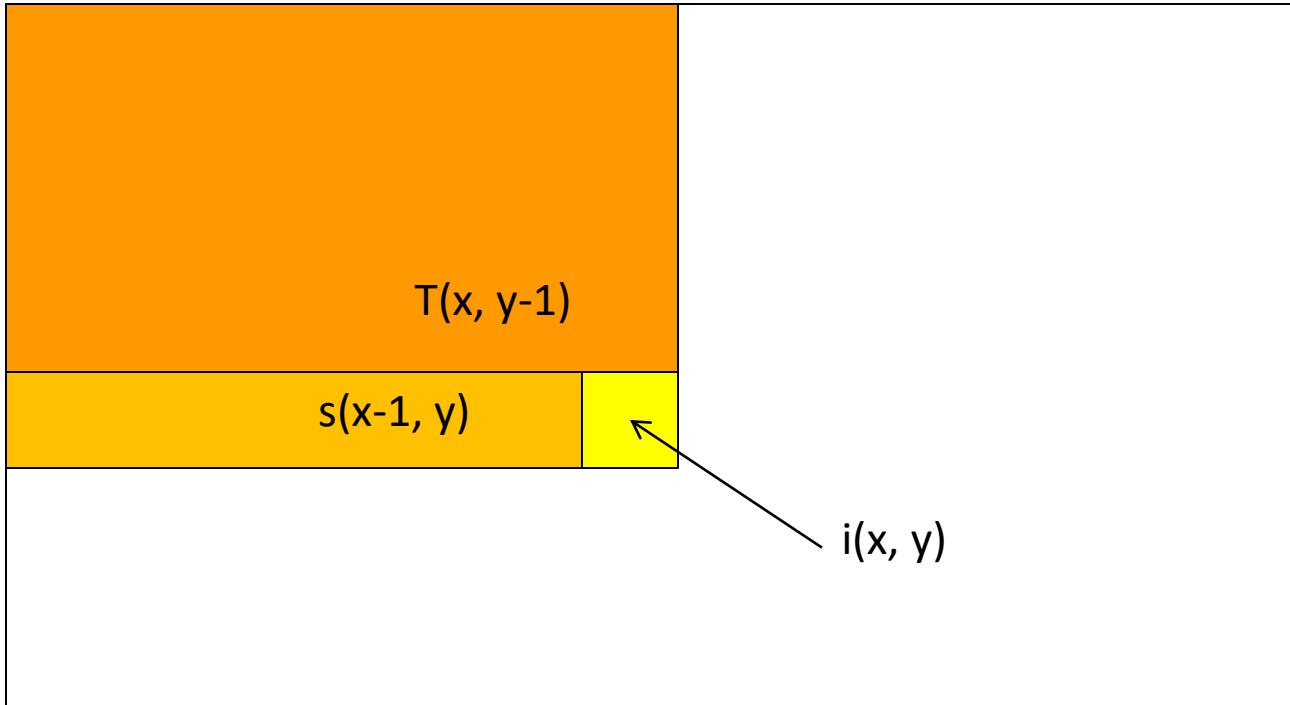
Etc.

Integral Images

- The *integral image* computes a value at each pixel (x,y) that is the sum of the pixel values above and to the left of (x,y) , inclusive
- This can quickly be computed in one pass through the image



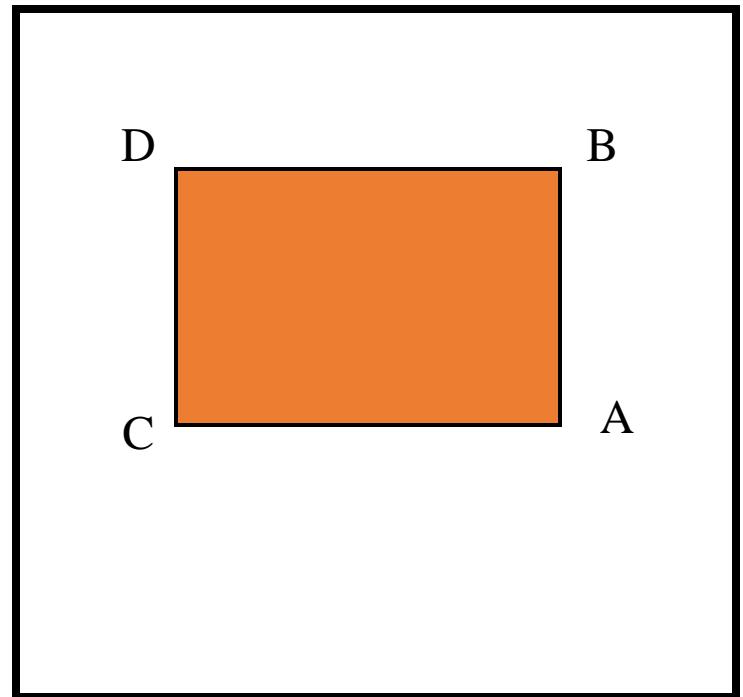
Integral Images



- Cumulative row sum: $s(x, y) = s(x-1, y) + i(x, y)$
- Integral image: $T(x, y) = T(x, y-1) + s(x, y)$

Integral Images

- Let A,B,C,D be the values of the integral image at the corners of a rectangle
- What is the sum of pixel values within the rectangle?

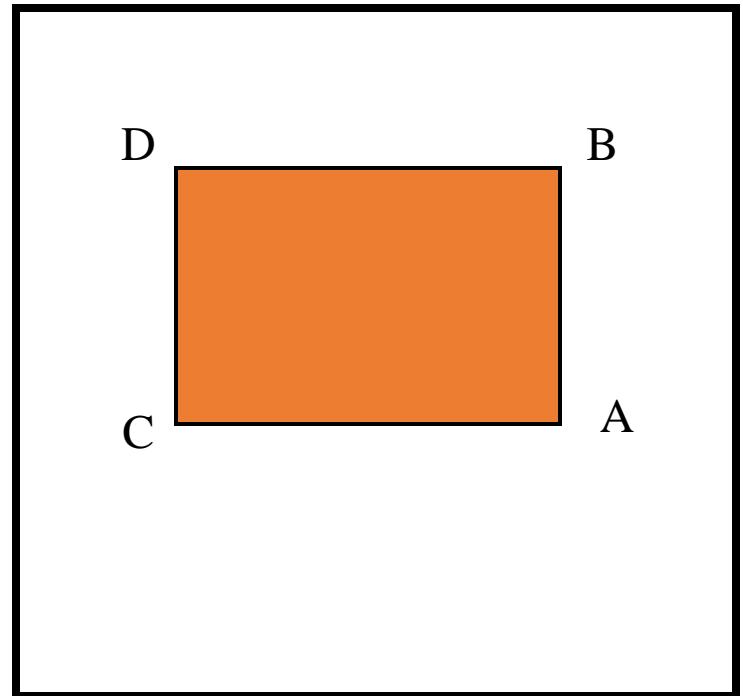


Integral Images

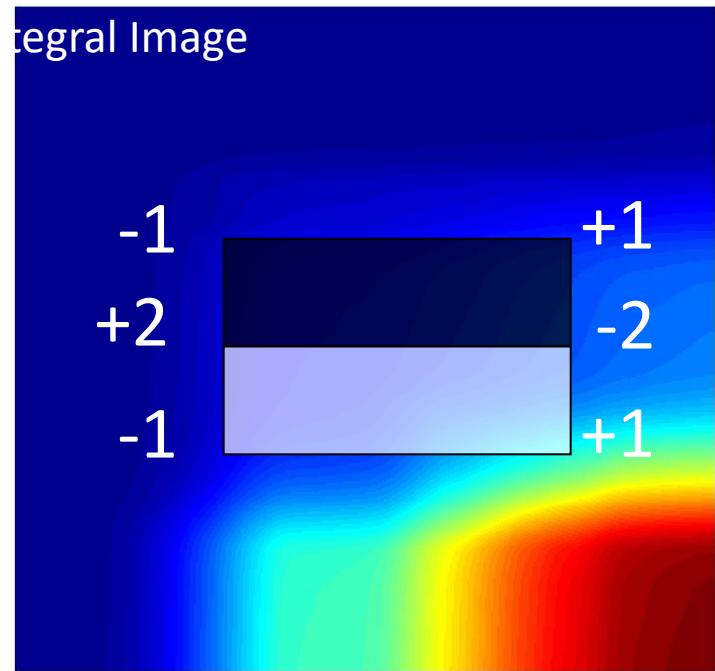
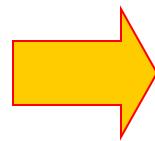
- Let A,B,C,D be the values of the integral image at the corners of a rectangle
- What is the sum of pixel values within the rectangle?

$$\text{sum} = A - B - C + D$$

- Only 3 additions are required for any size of rectangle!



Integral Images



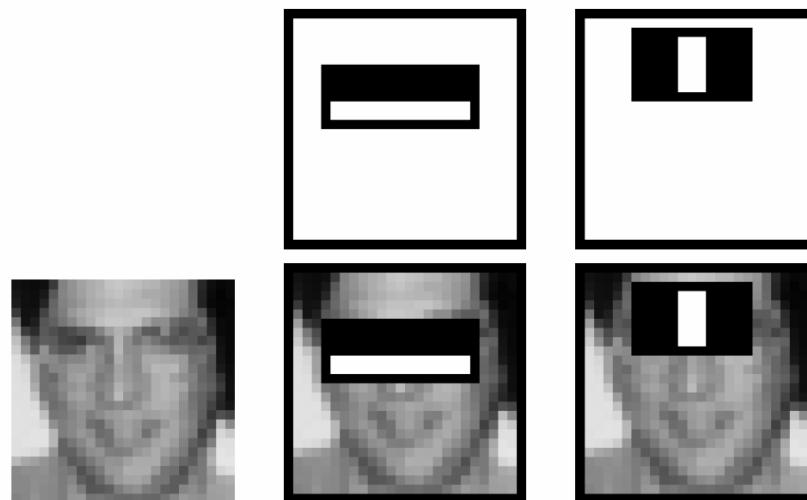
Feature Selection

- For a 24x24 detection region, the number of possible rectangle features is ~160,000!
- *Boosting* (AdaBoost) is a classification scheme that combines *weak learners* into a more accurate *ensemble classifier*

Feature Selection

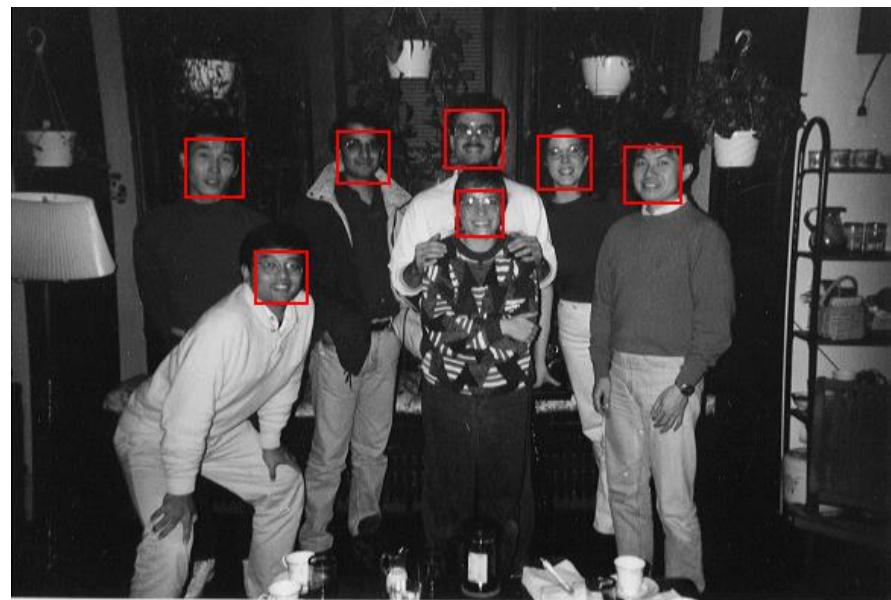
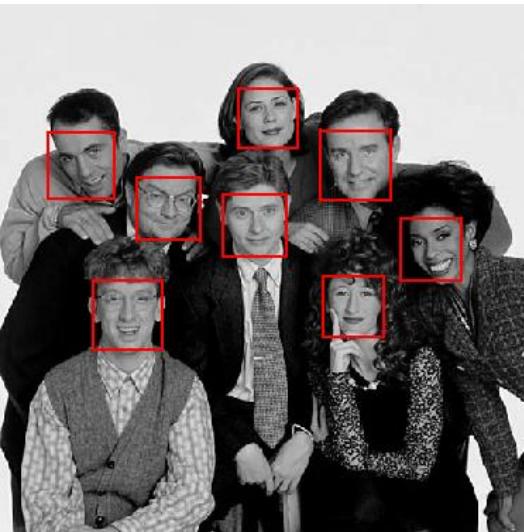
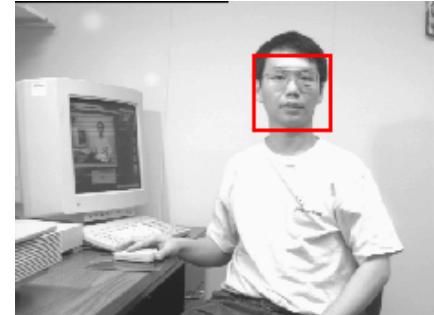
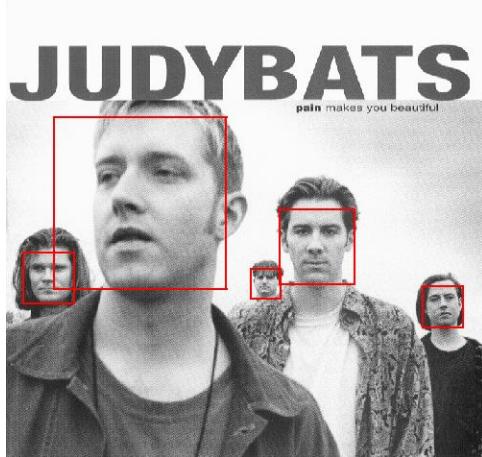
- For a 24x24 detection region, the number of possible rectangle features is ~160,000!
- *Boosting* (AdaBoost) is a classification scheme that combines *weak learners* into a more accurate *ensemble classifier*

