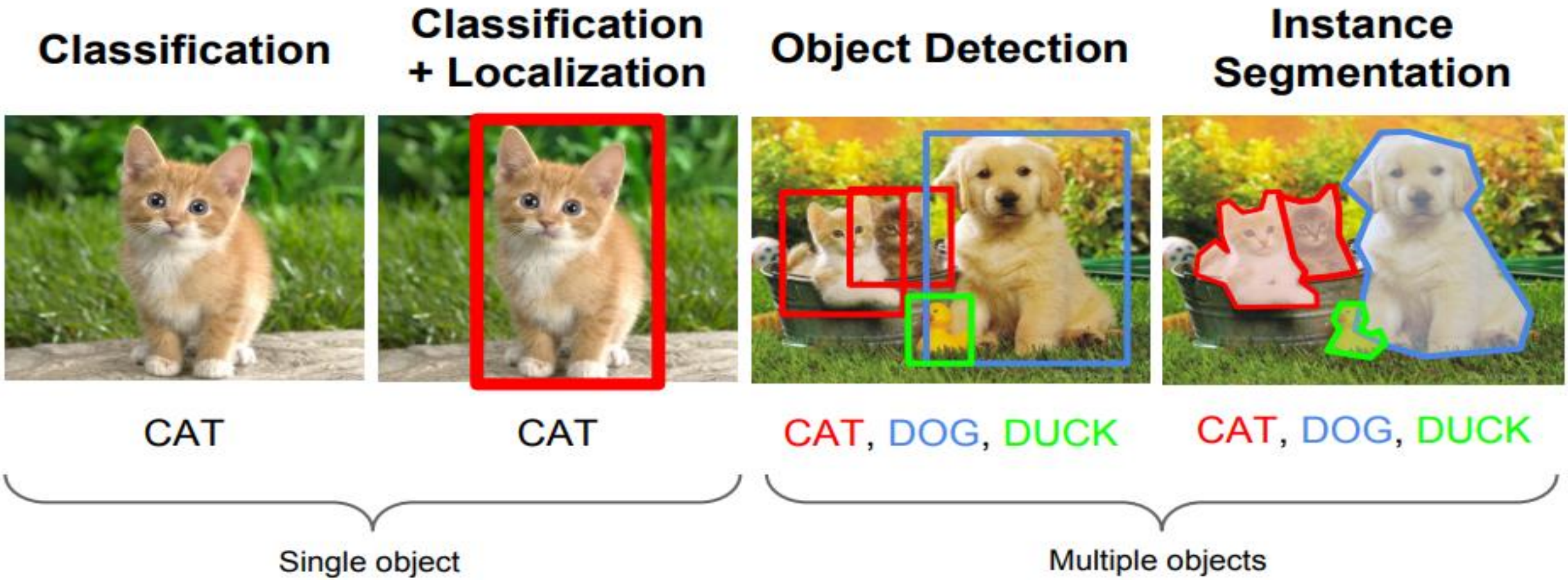
Object Detection using Deep Learning

Image credit

Roadmap



Slide from A. Karpathy

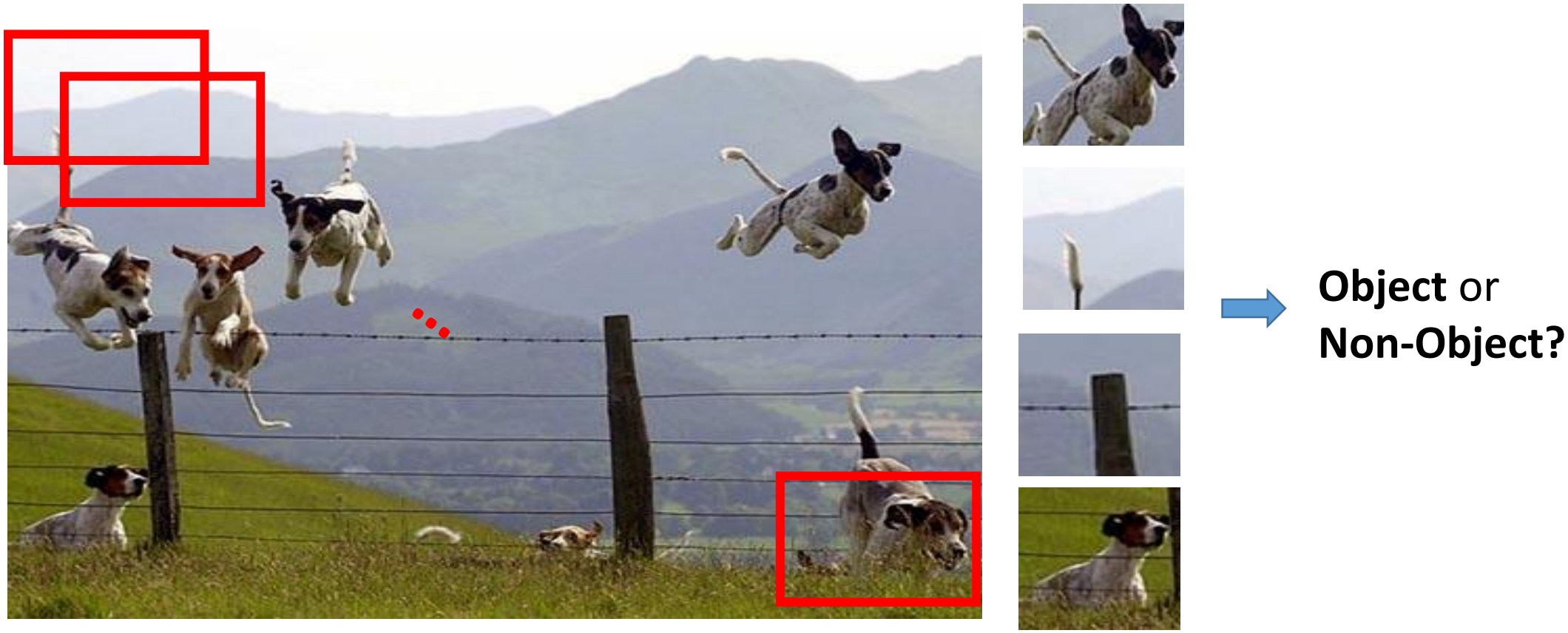
Today's class

- Object category detection

- Deep learning methods
  - Two-stage models: R-CNN
  - One-stage models: 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?**

What are the challenges of object detection?

- Images may contain more than one class, multiple instances from the same class
- Bounding box localization
- Evaluation

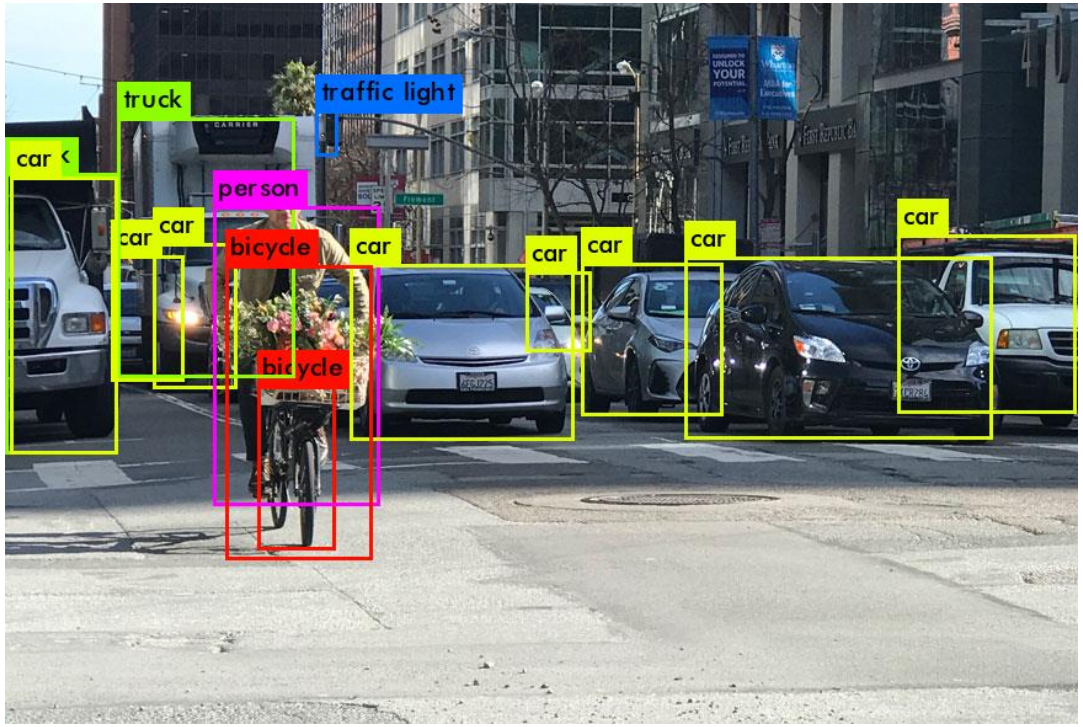
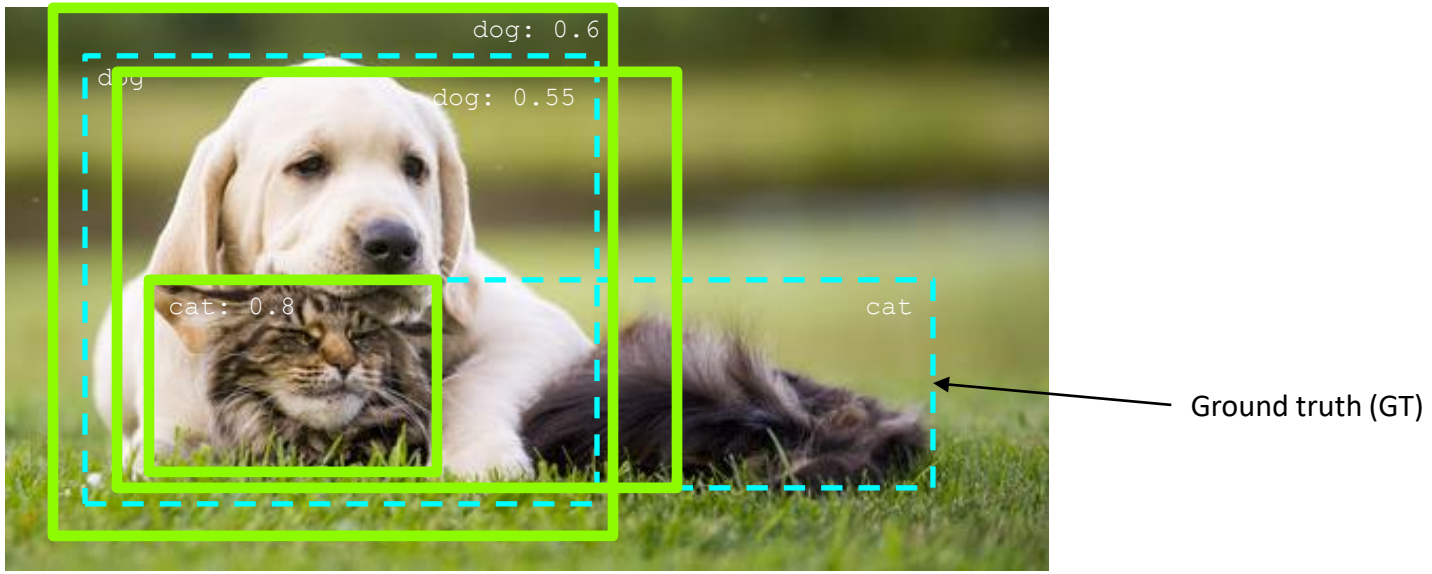


Image source

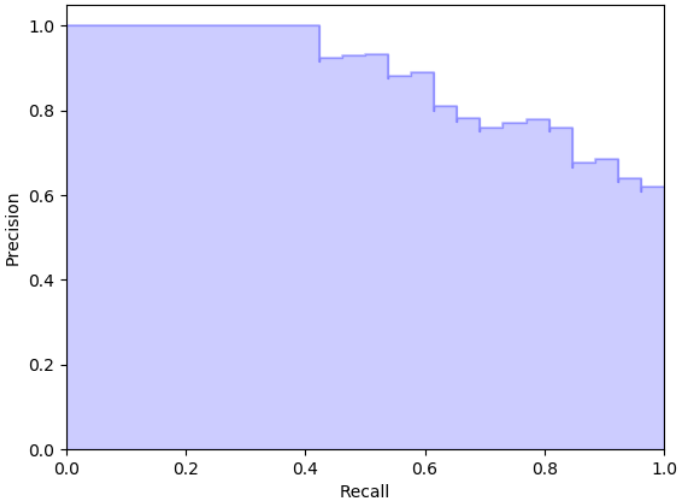
Object detection evaluation

- At test time, predict bounding boxes, class labels, and confidence scores
- For each detection, determine whether it is a true or false positive
  - PASCAL criterion: Area(GT ∩ Det) / Area(GT ∪ Det) > 0.5
  - For multiple detections of the same ground truth box, only one is considered a true positive



Object detection evaluation

- At test time, predict bounding boxes, class labels, and confidence scores
- For each detection, determine whether it is a true or false positive
- For each class, sort detections from highest to lowest confidence, plot Recall-Precision curve and compute Average Precision (area under the curve)
- Take mean of AP over classes to get mAP



Precision: true positive detections / total detections

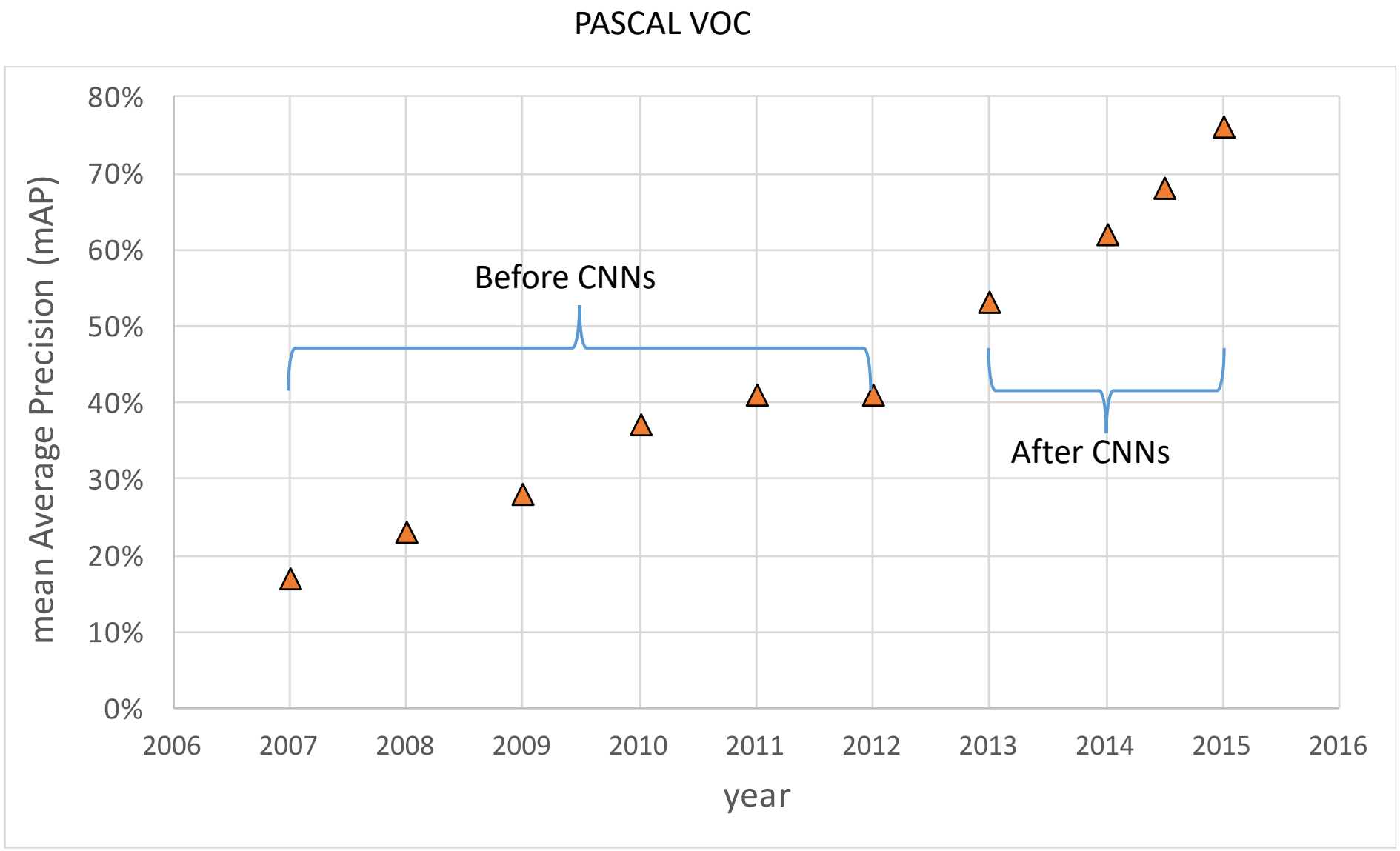
Recall: true positive detections / total positive test instances

PASCAL VOC Challenge (2005-2012)

- 20 challenge classes:
  - *Person*
  - *Animals:* bird, cat, cow, dog, horse, sheep
  - *Vehicles:* airplane, bicycle, boat, bus, car, motorbike, train
  - *Indoor:* bottle, chair, dining table, potted plant, sofa, tv/monitor
- Dataset size (by 2012): 11.5K training/validation images, 27K bounding boxes, 7K segmentations

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

Progress on PASCAL detection



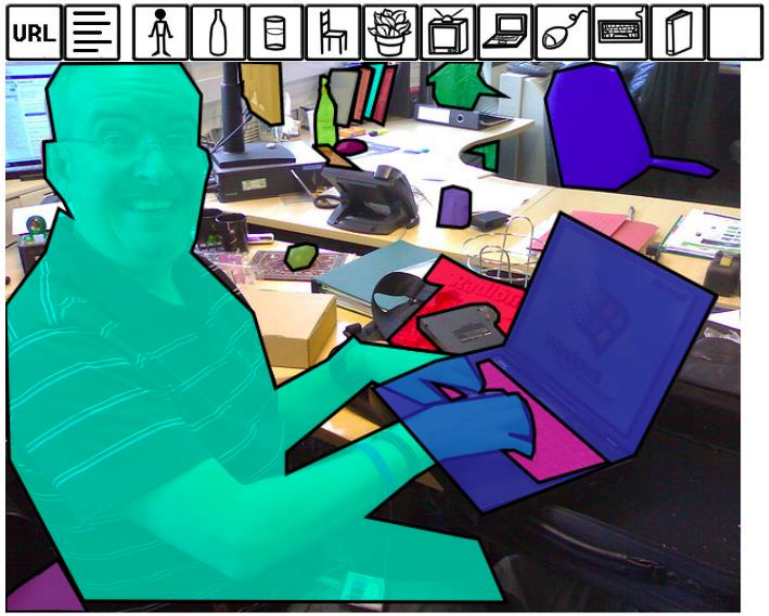
Current benchmark: COCO

What is COCO?