First two features selected by boosting



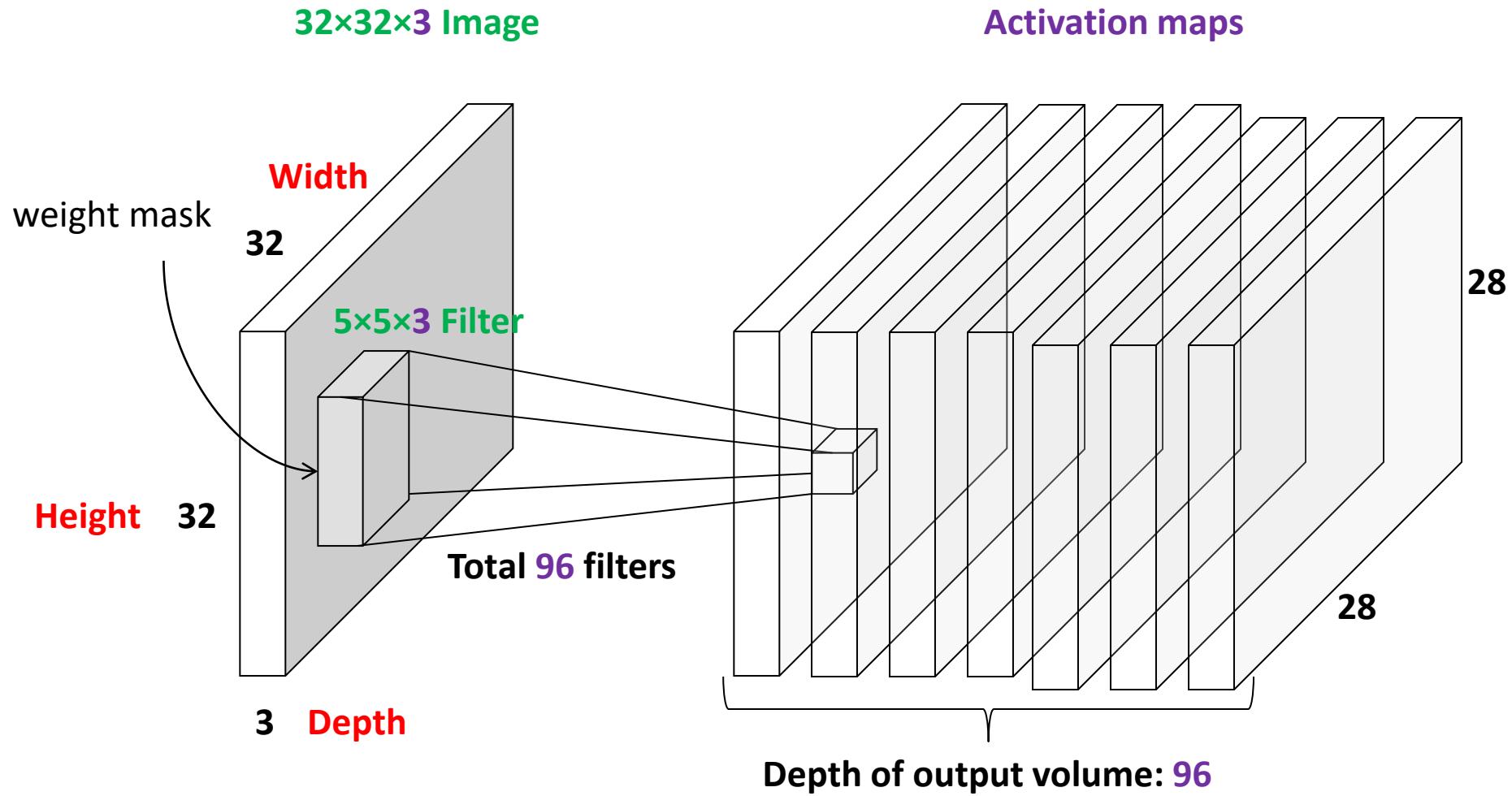
This feature combination can yield 100% detection rate and 50% false positive rate

Viola Jones Face Detection Results

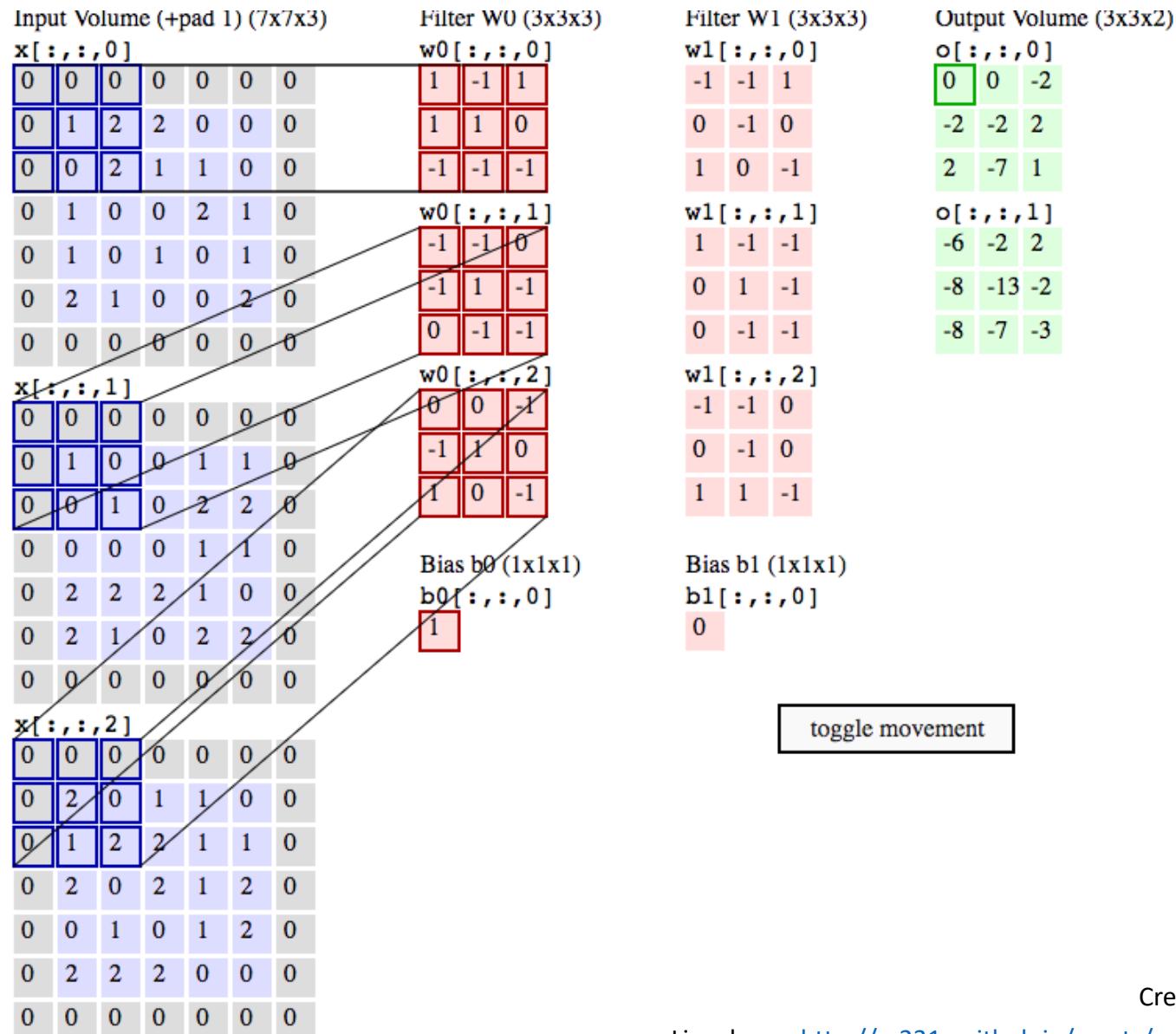


Convolutional Neural Network (Quick Recap)

Recap – Convolutional layer



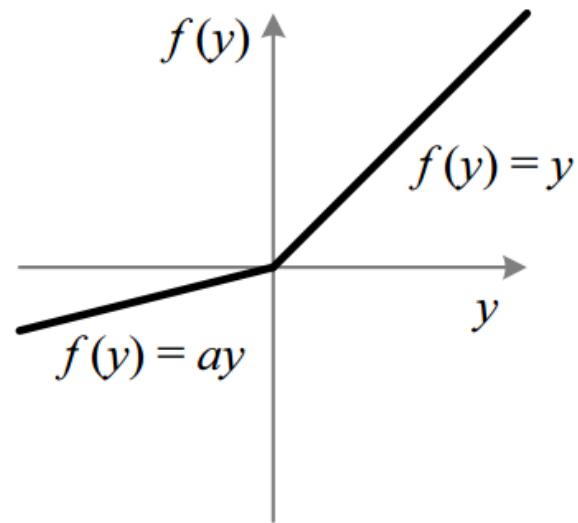
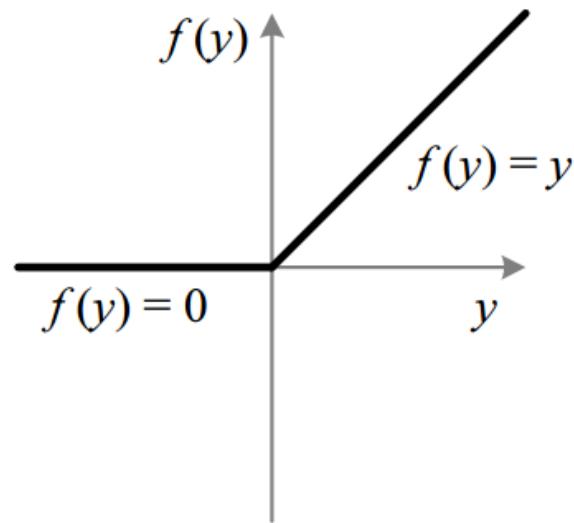
How it works?



Credit: Andrej Karpathy

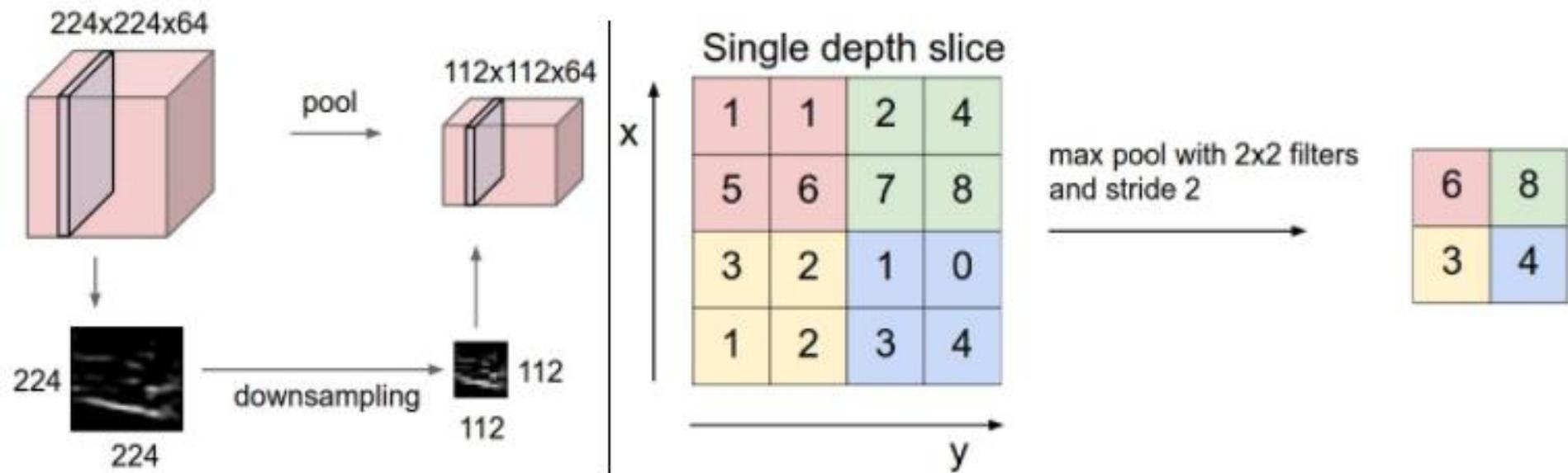
Live demo: <http://cs231n.github.io/assets/conv-demo/index.html>

Recap – Activation



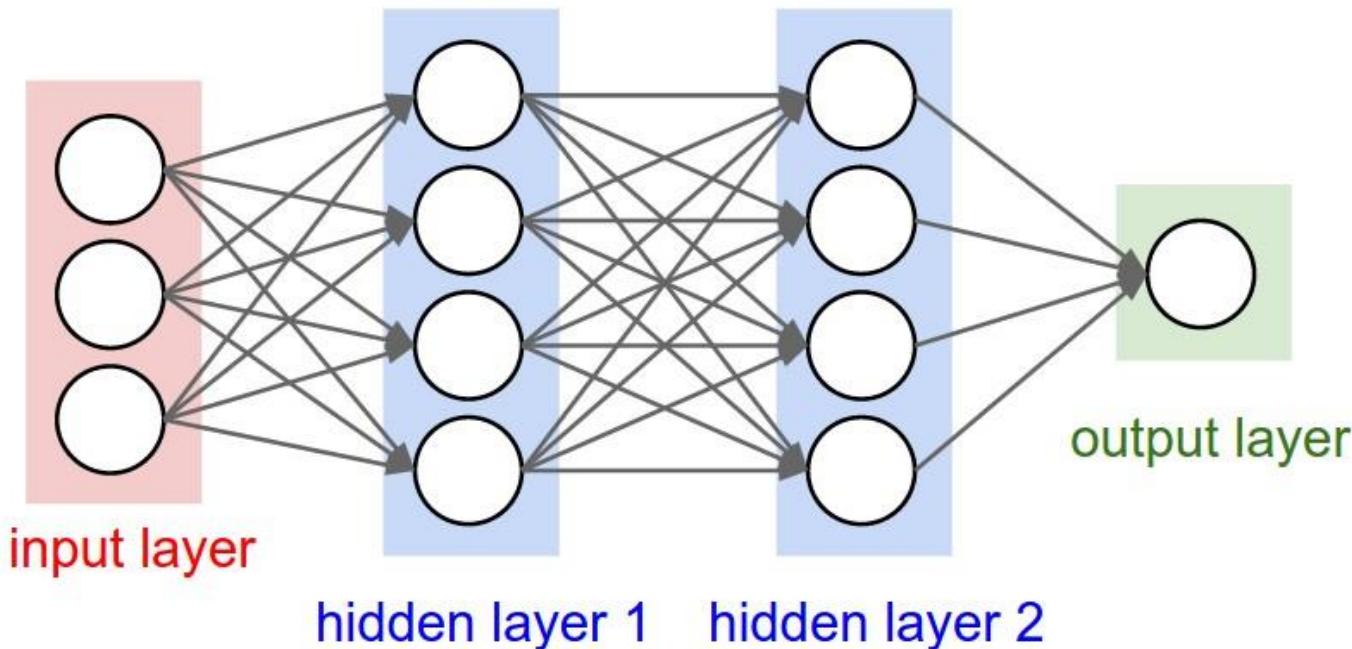
- Introduce the non-linearity

Recap – Pooling layer



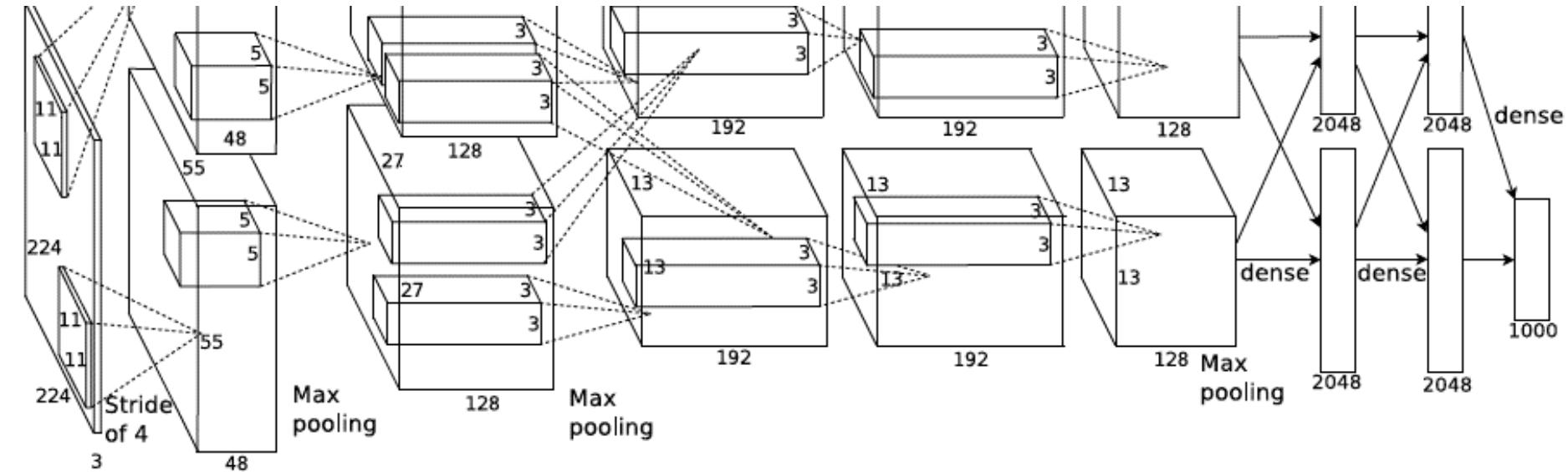
- Reduce the feature size
- Introduce a bit invariance (translation, rotation)