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints

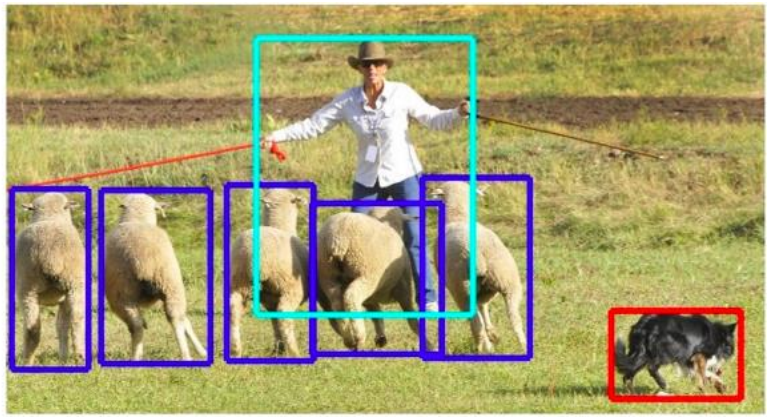
COCO
Common Objects in Context



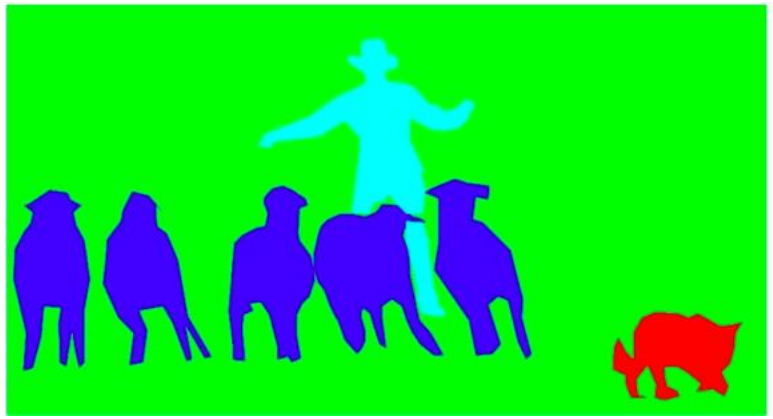
http://cocodataset.org/#home

COCO dataset: Tasks

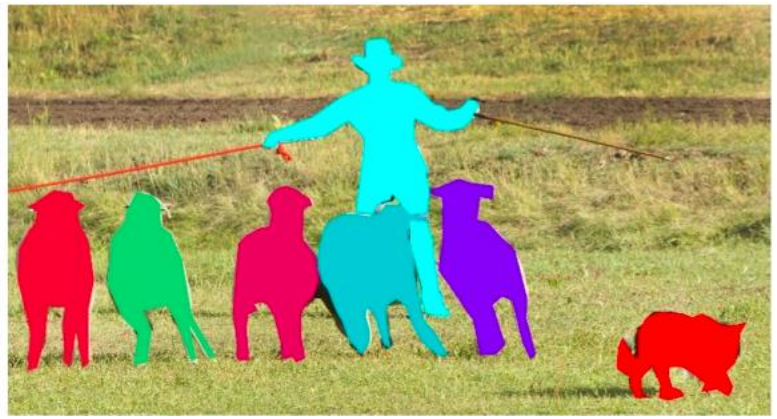
image classification



object detection



semantic segmentation

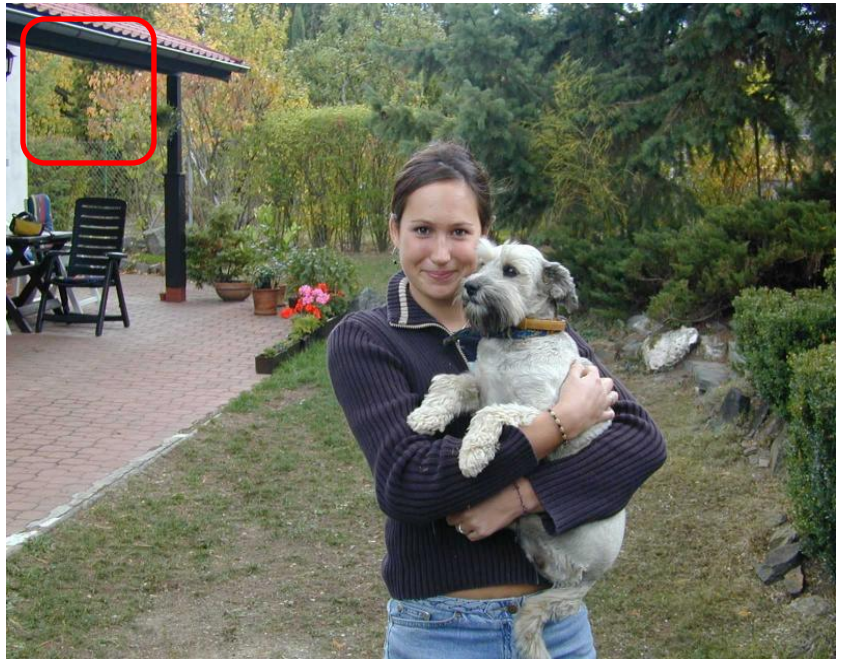


instance segmentation

- Also:
  - keypoint prediction,
  - captioning,
  - question answering
  - …

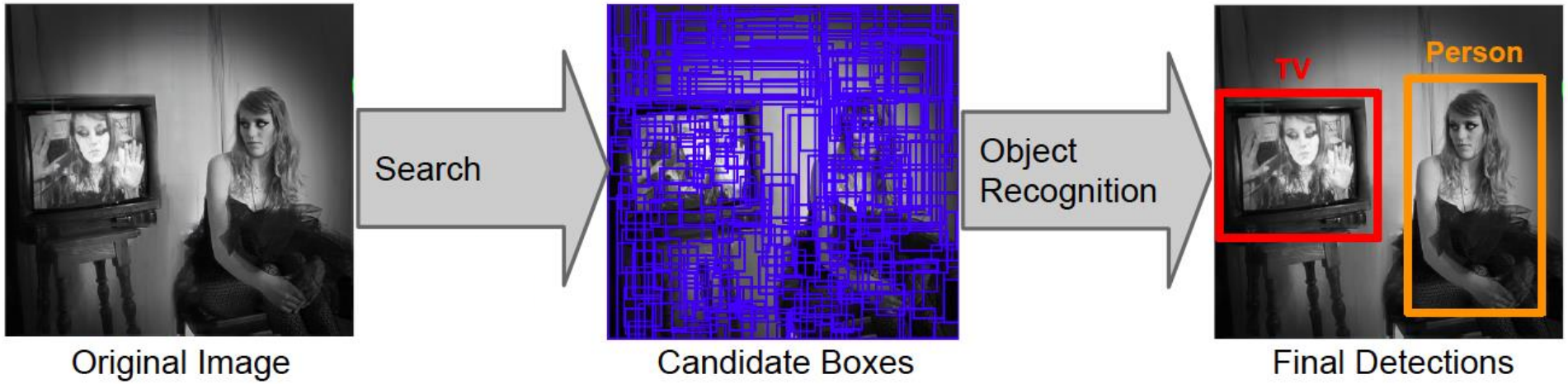
- Leaderboard: http://cocodataset.org/#detection-leaderboard
- Official COCO challenges no longer include detection
  - Emphasis has shifted to instance segmentation and dense semantic segmentation

Approaches to detection: Sliding windows



- Slide a window across the image and evaluate a detection model at each location
  - Thousands of windows to evaluate: efficiency and low false positive rates are essential
  - Difficult to extend to a large range of scales, aspect ratios

Approaches to detection: Object proposals



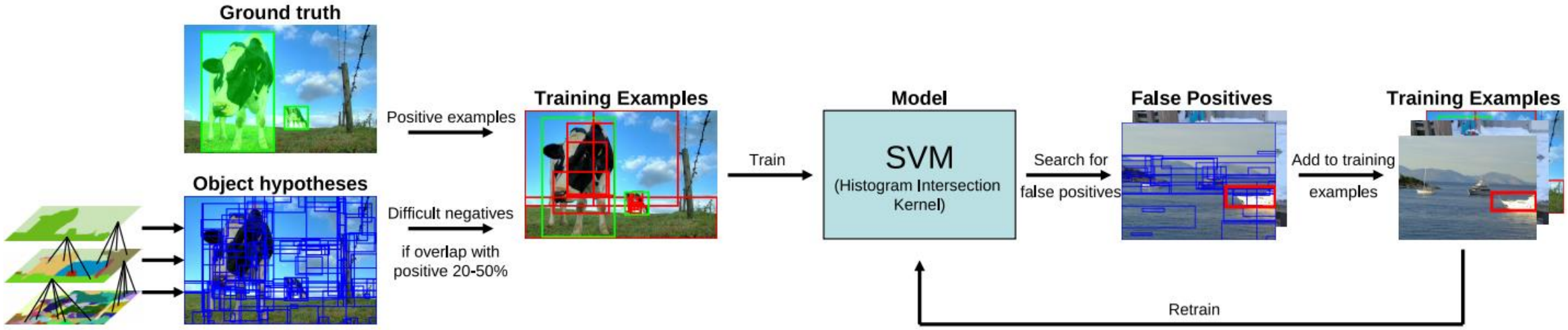
- Generate and evaluate a few hundred region proposals
  - Proposal mechanism can take advantage of low-level perceptual organization cues
  - Proposal mechanism can be category-specific or category-independent, hand-crafted or trained
  - Classifier can be slower but more powerful

Selective search for detection

- Use hierarchical segmentation: start with small *superpixels* and merge based on diverse cues

J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, Selective Search for Object Recognition, IJCV 2013

Selective search for detection



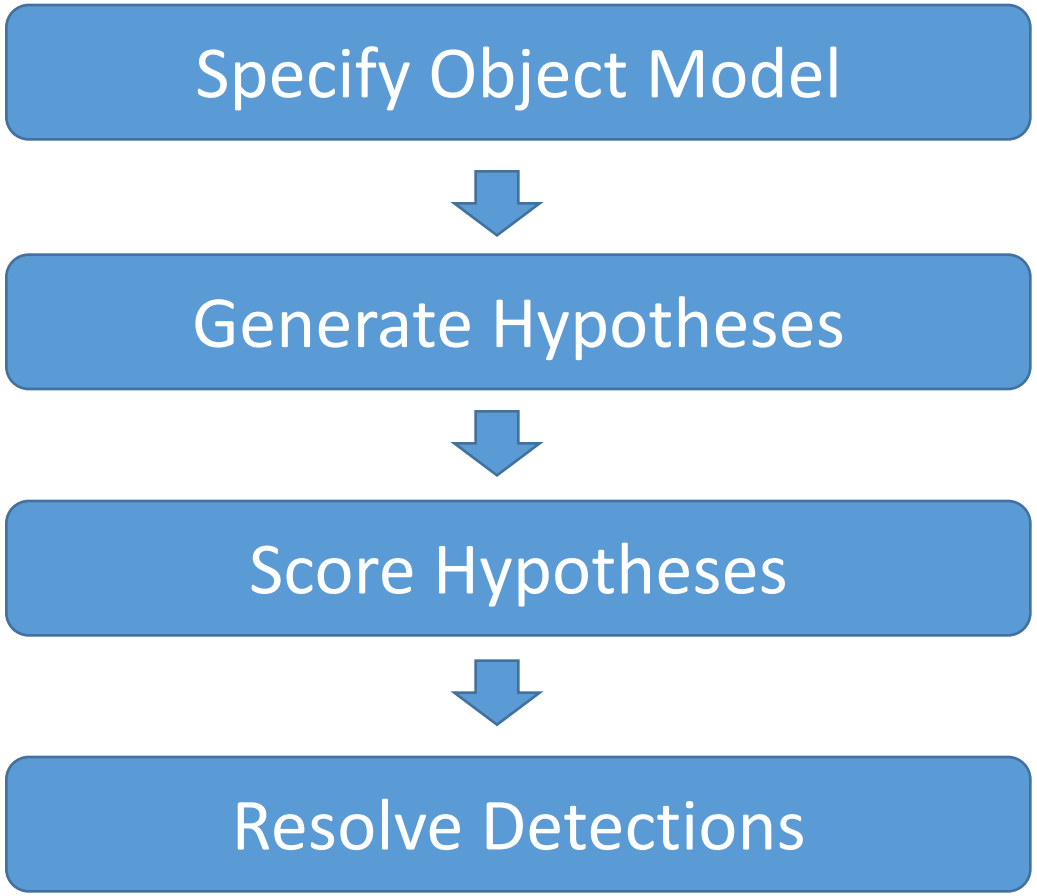
- Feature extraction: color SIFT, codebook of size 4K, spatial pyramid with four levels = 360K dimensions

J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, Selective Search for Object Recognition, IJCV 2013

Approaches to detection

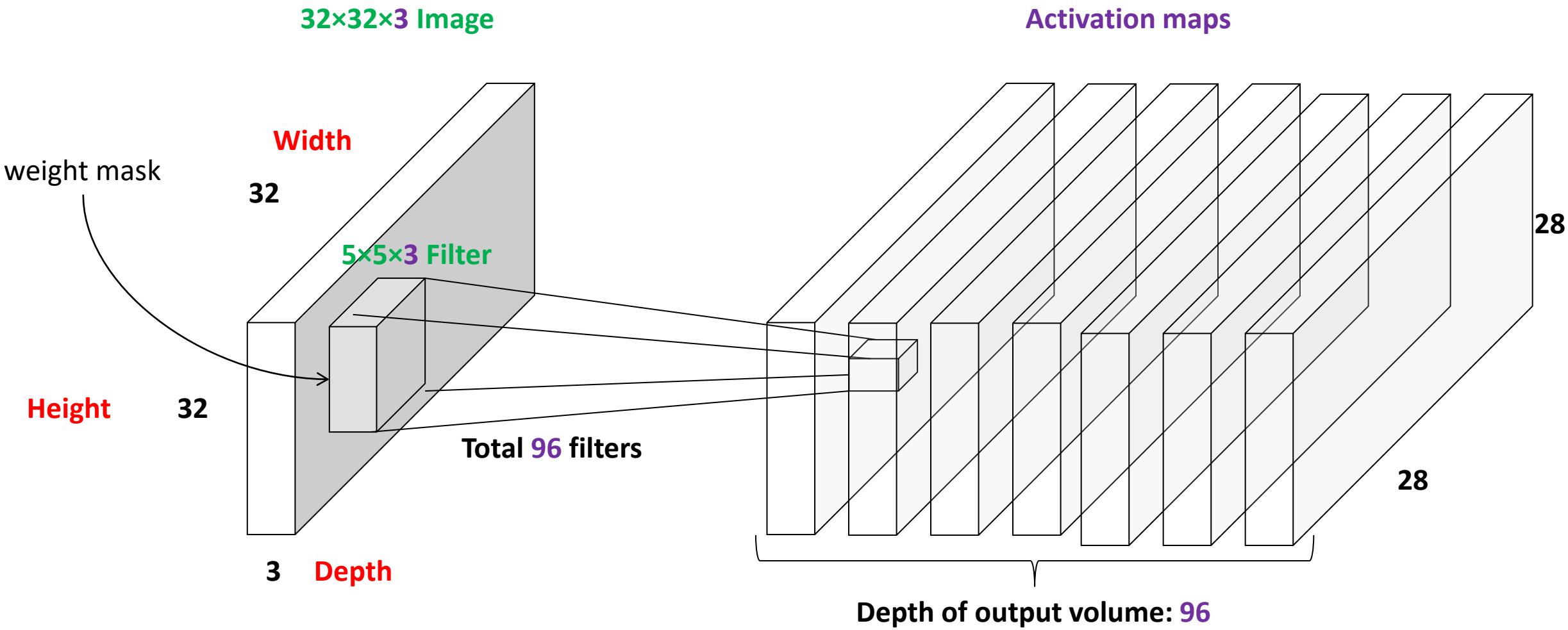
- Before ~2010, dominated by sliding windows
- 2010-2013: proposal-driven
- Deep learning approaches started as proposal-driven, but have evolved back toward sliding windows
- Most recently, “global” methods are becoming more common

General Process of Object Recognition

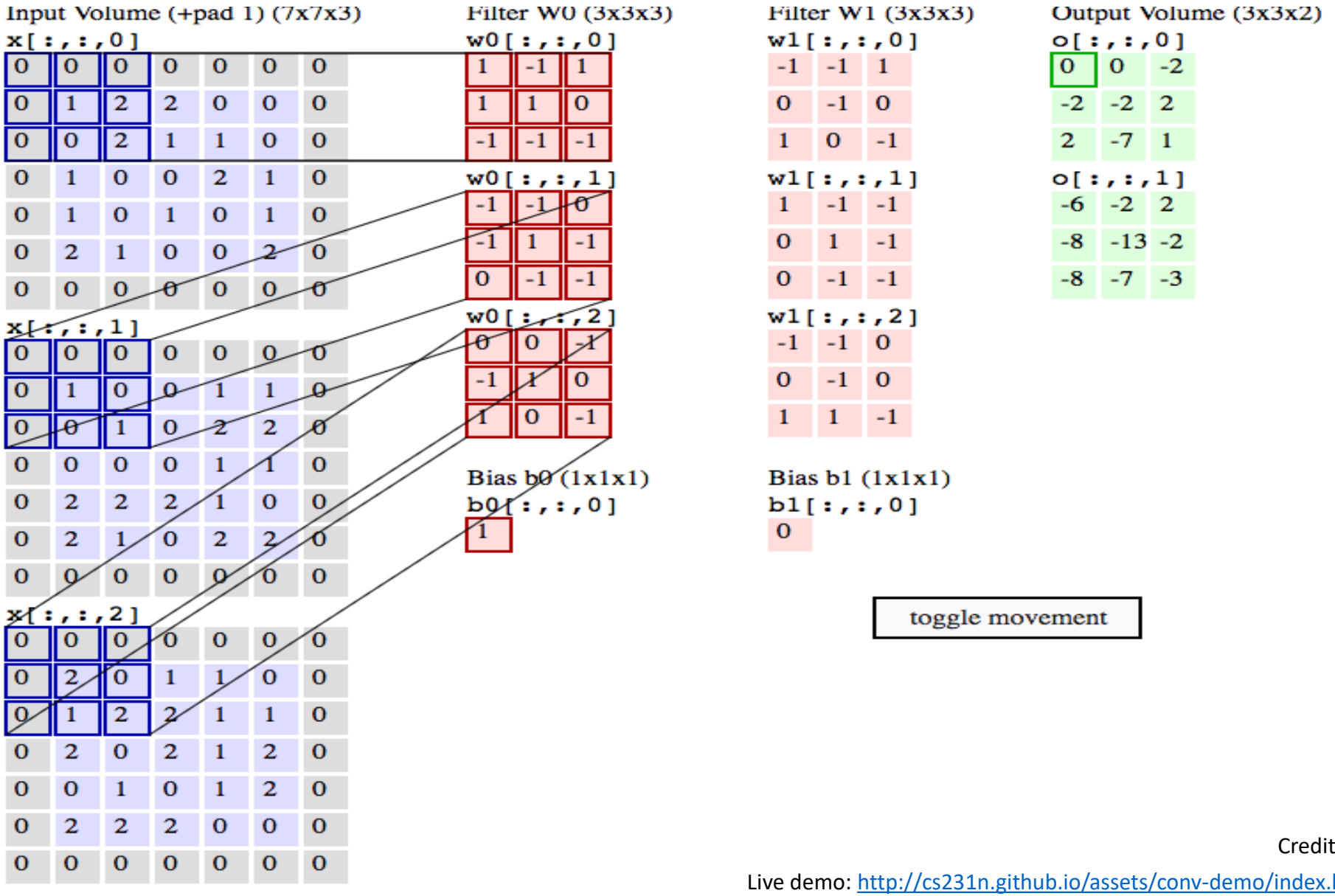


Convolutional Neural Network (Quick Recap)

Recap – Convolutional layer



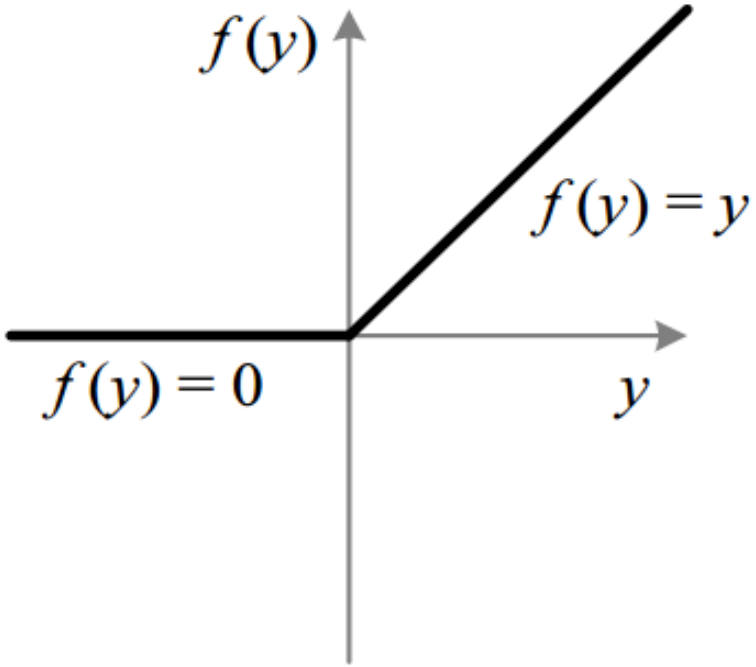
How it works?