Recap – Fully-connected layer



- Each output node is connected to all the input nodes
- Fixed number of input nodes
- Fixed number of output nodes

Put them all together



- Train the deep convolutional neural net with simple chain-rule (a.k.a back propagation)

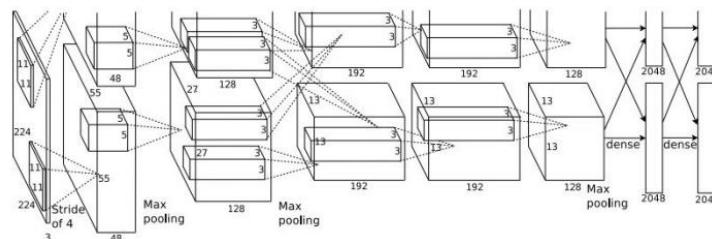
CNN methods for object detection

CNN as feature extractor



CNN as feature extractor

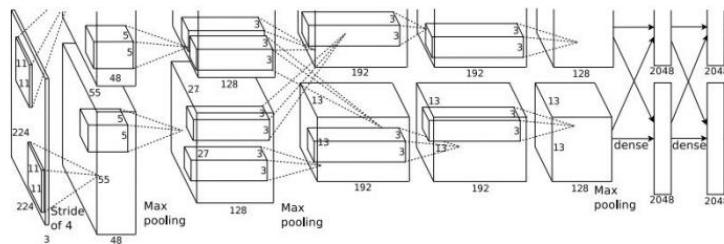
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? NO
Background? YES

CNN as feature extractor

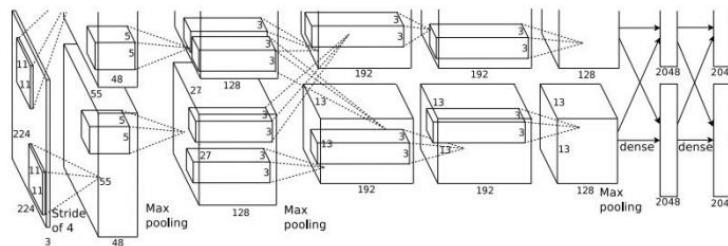
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

CNN as feature extractor

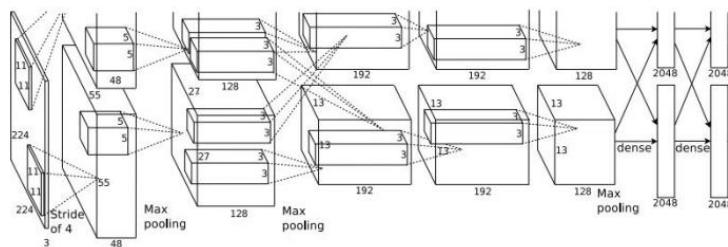
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

CNN as feature extractor

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? YES
Background? NO

CNN as feature extractor

- What could be the problems?

CNN as feature extractor

- What could be the problems?
 - Suppose we have a 600×600 image, if sliding window size is 20×20 , then have $(600-20+1) \times (600-20+1) = \sim 330,000$ windows

CNN as feature extractor

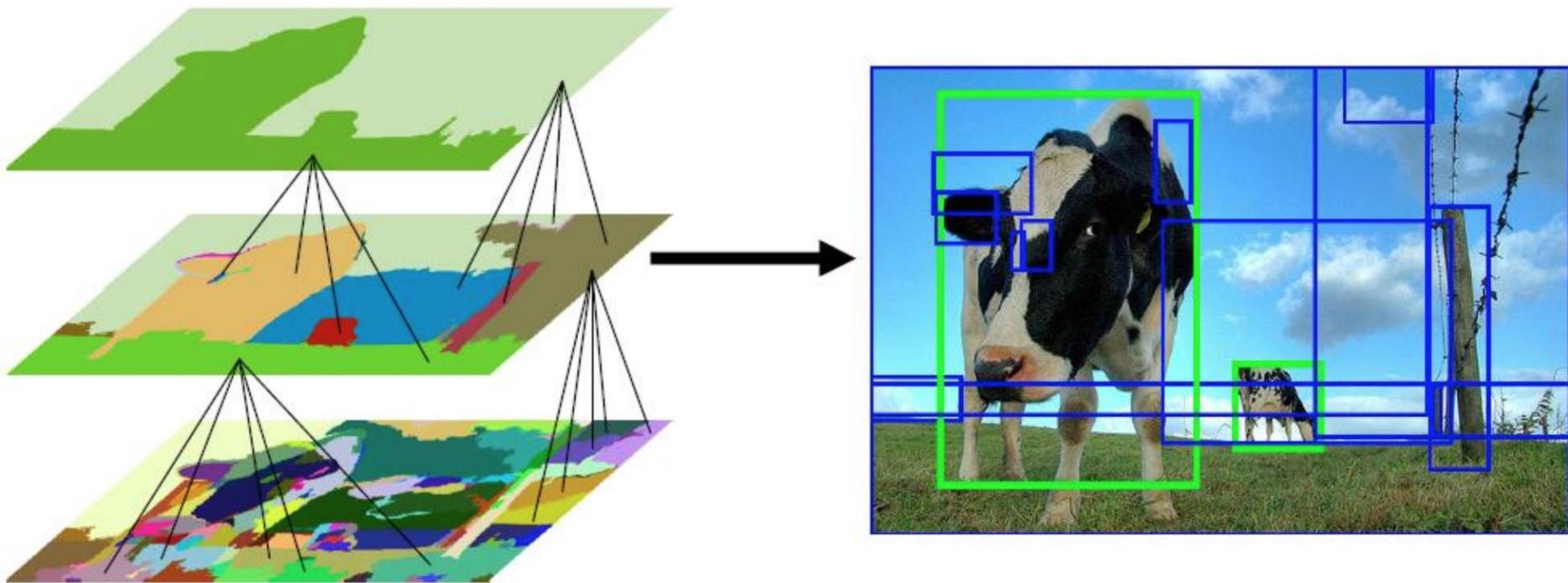
- What could be the problems?
 - Suppose we have a 600×600 image, if sliding window size is 20×20 , then have $(600-20+1) \times (600-20+1) = \sim 330,000$ windows
 - Sometimes we want to have more accurate results -
 - > multi-scale detection
 - Resize image
 - Multi-scale sliding window

CNN as feature extractor

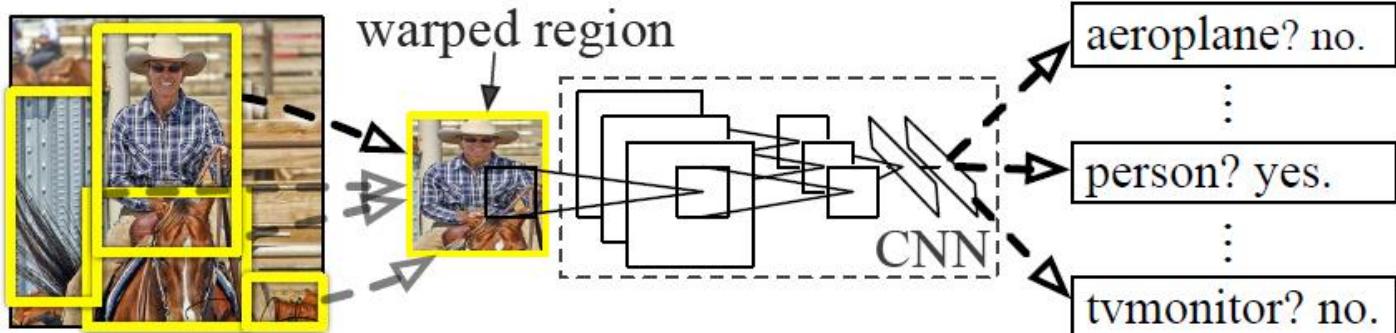
- What could be the problems?
 - Suppose we have a 600×600 image, if sliding window size is 20×20 , then have $(600-20+1) \times (600-20+1) = \sim 330,000$ windows
 - Sometimes we want to have more accurate results -
 - > multi-scale detection
 - Resize image
 - Multi-scale sliding window
 - For each image, we need to do the forward pass in the CNN for $\sim 330,000$ times. -> Slow!!!

Region Proposal

- Solution
 - Use some fast algorithms to filter out some regions first, only feed the potential region (region proposals) into CNN
 - E.g. selective search



R-CNN (Girshick et al. CVPR 2014)



1. Input image

2. Extract region proposals (~2k)

3. Compute CNN features

4. Classify regions

- Replace sliding windows with “selective search” region proposals (Uijlings et al. IJCV 2013)
- Extract rectangles around regions and resize to 227x227
- Extract features with fine-tuned CNN (that was initialized with network trained on ImageNet before training)
- Classify last layer of network features with SVM, refine bounding box localization (bbox regression) simultaneously

Bounding Box Regression

- Intuition

- If you observe part of the object, according to the seen examples, you should be able to refine the localization
- E.g. given the red box below, since you've seen many airplanes, you know this is not a good localization, you will adjust it to the green one



Bounding Box Regression

- Intuition
 - If you observe part of the object, according to the seen examples, you should be able to refine the localization
 - E.g. given the red box below, since you've seen many airplanes, you know this is not a good localization, you will adjust it to the green one



R-CNN (Girshick et al. CVPR 2014)

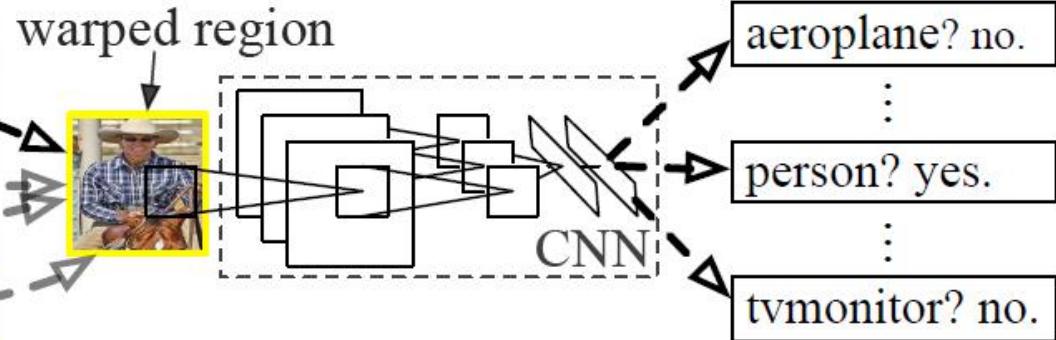
- What could be the problems?



1. Input image



2. Extract region proposals (~2k)



3. Compute CNN features

4. Classify regions

R-CNN (Girshick et al. CVPR 2014)

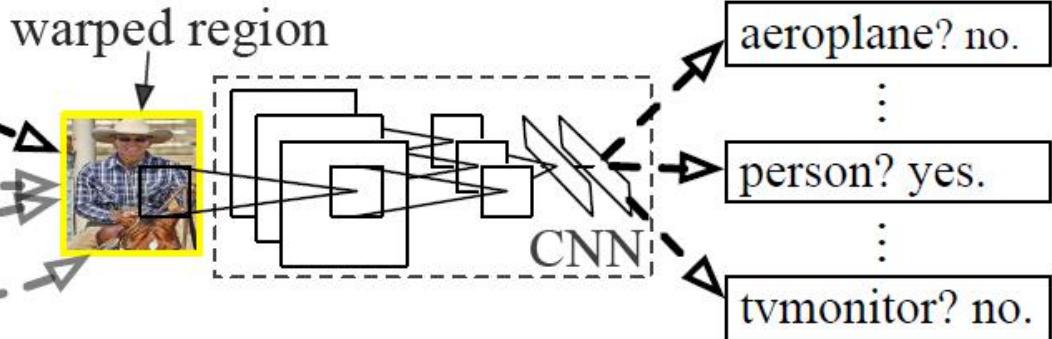
- What could be the problems?
 - Repetitive computation! For overlapping regions, we feed it multiple times into CNN



1. Input image



2. Extract region proposals (~2k)

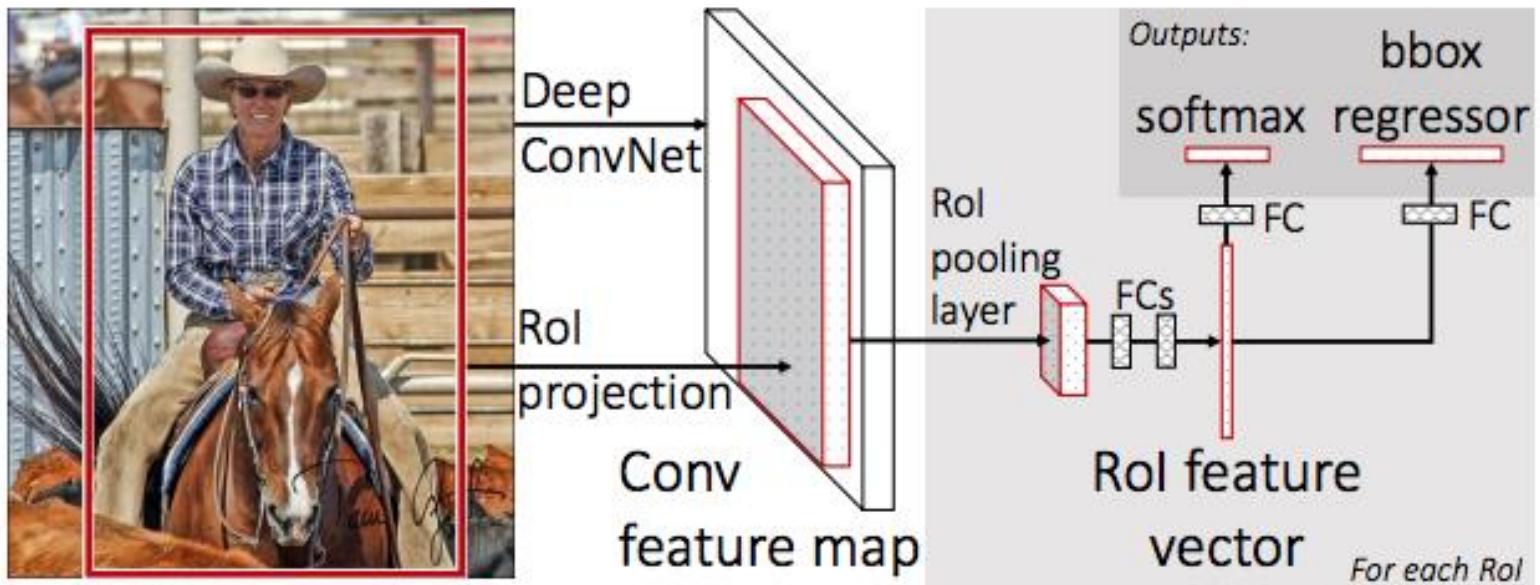


3. Compute CNN features

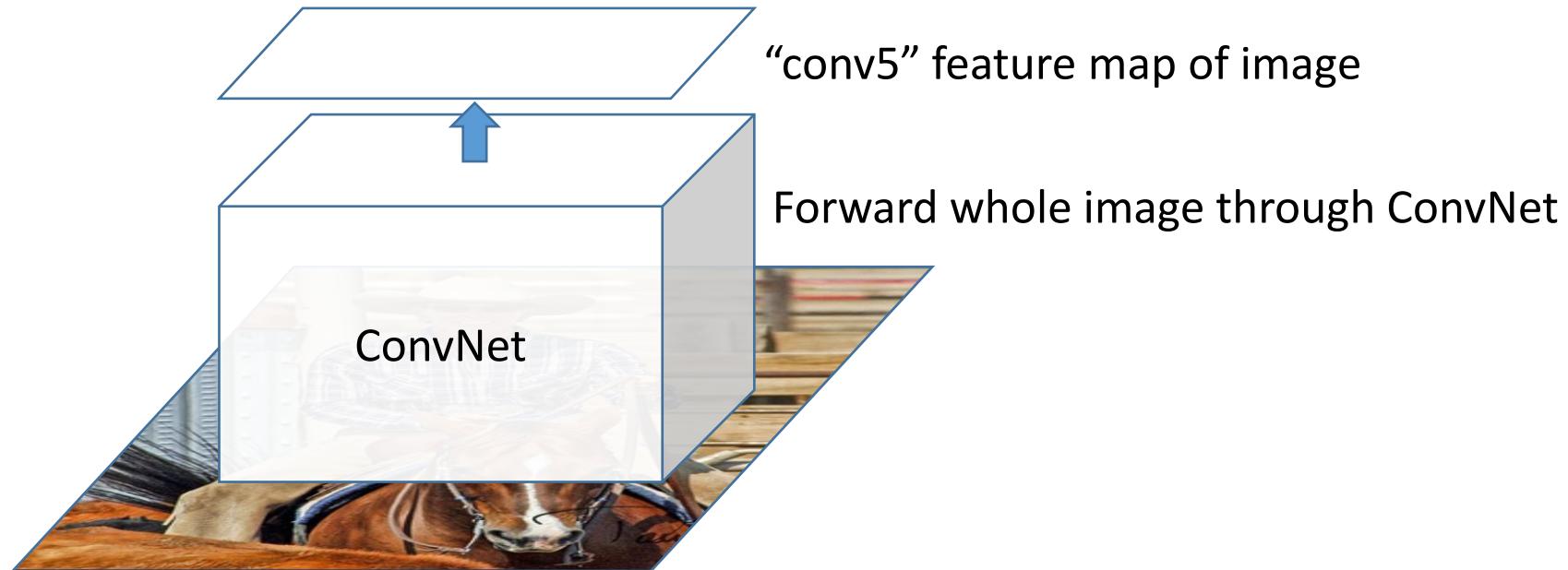
4. Classify regions

Fast R-CNN (Girshick ICCV 2015)