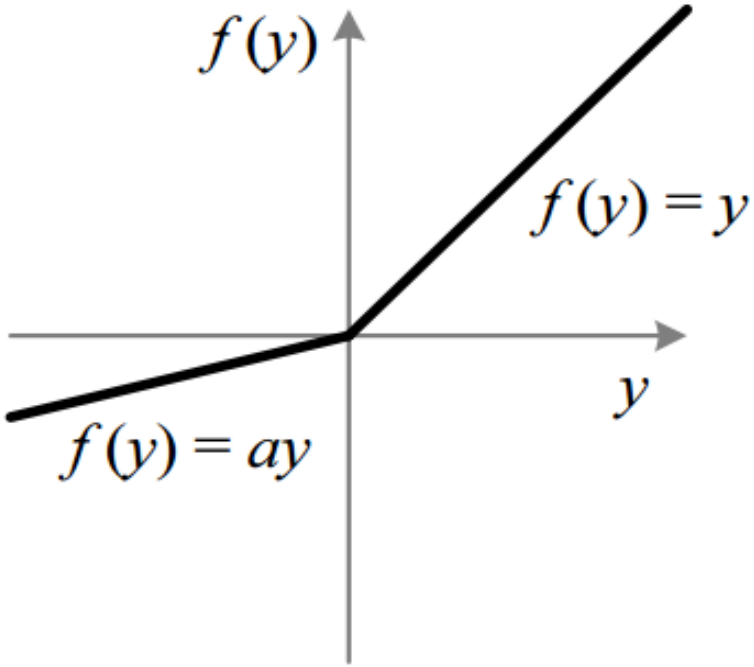


Credit: Andrej Karpathy

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

Recap – Activation

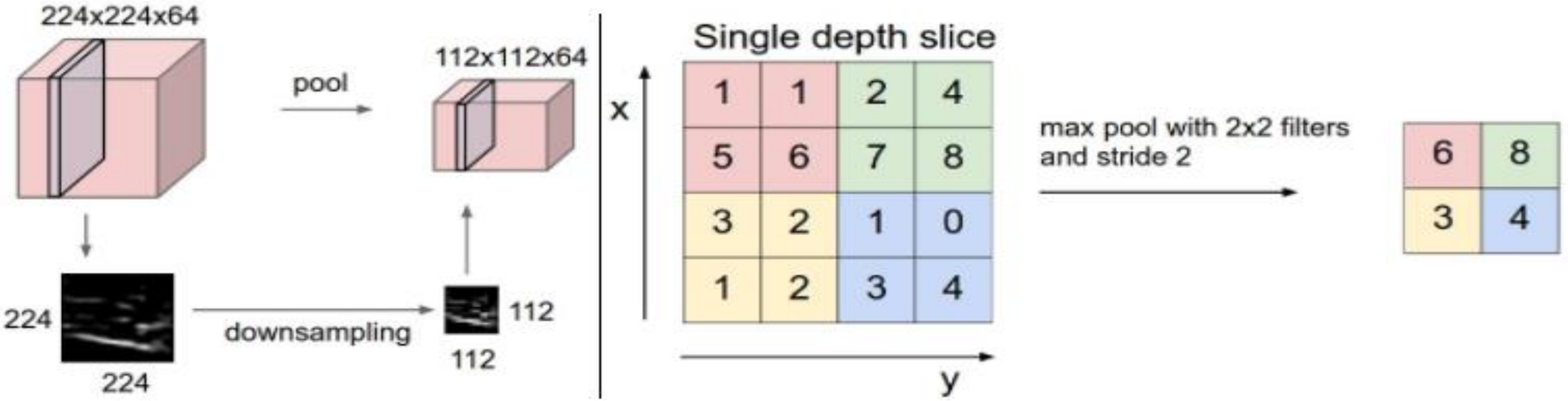




- Introduce the non-linearity

Credit: Andrej Karpathy

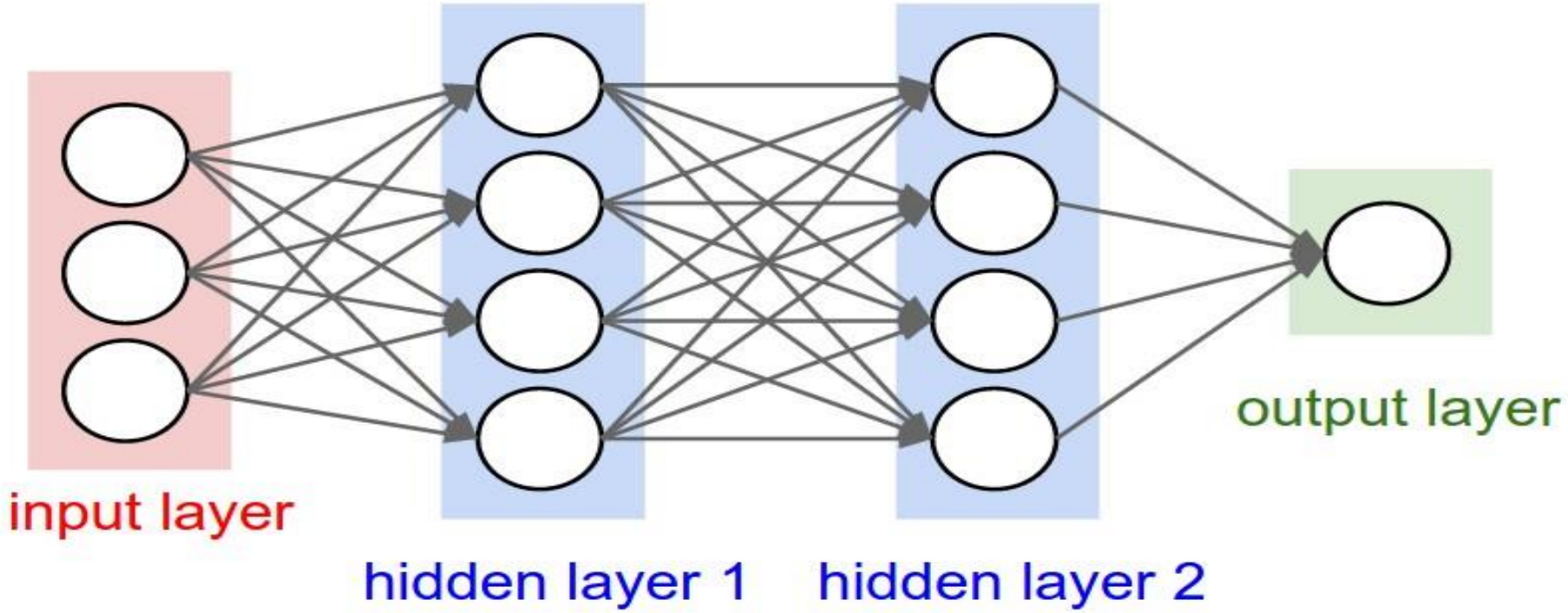
Recap – Pooling layer



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

Credit: Andrej Karpathy

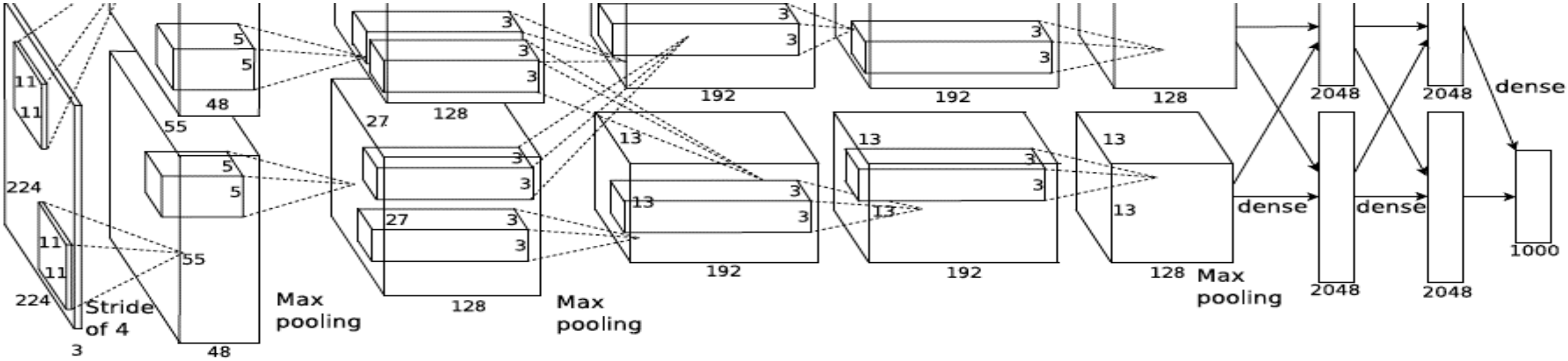
Recap – Fully-connected layer



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

Credit: Andrej Karpathy

Put them all together



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

Credit: Andrej Karpathy

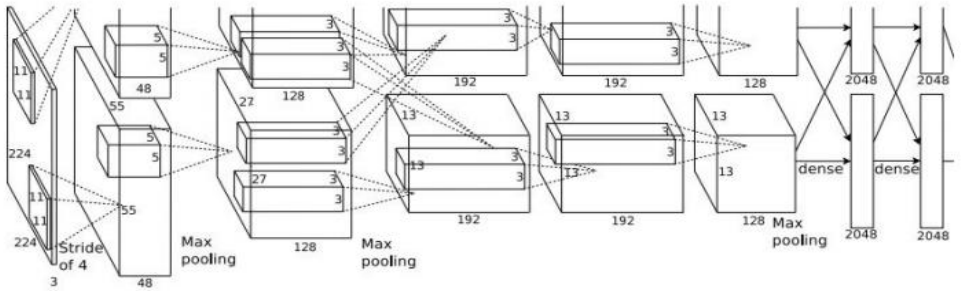
CNN methods for object detection

CNN as feature extractor

Image credit: Justin Johnson

CNN as feature extractor

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

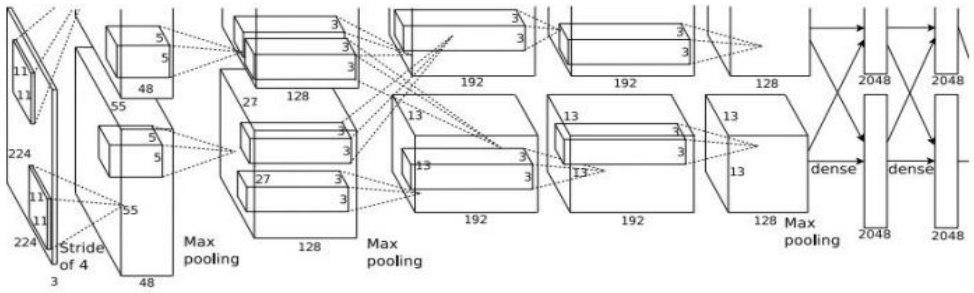


Dog? NO
Cat? NO
Background? YES

Slides by Justin Johnson

CNN as feature extractor

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

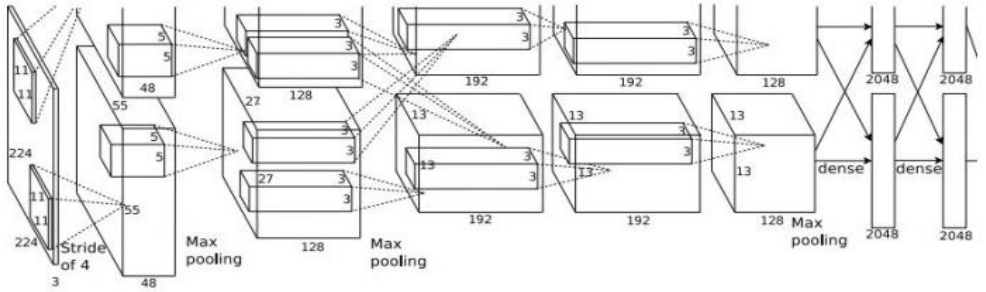


Dog? YES
Cat? NO
Background? NO

Slides by Justin Johnson

CNN as feature extractor

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

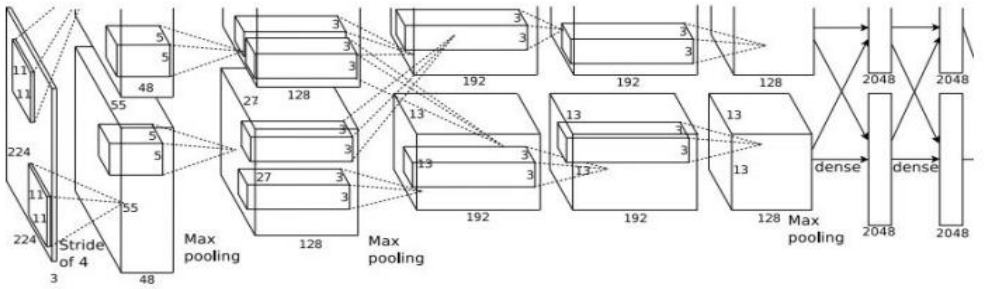


Dog? YES
Cat? NO
Background? NO

Slides by Justin Johnson

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

Slides by Justin Johnson

CNN as feature extractor

- What could be the problems?

CNN as feature extractor

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

CNN as feature extractor

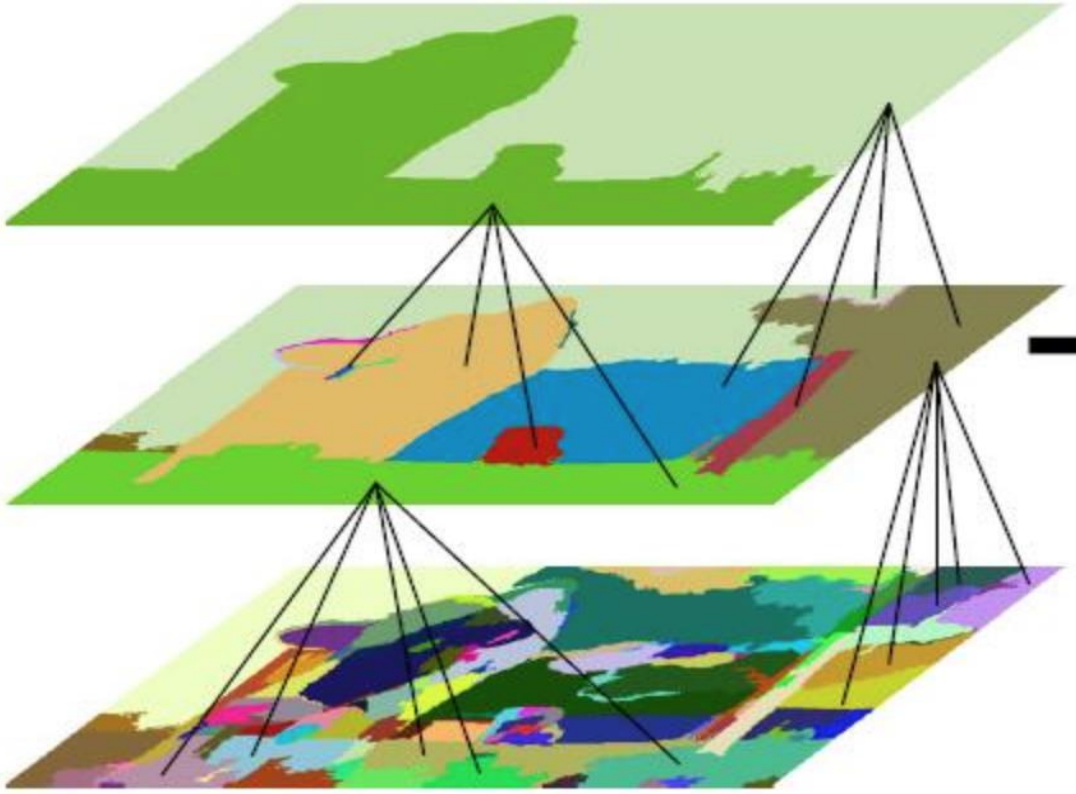
- What could be the problems?
  - Suppose we have a 600 x 600 image, if sliding window size is 20 x 20, then have (600-20+1) x (600-20+1) = ~330,000 windows
  - Sometimes we want to have more accurate results -> multi-scale detection
    - Resize image
    - Multi-scale sliding window

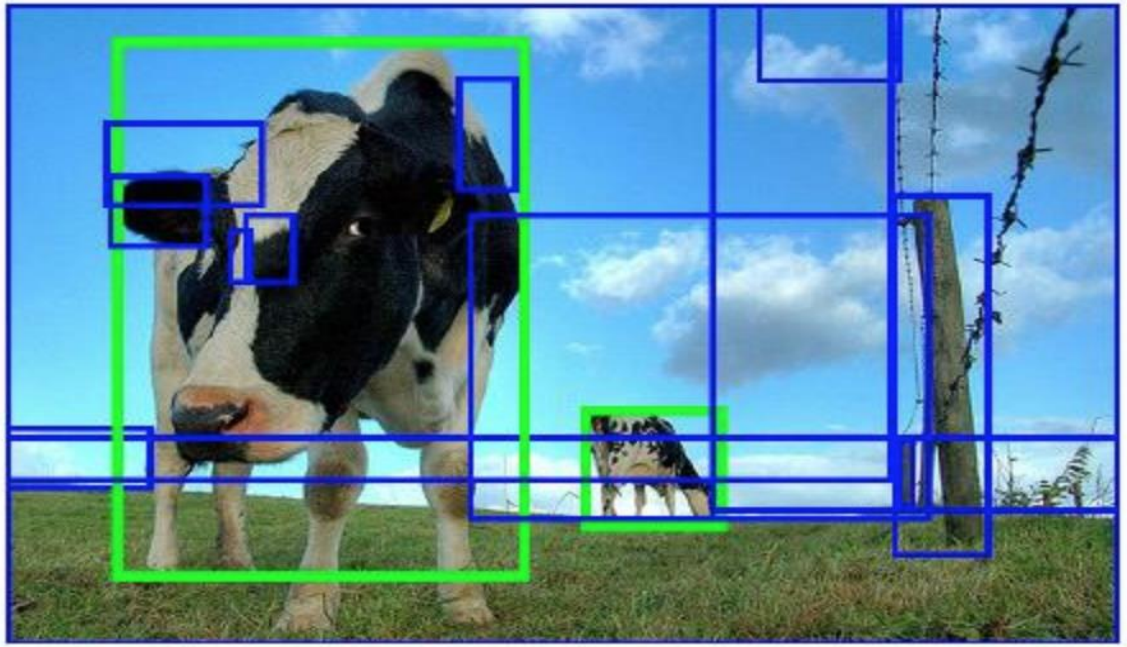
CNN as feature extractor

- What could be the problems?
  - Suppose we have a 600 x 600 image, if sliding window size is 20 x 20, then have (600-20+1) x (600-20+1) = ~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 ~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

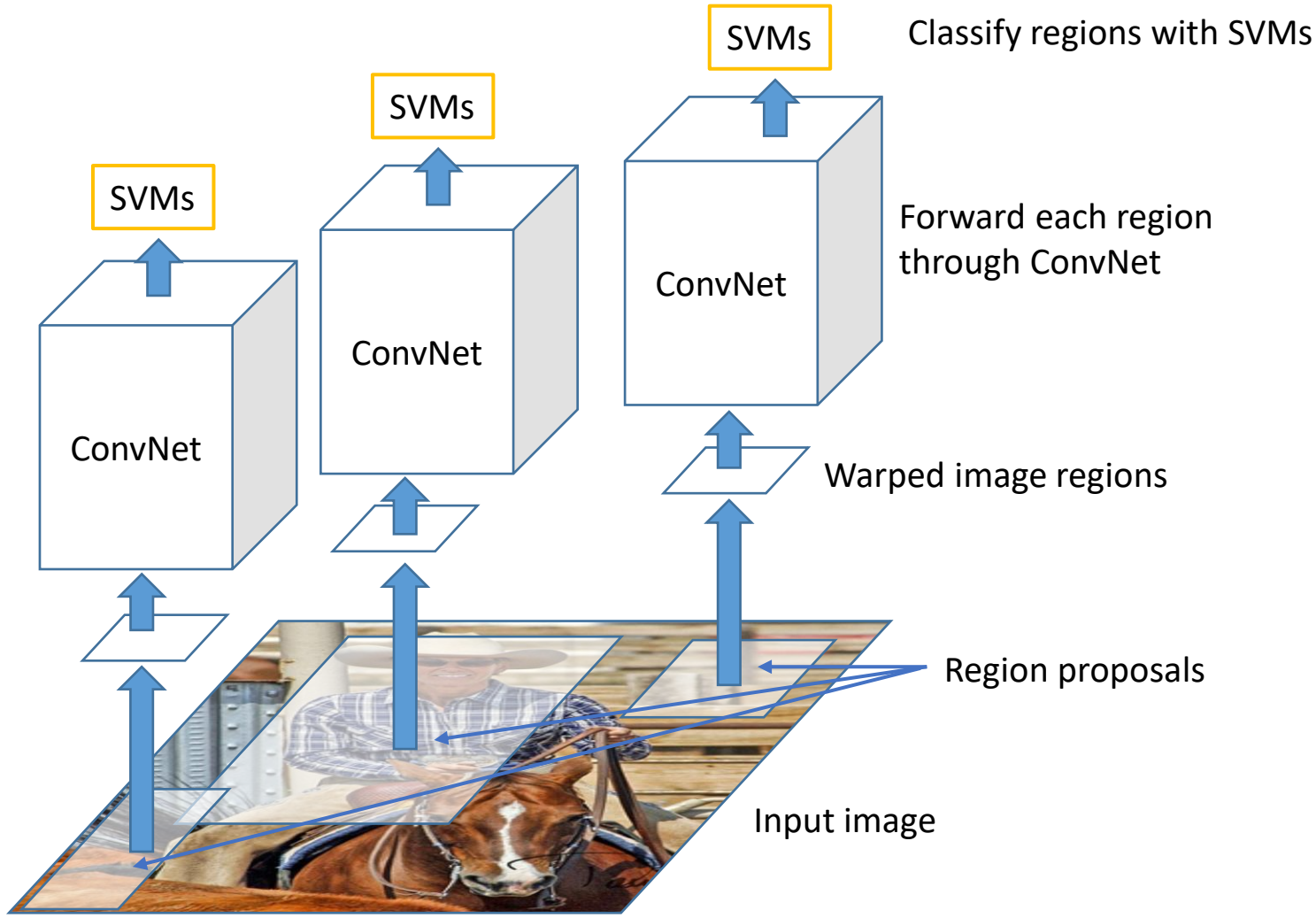




Uijilings et al. IJCV 2013

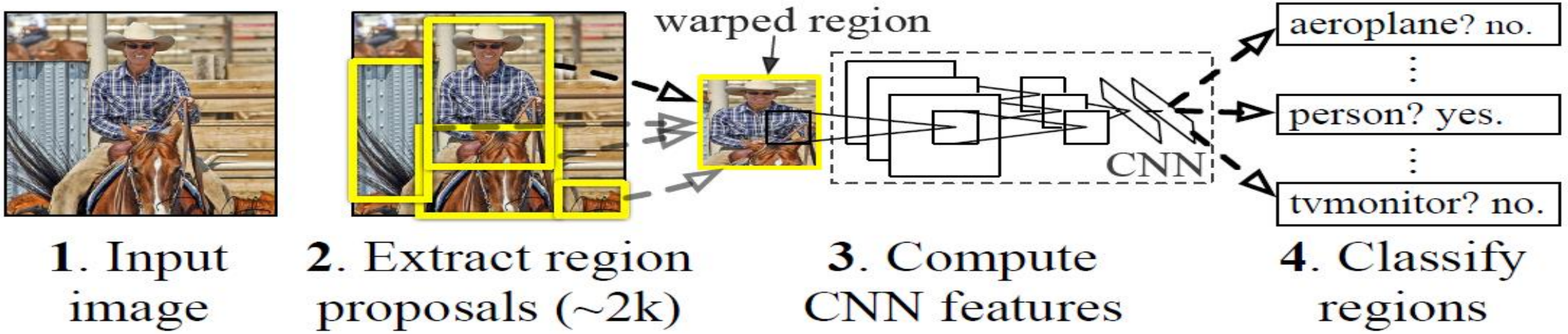
R-CNN: Region proposals + CNN features

Source: R. Girshick



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

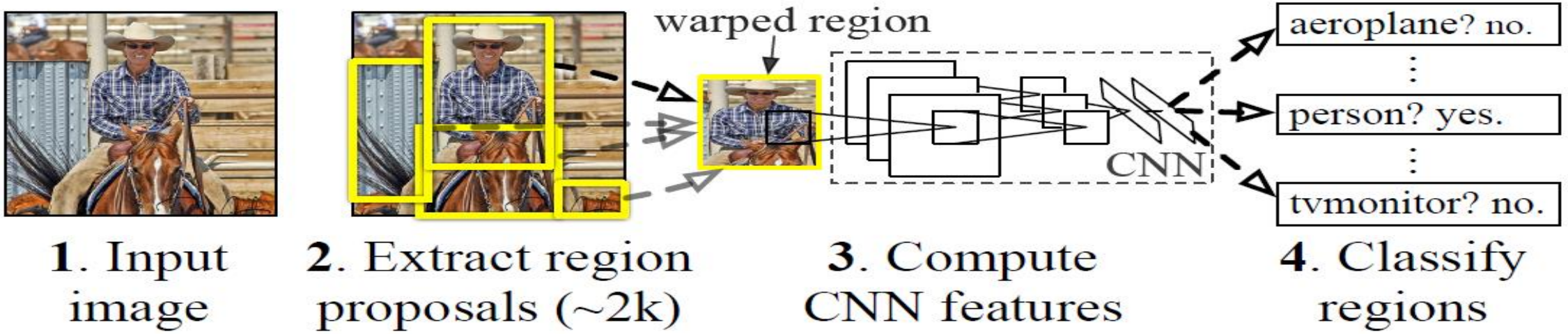
R-CNN (Girshick et al. CVPR 2014)



- Replace sliding windows with “selective search” region proposals (Uijilings 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

http://arxiv.org/pdf/1311.2524.pdf

R-CNN (Girshick et al. CVPR 2014)



- **Regions**: ~2000 Selective Search proposals
- **Network**: AlexNet *pre-trained* on ImageNet (1000 classes), *fine-tuned* on PASCAL (21 classes)
- **Final detector**: warp proposal regions, extract fc7 network activations (4096 dimensions), classify with linear SVM
- **Bounding box regression** to refine box locations
- **Performance:** mAP of **53.7%** on PASCAL 2010
  (vs. **35.1%** for Selective Search and **33.4%** for Deformable Part Models)

http://arxiv.org/pdf/1311.2524.pdf

R-CNN pros and cons

- Pros
  - Much more accurate than previous approaches!
  - Any deep architecture can immediately be “plugged in”
- Cons
  - Not a single end-to-end system
    - Fine-tune network with softmax classifier (log loss)
    - Train post-hoc linear SVMs (hinge loss)
    - Train post-hoc bounding-box regressions (least squares)
  - Training was slow (84h), took up a lot of storage
    - 2000 CNN passes per image
  - Inference (detection) was slow (47s / image with VGG16)

Bounding Box Regression

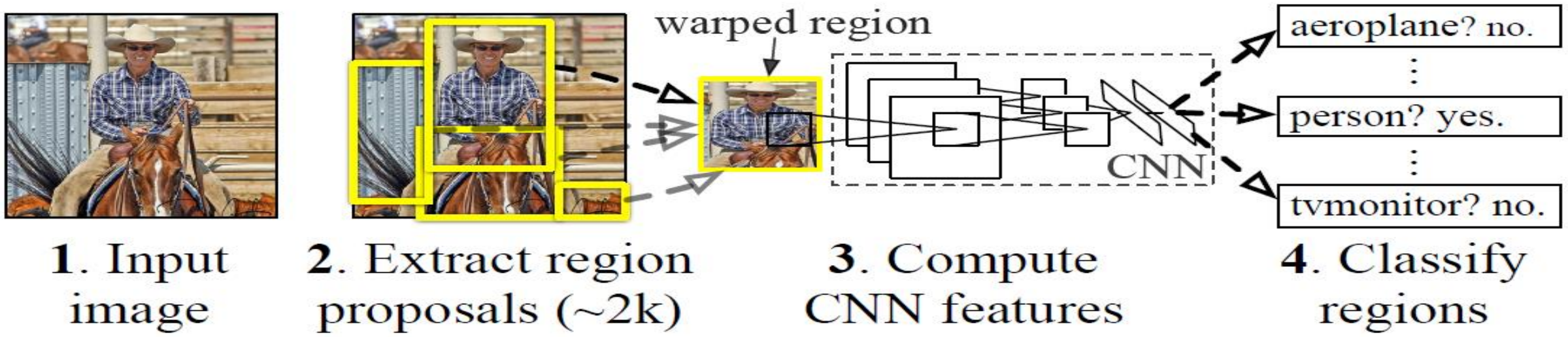
- Intuition
  - If you observe part of the object, according to the seen examples, you should be able to refine the localization
  - E.g. given the red box below, since you've seen many airplanes, you know this is not a good localization, you will adjust it to the green one

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

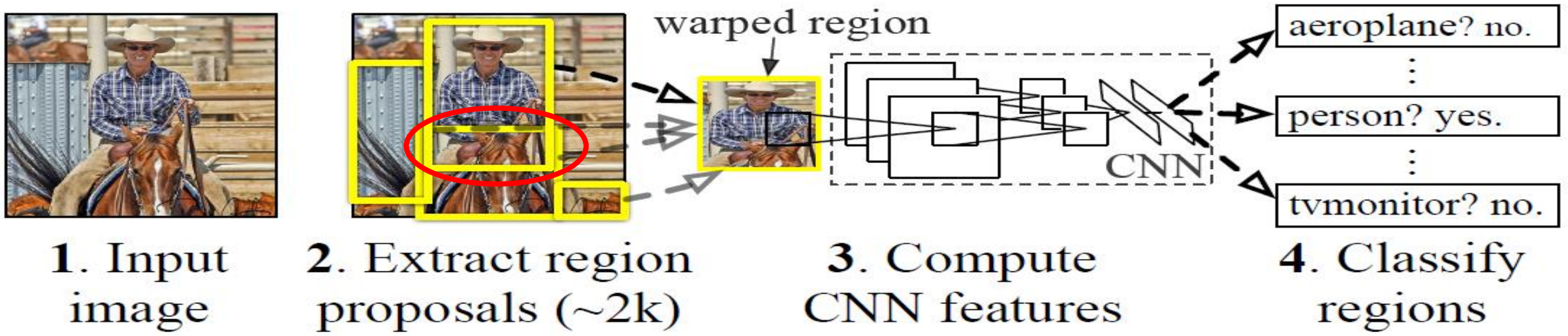
R-CNN (Girshick et al. CVPR 2014)

- What could be the problems?



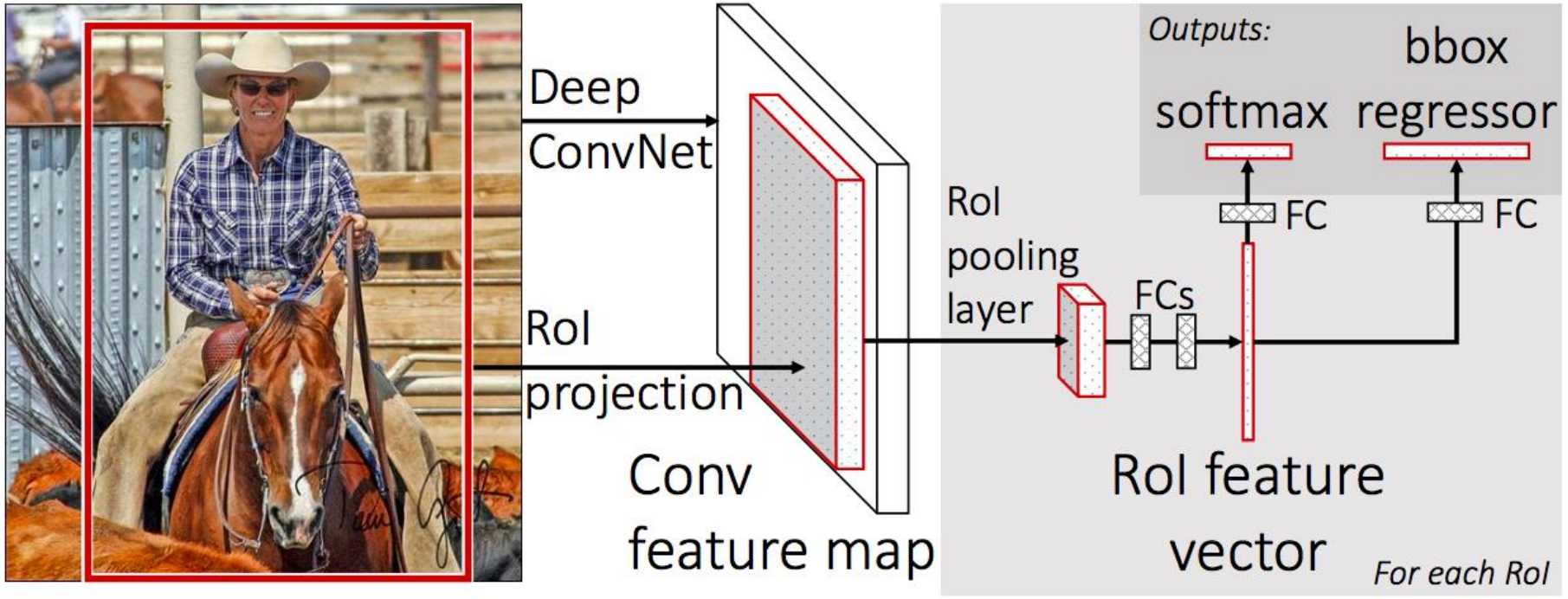
R-CNN (Girshick et al. CVPR 2014)

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



Fast R-CNN (Girshick ICCV 2015)

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

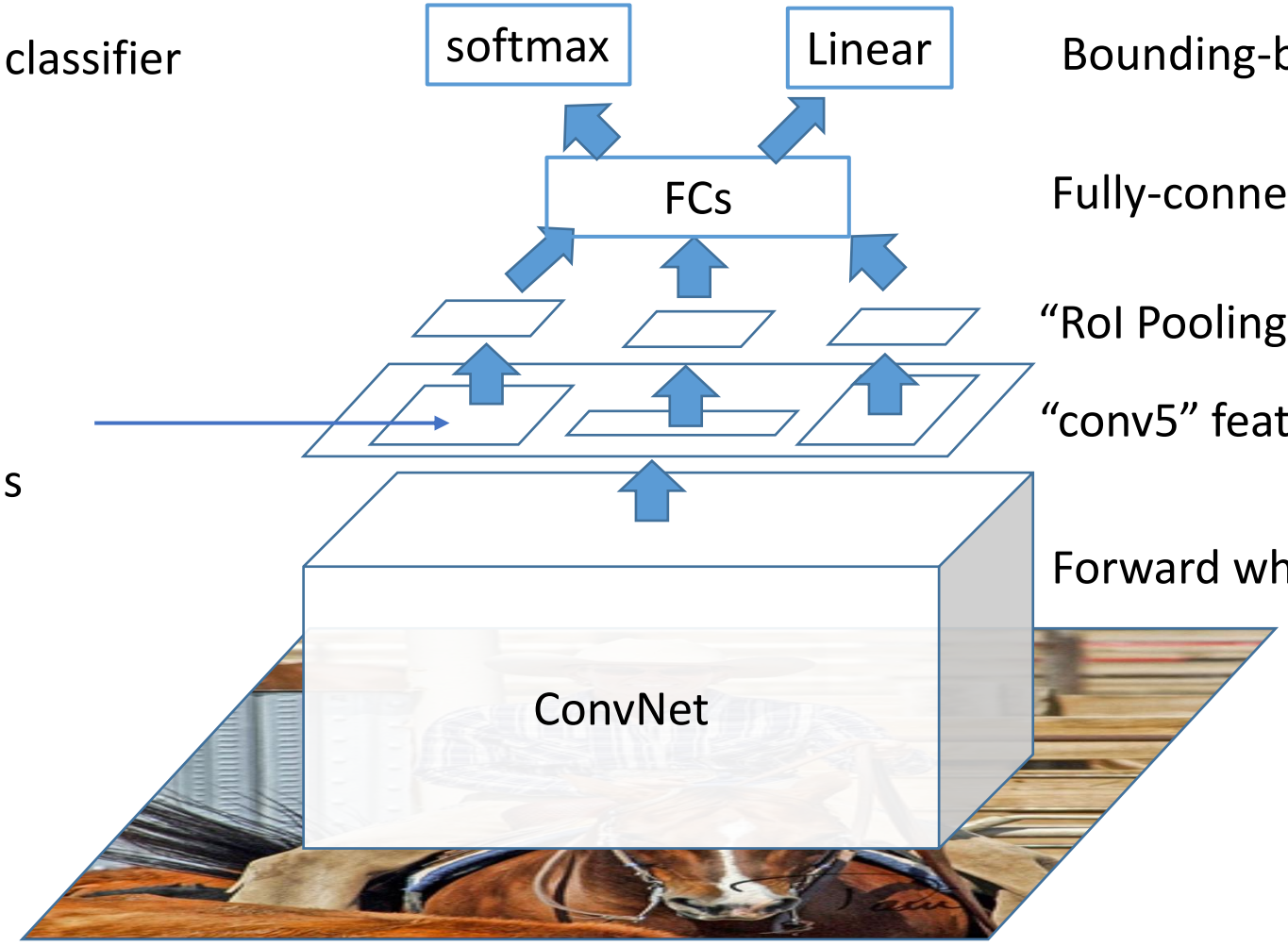


- For each RoI, network predicts probabilities for $C + 1$ classes (class 0 is background) and four bounding box offsets for C classes

https://arxiv.org/pdf/1504.08083.pdf

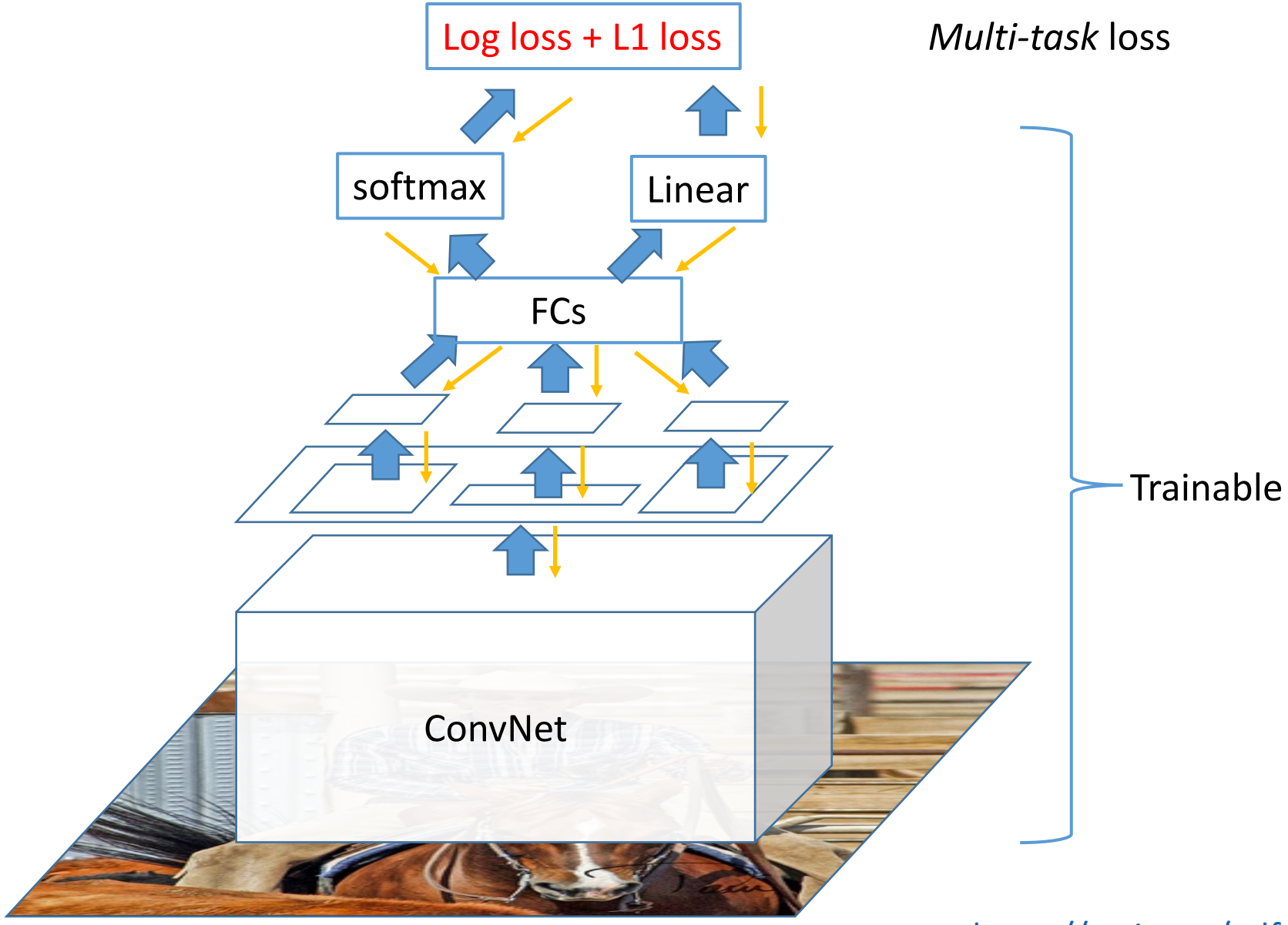
Fast R-CNN (Girshick ICCV 2015)

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



https://arxiv.org/pdf/1504.08083.pdf

Fast R-CNN (Girshick ICCV 2015)



https://arxiv.org/pdf/1504.08083.pdf

Fast R-CNN results

	Fast R-CNN	R-CNN
Train time (h)	**9.5**	84
- Speedup	**8.8x**	
Test time / image	**0.32s**	47.0s
- Test speedup	**146x**	
mAP	**66.9%**	66.0%

(vs. 53.7% for AlexNet)

Timings exclude object proposal time, which is equal for all methods.
All methods use VGG16.