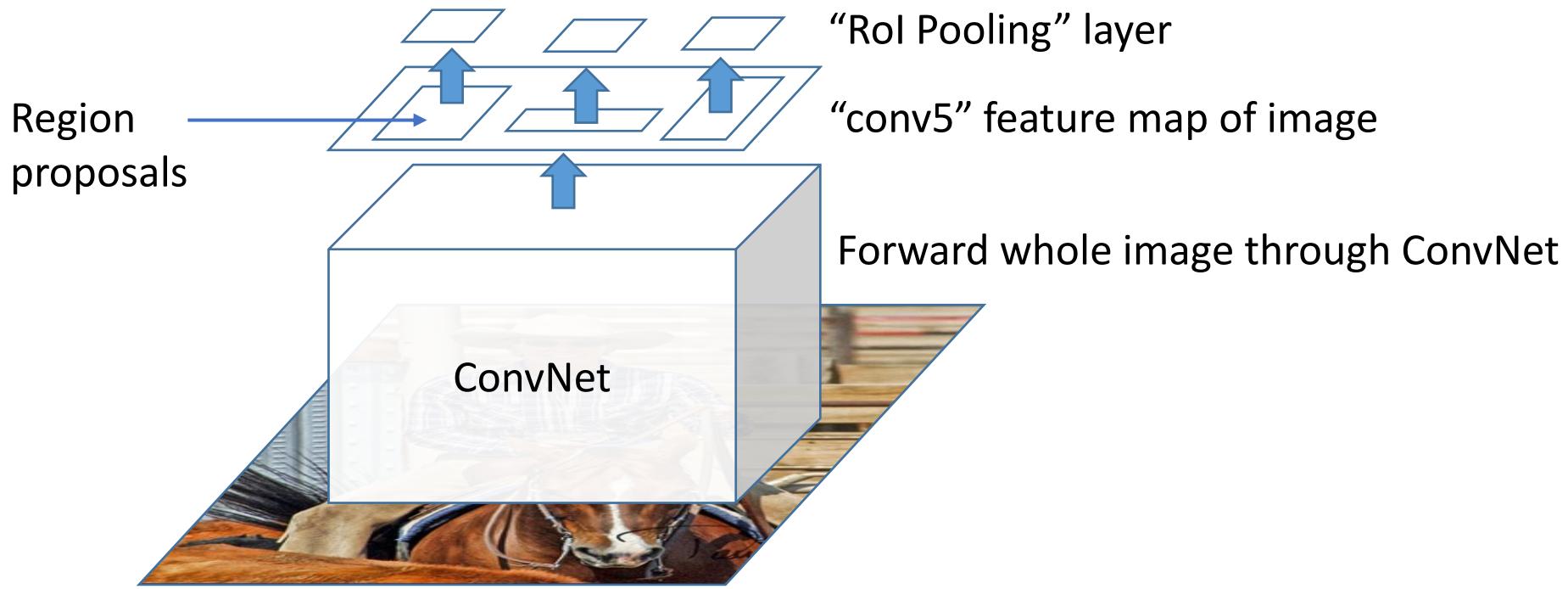
- Solution
 - Why not feed the whole image into CNN only once! Then crop features instead of image itself



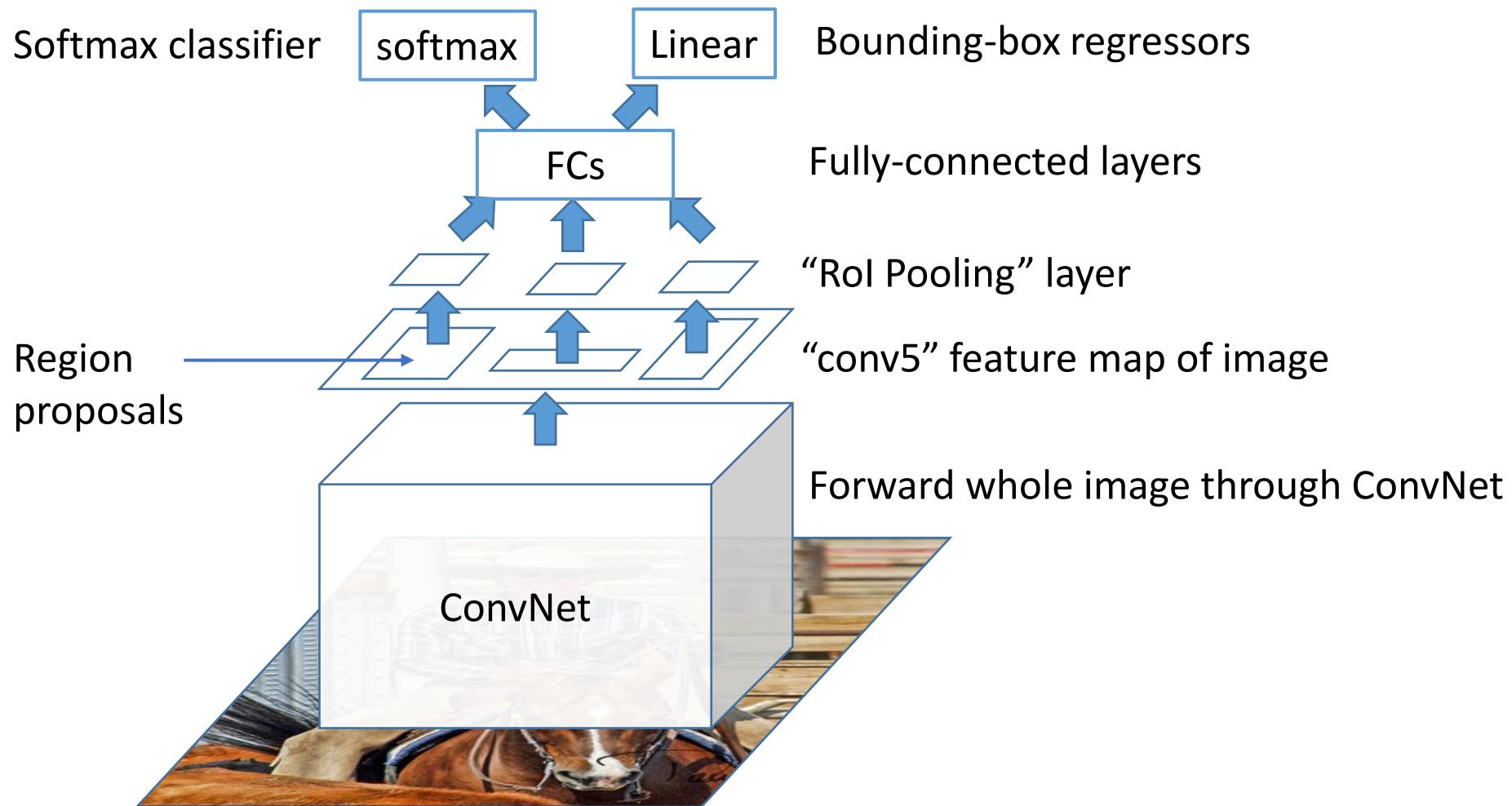
Fast R-CNN (Girshick ICCV 2015)



Fast R-CNN (Girshick ICCV 2015)

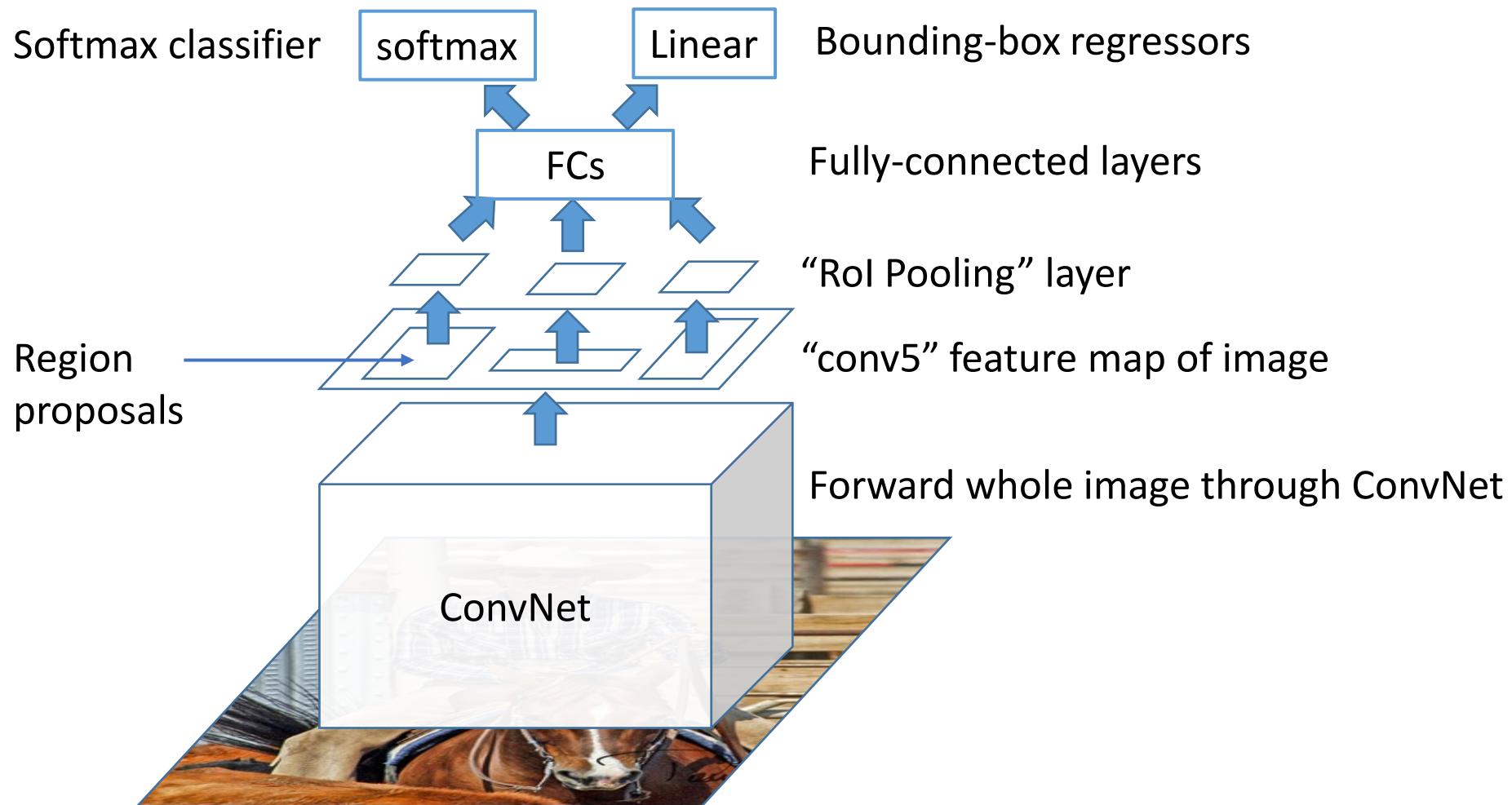


Fast R-CNN (Girshick ICCV 2015)

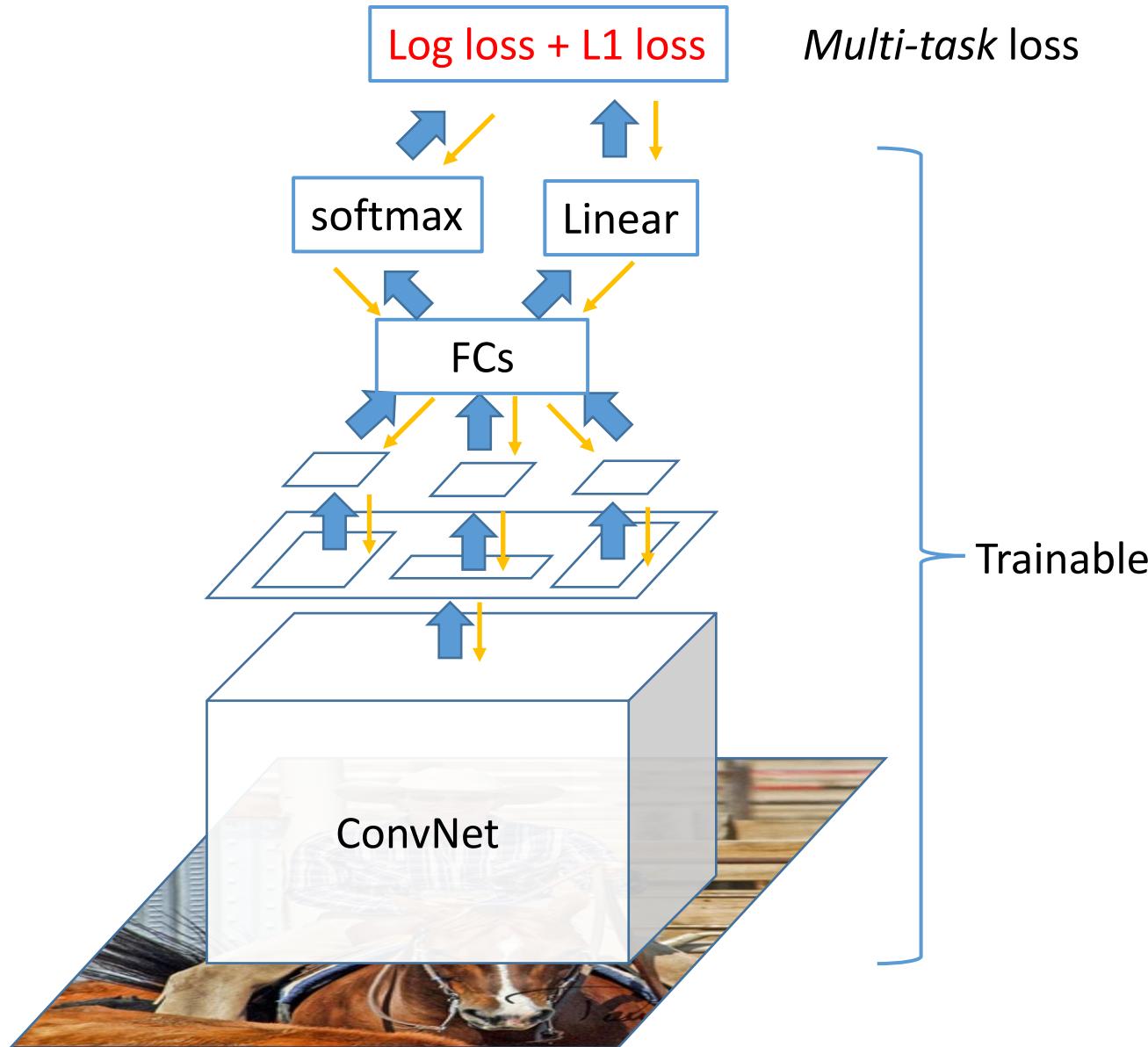


Fast R-CNN (Girshick ICCV 2015)