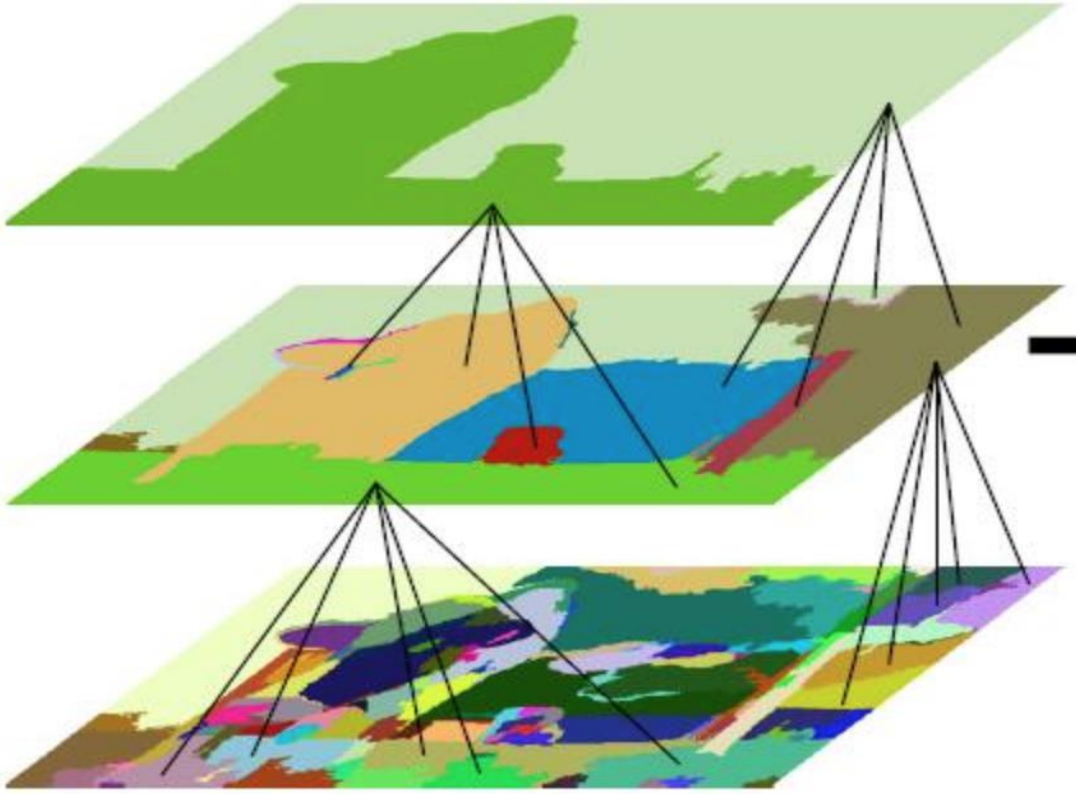
Source: R. Girshick

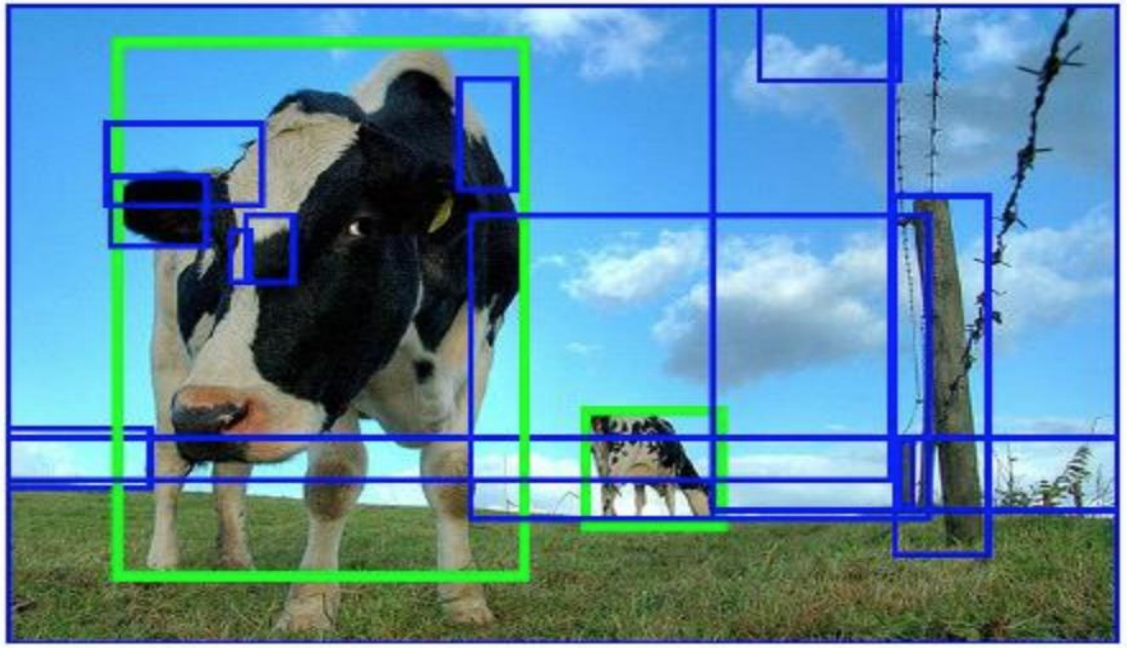
Fast R-CNN (Girshick ICCV 2015)

- 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!



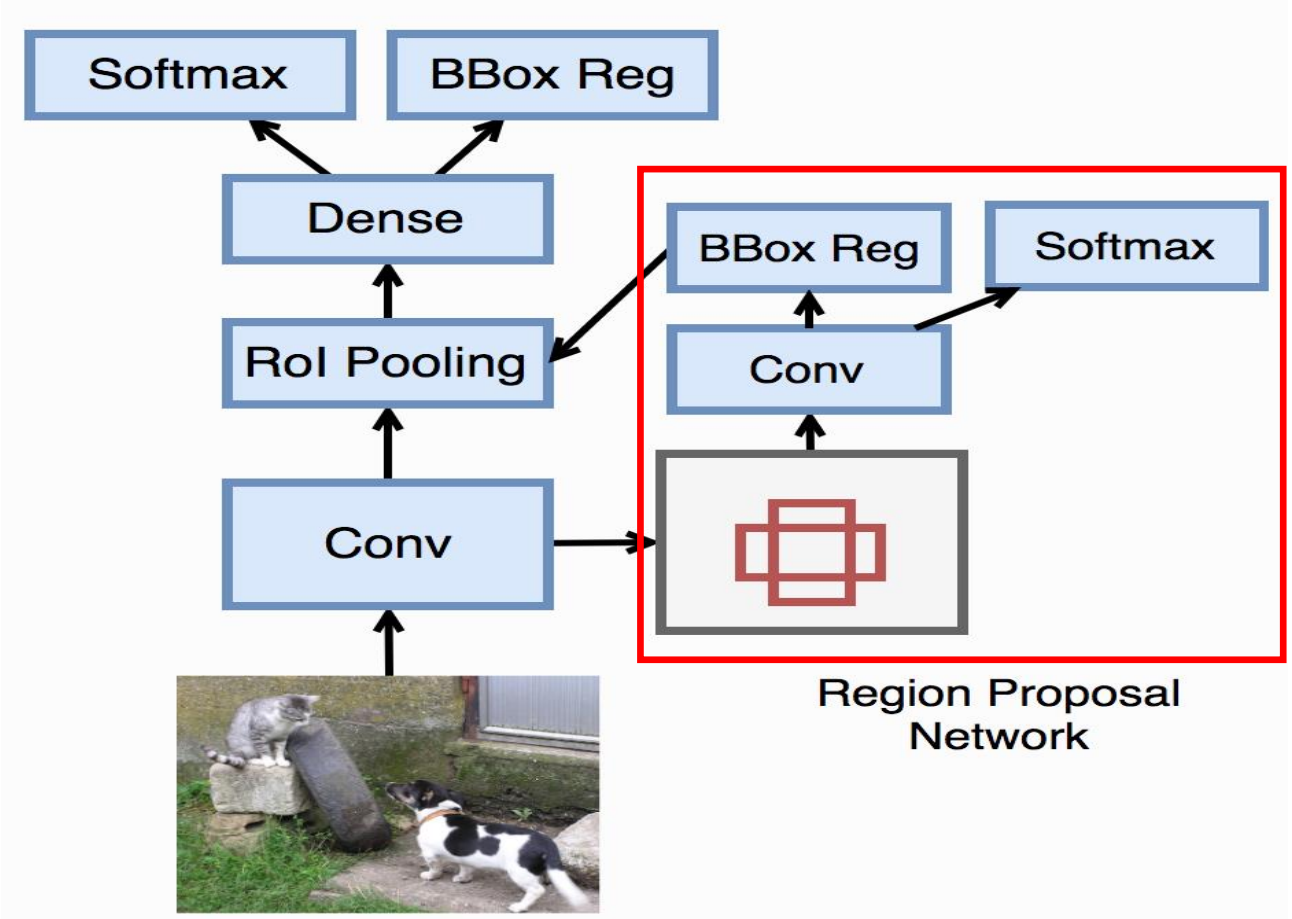


Uijlings et al. IJCV 2013

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

- Solution
  - Why not generate region proposals using CNN??!

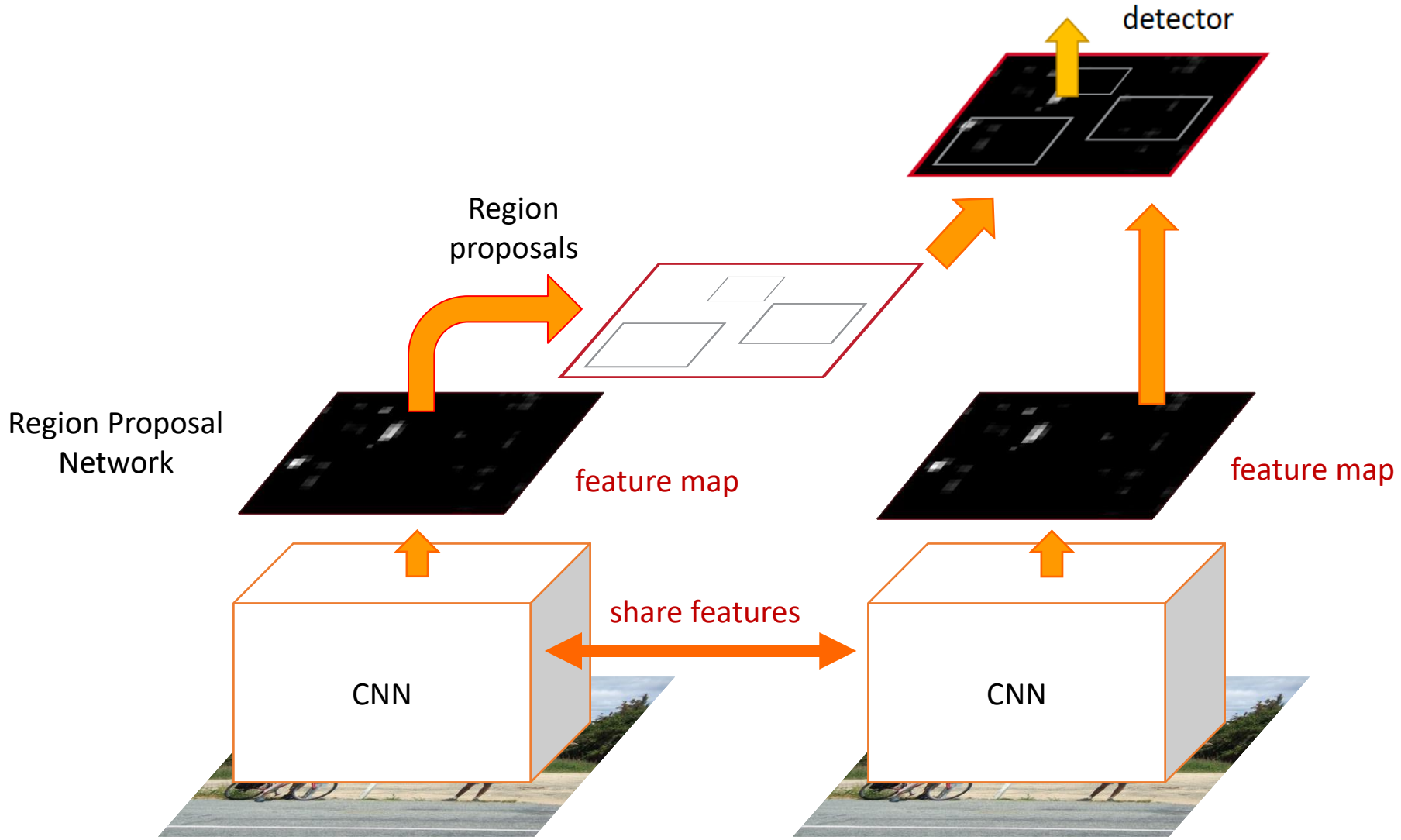
    -> RPN



https://arxiv.org/pdf/1506.01497.pdf

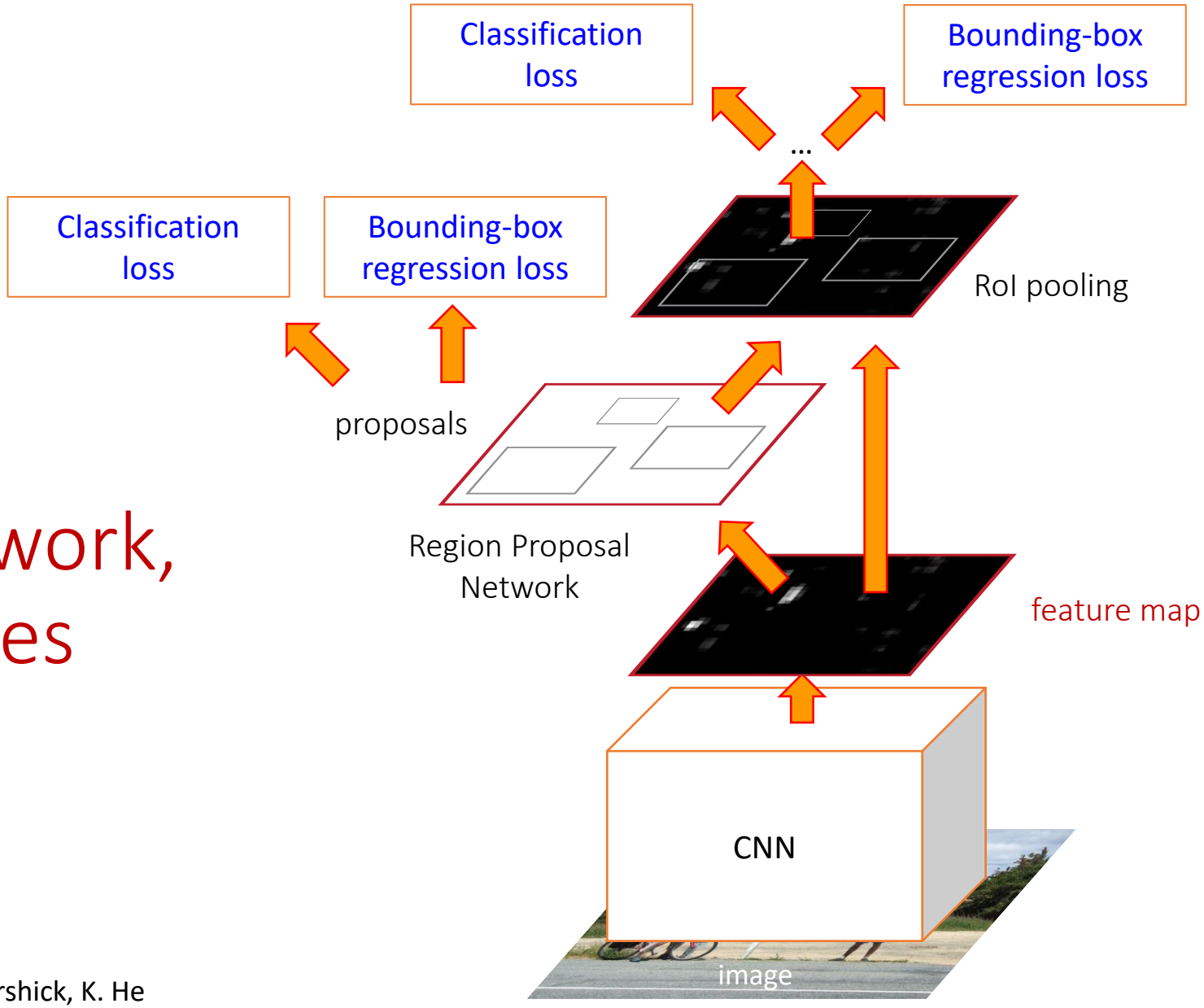
Image credit: http://zh.gluon.ai/chapter_computer-vision/object-detection.html

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



S. Ren, K. He, R. Girshick, and J. Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, NIPS 2015

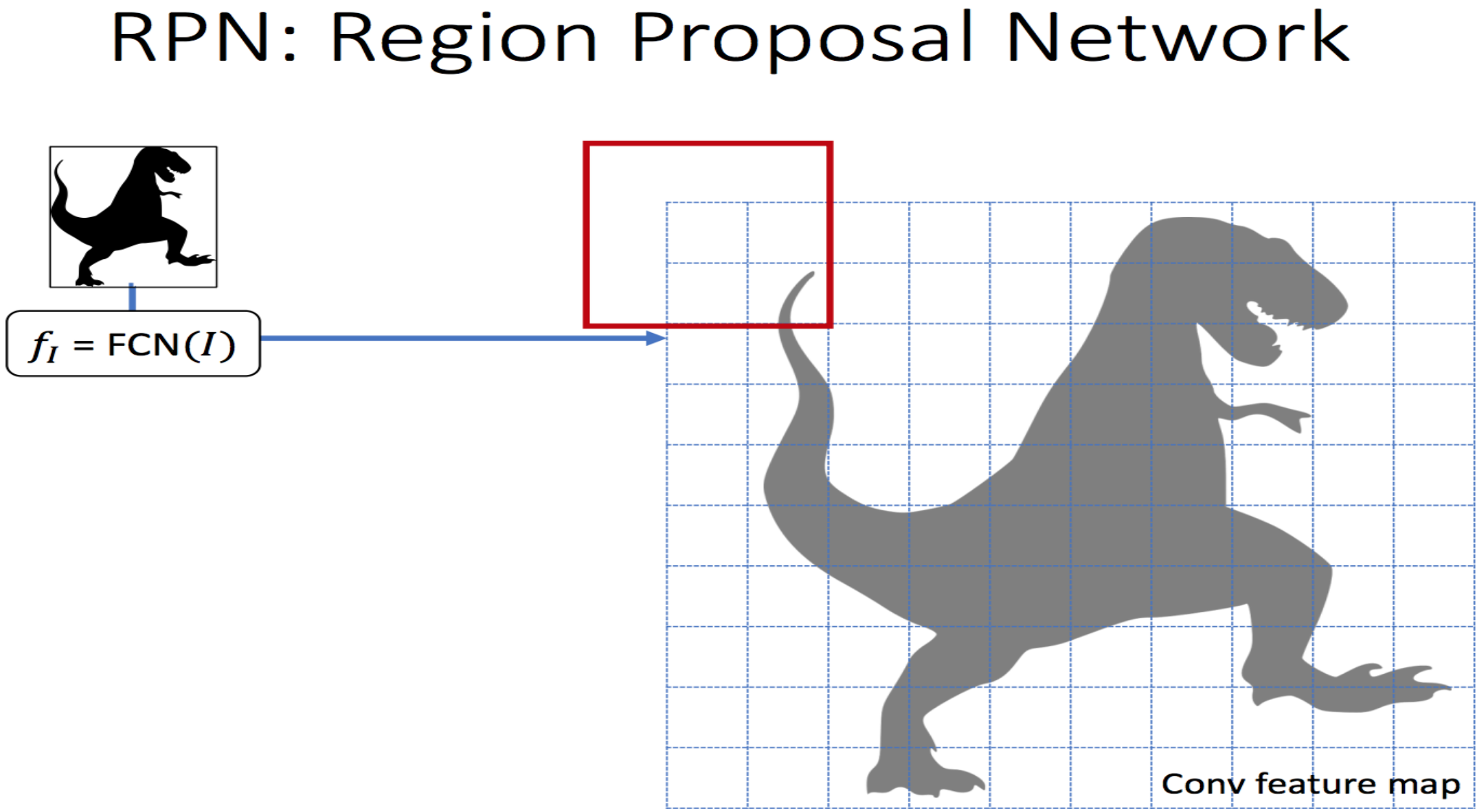
Faster R-CNN (Ren et al. NIPS 2015)



One network, four losses

Source: R. Girshick, K. He

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

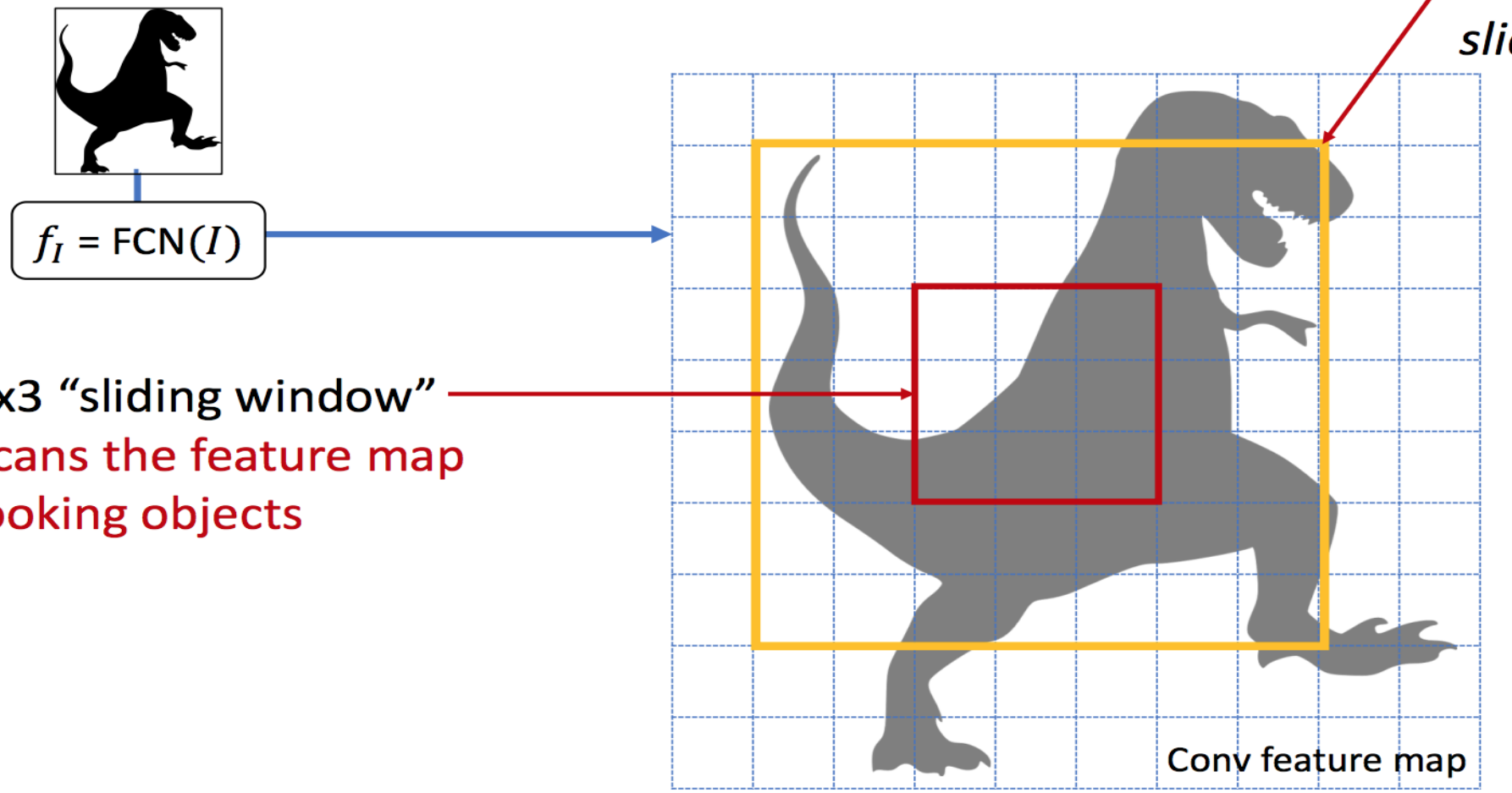


https://arxiv.org/pdf/1506.01497.pdf

Slides by Ross Girshick

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

RPN: Anchor Box

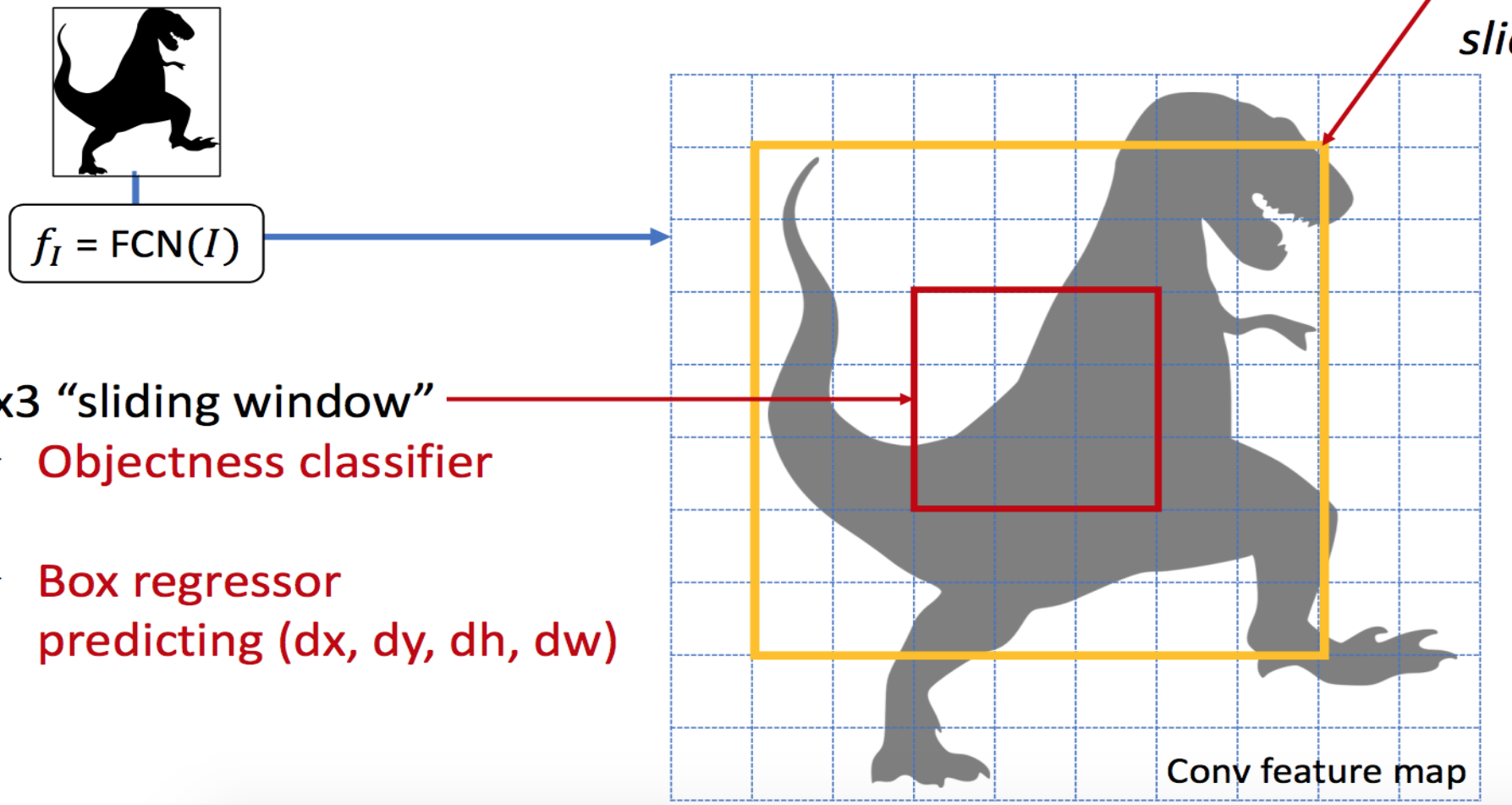


https://arxiv.org/pdf/1506.01497.pdf

Slides by Ross Girshick

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

RPN: Anchor Box



- Objectness classifier
- Box regressor predicting (dx, dy, dh, dw)

https://arxiv.org/pdf/1506.01497.pdf

Slides by Ross Girshick

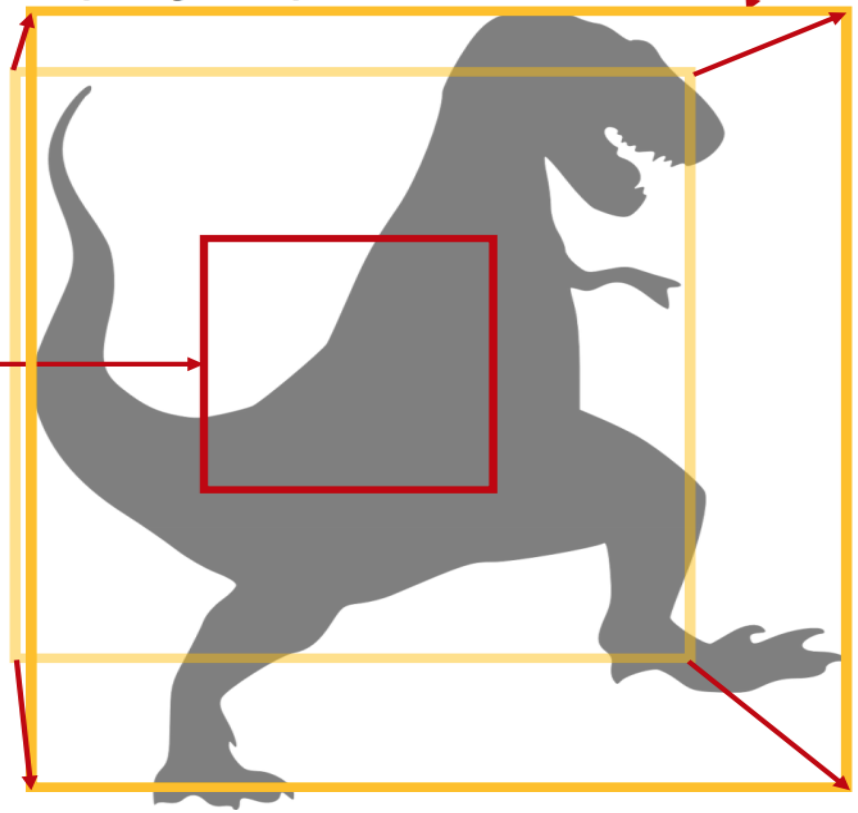
Faster R-CNN (Ren et al. NIPS 2015)

RPN: Prediction (on object)

Objectness score

P(object) = 0.94

Anchor box: transformed by box regressor



3x3 “sliding window”

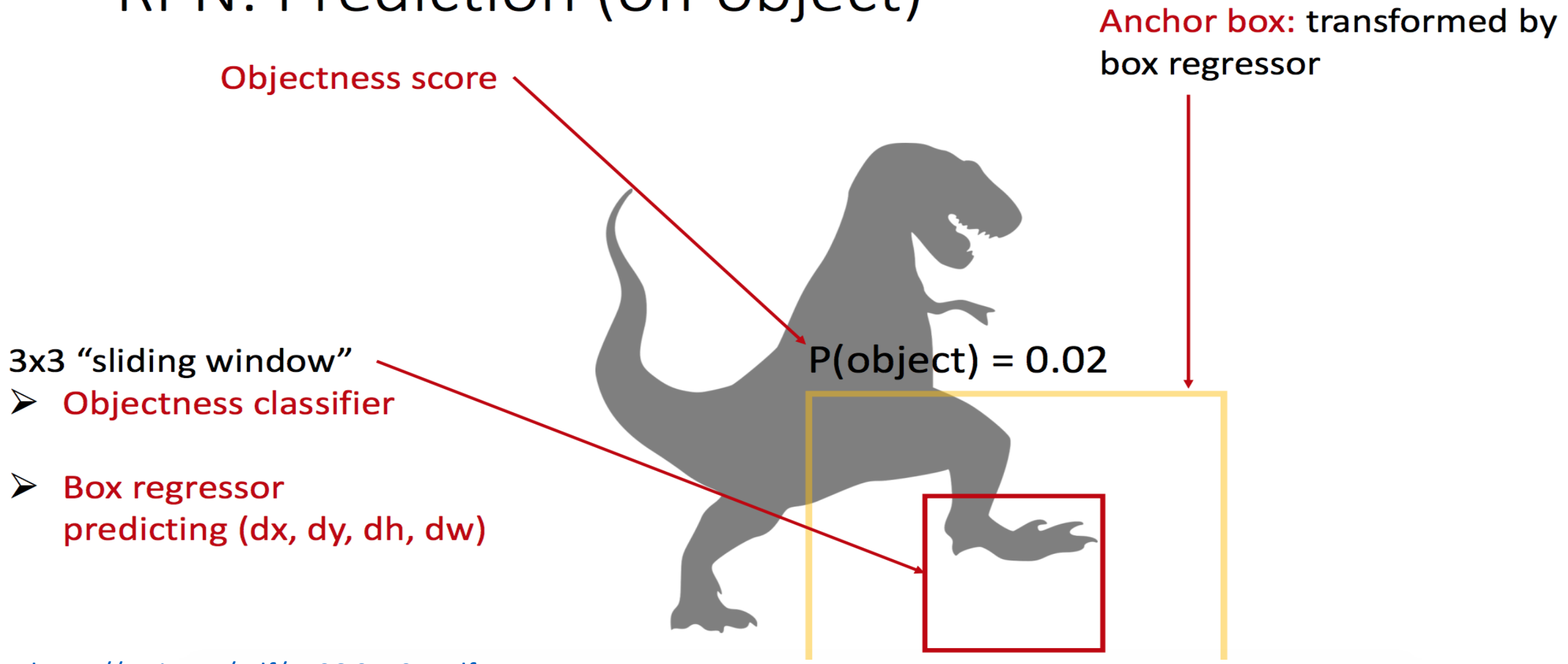
- Objectness classifier
- Box regressor predicting (dx, dy, dh, dw)

https://arxiv.org/pdf/1506.01497.pdf

Slides by Ross Girshick

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

RPN: Prediction (off object)

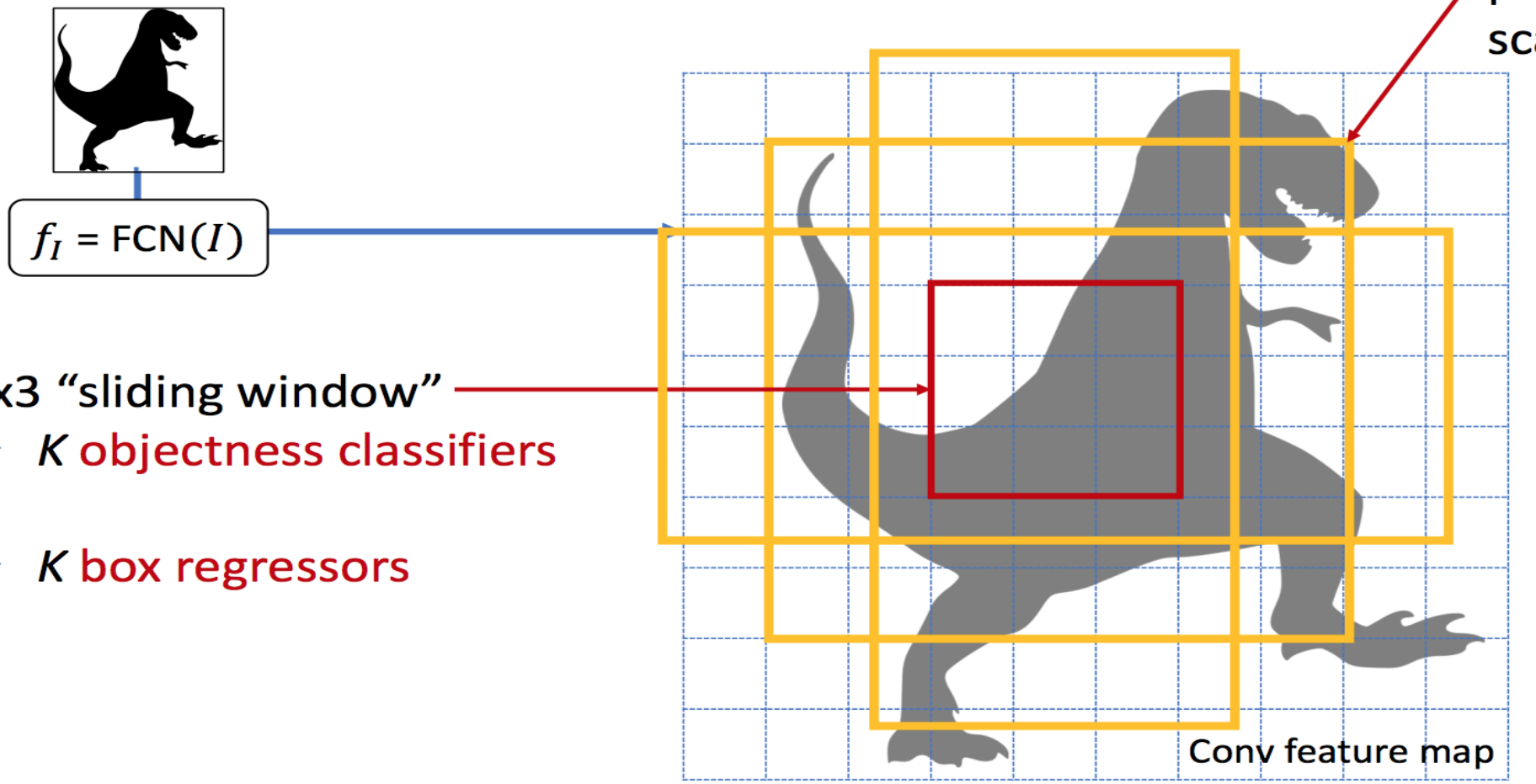


https://arxiv.org/pdf/1506.01497.pdf

Slides by Ross Girshick

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

RPN: Multiple Anchors

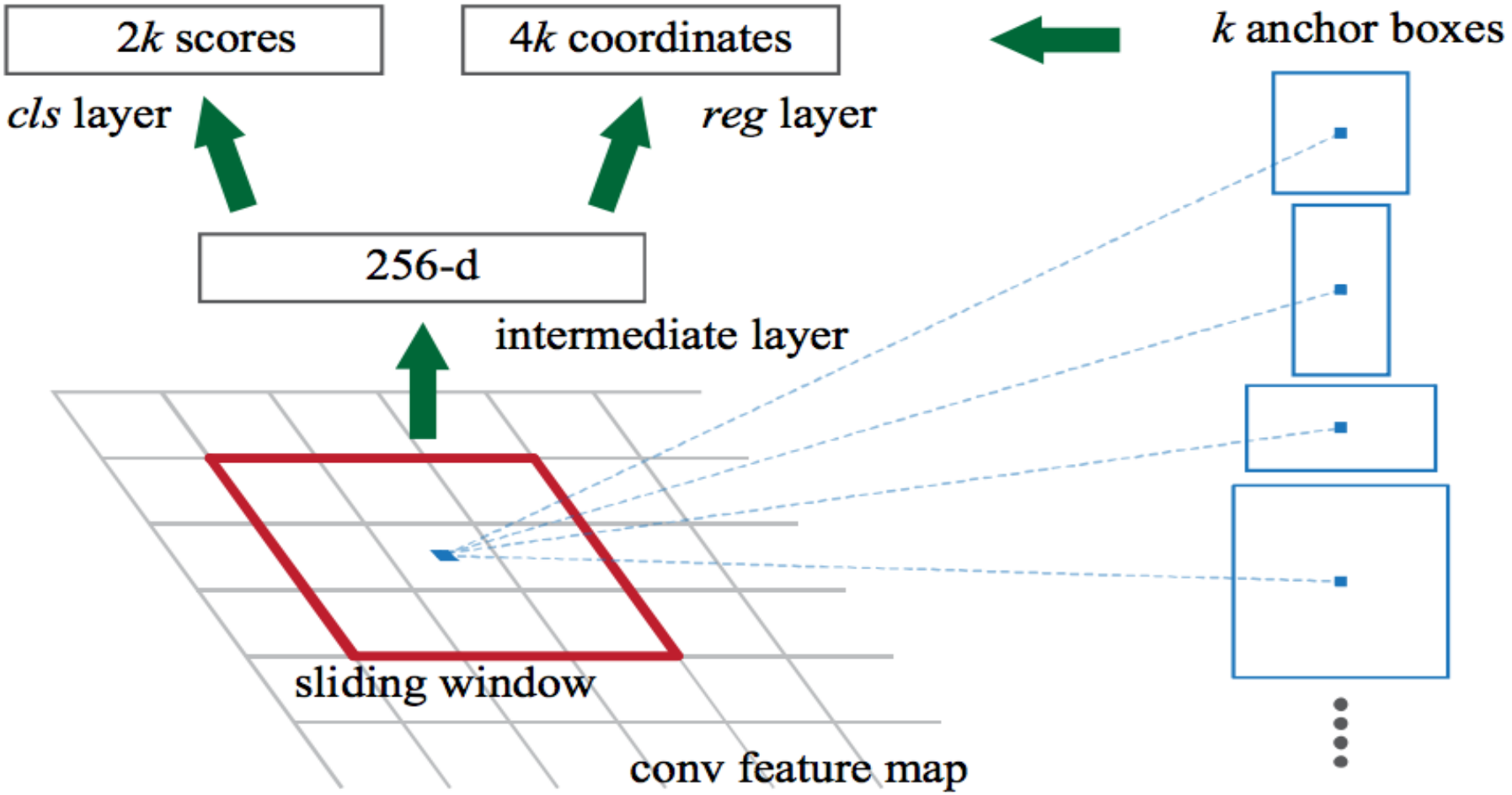


https://arxiv.org/pdf/1506.01497.pdf

Slides by Ross Girshick

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

Region proposal network



https://arxiv.org/pdf/1506.01497.pdf

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

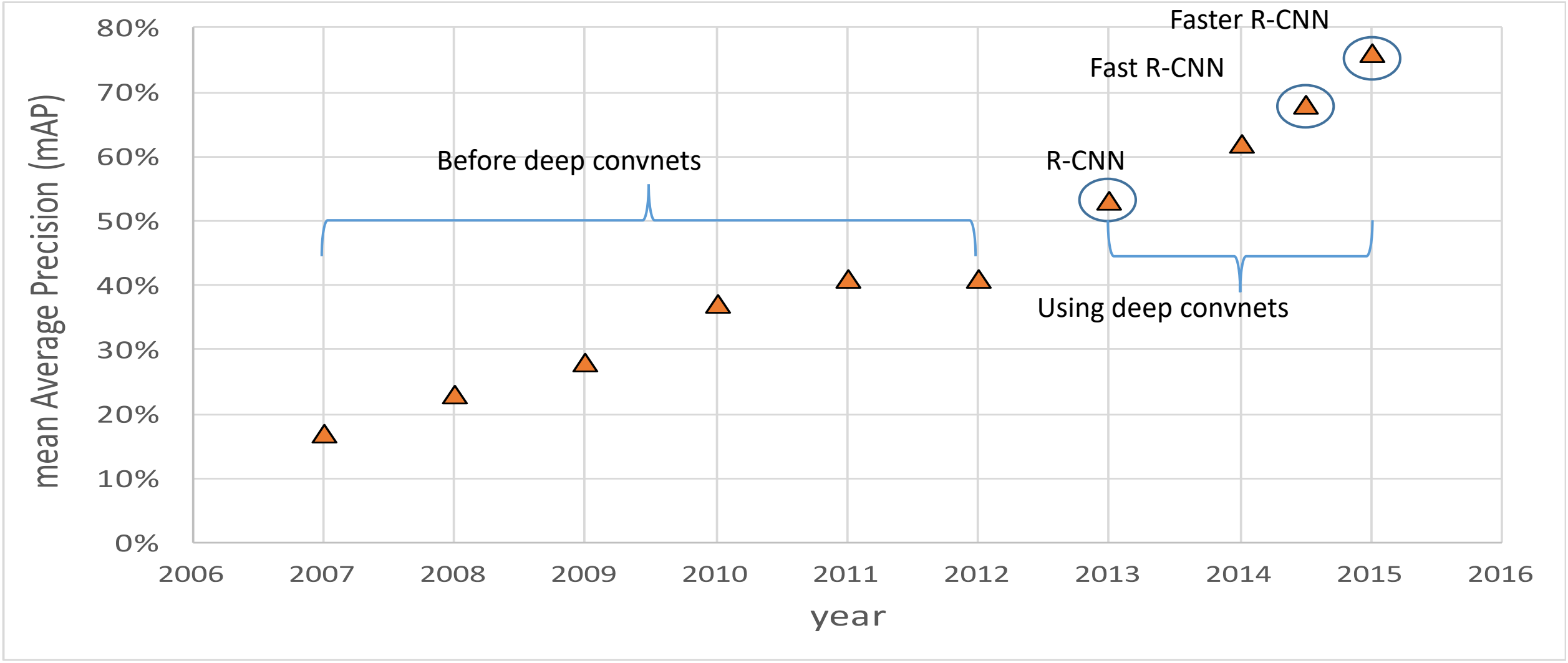
Faster R-CNN results

system	time	07 data	07+12 data
R-CNN	~50s	66.0	-
Fast R-CNN	~2s	66.9	70.0
Faster R-CNN	198ms	**69.9**	**73.2**