Rather than using post-hoc bounding-box regressors, bounding-box regression is implemented as an additional linear layer in the network



Fast R-CNN (Girshick ICCV 2015)

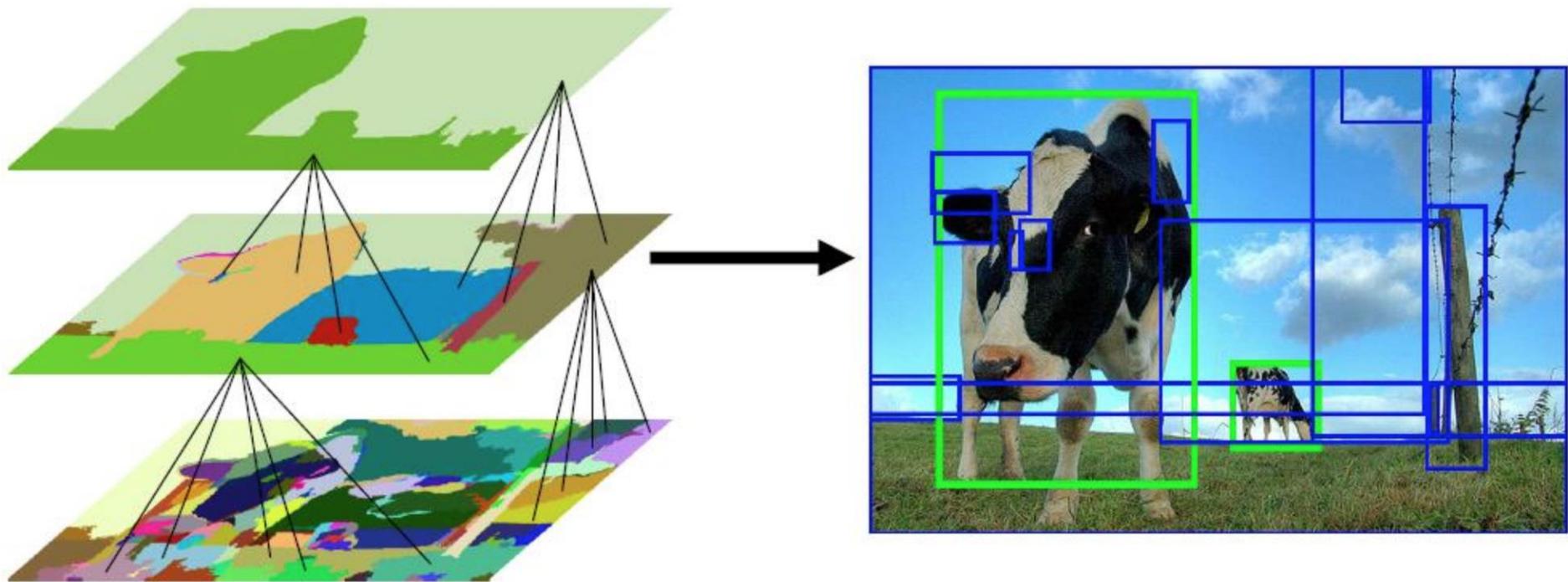


Fast R-CNN (Girshick ICCV 2015)

- What could be the problems?

Fast R-CNN (Girshick ICCV 2015)

- What could be the problems?
 - Why we need the region proposal pre-processing step? That's not “deep learning” at all. Not cool!



Faster R-CNN (Ren et al. NIPS 2015)

- Solution

- Why not generate region proposals using CNN??!

-> RPN

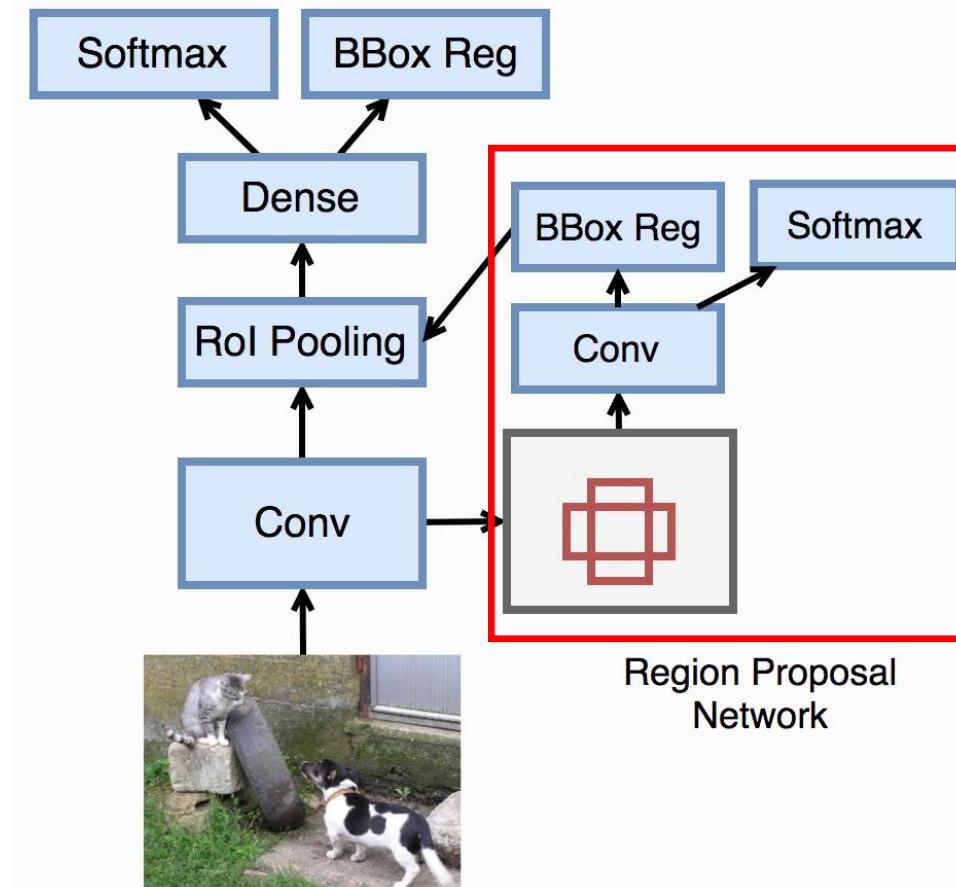
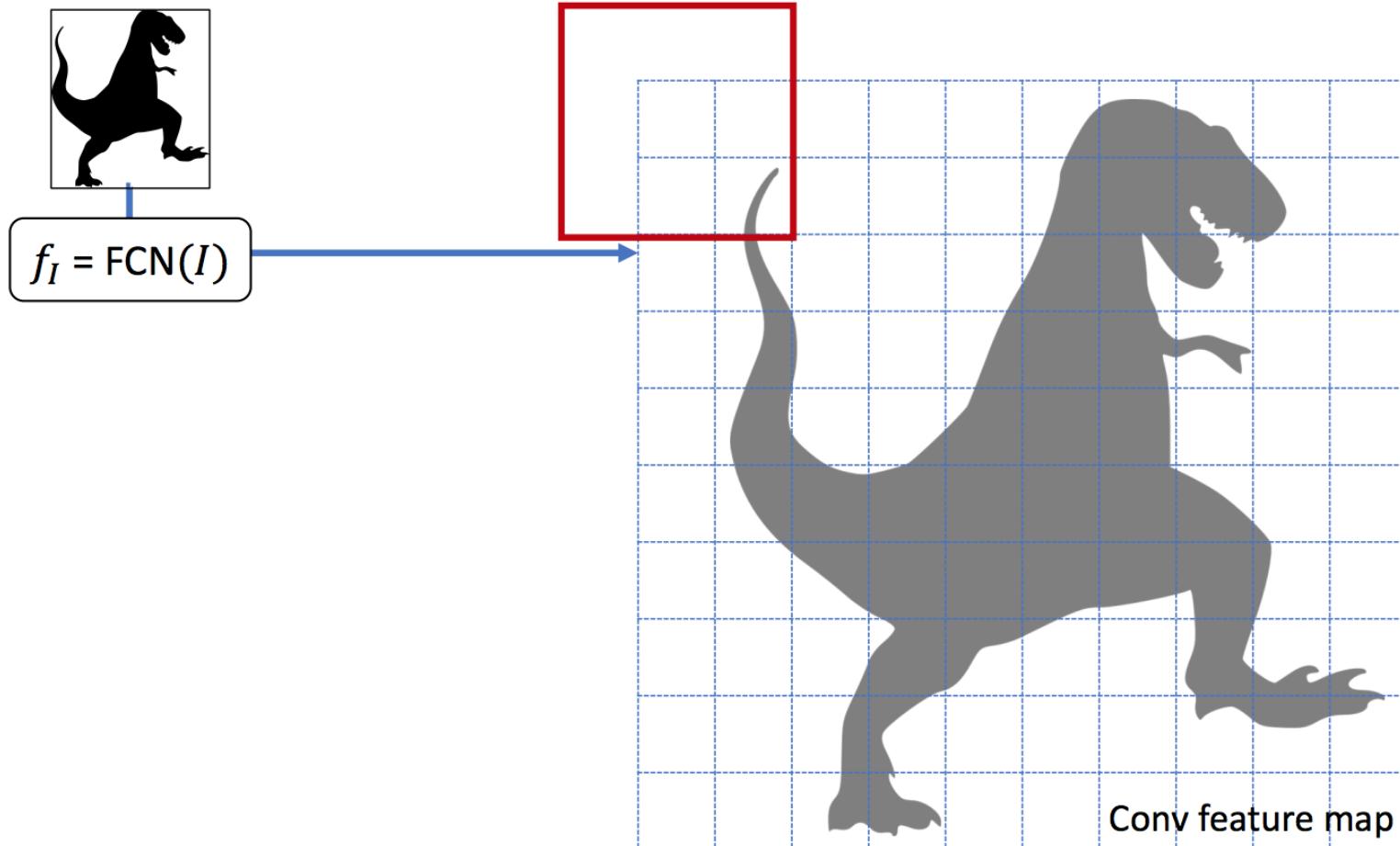


Image credit:

http://zh.gluon.ai/chapter_computer-vision/object-detection.html

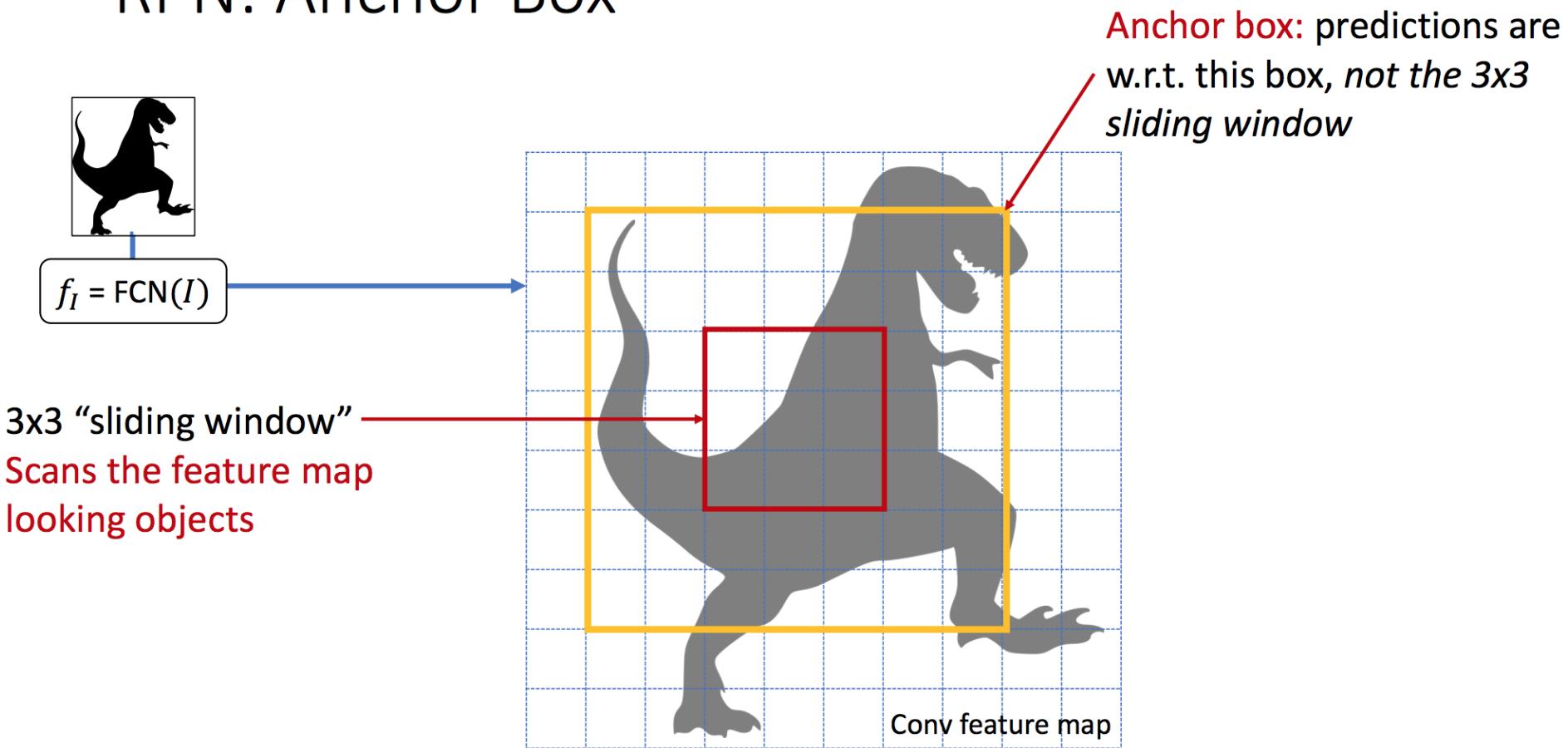
Faster R-CNN (Ren et al. NIPS 2015)

RPN: Region Proposal Network



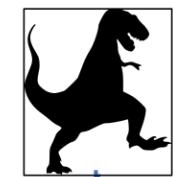
Faster R-CNN (Ren et al. NIPS 2015)

RPN: Anchor Box

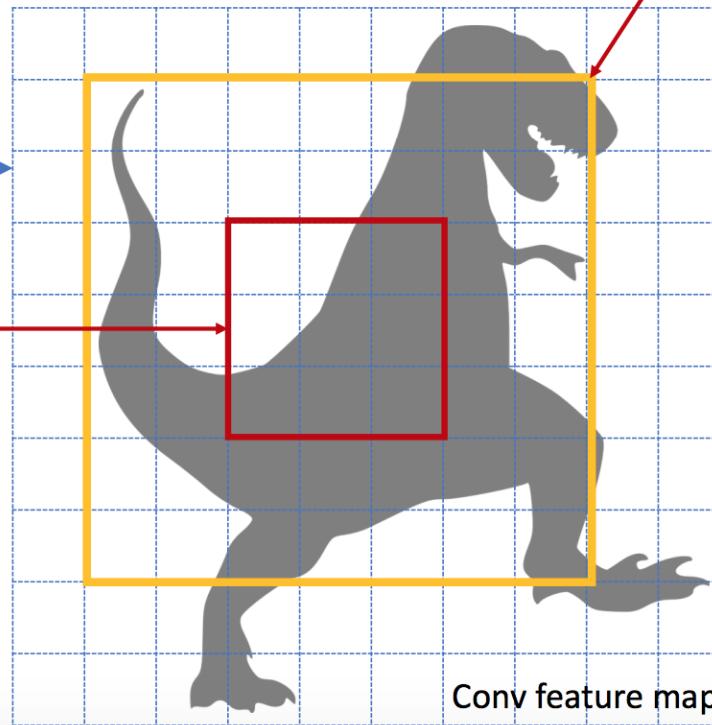


Faster R-CNN (Ren et al. NIPS 2015)

RPN: Anchor Box



$$f_I = \text{FCN}(I)$$

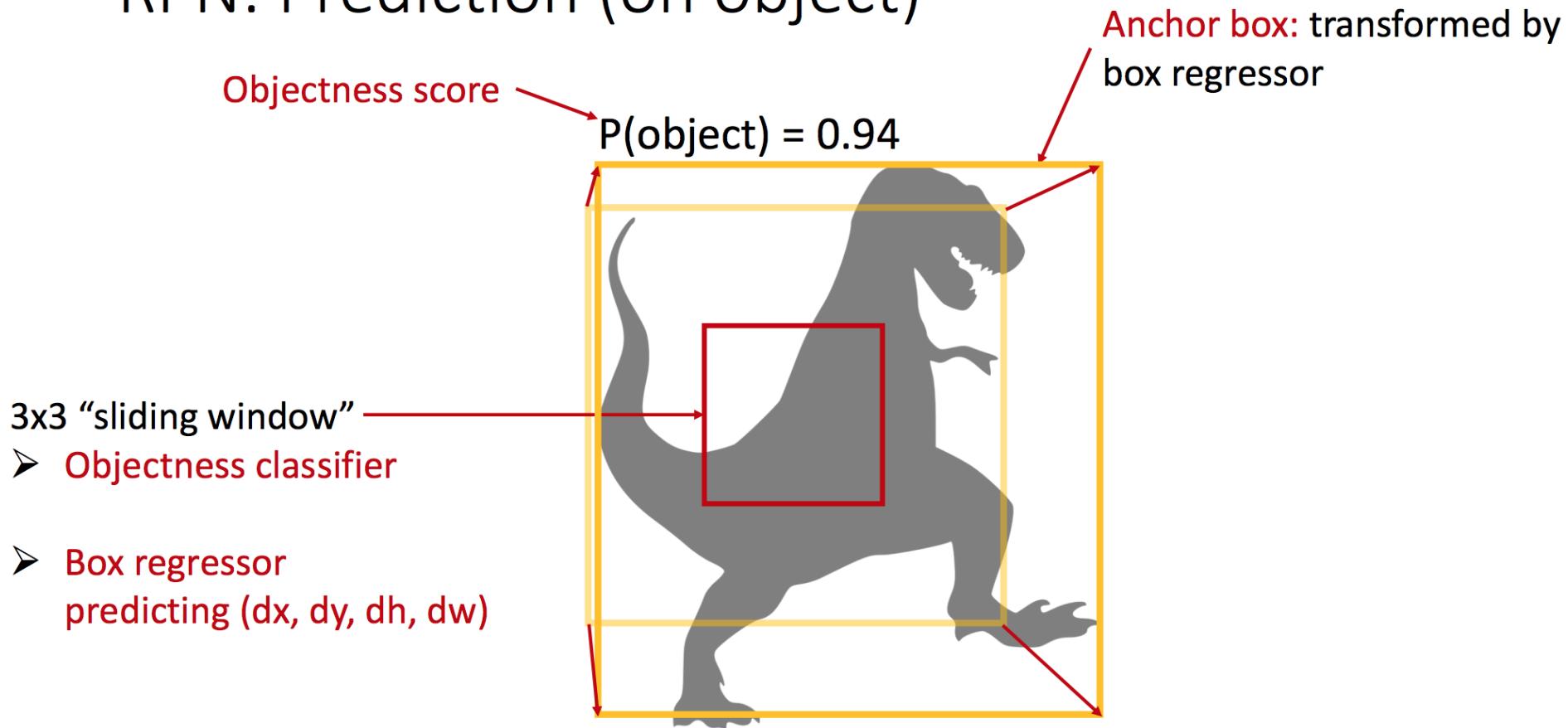


Anchor box: predictions are
w.r.t. this box, *not the 3x3
sliding window*

- 3x3 "sliding window"
- Objectness classifier
 - Box regressor predicting (dx, dy, dh, dw)

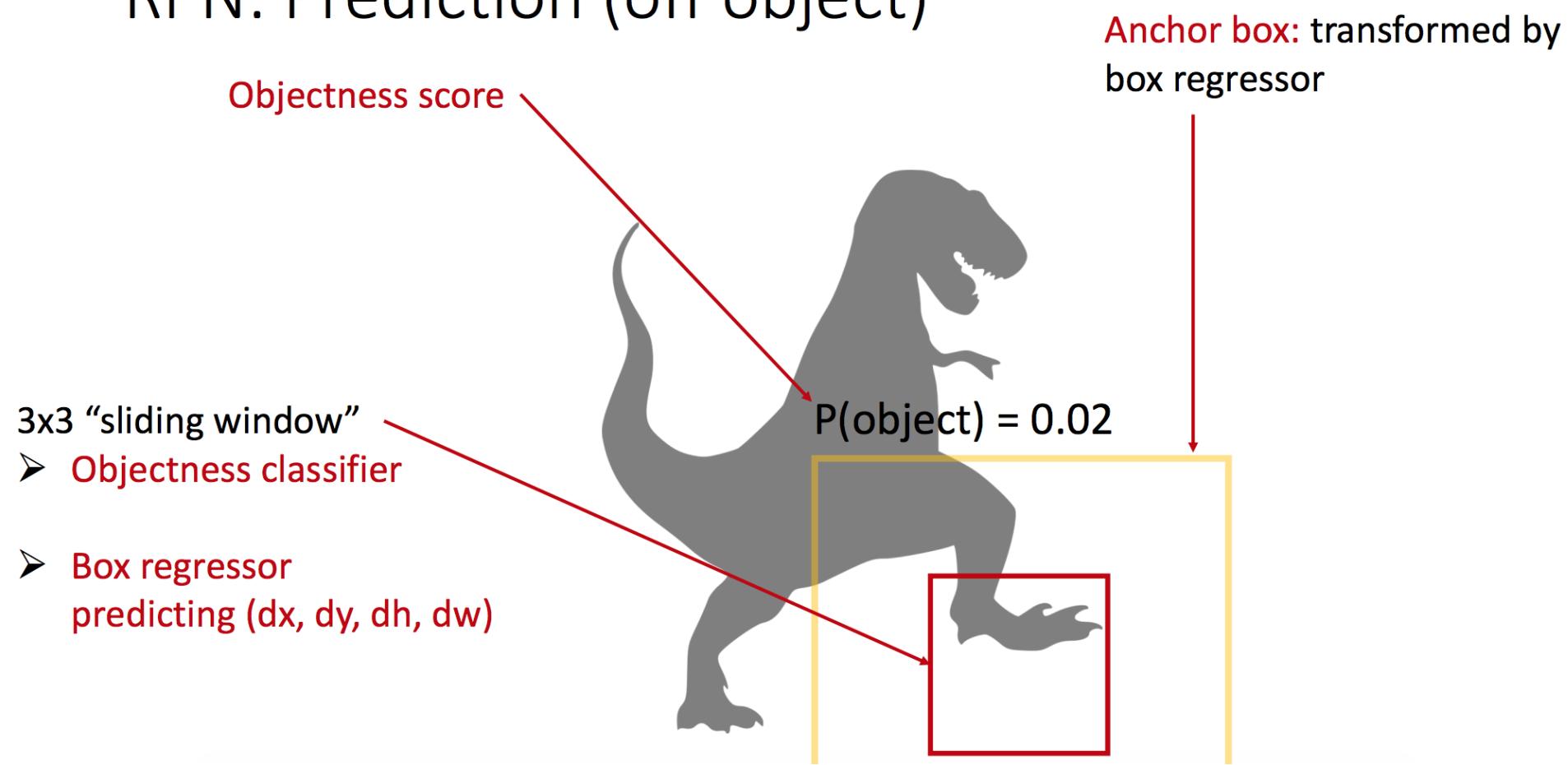
Faster R-CNN (Ren et al. NIPS 2015)

RPN: Prediction (on object)



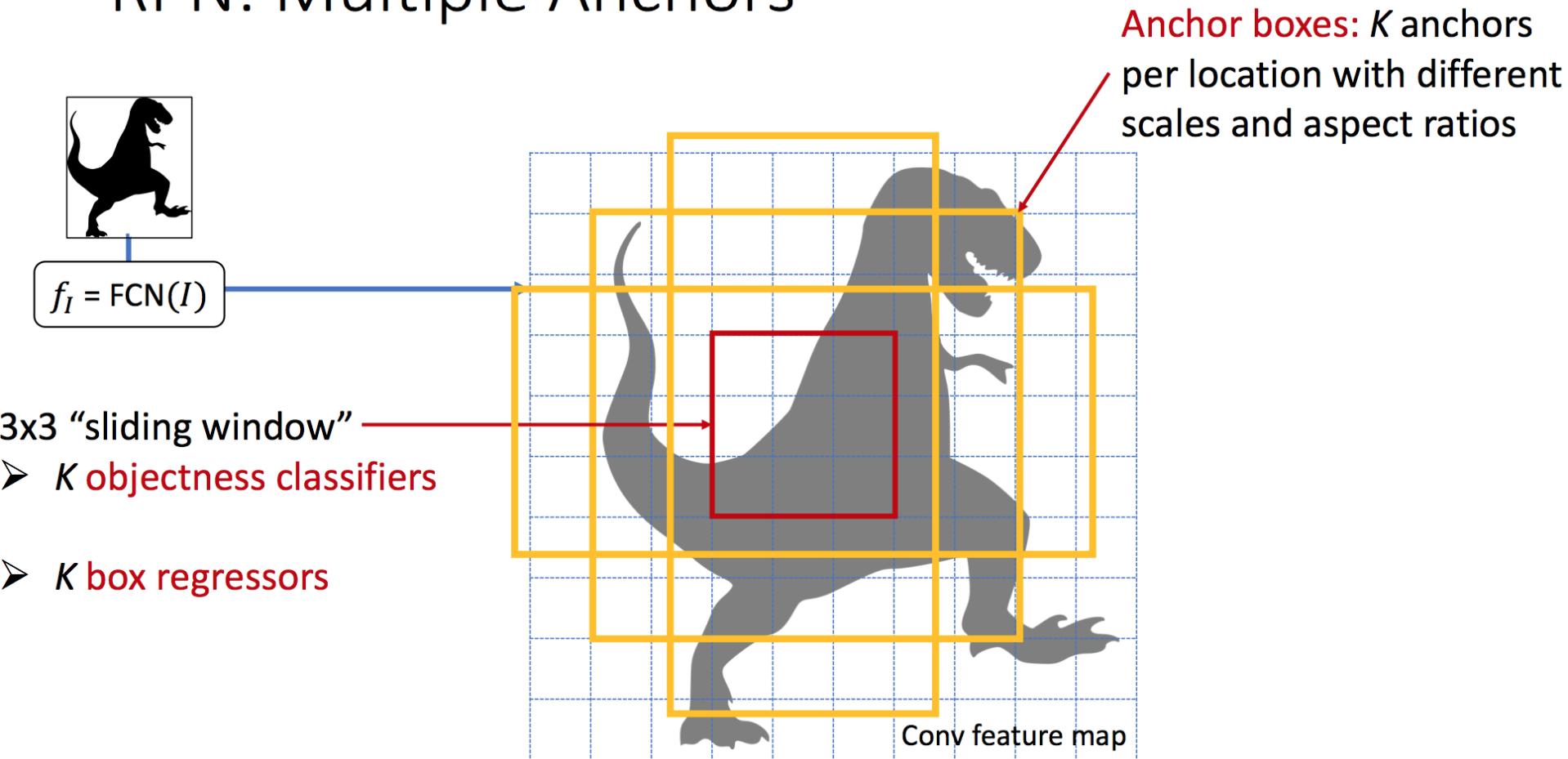
Faster R-CNN (Ren et al. NIPS 2015)

RPN: Prediction (off object)



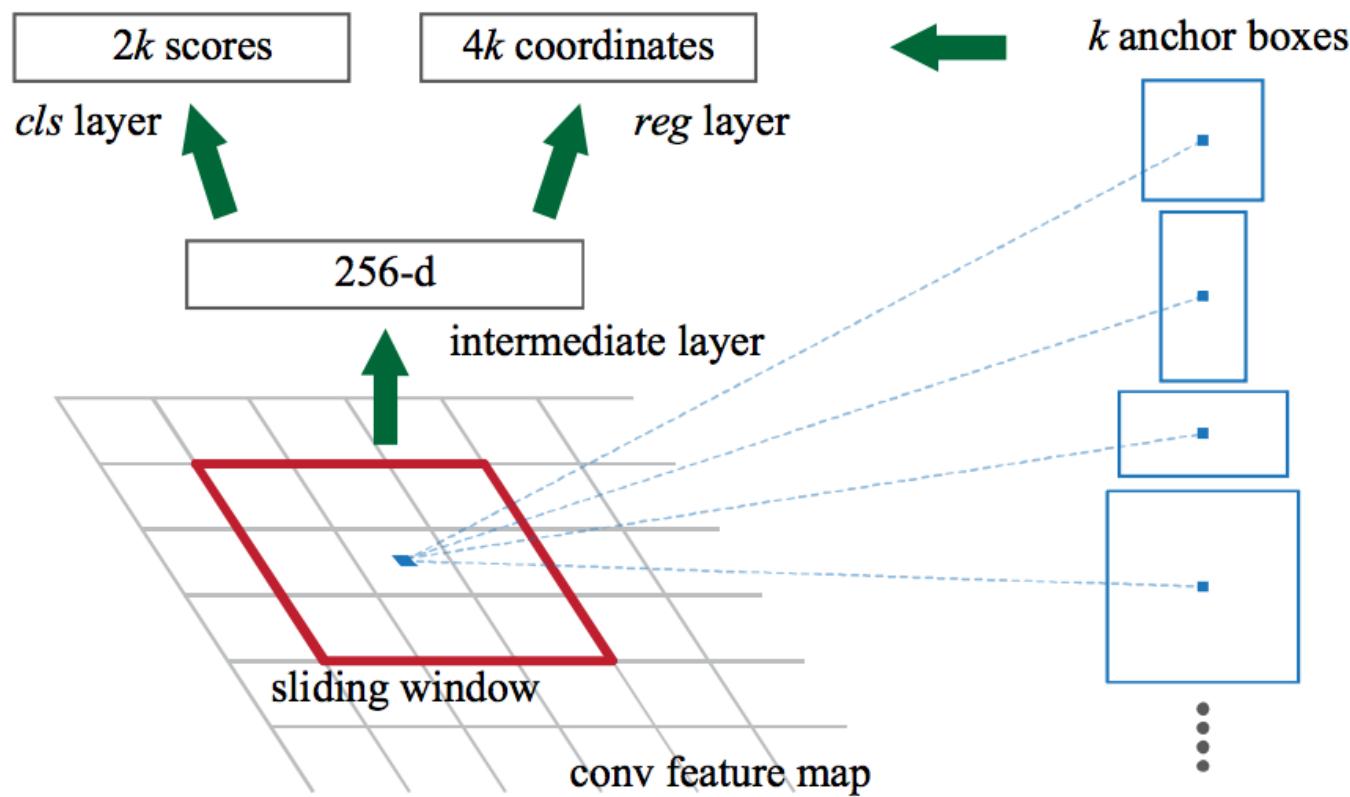
Faster R-CNN (Ren et al. NIPS 2015)