detection mAP on PASCAL VOC 2007, with VGG-16 pre-trained on ImageNet

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

https://arxiv.org/pdf/1506.01497.pdf

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 images and videos

https://arxiv.org/pdf/1506.01497.pdf

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

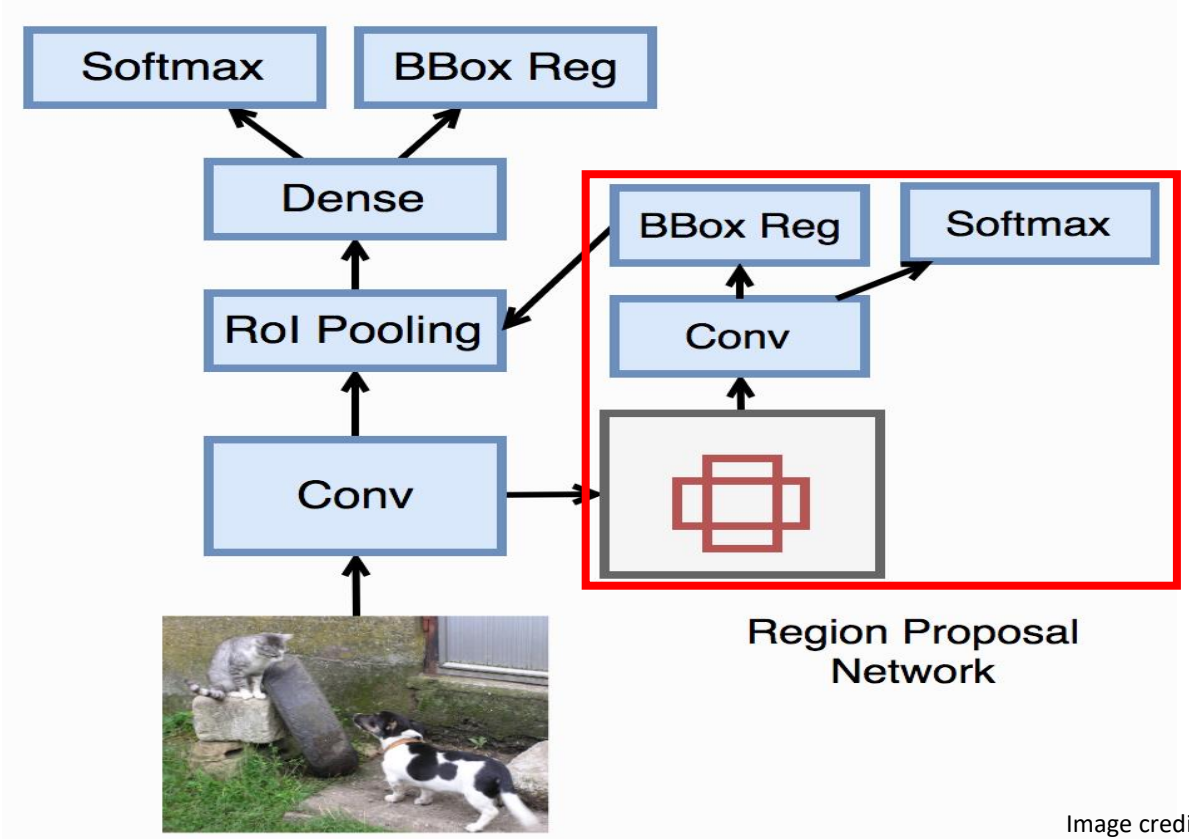
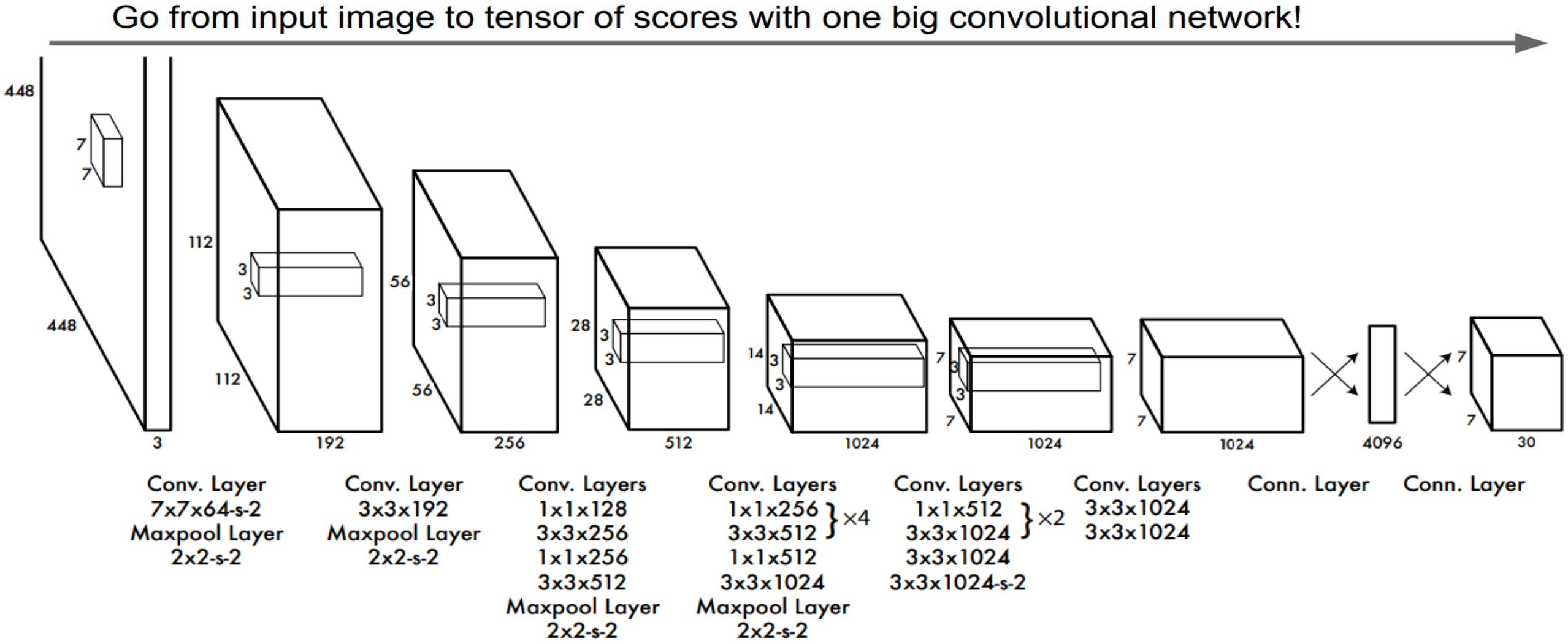


Image credit:
http://zh.gluon.ai/chapter_computer-vision/object-detection.html

You Only Look Once (YOLO)

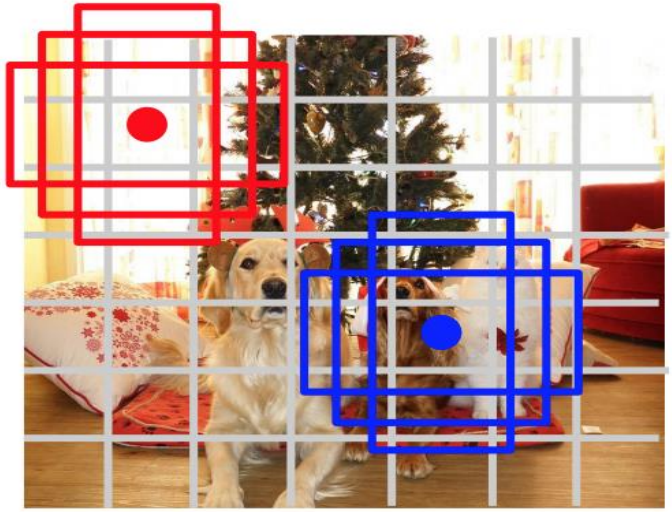


Redmon et al. You only look once: Unified, real-time object detection. CVPR 2016.

You Only Look Once (YOLO)

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

Input image
3 x H x W



Divide image into grid
7 x 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 x 7 x (5 * B + C)

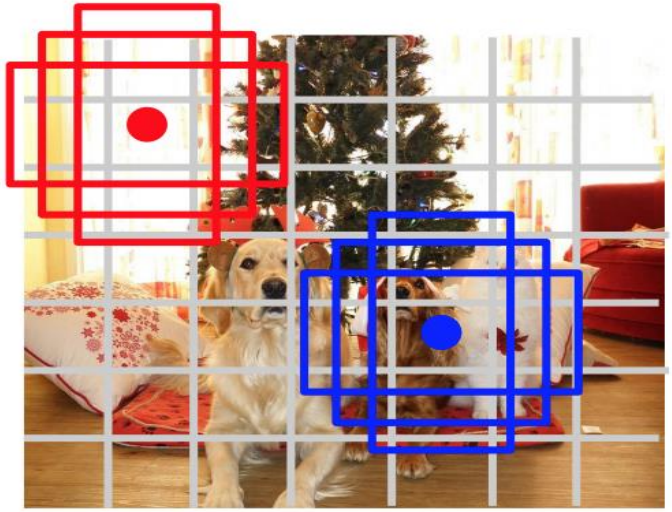
Redmon et al. You only look once: Unified, real-time object detection. cvpr 2016.

Slides by Justin Johnson

You Only Look Once (YOLO)

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

Input image
3 x H x W



Divide image into grid
7 x 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 x 7 x (5 * B + C)

B = 2 in experiments

C = 20 in PASCAL VOC

Final prediction=7×7×30 tensor.

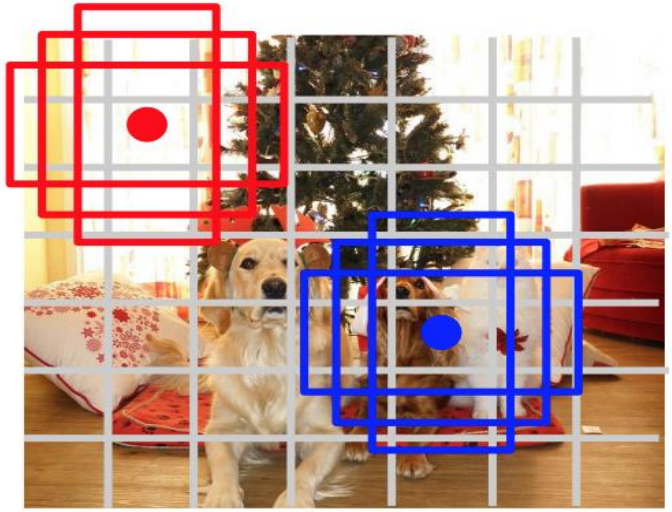
Redmon et al. You only look once: Unified, real-time object detection. cvpr 2016.

Slides by Justin Johnson

You Only Look Once (YOLO)

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

Input image
3 x H x W



Divide image into grid
7 x 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 x 7 x (5 * B + C)

- Very efficient but lower accuracy

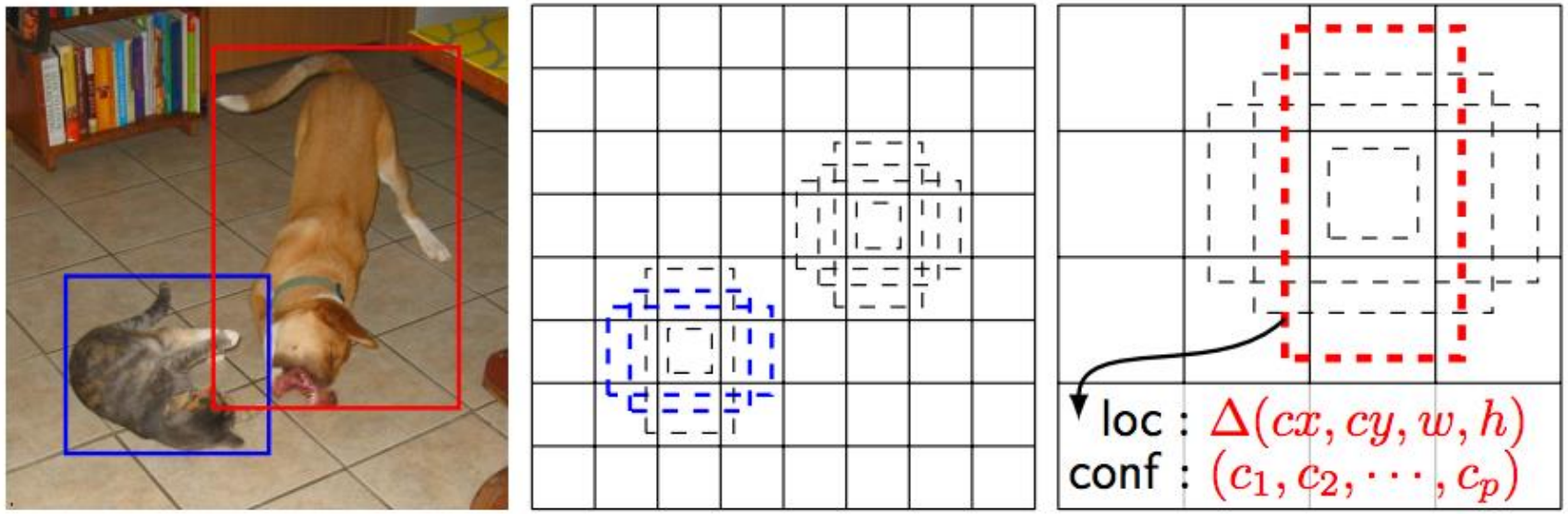
Redmon et al. You only look once: Unified, real-time object detection. cvpr 2016.

Slides by Justin Johnson

You Only Look Once (YOLO)

- Each grid cell predicts only two boxes and can only have one class – this limits the number of nearby objects that can be predicted
- Localization accuracy suffers compared to Fast(er) R-CNN due to coarser features, errors on small boxes
- 7x speedup over Faster R-CNN (45-155 FPS vs. 7-18 FPS)

SSD: Single Shot MultiBox Detector

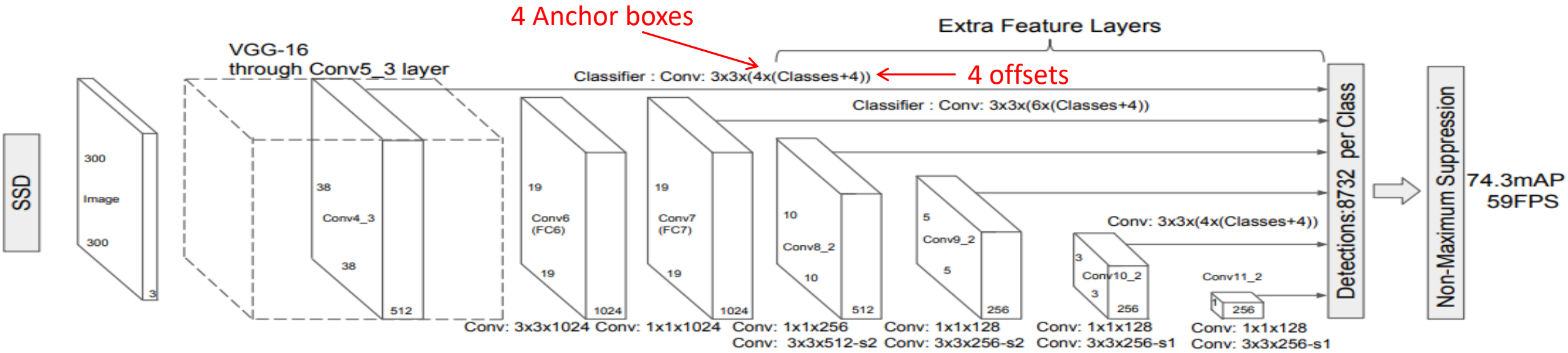


(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

- Similarly to YOLO, predict bounding boxes directly from conv maps
- Unlike YOLO, do not use FC layers and predict different size boxes from conv maps at different resolutions
- Similarly to RPN, use anchors

W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. Berg, SSD: Single Shot MultiBox Detector, ECCV 2016.

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.

Liu et al. SSD: Single Shot MultiBox Detector, ECCV 2016.

SSD: Single Shot MultiBox Detector

Source: http://cv-tricks.com/object-detection/faster-r-cnn-yolo-ssd/

One-stage detection

- What could be the problems?

One-stage detection

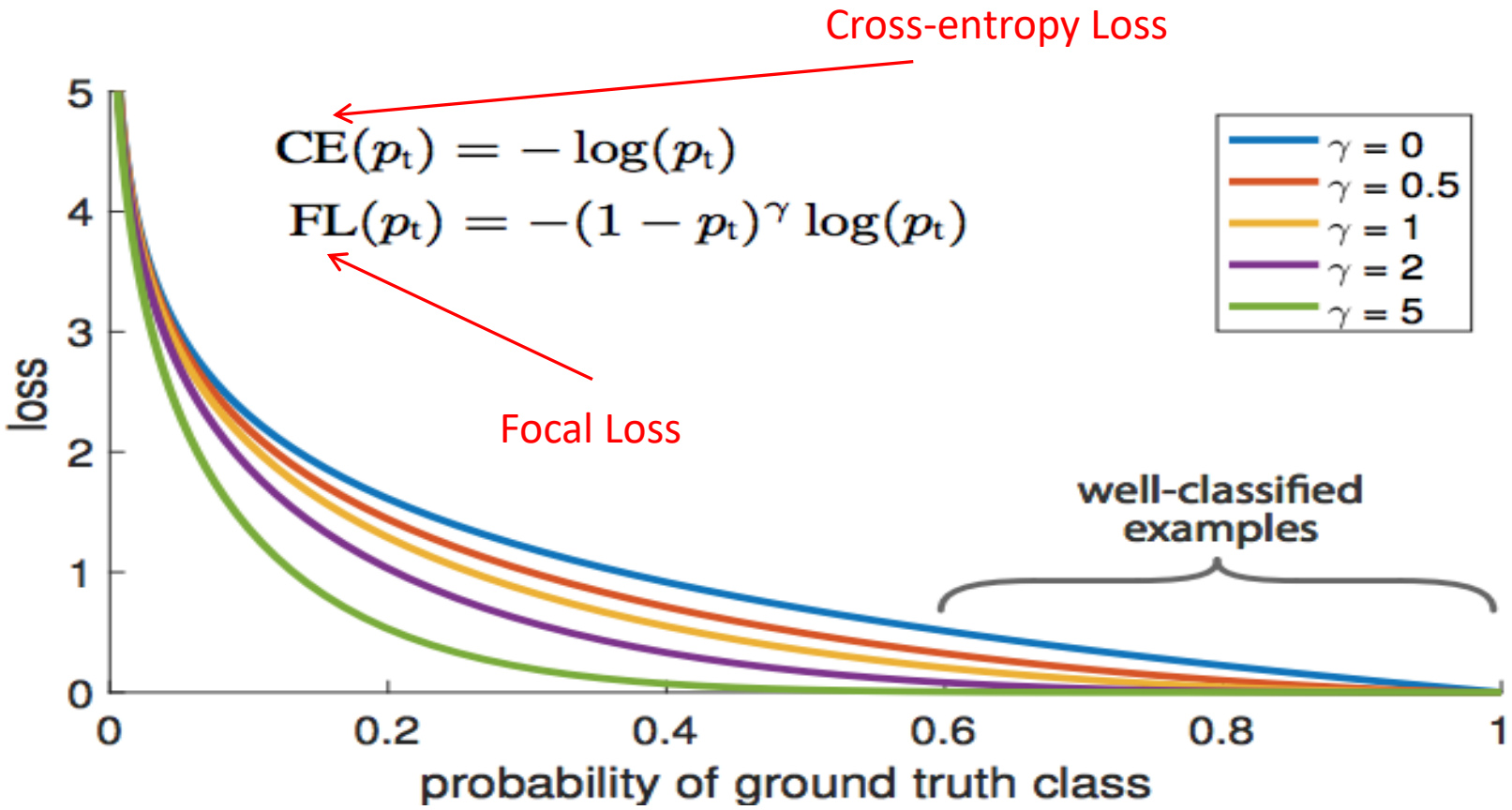
- What could be the problems?
  - The extreme foreground-background class imbalance
    -> we have a lot more negative examples.