RPN: Multiple Anchors



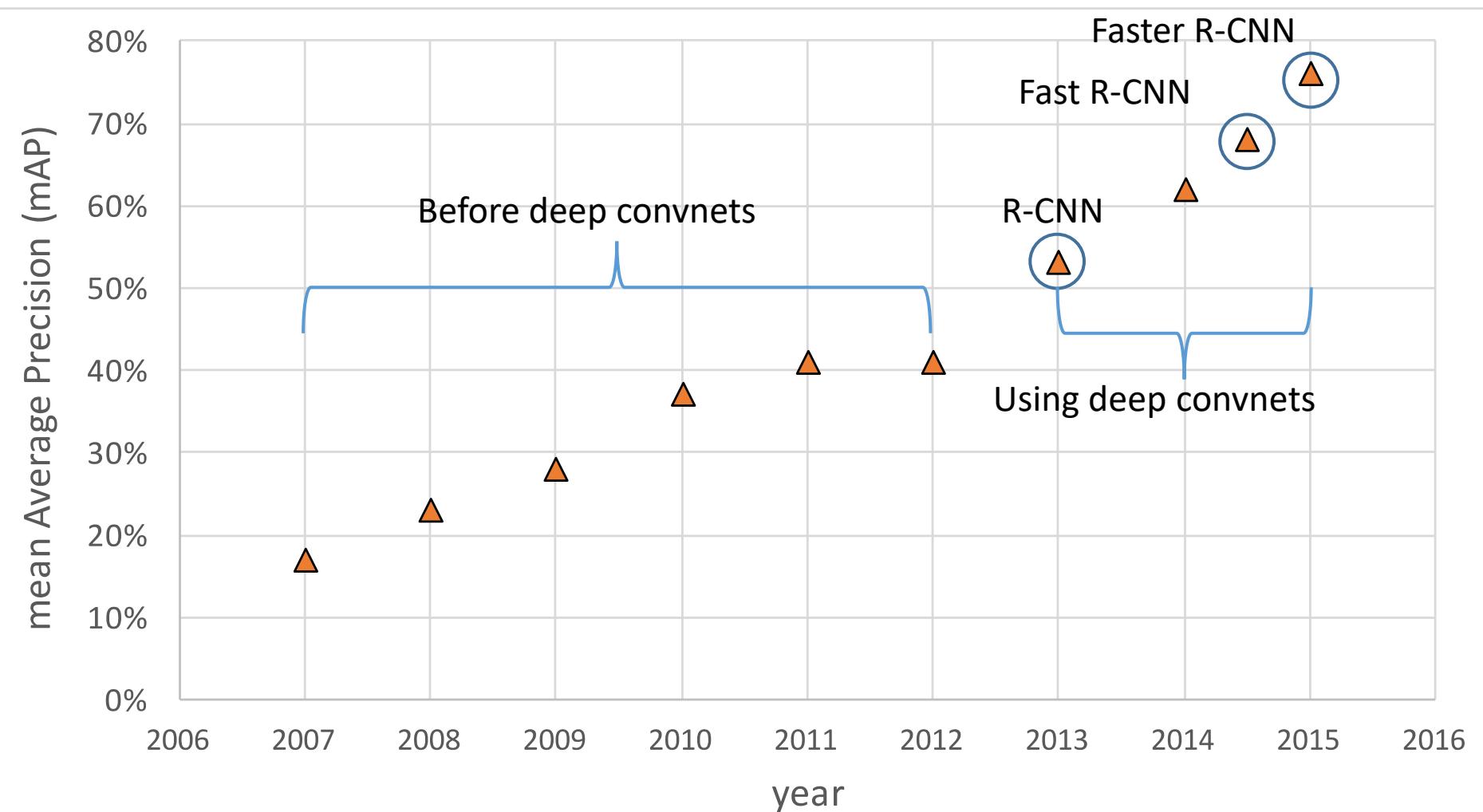
Faster R-CNN (Ren et al. NIPS 2015)

Region proposal network



Faster R-CNN (Ren et al. NIPS 2015)

Progress on PASCAL VOC database



Faster R-CNN (Ren et al. NIPS 2015)

- What could be the problems

Faster R-CNN (Ren et al. NIPS 2015)

- What could be the problems
 - Two-stage detection pipeline is still too slow to apply on real-time videos

One-stage detection

- Solution

- Don't generate object proposals!
- Consider a tiny subset of the output space by design; directly classify this small set of boxes

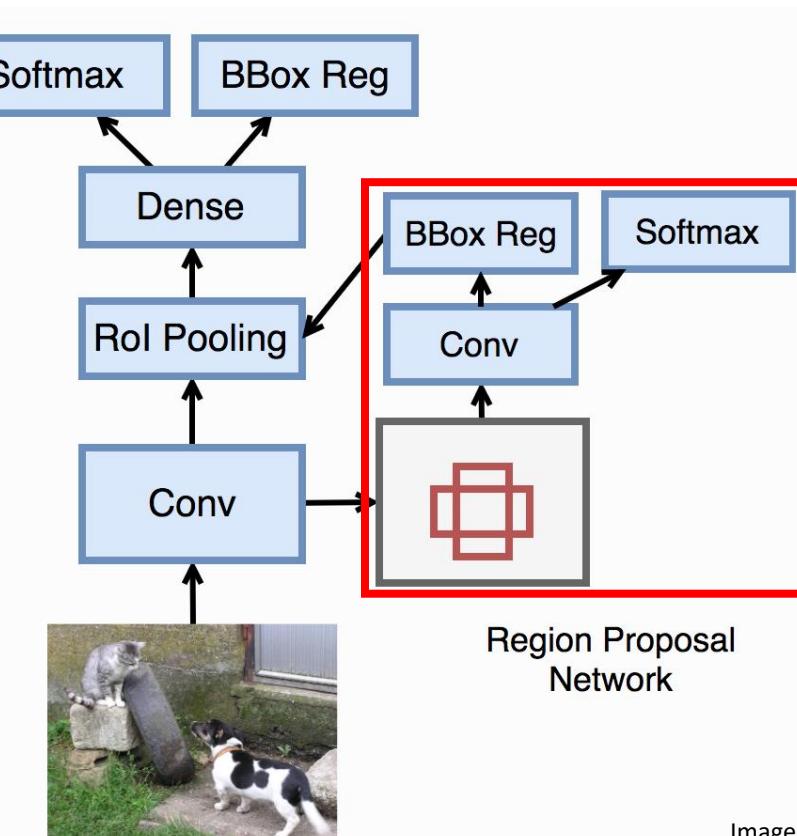
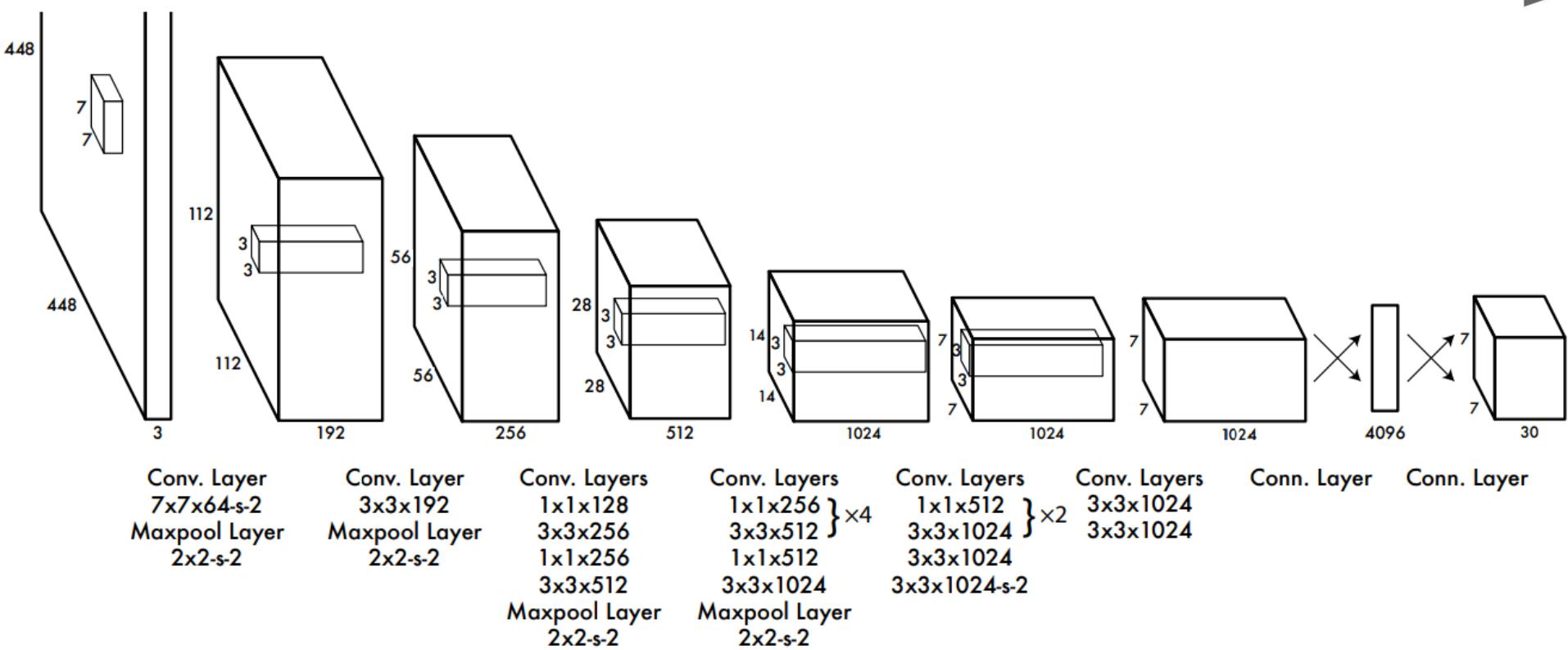


Image credit:

http://zh.gluon.ai/chapter_computer-vision/object-detection.html

You Only Look Once (YOLO)

Go from input image to tensor of scores with one big convolutional network!

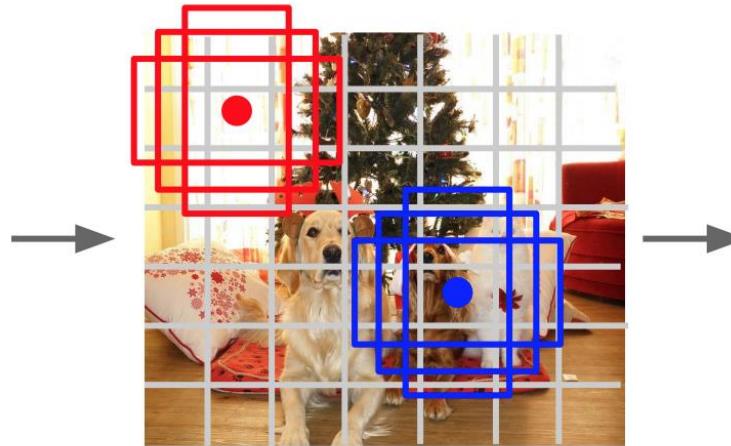


You Only Look Once (YOLO)

Go from input image to tensor of scores with one big convolutional network!



Input image
 $3 \times H \times W$



Divide image into grid
 7×7

Image a set of **base boxes**
centered at each grid cell

Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers:
(dx , dy , dh , dw , confidence)
- Predict scores for each of C classes (including background as a class)

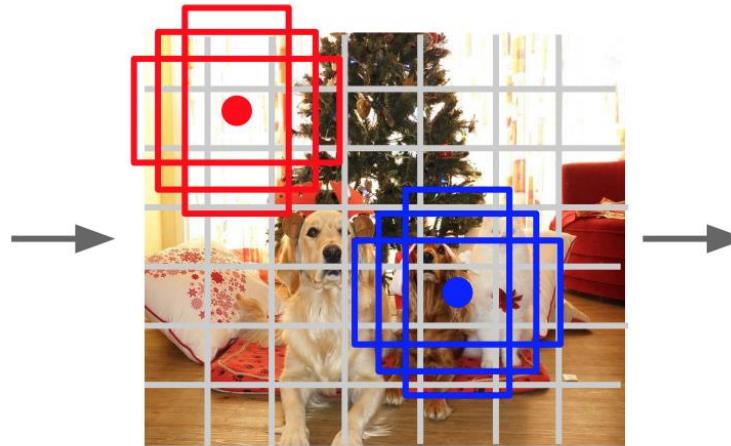
Output:
 $7 \times 7 \times (5 * B + C)$

You Only Look Once (YOLO)

Go from input image to tensor of scores with one big convolutional network!



Input image
 $3 \times H \times W$



Divide image into grid
 7×7

Image a set of **base boxes**
centered at each grid cell

Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers:
(dx , dy , dh , dw , confidence)
- Predict scores for each of C classes (including background as a class)

Output:
 $7 \times 7 \times (5 * B + C)$

$B = 2$ in experiments

$C = 20$ in PASCAL VOC

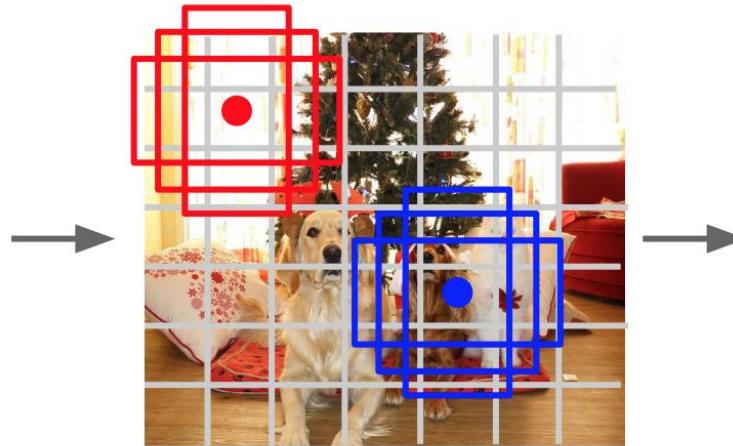
Final prediction= $7 \times 7 \times 30$ tensor.

You Only Look Once (YOLO)

Go from input image to tensor of scores with one big convolutional network!



Input image
 $3 \times H \times W$



Divide image into grid
 7×7

Image a set of **base boxes**
centered at each grid cell

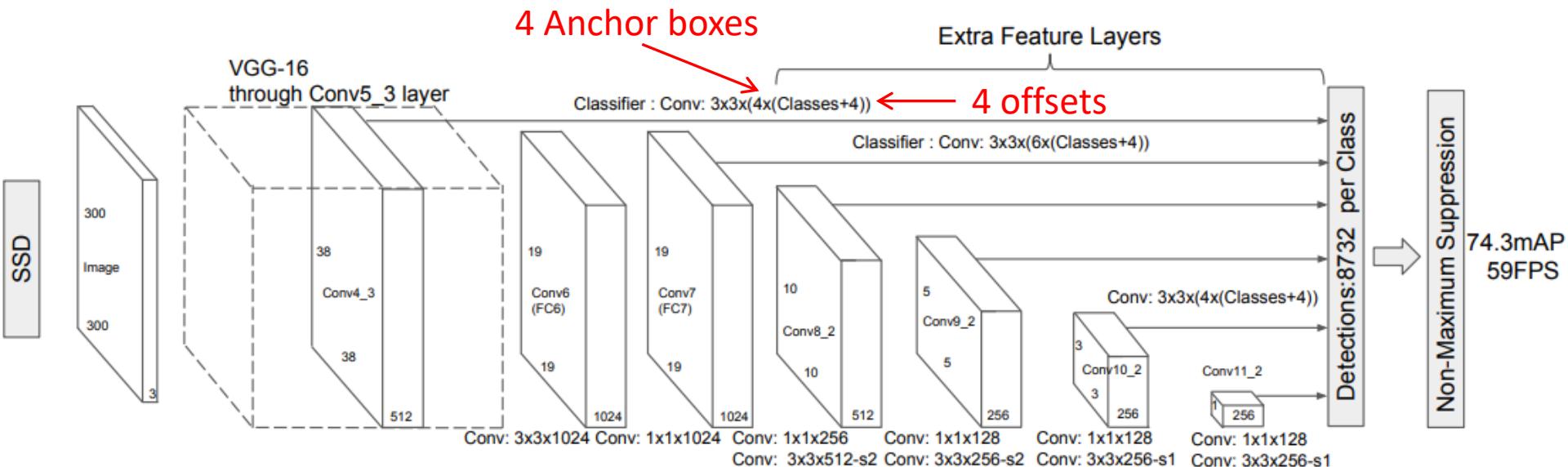
Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers:
(dx , dy , dh , dw , confidence)
- Predict scores for each of C classes (including background as a class)

Output:
 $7 \times 7 \times (5 * B + C)$

•Very efficient but lower accuracy

SSD: Single Shot MultiBox Detector



- Run a small 3×3 sized convolutional kernel to predict the bounding boxes and classification probability.
- SSD also uses anchor boxes at various aspect ratio similar to Faster-RCNN and learns the off-set rather than learning the box.
- In order to handle the scale, SSD predicts bounding boxes after multiple convolutional layers.

SSD: Single Shot MultiBox Detector



One-stage detection

- What could be the problems?

One-stage detection

- What could be the problems?
 - The extreme foreground-background class imbalance
-> we have a lot more negative examples.

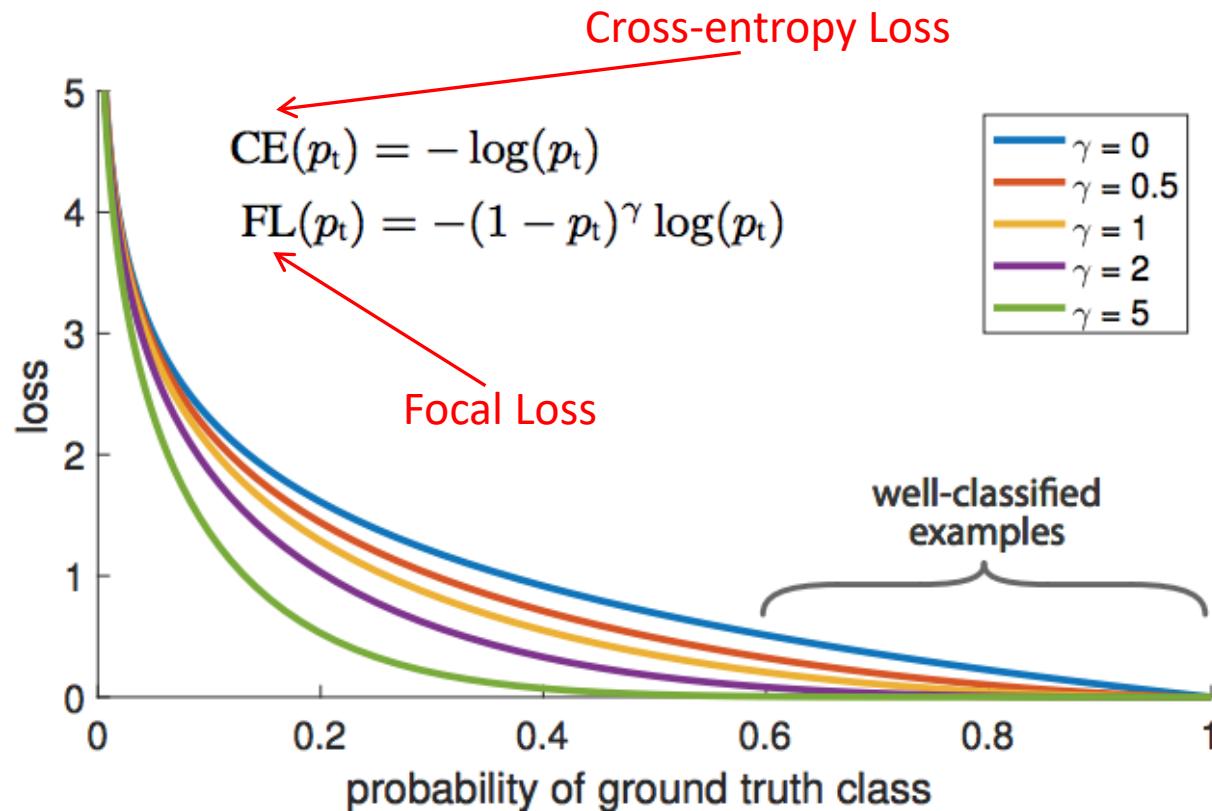
One-stage detection

- What could be the problems?
 - The extreme foreground-background class imbalance
-> we have a lot more negative examples.
 - The vast number of easy negatives overwhelms the detector during training.

RetinaNet (Lin et al. ICCV 2017)

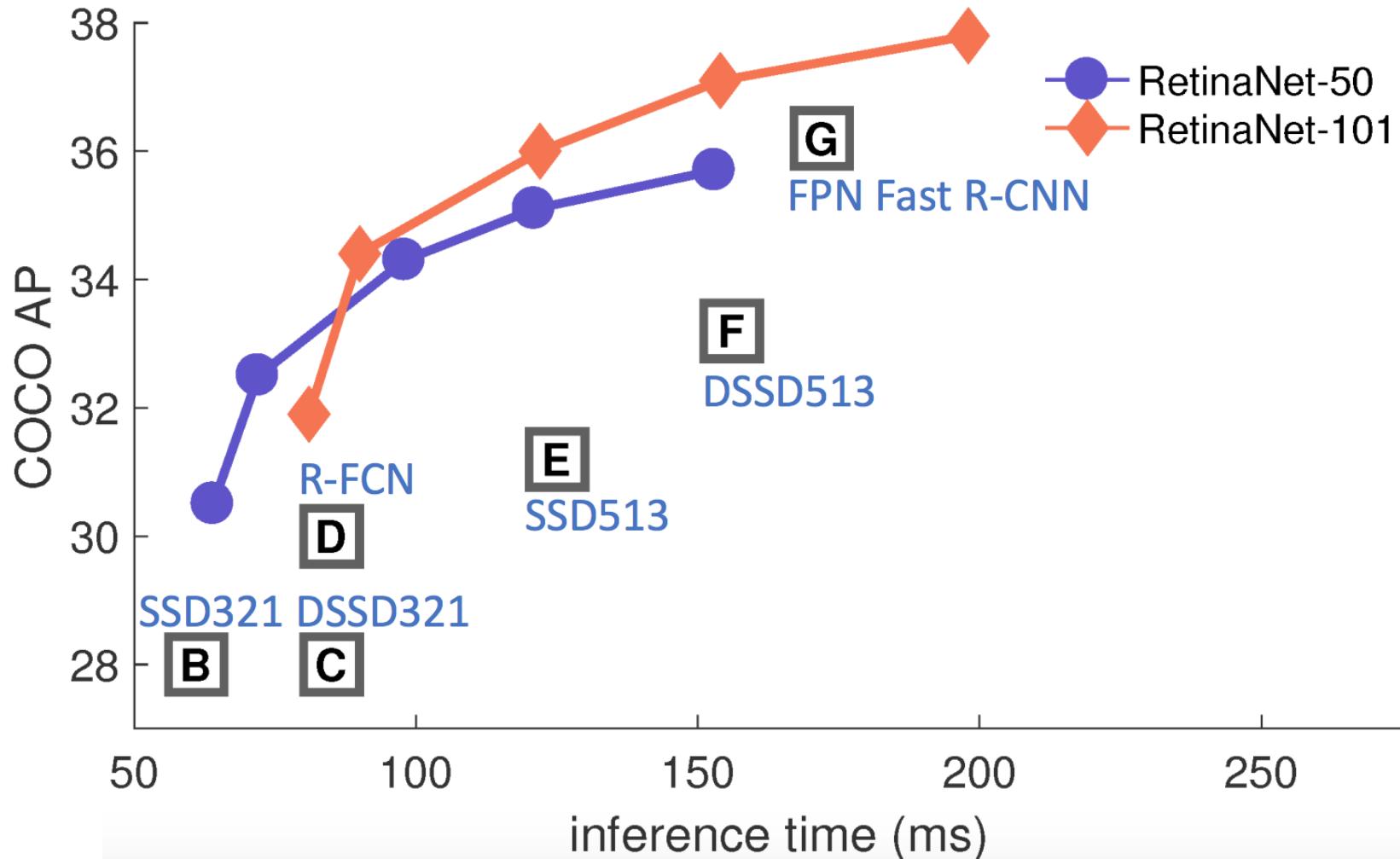
- Solution

- For easy negative examples, down-weight the loss, so that the gradients from these example have smaller impact to the model

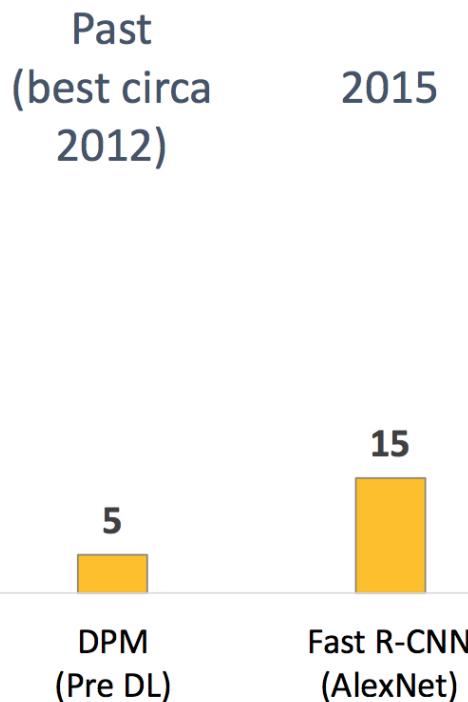


RetinaNet (Lin et al. ICCV 2017)

Speed/Accuracy Tradeoff

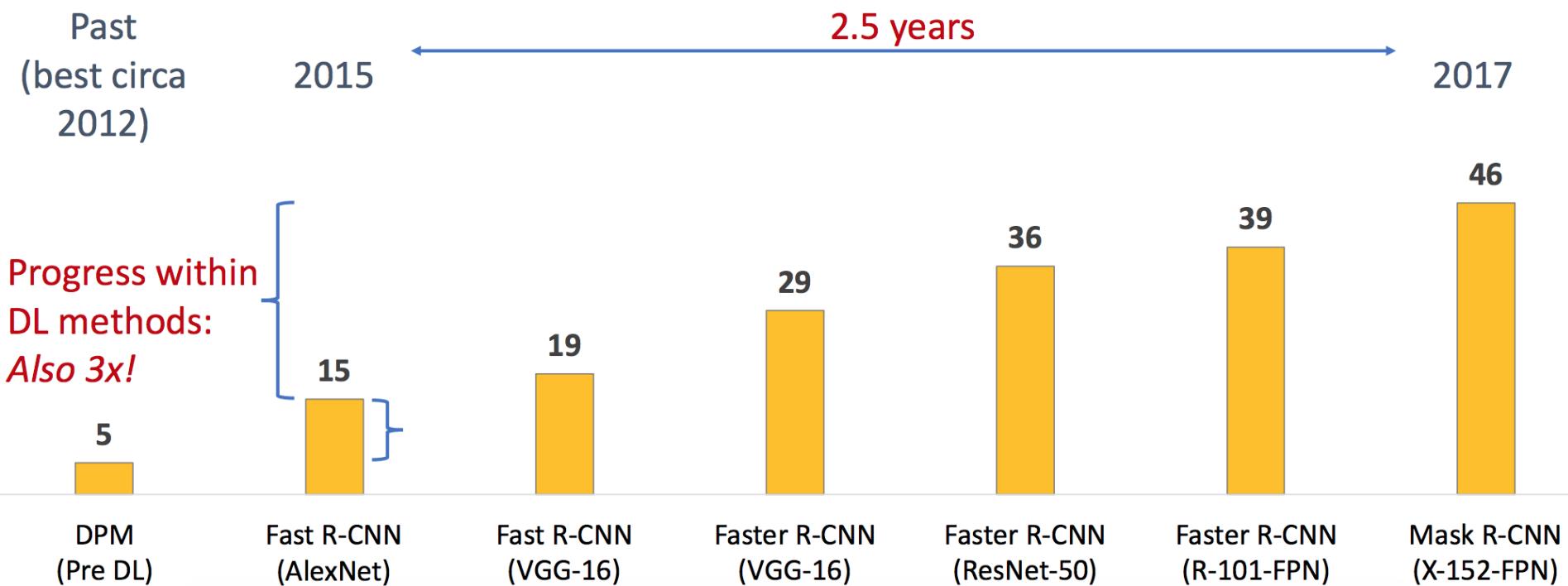


COCO Object Detection Average Precision (%)

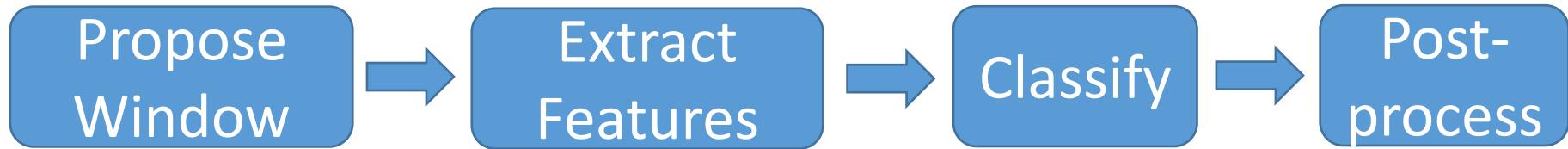


Movement to
Deep Learning methods:
3x improvement in AP

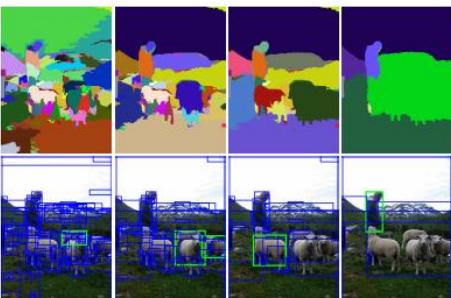
COCO Object Detection Average Precision (%)



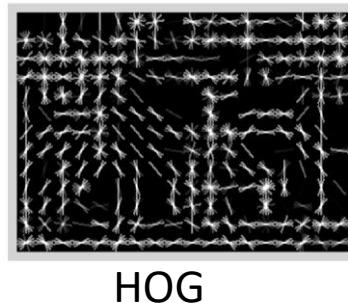
Summary: statistical templates



Sliding window: scan image pyramid



Region proposals:
edge/region-based, resize
to fixed window



HOG

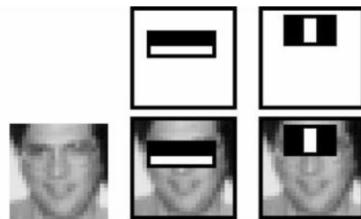
SVM

Boosted stumps

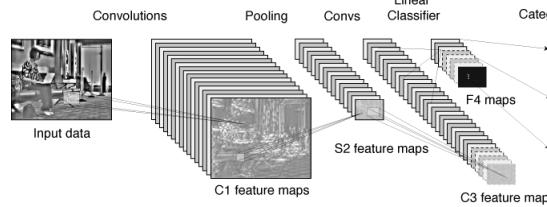
Neural network

Non-max suppression

Segment or refine localization



Fast randomized features



CNN features

Acknowledgements

- Thanks to the following researchers for making their teaching/research material online
 - Forsyth
 - Steve Seitz
 - Noah Snavely
 - J.B. Huang
 - Derek Hoiem
 - J. Hays
 - J. Johnson
 - R. Girshick
 - S. Lazebnik
 - K. Grauman
 - Antonio Torralba
 - Rob Fergus
 - Leibe
 - And many more

Next class

- Image Segmentation