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.

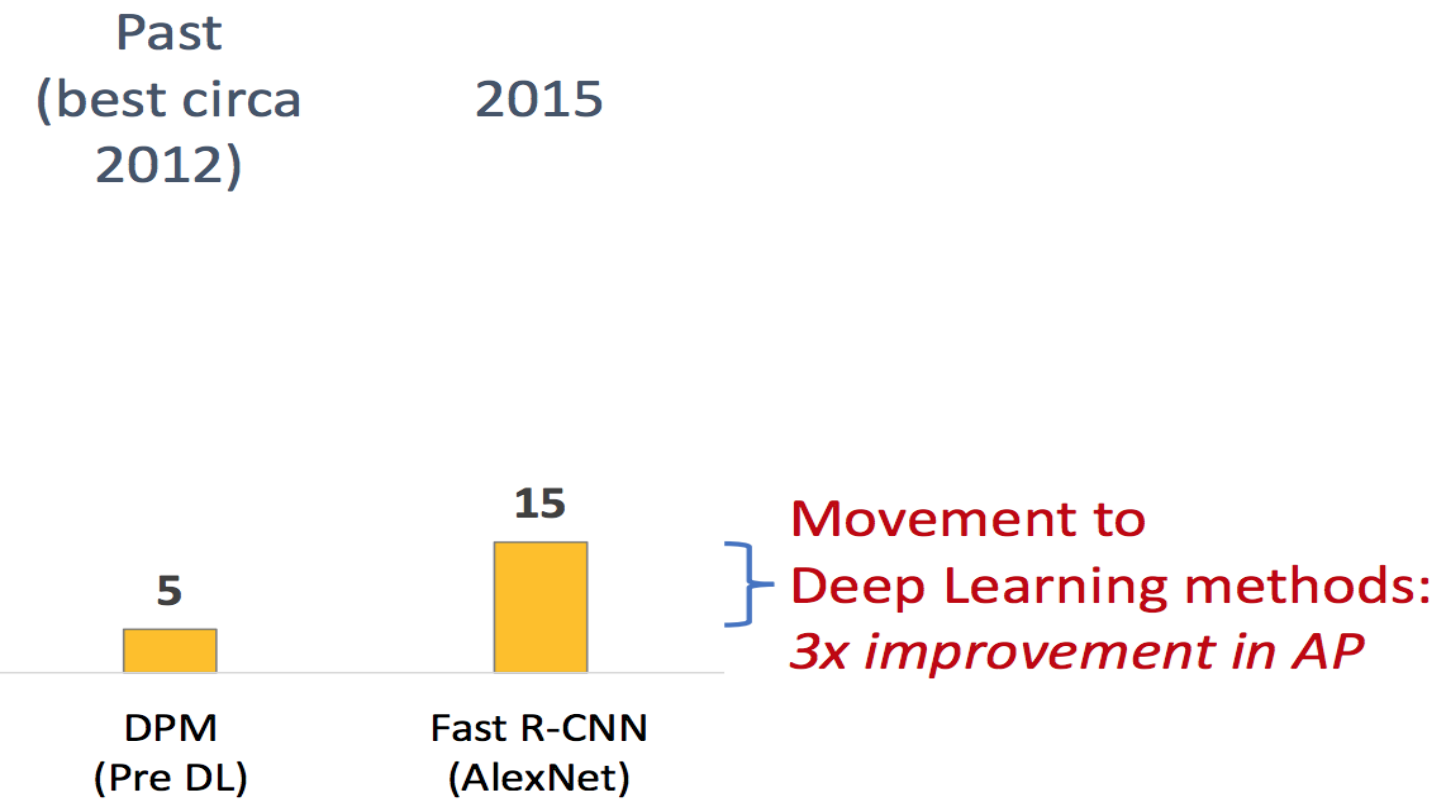
RetinaNet (Lin et al. ICCV 2017)

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

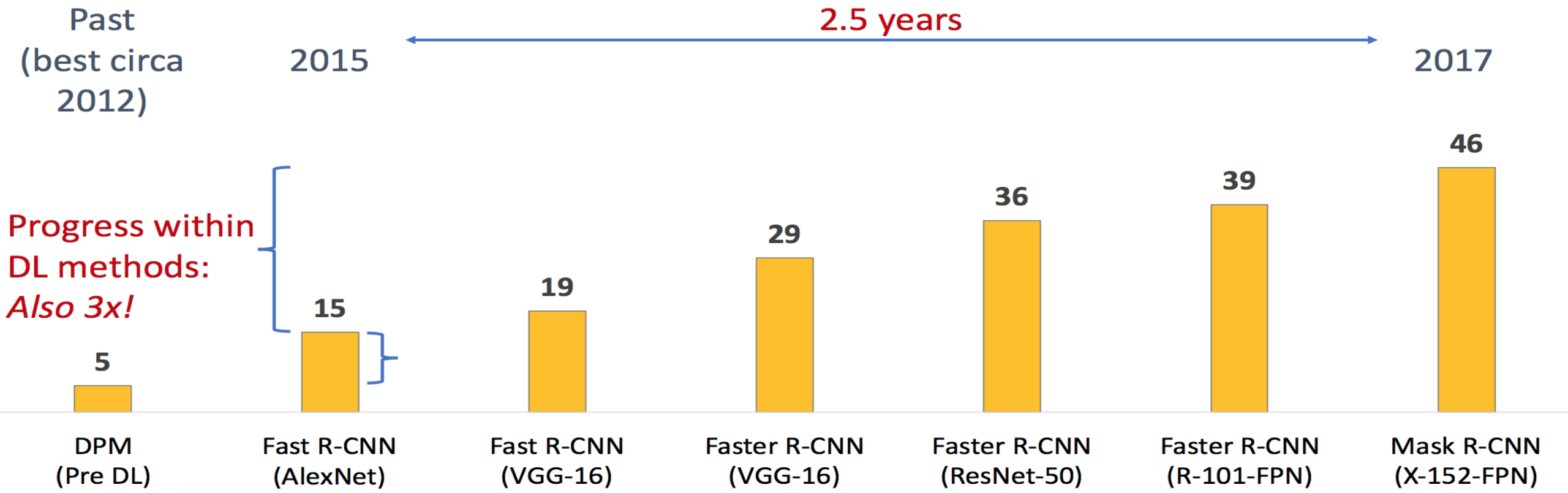


Lin et al. Focal loss for dense object detection. *ICCV* 2017. https://arxiv.org/pdf/1708.02002.pdf

COCO Object Detection Average Precision (%)



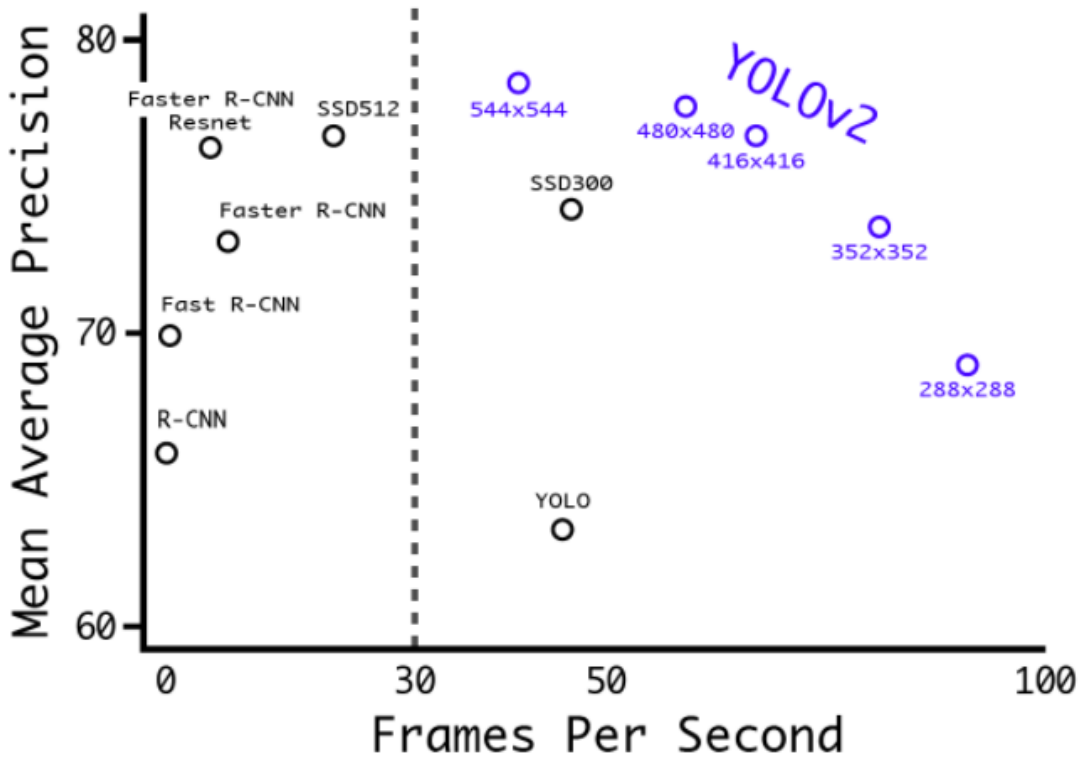
COCO Object Detection Average Precision (%)



YOLO v2

- Remove FC layer, do convolutional prediction with anchor boxes instead
- Increase resolution of input images and conv feature maps
- Improve accuracy using batch normalization and other tricks

VOC 2007 results



YouTube demo

J. Redmon and A. Farhadi, YOLO9000: Better, Faster, Stronger, CVPR 2017

YOLO v3

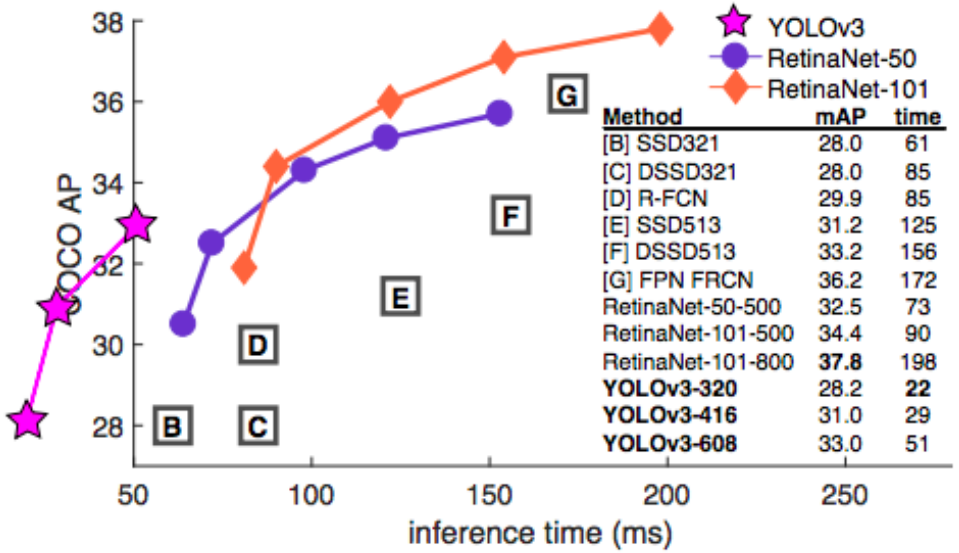
YOLOv3: An Incremental Improvement

Joseph Redmon, Ali Farhadi
University of Washington

Abstract

We present some updates to YOLO! We made a bunch of little design changes to make it better. We also trained this new network that's pretty swell. It's a little bigger than last time but more accurate. It's still fast though, don't worry. At 320 × 320 YOLOv3 runs in 22 ms at 28.2 mAP, as accurate as SSD but three times faster. When we look at the old .5 IOU mAP detection metric YOLOv3 is quite good. It achieves 57.9 AP_{50} in 51 ms on a Titan X, compared to 57.5 AP_{50} in 198 ms by RetinaNet, similar performance but 3.8× faster. As always, all the code is online at https://pjreddie.com/yolo/.

1. Introduction



Method	mAP	time
[B] SSD321	28.0	61
[C] DSSD321	28.0	85
[D] R-FCN	29.9	85
[E] SSD513	31.2	125
[F] DSSD513	33.2	156
[G] FPN FRCN	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	**37.8**	198
YOLOv3-320	28.2	**22**
YOLOv3-416	31.0	29
YOLOv3-608	33.0	51

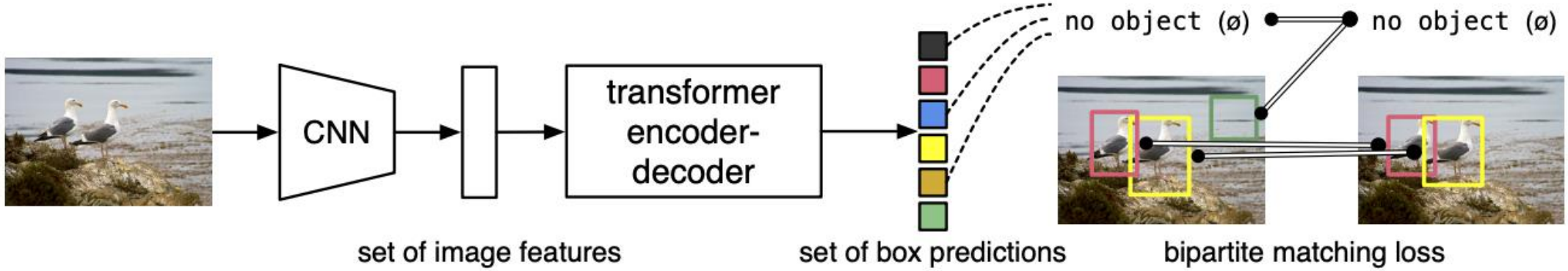
Figure 1. We adapt this figure from the Focal Loss paper [9]. YOLOv3 runs significantly faster than other detection methods with comparable performance. Times from either an M40 or Titan X, they are basically the same GPU.

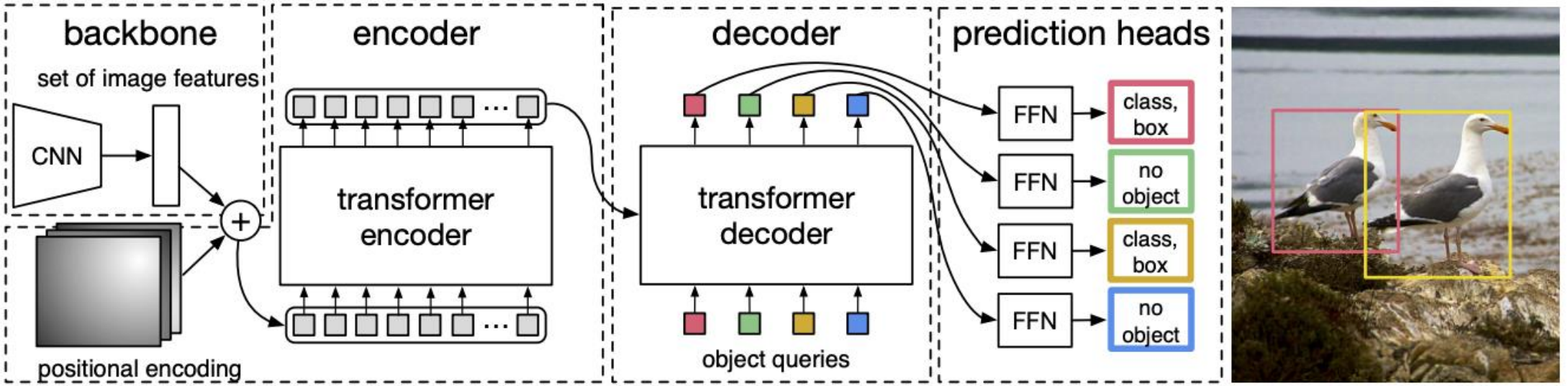
https://pjreddie.com/media/files/papers/YOLOv3.pdf

Summary so far

- R-CNN: region proposals + CNN on cropped, resampled regions
- Fast R-CNN: region proposals + RoI pooling on top of a conv feature map
- Faster R-CNN: RPN + RoI pooling
- Next generation of detectors: YOLO, SSD, RetinaNet
  - Direct prediction of BB offsets, class scores on top of conv feature maps
  - Get better context by combining feature maps at multiple resolutions
- Most recent developments: architectures borrowed from dense prediction, transformers

Detection Transformer (DETR)





N. Carion et al. End-to-end object detection with transformers. arXiv 2020

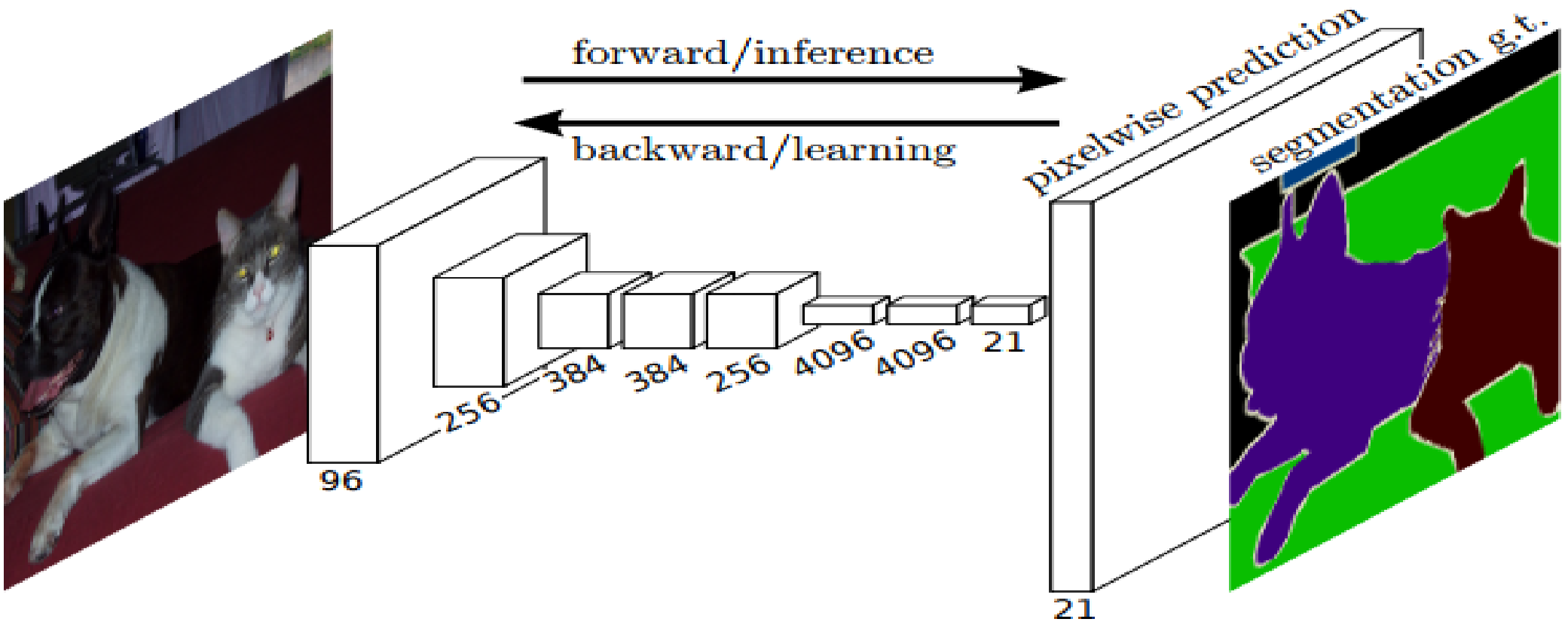
Acknowledgement

Thanks to the following courses and corresponding researchers for making their teaching/research material online

- Deep Learning, Stanford University
- Introduction to Deep Learning, University of Illinois at Urbana-Champaign
- Introduction to Deep Learning, Carnegie Mellon University
- Convolutional Neural Networks for Visual Recognition, Stanford University
- Natural Language Processing with Deep Learning, Stanford University
- And Many More ……

Next class

- Image Segmentation



https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf