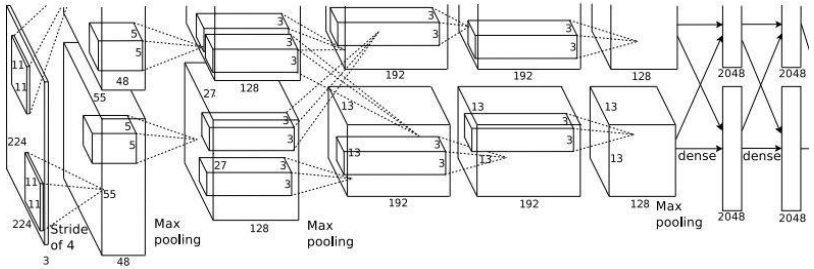
AER 1515 – Perception for Robotics

Lecture 10– Visual Classification, Segmentation and Object Detection

Prof. Steven Waslander

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Canonical Task Image Classification



Vector:
4096

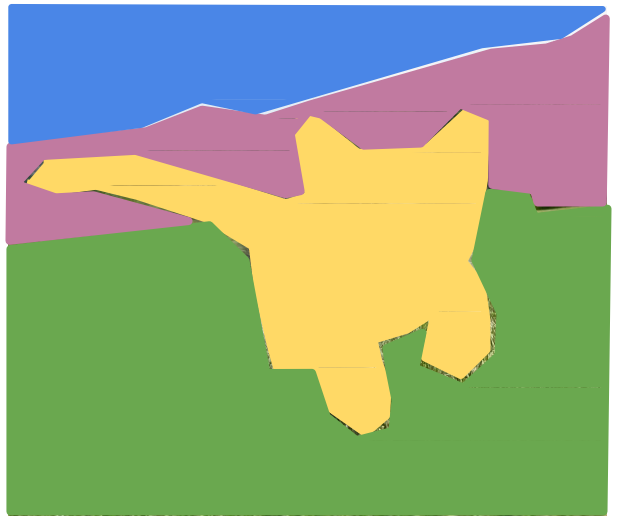
Fully-Connected:
4096 to 1000

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Slides from Fei-Fei Li's CS231n

Other Computer Vision Tasks

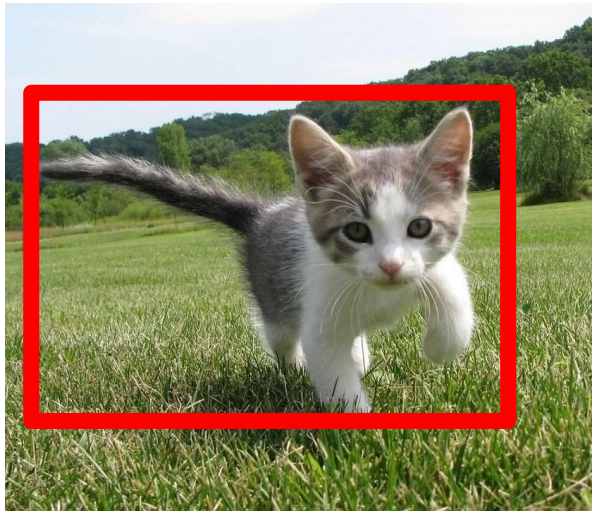
Semantic Segmentation



GRASS, CAT, TREE, SKY

No objects, just pixels

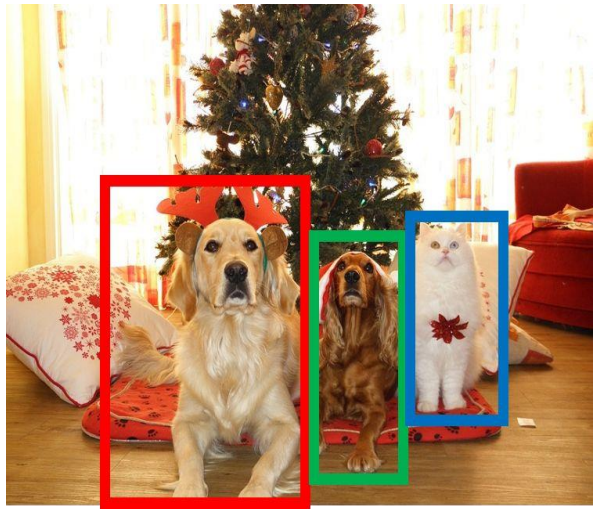
Classification + Localization



CAT

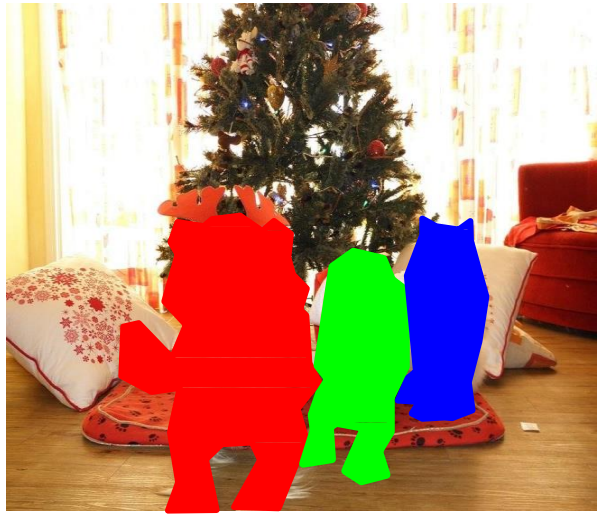
Single Object

Object Detection



DOG, DOG, CAT

Instance Segmentation



DOG, DOG, CAT

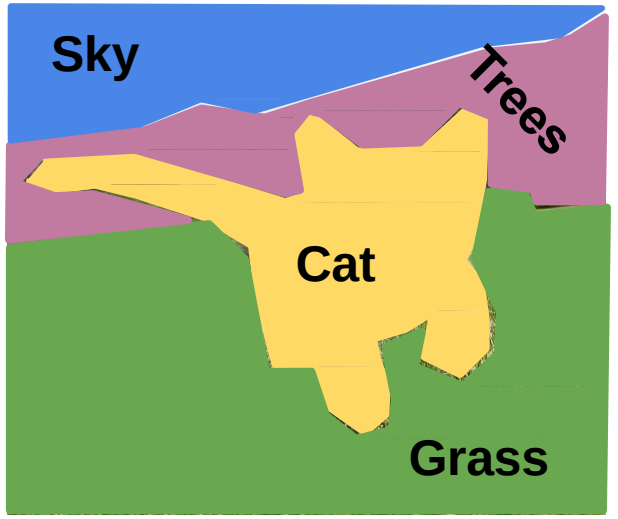
Multiple Object

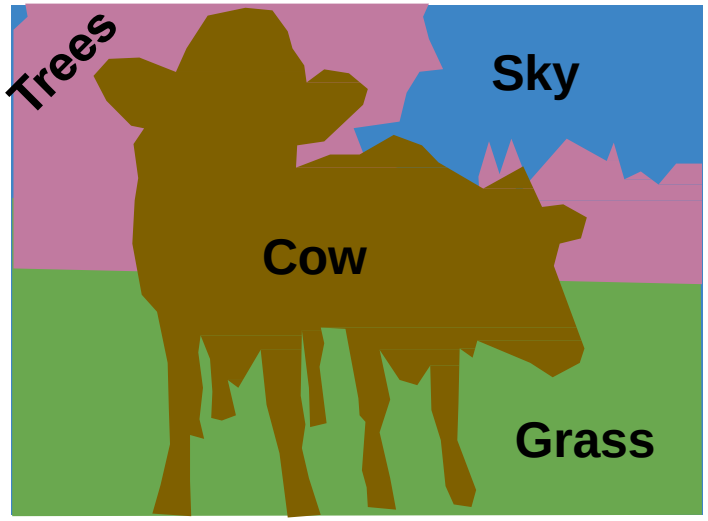
Semantic Segmentation

- Don’t differentiate instances, only care about pixel labels

- Provide class labels to inform decision making for robotics
  - Which part to grab
  - Object extent
  - Motion models

- Can be used as a pre-processing step for many subsequent perception tasks



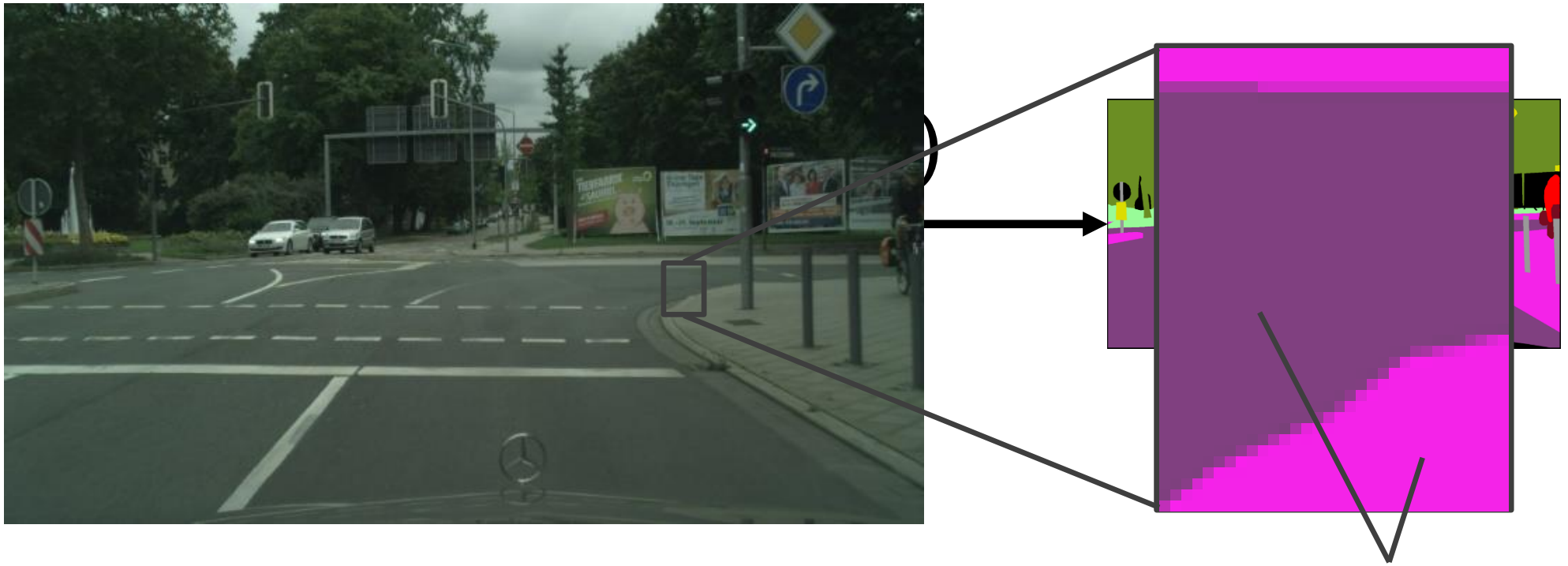


The Semantic Segmentation Problem – Autonomous Driving Example

- Road
- Sidewalk
- Pole
- Traffic Light
- Traffic Signs
- Vegetation
- Terrain
- Sky
- Car
- Background

Slides from Coursera Course

Mathematical Problem Formulation



$$f(x;\theta) = [S_{class_1}, \dots, S_{class_k}]$$

$S_{road} = 0.92$

$S_{sidewalk} = 0.02$

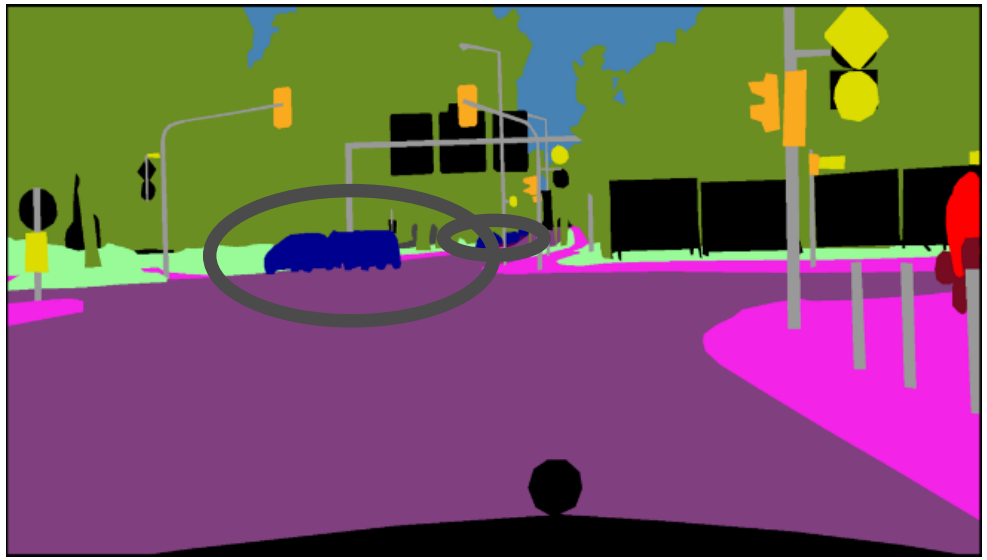
.

.

$S_{Background} = 0.03$

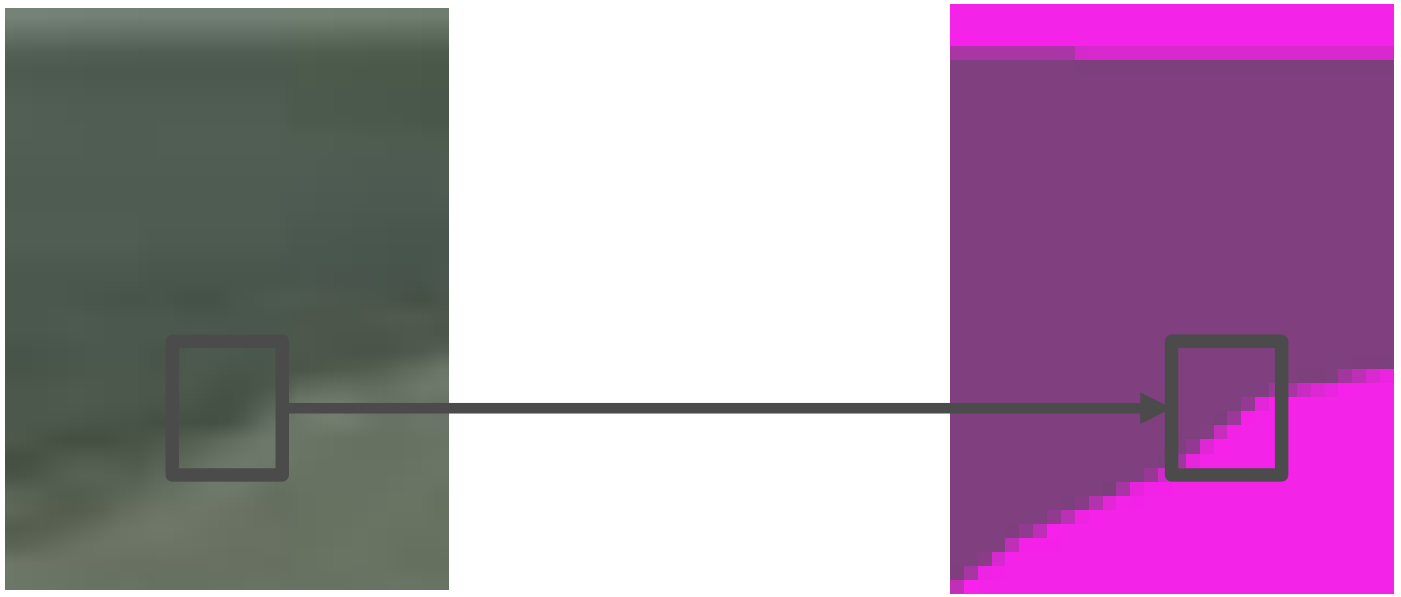
Semantic Segmentation is Not Trivial !

- Occlusion, truncation, scale, and illumination changes

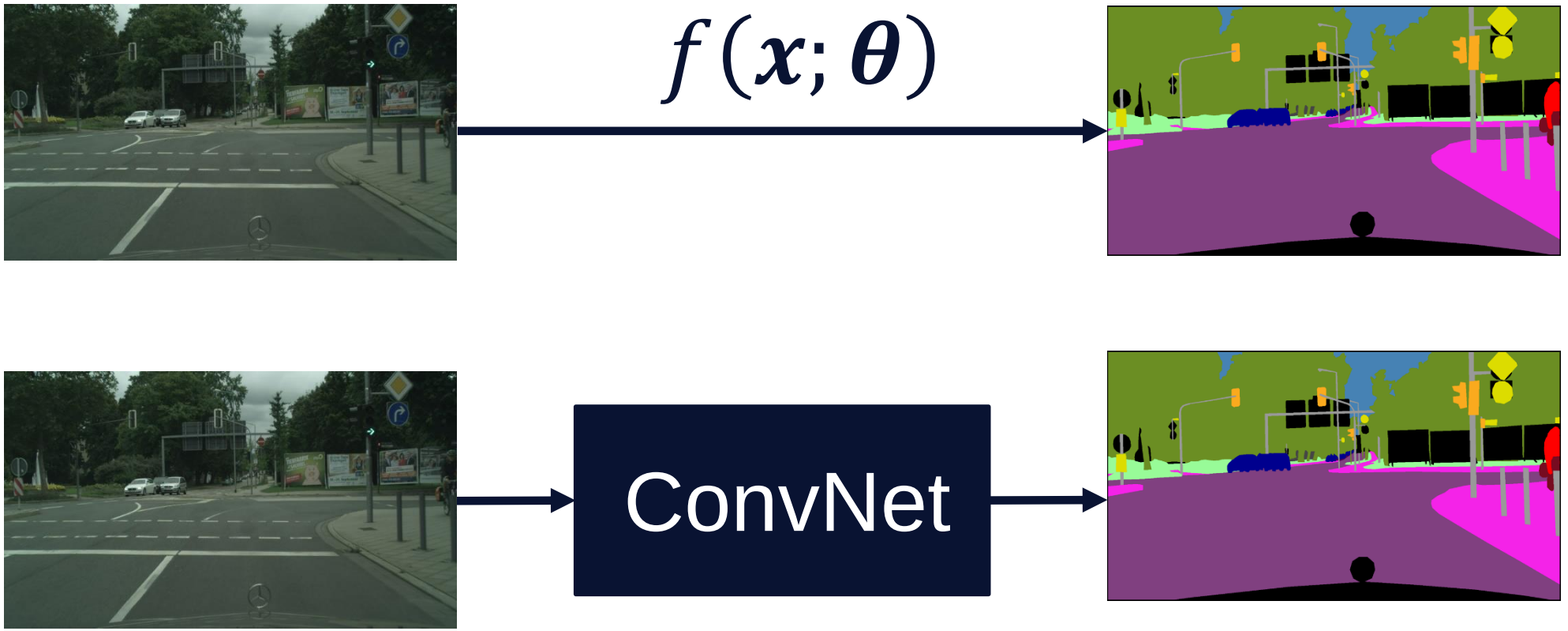


Semantic Segmentation is Not Trivial !

- Occlusion, truncation, scale, and illumination changes
- Smooth boundaries



ConvNets For Semantic Segmentation



Evaluation Metrics

- **True Positive (TP):** The number of correctly classified pixels belonging to class X
- **False Positive (FP):** The number of pixels that **do not belong** to class X in ground truth but **are classified** that class by the algorithm
- **False Negative (FN):** The number of pixels that **do belong** to class X in ground truth, but **are not classified** as that class by the algorithm

$$IOU_{class} = \frac{TP}{TP + FP + FN}$$

Evaluation Metrics

Ground Truth

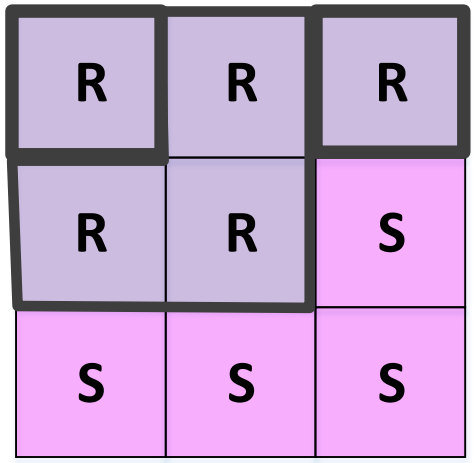
R	R	R
R	R	S
S	S	S

Prediction

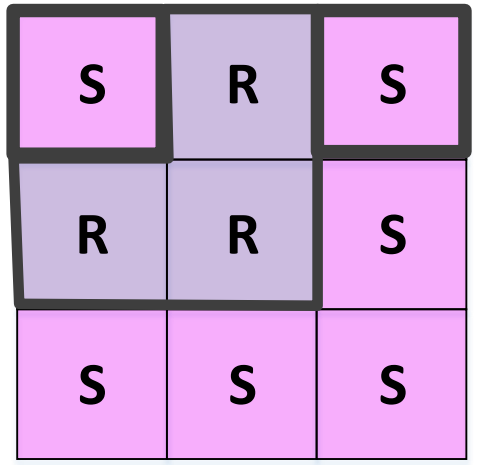
S	R	S
R	R	S
S	S	S

Evaluation Metrics

Ground Truth



Prediction



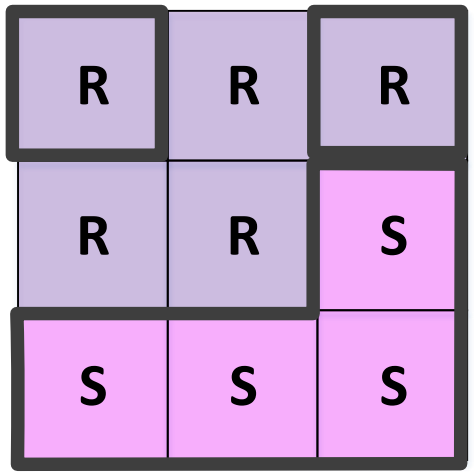
Class: Road

$$TP = 3$$
$$FP = 0$$
$$FN = 2$$

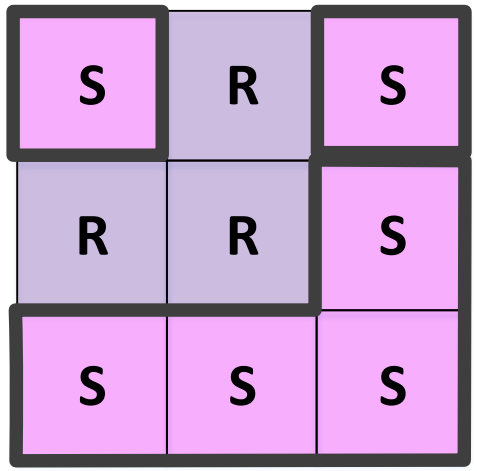
$$IOU_{Road} = \frac{3}{3+0+2} = \frac{3}{5}$$

Evaluation Metrics

Ground Truth



Prediction



Class: Sidewalk

$$\boldsymbol{TP} = \mathbf{4}$$
$$\boldsymbol{FP} = \mathbf{2}$$
$$\boldsymbol{FN} = \mathbf{0}$$

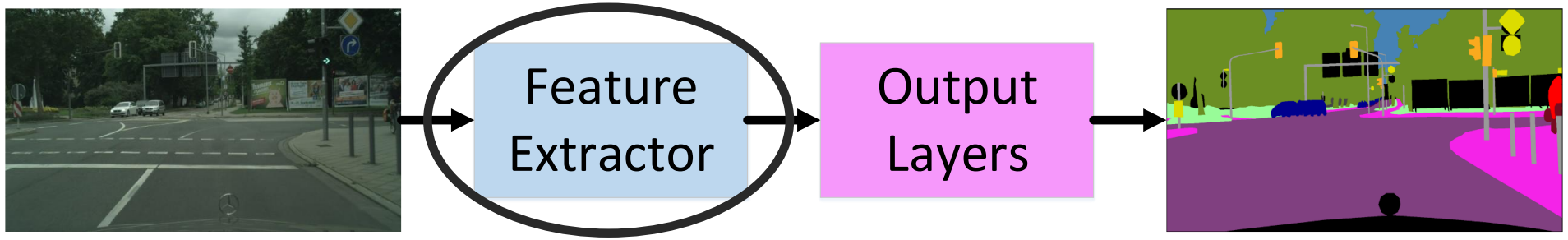
$$IOU_{Road} = \frac{4}{4+2+0} = \frac{4}{6}$$

Evaluation Metrics

- **Class IOU** over all the data is calculated by computing the sum of TP, FP, FN for all images first
- Averaging the class IOU is usually not a very good idea!

- CityScapes Segmentation Dataset

ConvNets For Semantic Segmentation



The Feature Extractor

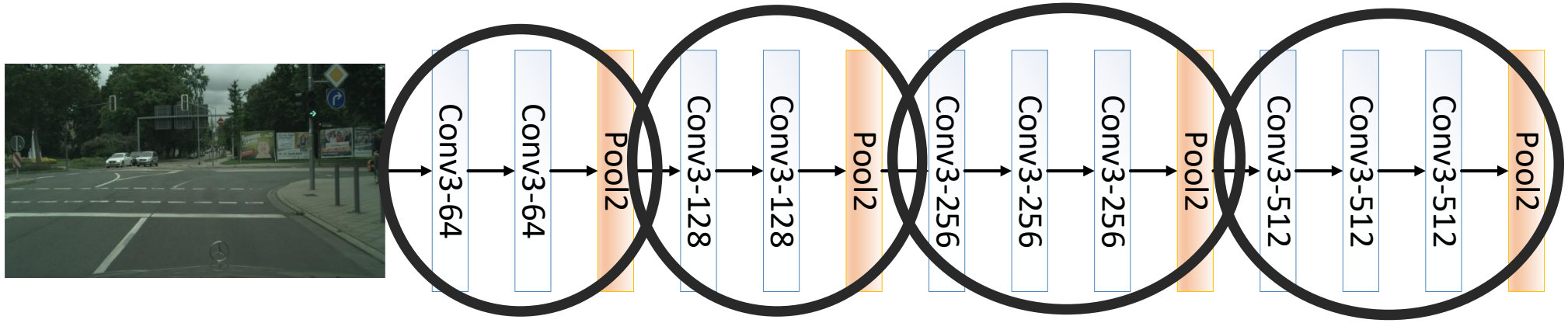
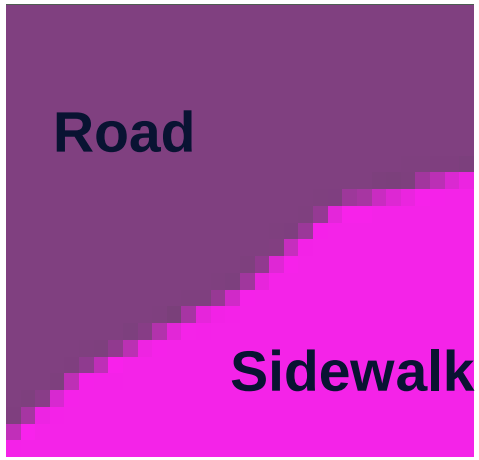


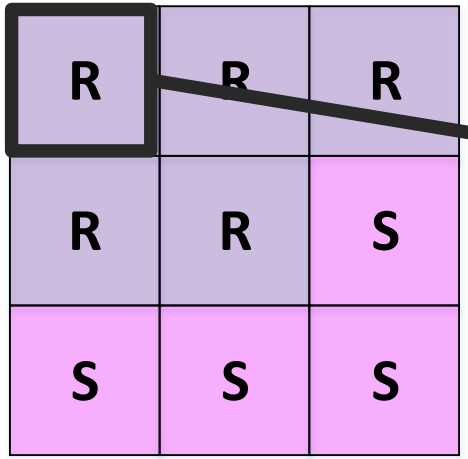
	Image	Conv1	Conv2	Conv3	Conv4
Width	M	M/2	M/4	M/8	M/16
Height	N	N/2	N/4	N/8	N/16
Depth	3	64	128	256	512

Output Representation

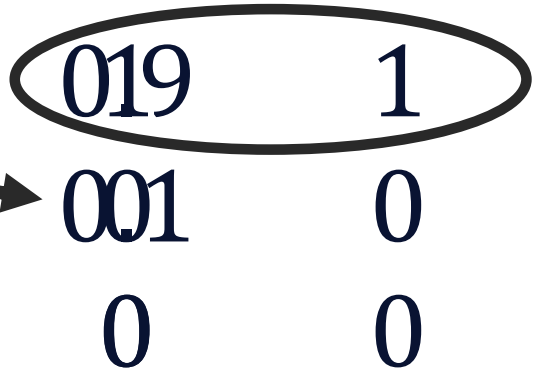
Ground Truth



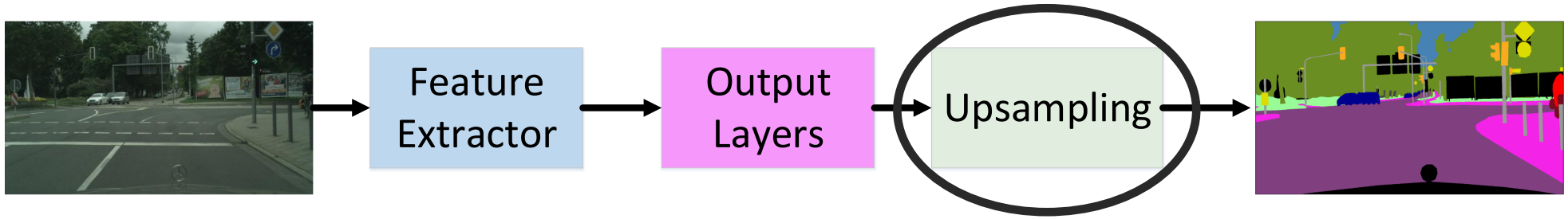
Class



One Hot



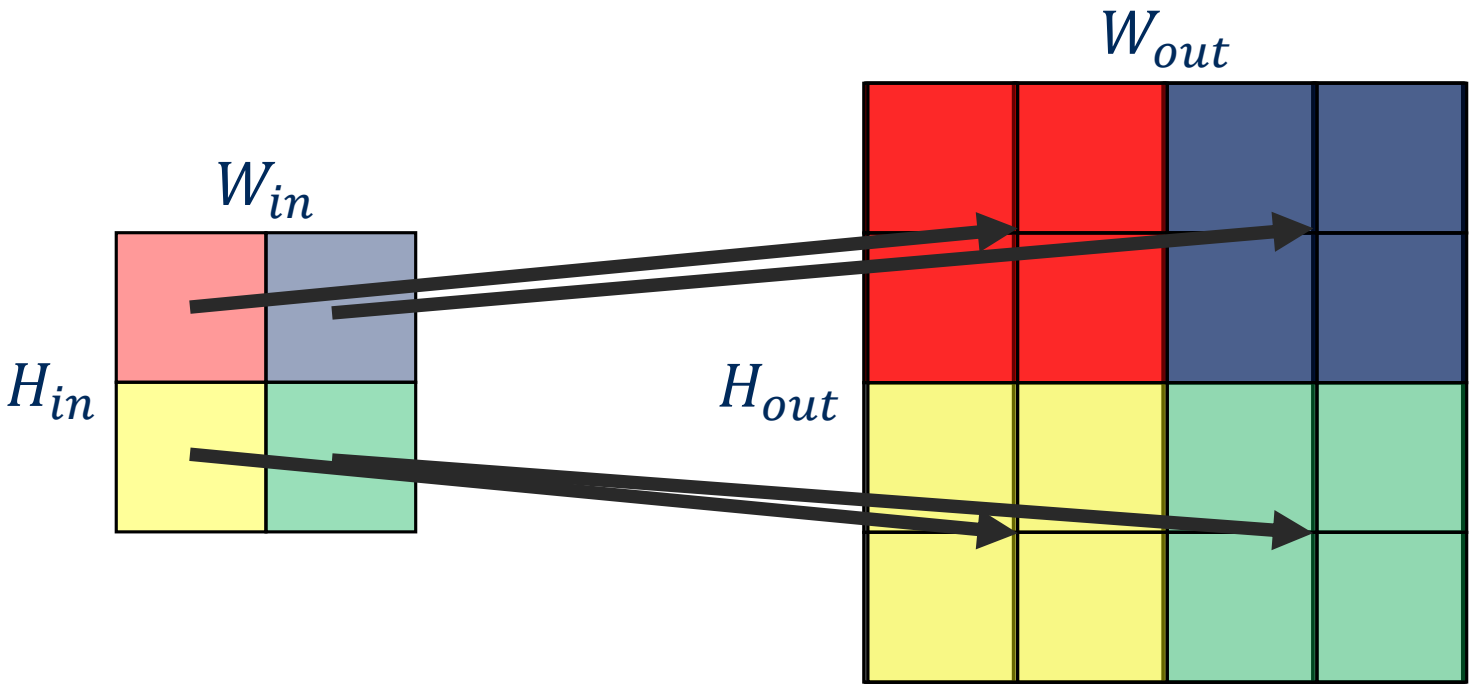
Upsampling the Output



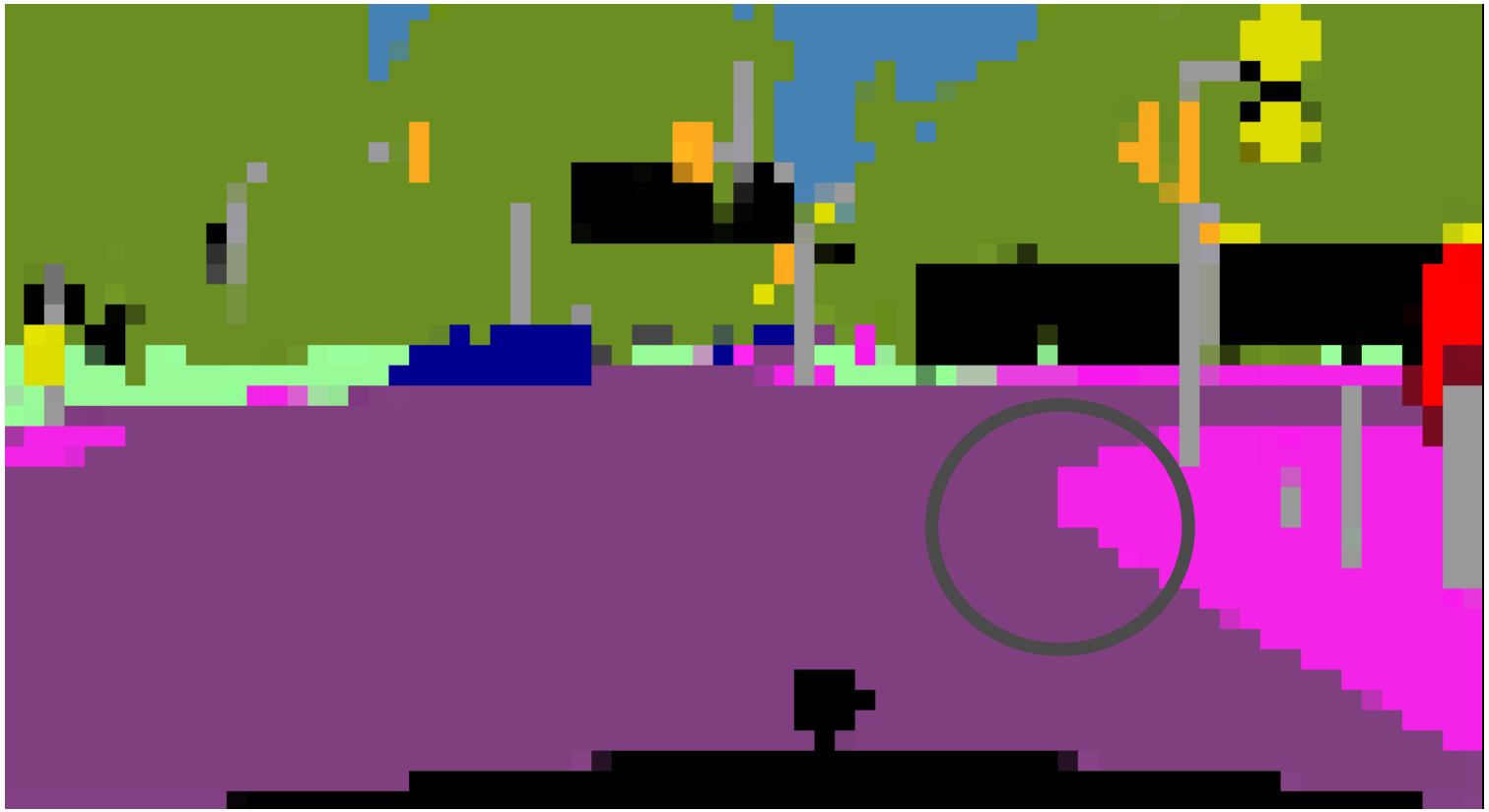
Upsampling Layer

- Upsampling Multiplier $\boldsymbol{S}$

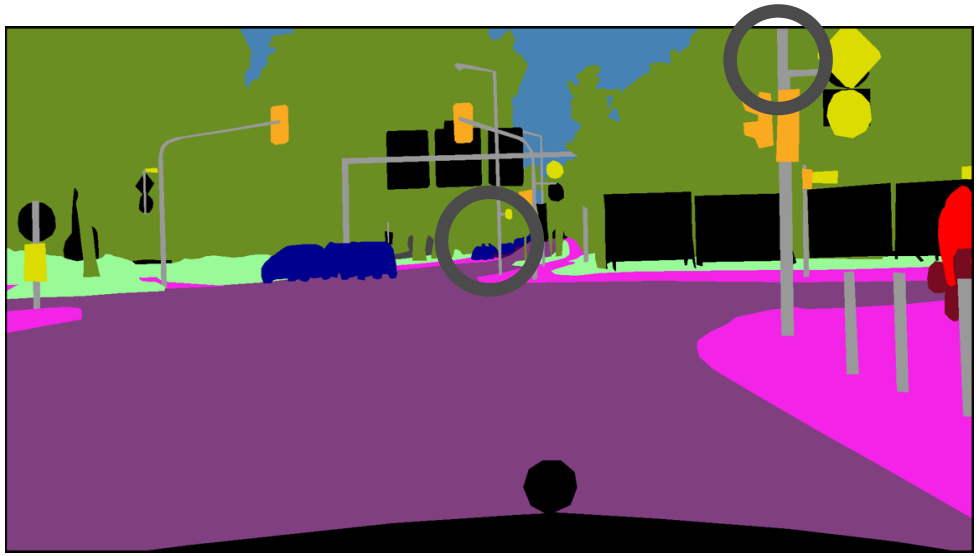
- $\boldsymbol{W_{out} = S \times W_{in}}$
- $\boldsymbol{H_{out} = S \times H_{in}}$
- $\boldsymbol{D_{out} = D_{in}}$

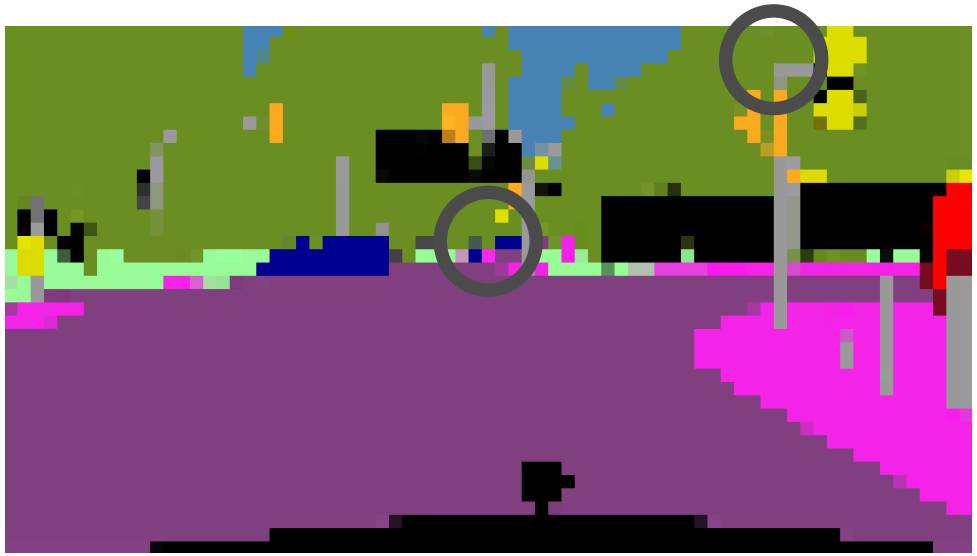


Upsampling The Output

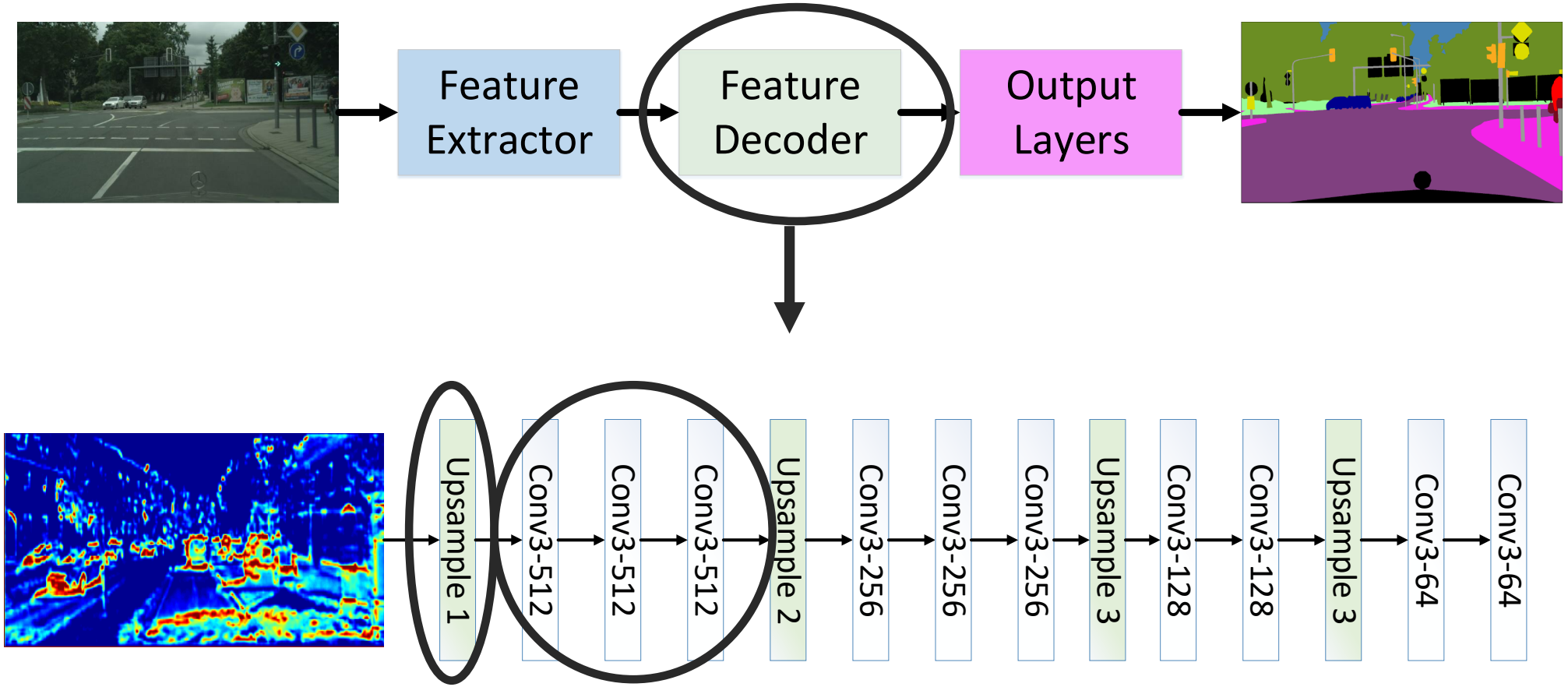


Upsampling The Output

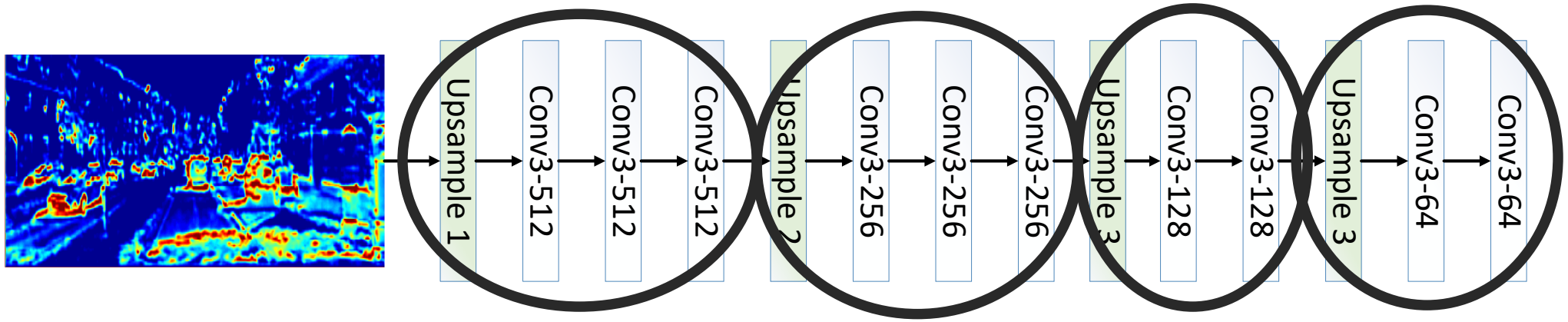




Learning Same Resolution Feature Maps



The Feature Decoder



	Feature Map	Deconv1	Deconv2	Deconv3	Deconv4
Width	M/16	M/8	M/4	M/2	M
Height	N/16	N/8	N/4	N/2	N
Depth	512	512	256	128	64

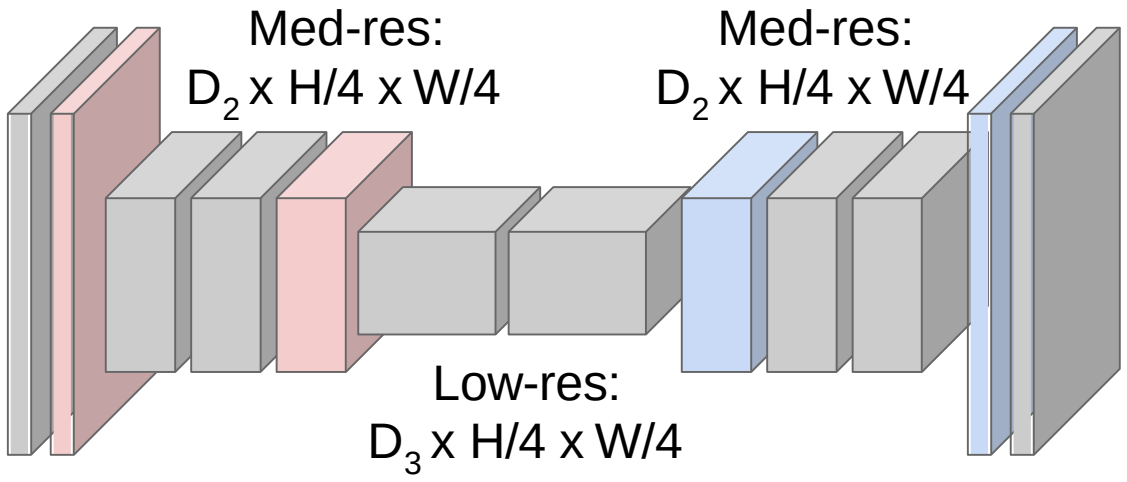
Semantic Segmentation Idea: Fully Convolutional

Downsampling:
Pooling, strided convolution

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

Upsampling:
Unpooling or strided transpose convolution

Input:
3 x H x W



High-res:
D_1 x H/2 x W/2

High-res:
D_1 x H/2 x W/2

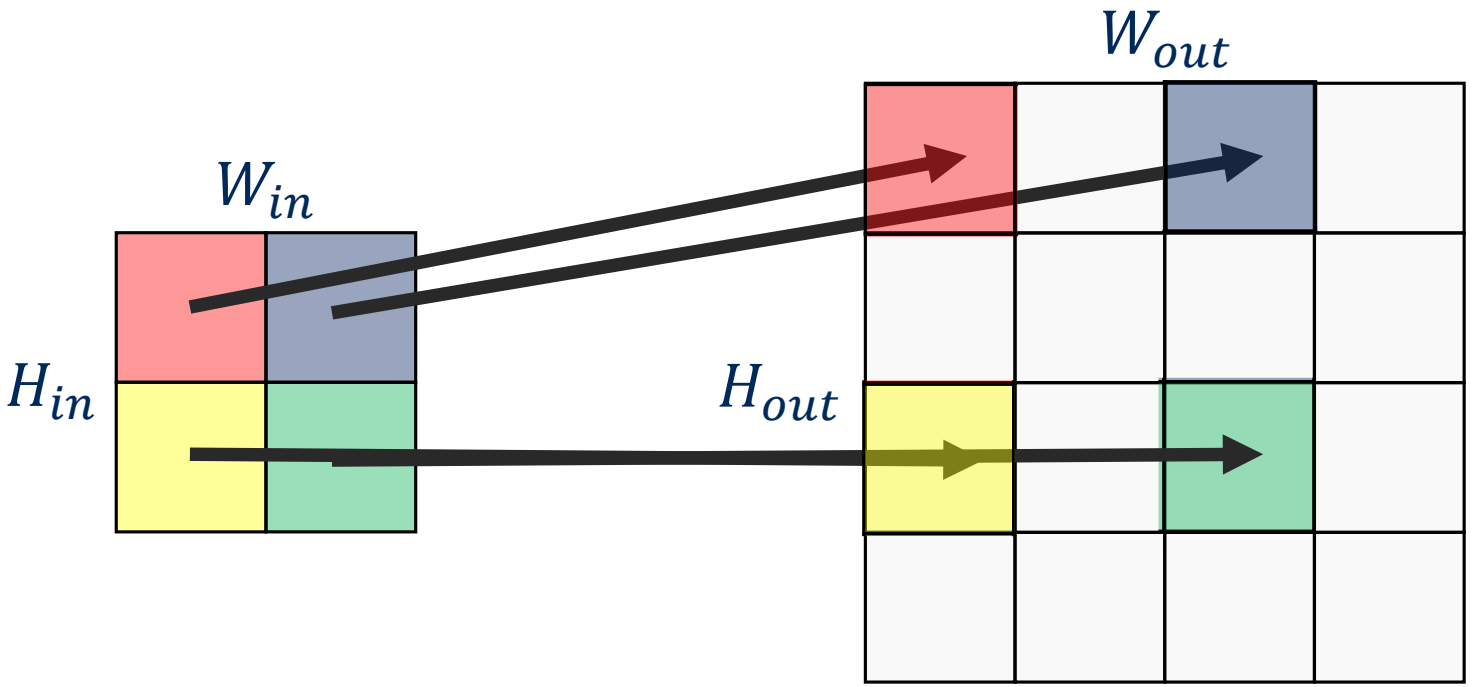
Predictions:
H x W

Long, Shelhamer, and Darrell, “Fully Convolutional Networks for Semantic Segmentation”, CVPR 2015
Noh et al, “Learning Deconvolution Network for Semantic Segmentation”, ICCV 2015

Upsampling Layer

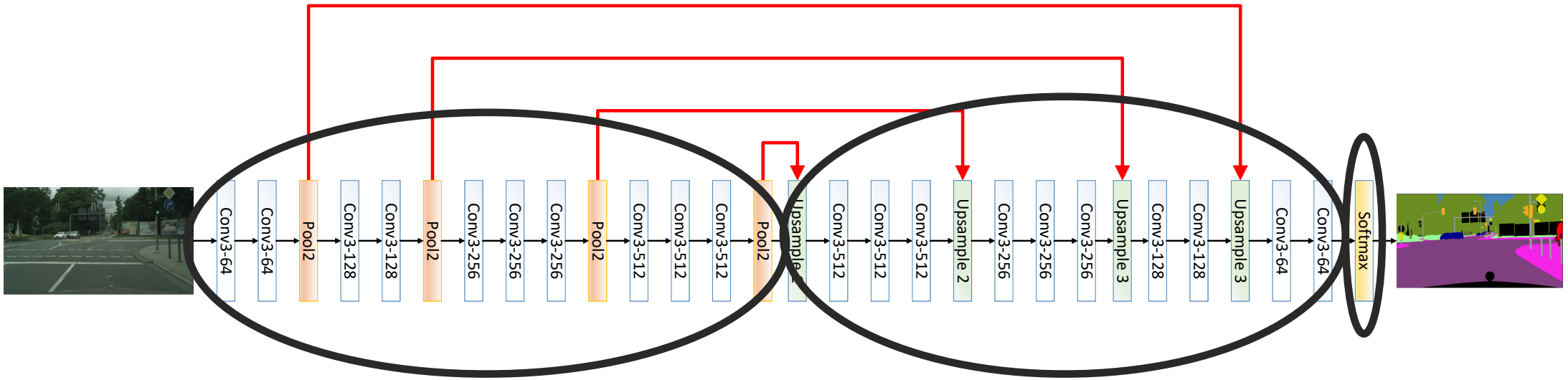
- Upsampling Multiplier $\boldsymbol{S}$

- $\boldsymbol{W_{out} = S \times W_{in}}$
- $\boldsymbol{H_{out} = S \times H_{in}}$
- $\boldsymbol{D_{out} = D_{in}}$

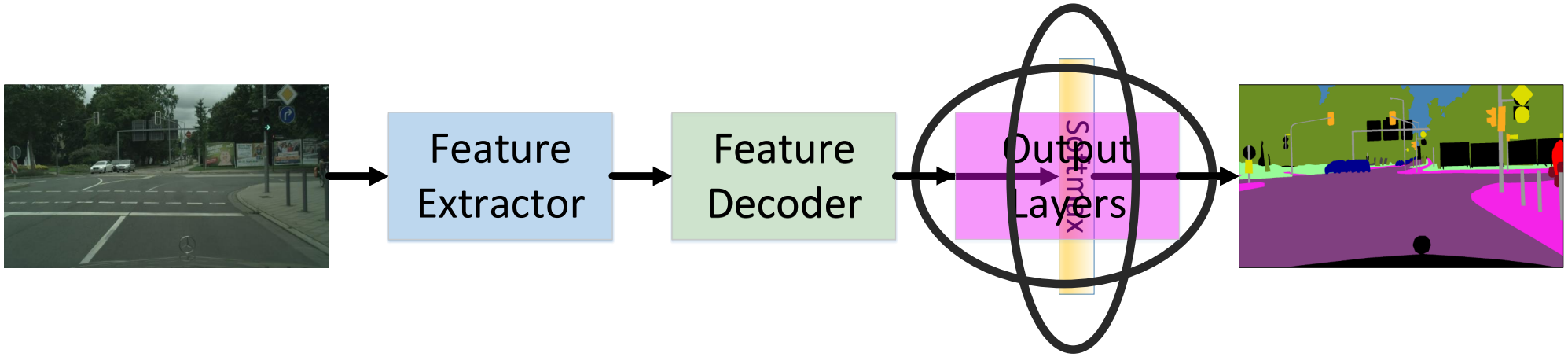


ConvNets For Semantic Segmentation

- Up-convolution, feature pyramids



Learning Same Resolution Feature Maps



Softmax:

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for classes } i = 1, \ldots, K \text{ and scores } \mathbf{z} = (z_1, \ldots, z_K) \in \mathbb{R}^K$$

Classification Loss

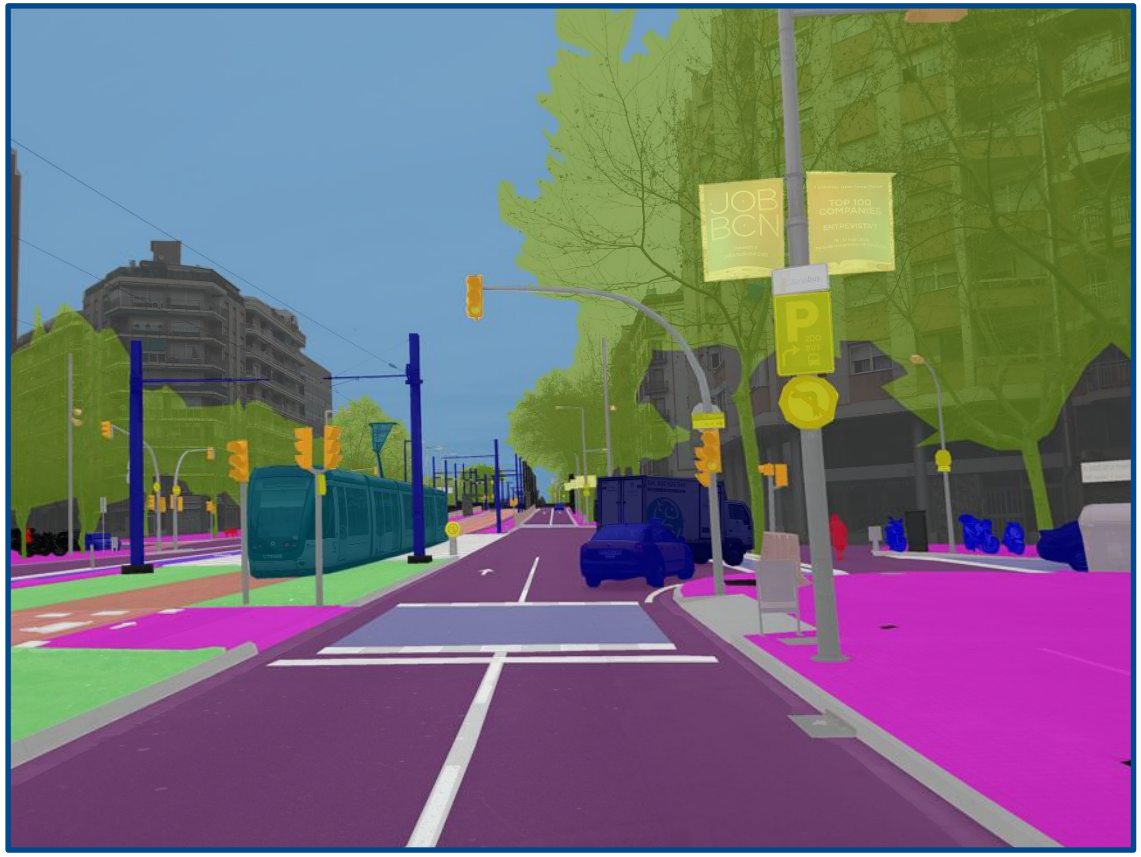
- Cross-entropy loss (recall cross entropy from Lecture 3, also known as negative log likelihood)

$$\begin{aligned} L_{cls} &= \frac{1}{N_{total}} \sum_i H\left(s_i^*, s_i\right) \\ &= -\frac{1}{N_{total}} \sum_i \sum_k s_{ik}^* \log s_{ik} \end{aligned}$$

  - N_{total} is the number of pixels in all images of our minibatch
  - s_i is the softmax output of the neural network
  - s_i^* is the ground truth classification probability distributions, defined as a one-hot encoding for the true class

$$\begin{aligned} H(p,q) &= \int p(x) \ln\left(\frac{p(x)}{q(x)}\right) + p(x) \ln\left(\frac{1}{p(x)}\right) dx \\ &= \mathbb{KL}(p||q) + H(p) \end{aligned}$$

Semantic Segmentation Results

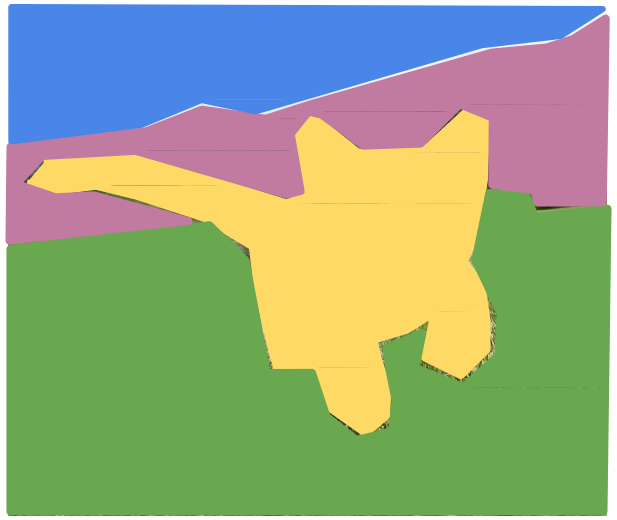


Data From:
https://www.mapillary.com/dataset/vistas?pKey=rwbBtYKofke2NeLIvj8j-A

More Results

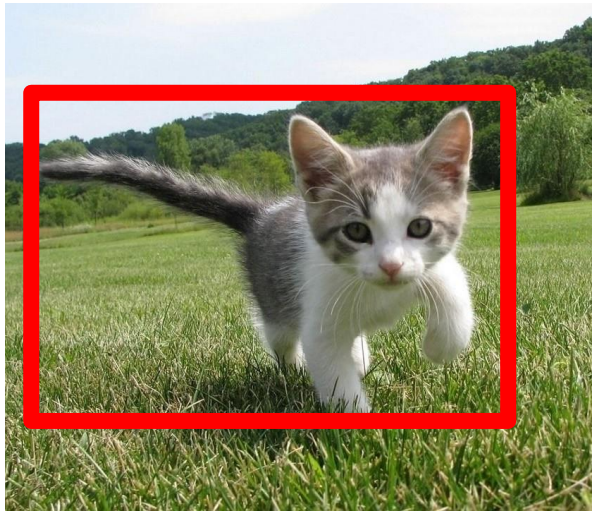
- Segnet on Kitti
- ICNet on CityScapes
- FrameNet on SemanticKitti Lidar (new)

Classification + Localization



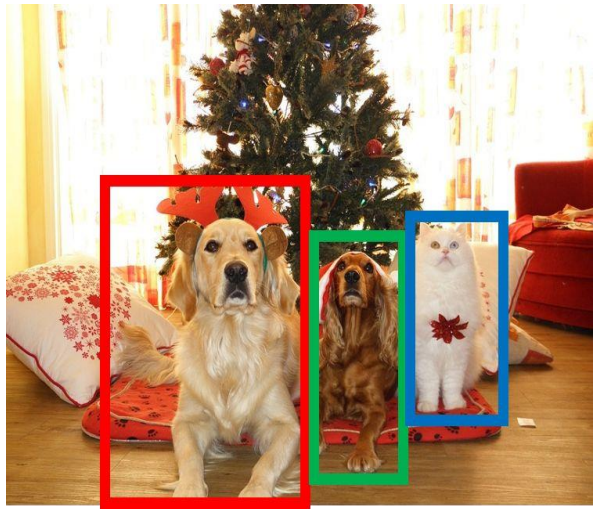
GRASS, CAT, TREE, SKY

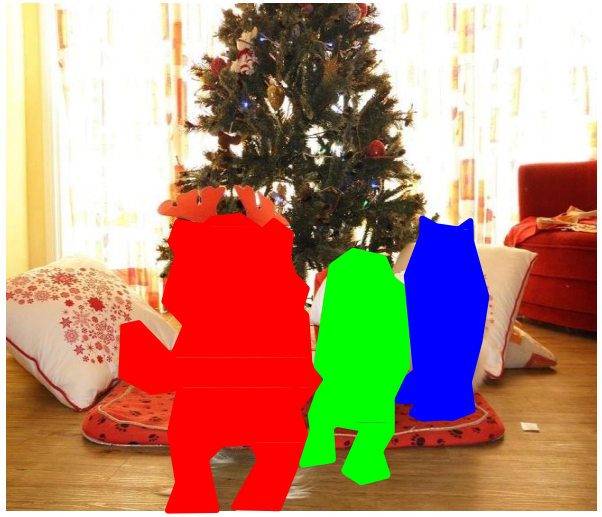
No objects, just pixels



CAT

Single Object



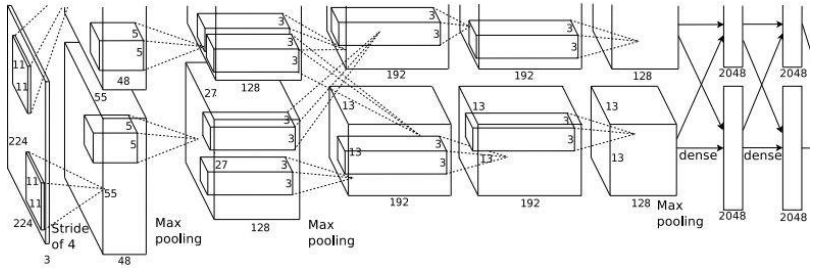


DOG, DOG, CAT

DOG, DOG, CAT

Multiple Object

Classification + Localization



Fully Connected: 4096 to 1000

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

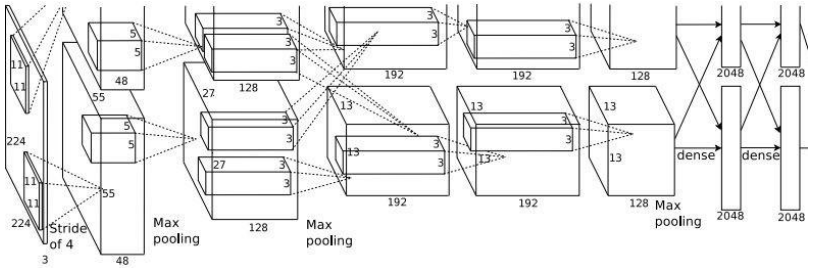
Vector: 4096

Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

Treat localization as a regression problem!

Classification + Localization



Correct label:
Cat

Fully Connected:
4096 to 1000

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Softmax Loss

Vector:
4096

Fully Connected:
4096 to 4

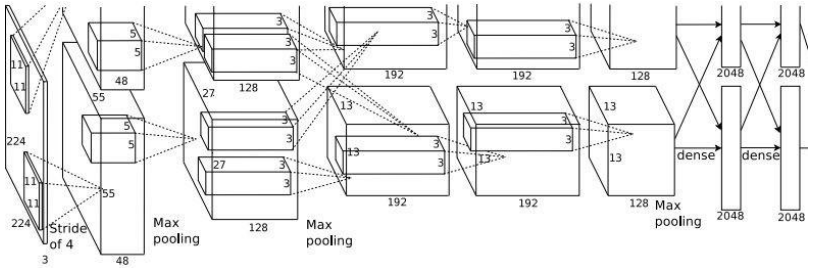
Box Coordinates
(x, y, w, h)

L2 Loss

Correct box:
(x', y', w', h')

Treat localization as a regression problem!

Classification + Localization



Vector:
4096

Fully Connected:
4096 to 1000

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Softmax Loss

Correct label:
Cat

Multitask Loss

Loss

Fully Connected:
4096 to 4

Box Coordinates
(x, y, w, h)

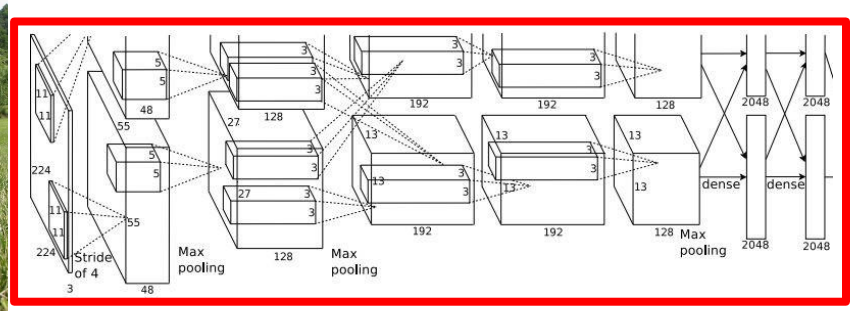
L2 Loss

Correct box:
(x', y', w', h')

Treat localization as a regression problem!

Classification + Localization

This image is CC0 public domain



Often pretrained on ImageNet
(Transfer learning)

Vector: 4096

Fully Connected: 4096 to 1000

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Softmax Loss

Correct label:
Cat

Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

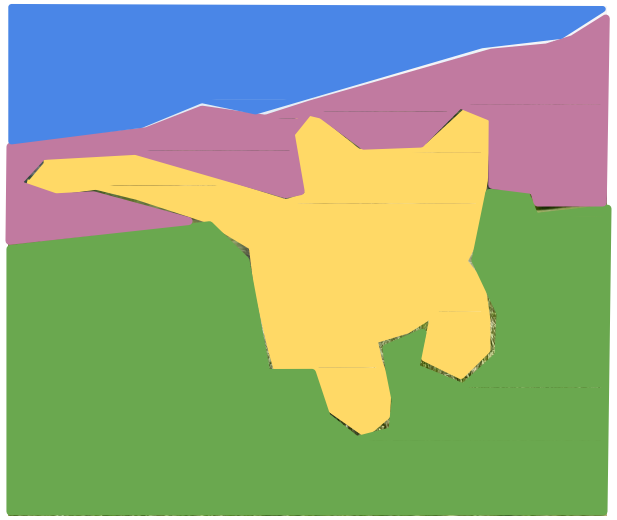
L2 Loss

Correct box:
(x', y', w', h')

+ → **Loss**

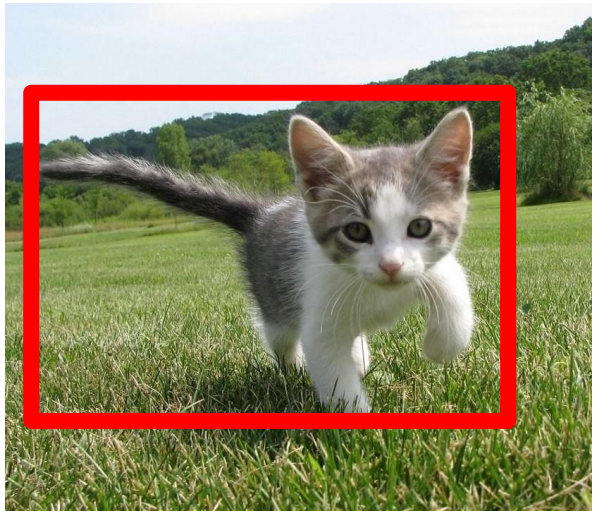
Treat localization as a regression problem!

Object Detection



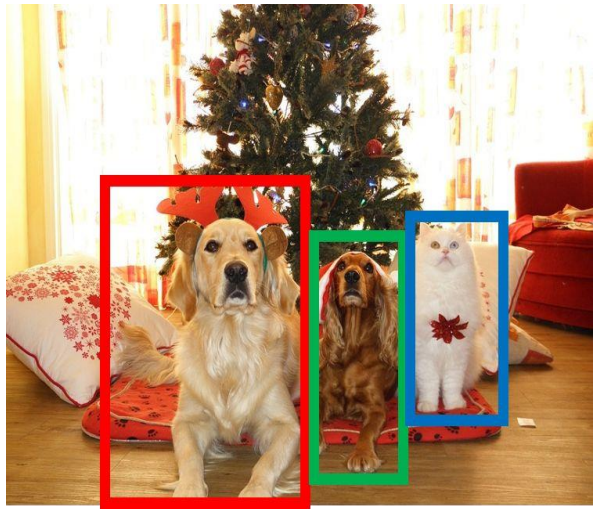
GRASS, CAT, TREE, SKY

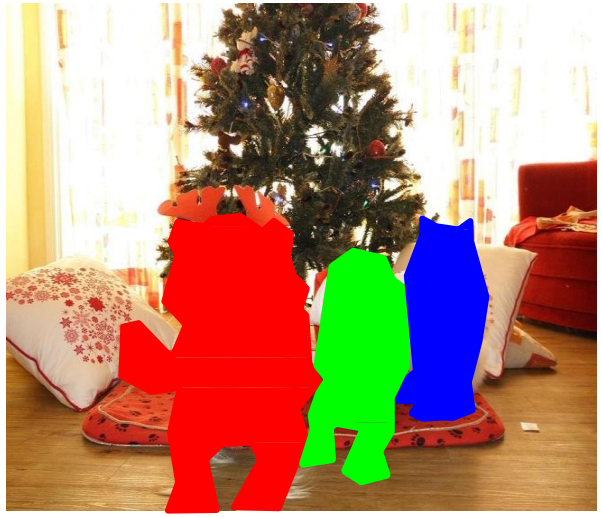
No objects, just pixels



CAT

Single Object





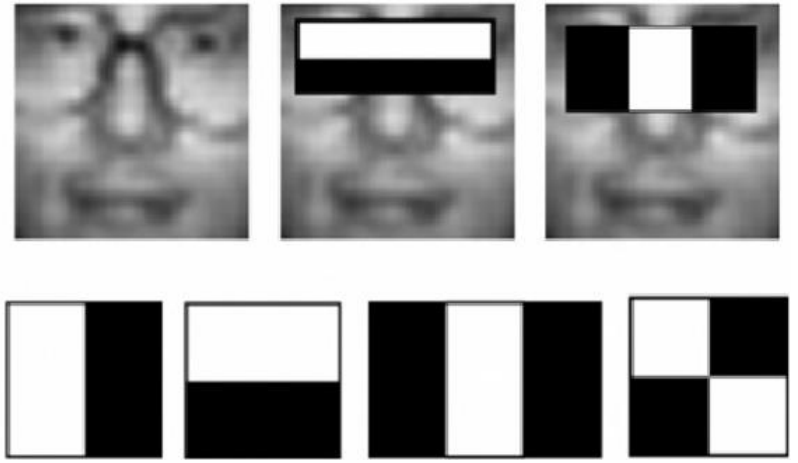
DOG, DOG, CAT

Multiple Object

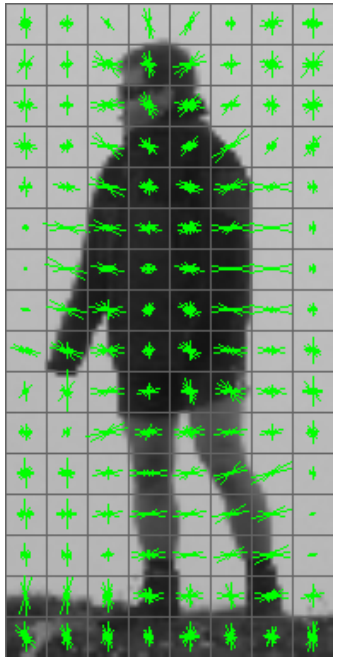
Brief History of Object Detection

- 2001 – Viola, Jones – Viola-Jones Object Detection Framework
- 2005 – Dalal, Triggs – Histogram of Oriented Gradients
- 2009 – Deng, Dong, Socher, Li, Li and Li, Imagenet
- 2012 – Krizhevsky, Sutskever, Hinton - Alexnet

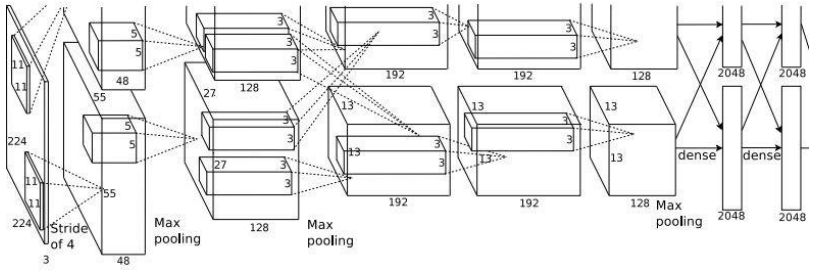
Viola-Jones



HOG

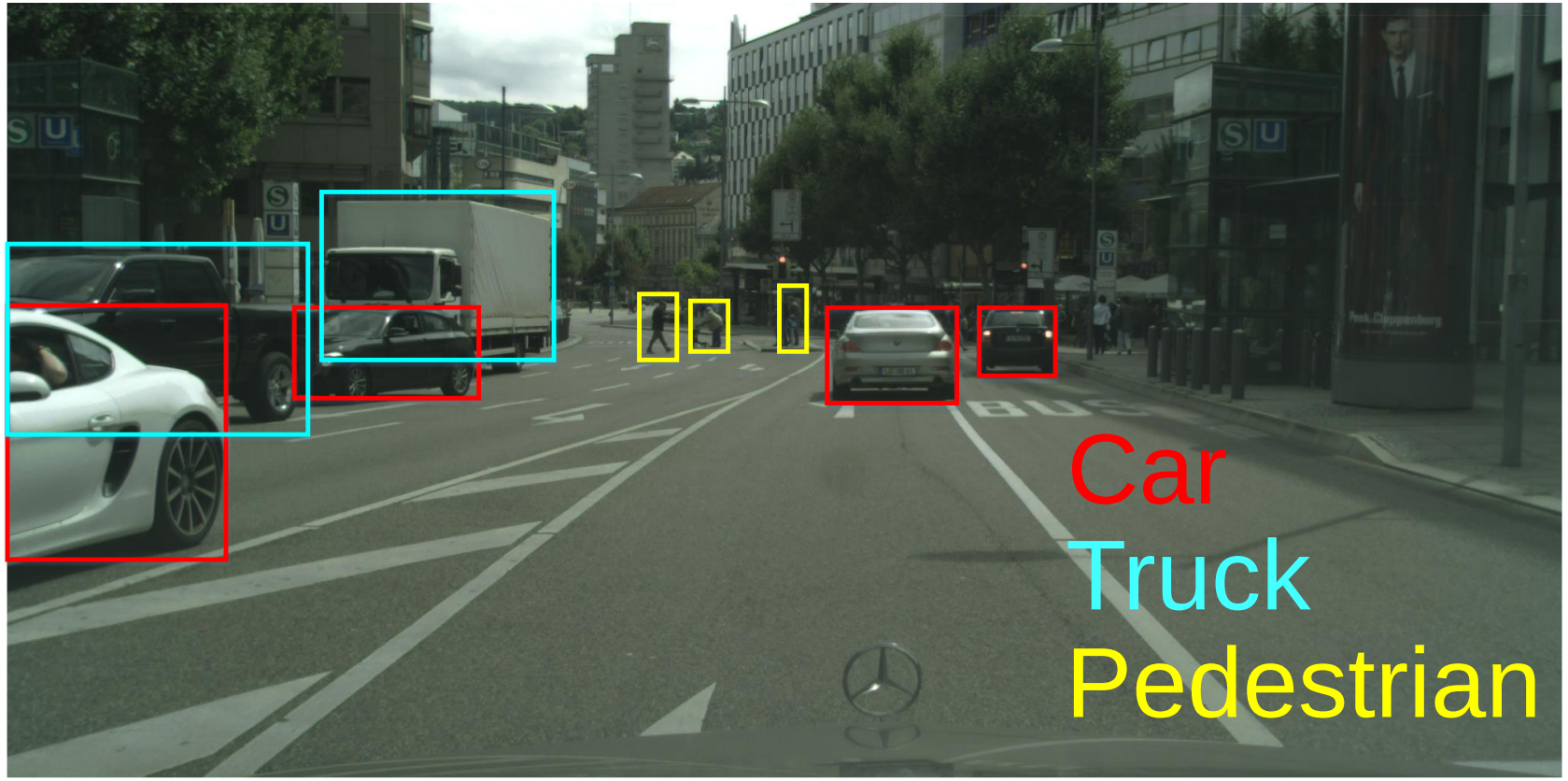


Alexnet



Imagenet

The Object Detection Problem

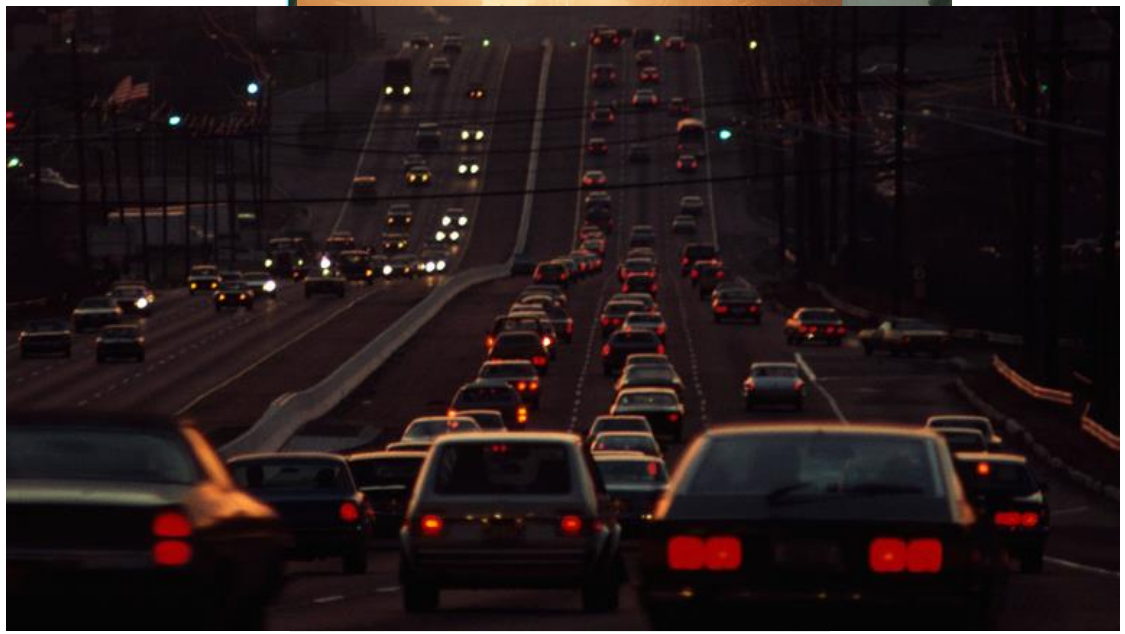


Object Detection Is Not Trivial !

- **Extent of objects is not fully observed!**
  - **Occlusion:** Background objects covered by foreground objects
  - **Truncation:** Objects are out of image boundaries

- **Scale:** Object size gets smaller as the object moves farther away

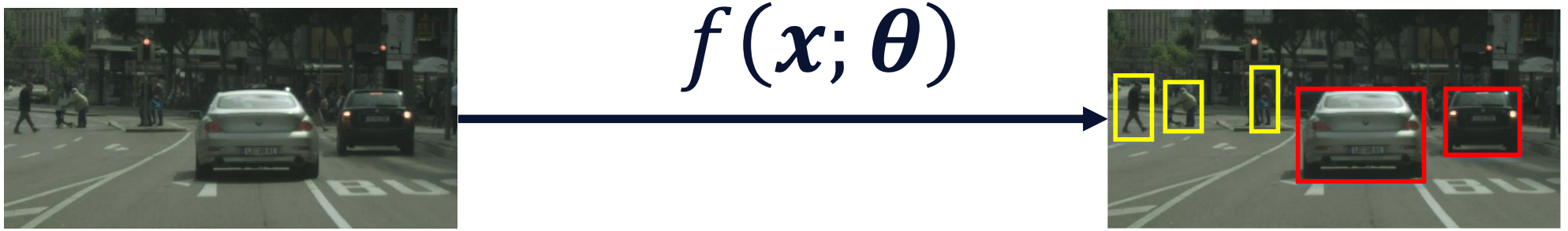
- Illumination Changes:
  - **Too bright**
  - **Too dark**



Mathematical Problem Formulation

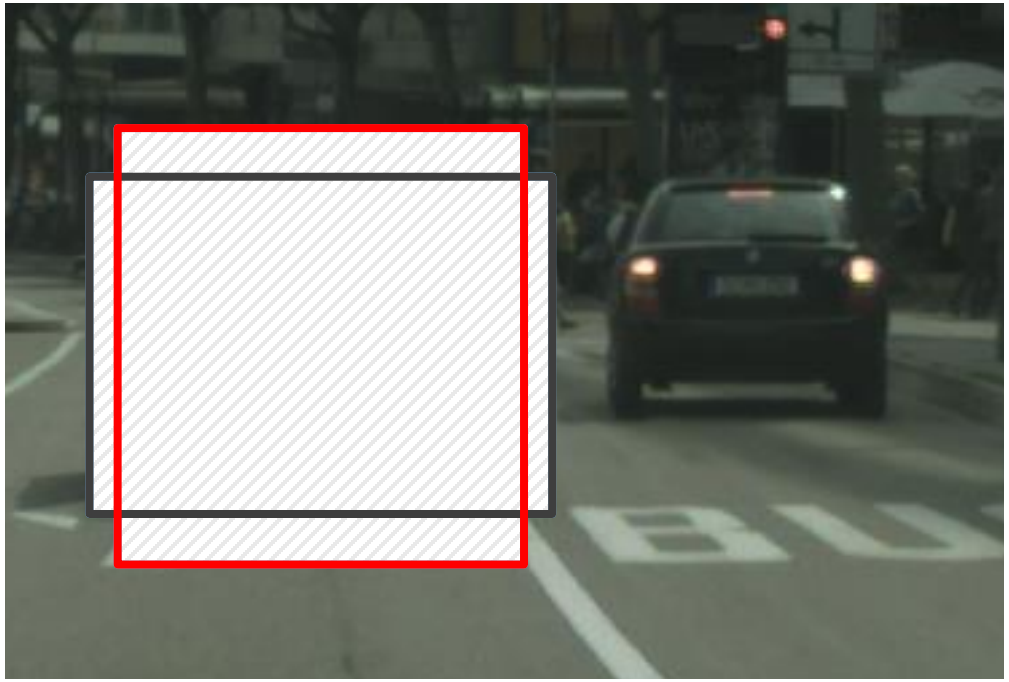
$$f(x;\theta) = [x_{min}, y_{min}, x_{max}, y_{max}, S_{class_1} \dots, S_{class_k}]$$

ConvNets For Object Detection



Evaluation Metrics

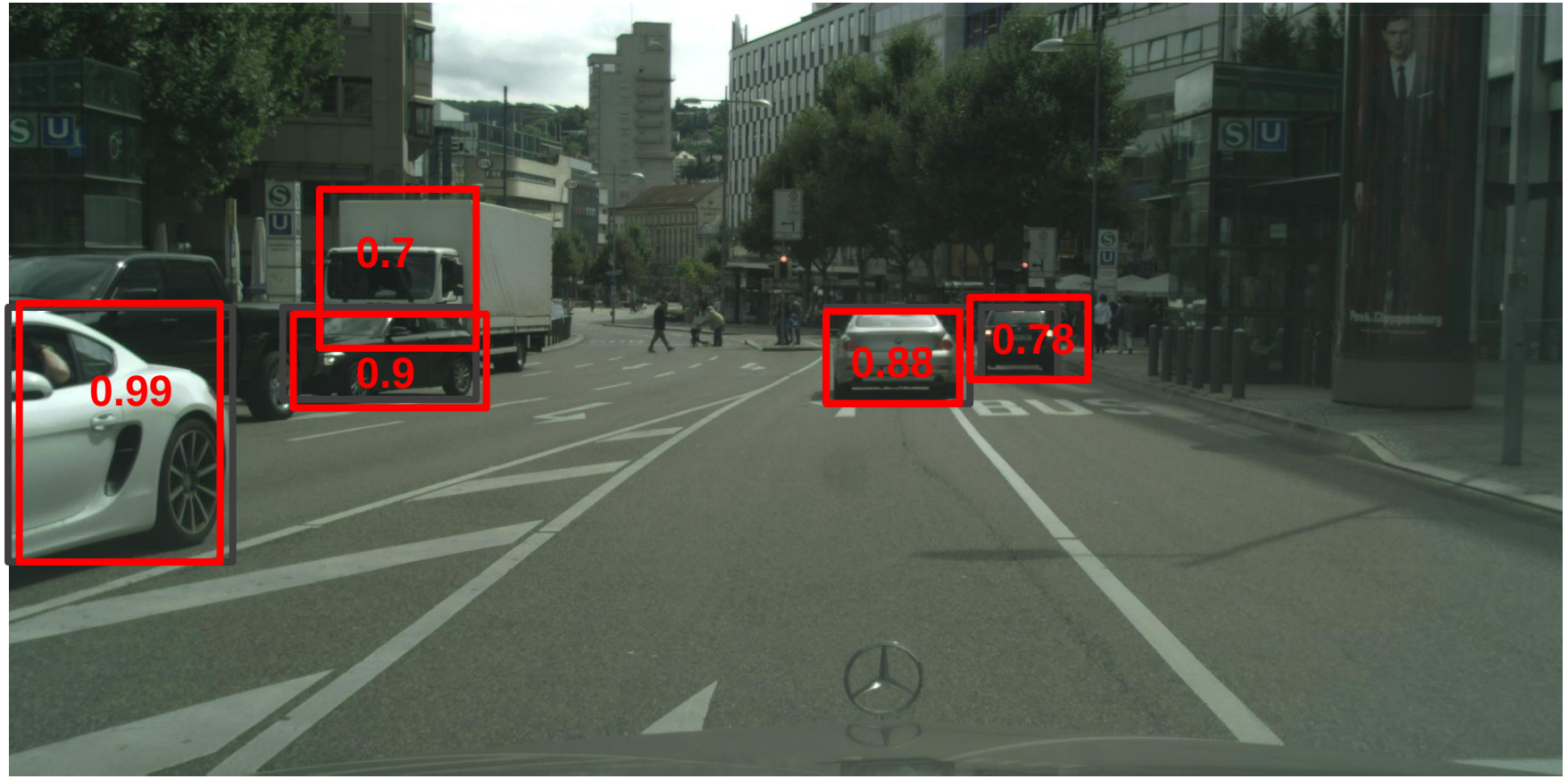
- **Intersection-Over-Union (IOU):** area of intersection of **predicted box** with a **ground truth box**, divided by the area of their union



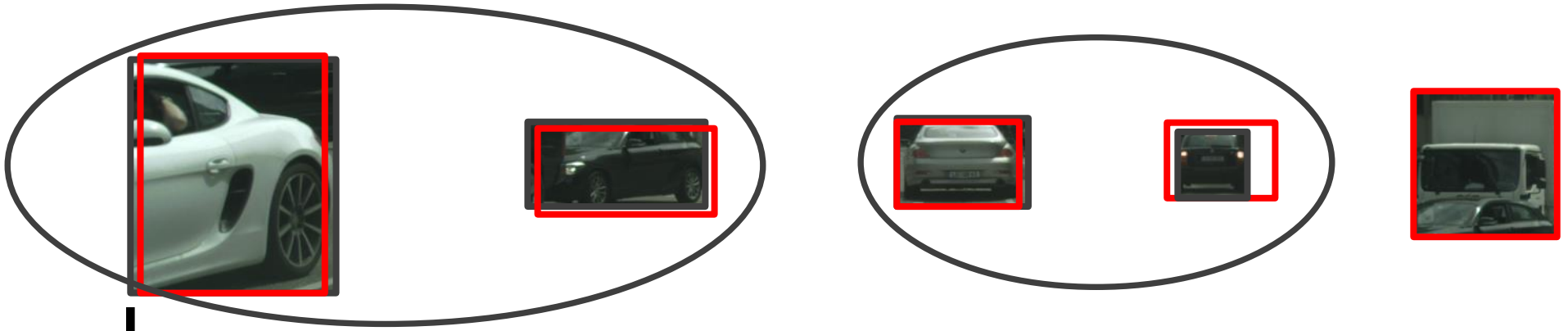
Evaluation Metrics

- **True Positive (TP):** Object class score > score threshold, and IOU > IOU threshold
- **False Positive (FP):** Object class score > score threshold, and IOU < IOU threshold
- **False Negative (FN):** Number of ground truth objects not detected by the algorithm
- **Precision:** TP/ (TP + FP)
- **Recall:** TP / (TP + FN)
- **Precision Recall Curve (PR-Curve):** Use multiple object class score thresholds to compute precision and recall. Plot the values with precision on y-axis, and recall on x-axis
- **Average Precision (AP):** Area under PR-Curve for a single class. Usually approximated using ~~**11**~~ **40** recall points

Example



Example

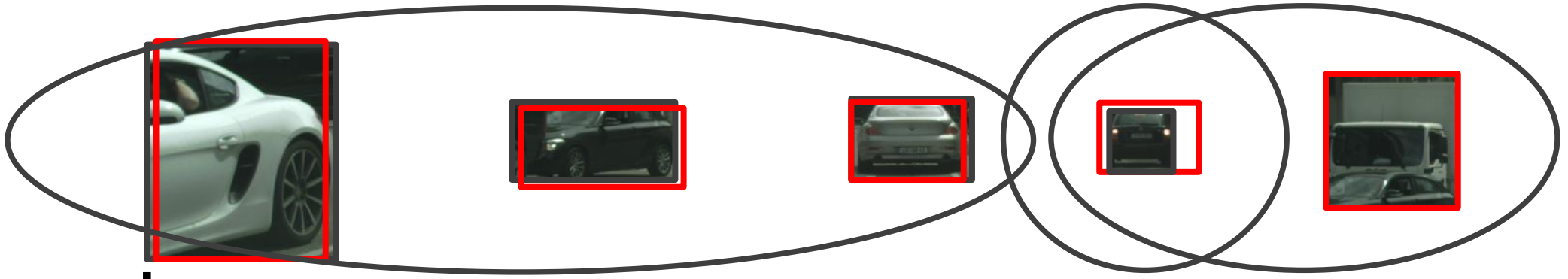


S_{car}	0.99	0.9	0.88	0.78	0.7
IOU	0.95	0.9	0.8	0.3	0

- Score Threshold: 0.9
- IOU Threshold: 0.7

- TP = 2
- FP = 0
- FN = 2
- Precision = 2/2 = 1
- Recall = 2/4 = 0.5

Example



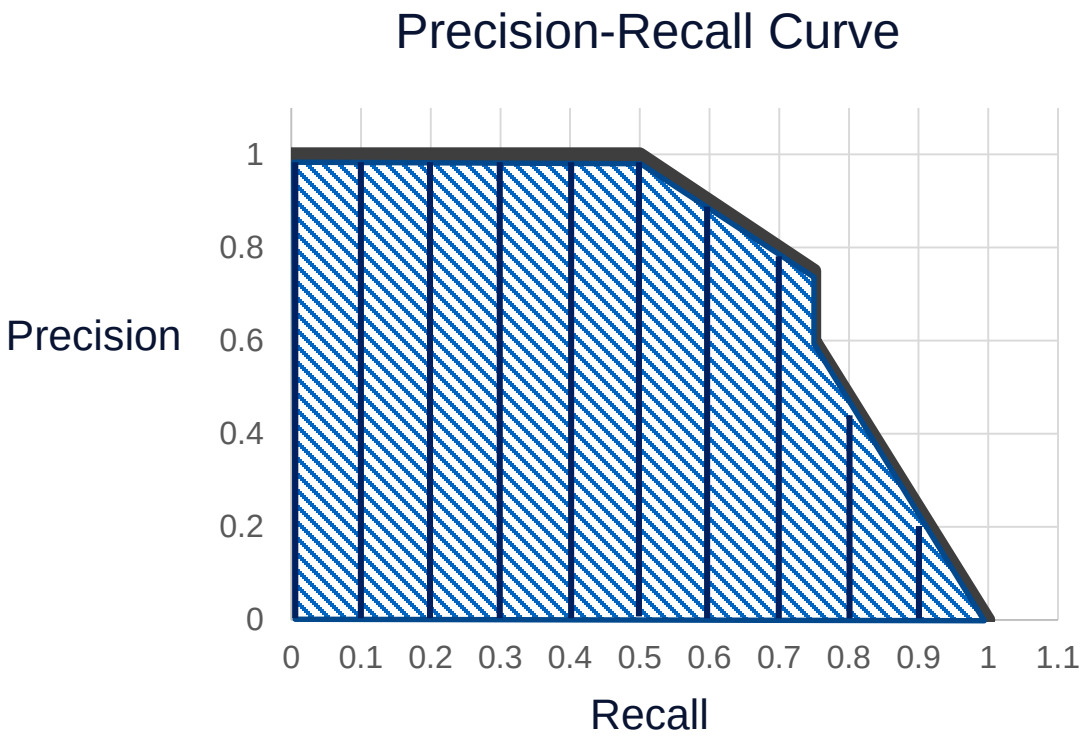
S_{car}	0.99	0.9	0.88	0.78	0.7
IOU	0.95	0.9	0.8	0.3	0

- Score Threshold: 0.7
- IOU Threshold: 0.7

- TP = 3
- FP = 2
- FN = 1
- Precision = 3/5 = 0.6
- Recall = 3/4 = 0.75

Example

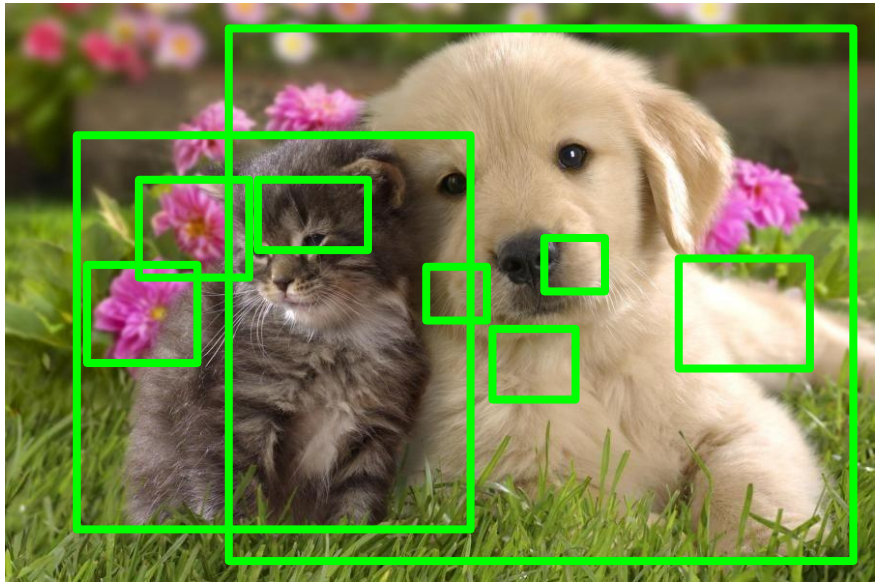
Score threshold	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
Precision	1	0.75	0.6	0.6	0.6	0.6	0.6	0.6	0.6
Recall	0.5	0.75	0.75	0.75	0.75	0.75	0.75	0.75	0.75



$$AP = \frac{1}{11}\sum_{r=0}^{11} p_r \approx 0.75$$

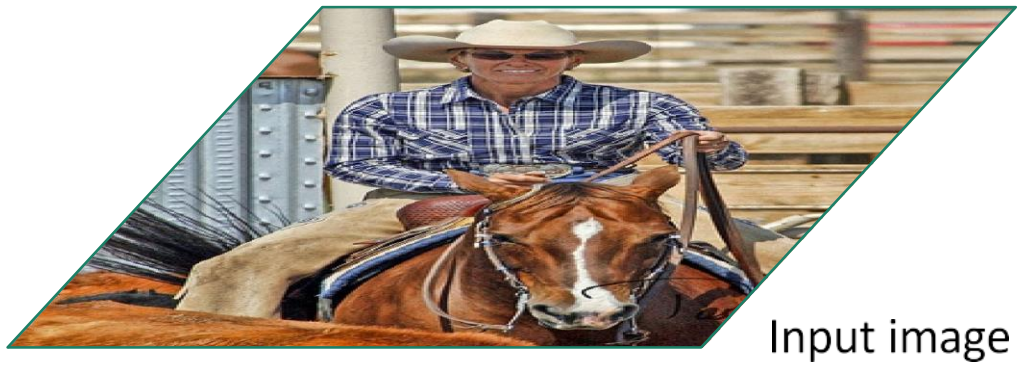
Region Proposals

- Find “blobby” image regions that are likely to contain objects
- Relatively fast to run; e.g. Selective Search gives 1000 region proposals in a few seconds on CPU



Alexe et al, “Measuring the objectness of image windows”, TPAMI 2012
Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013
Cheng et al, “BING: Binarized normed gradients for objectness estimation at 300fps”, CVPR 2014
Zitnick and Dollar, “Edge boxes: Locating object proposals from edges”, ECCV 2014

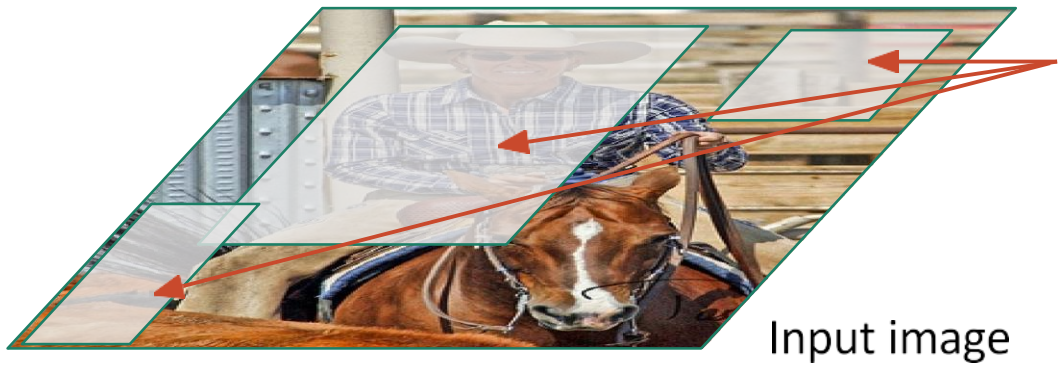
R-CNN



Input image

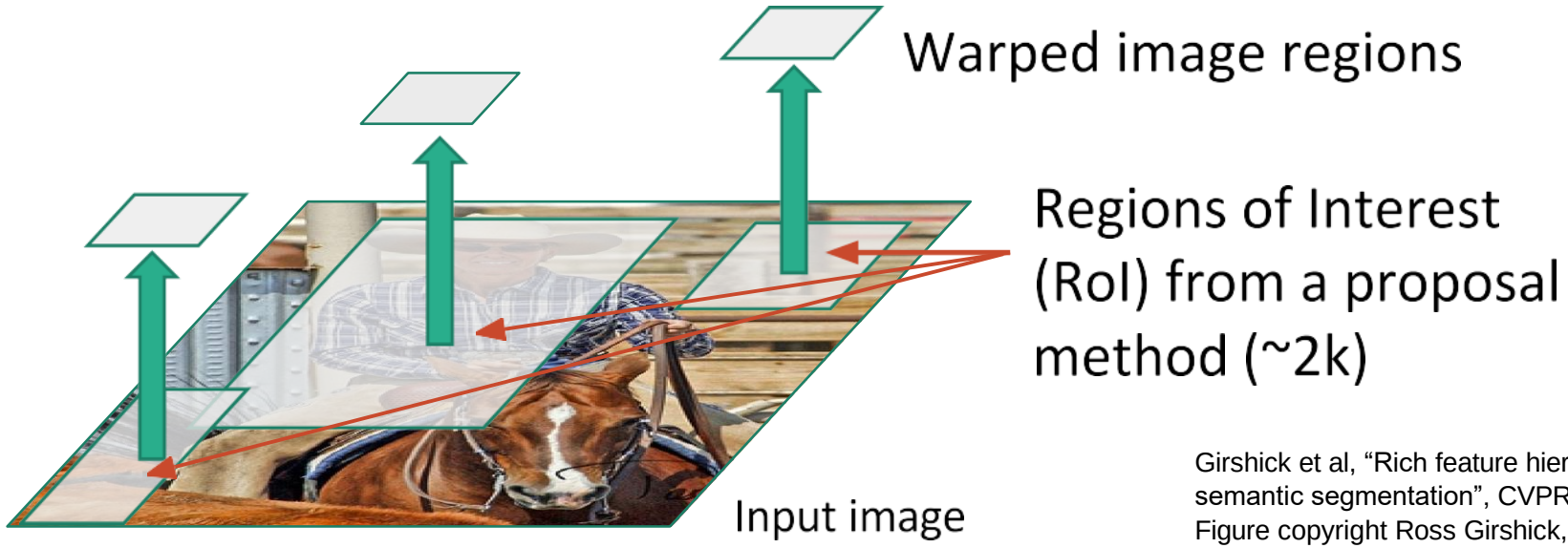
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.

R-CNN



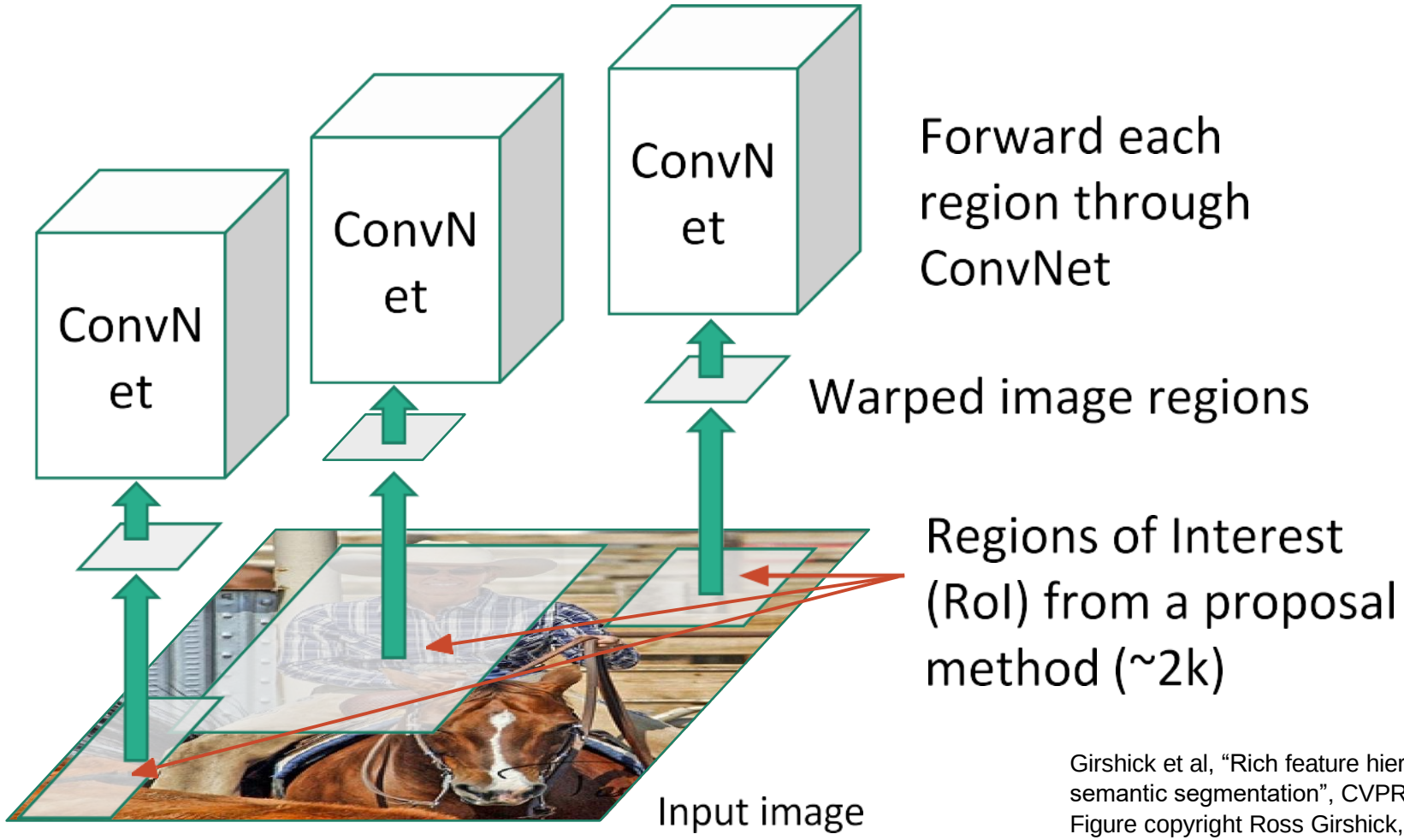
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.

R-CNN



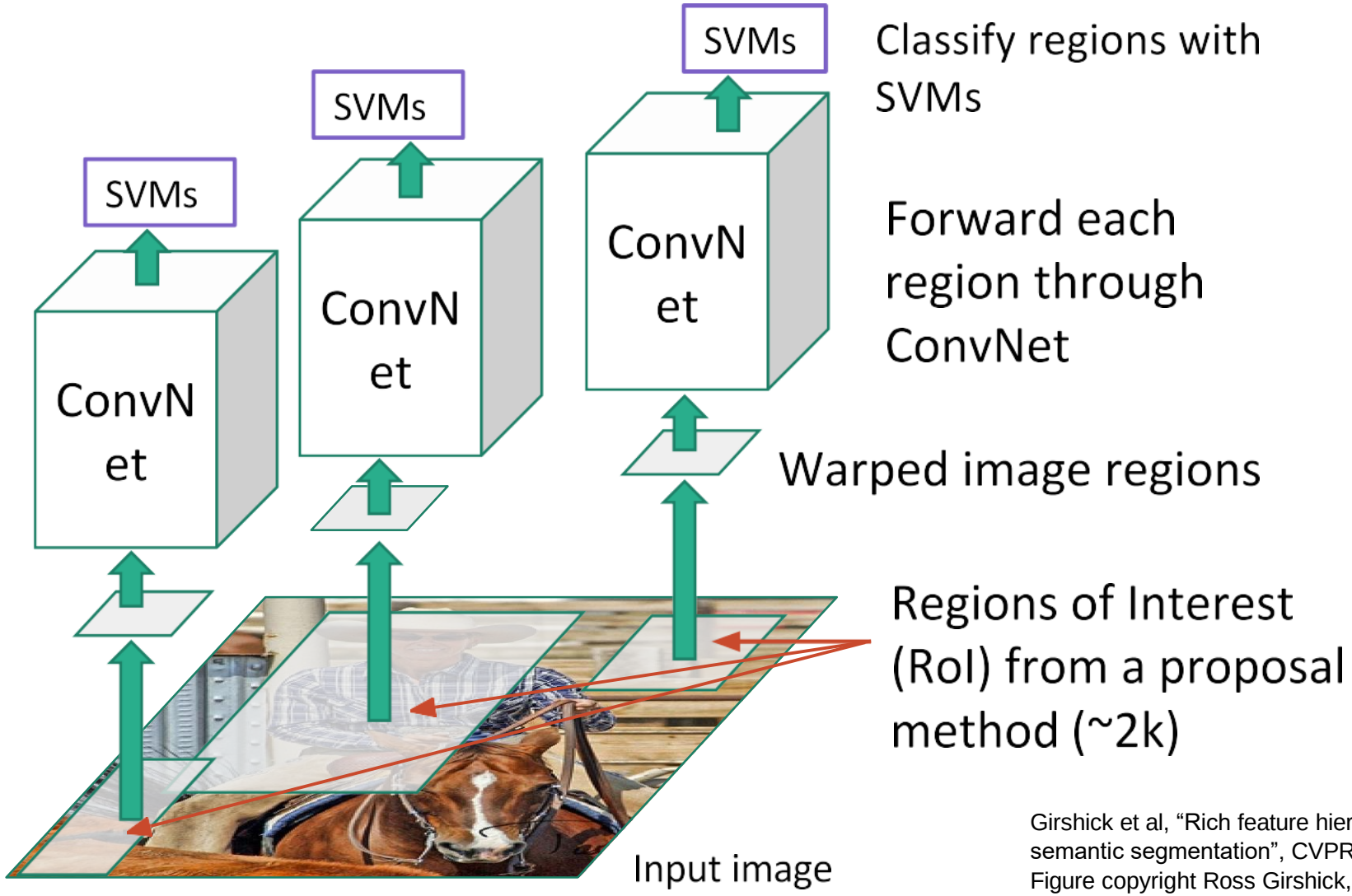
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN



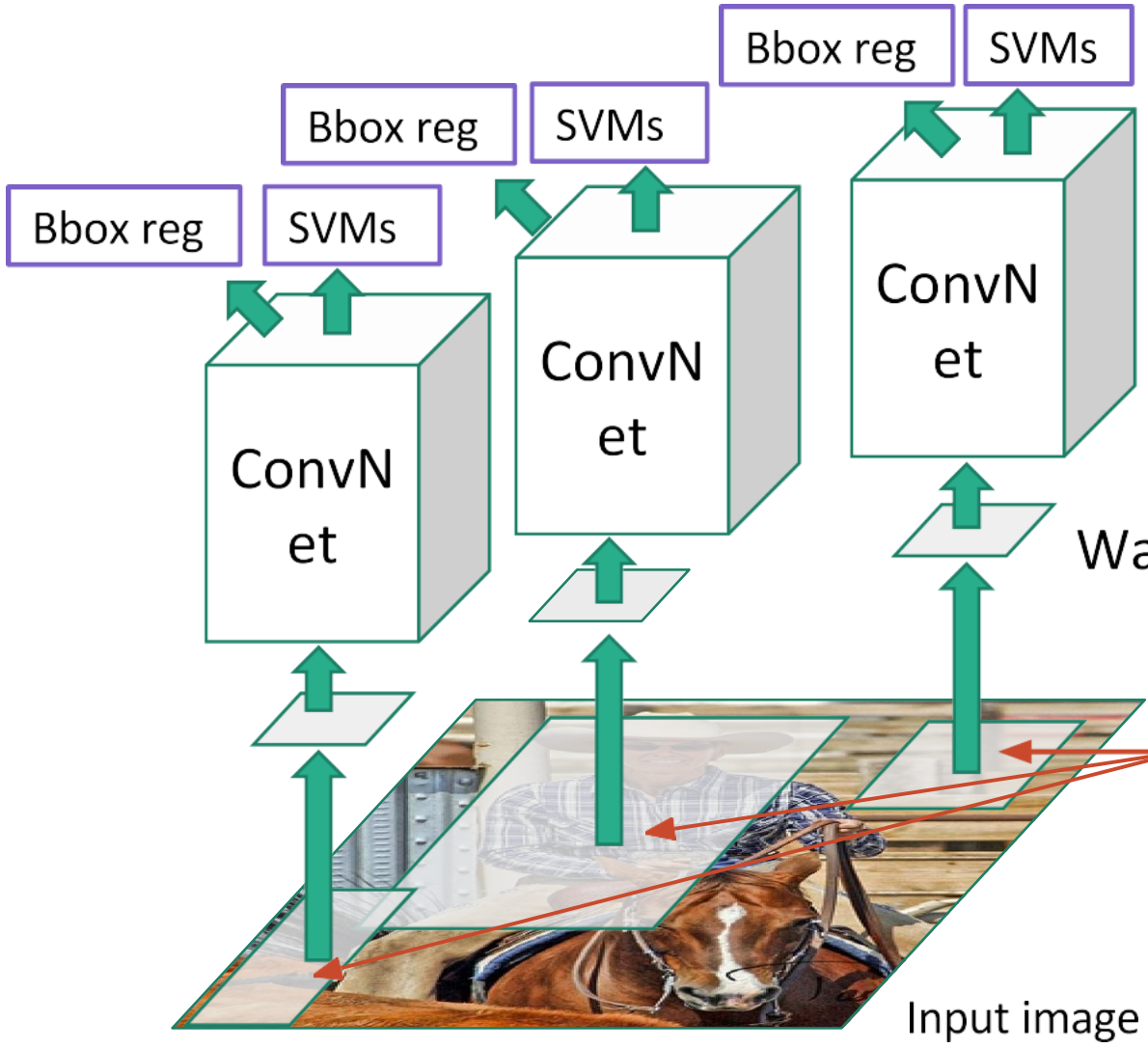
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN



Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN



Linear Regression for bounding box offsets

Classify regions with SVMs

Forward each region through ConvNet

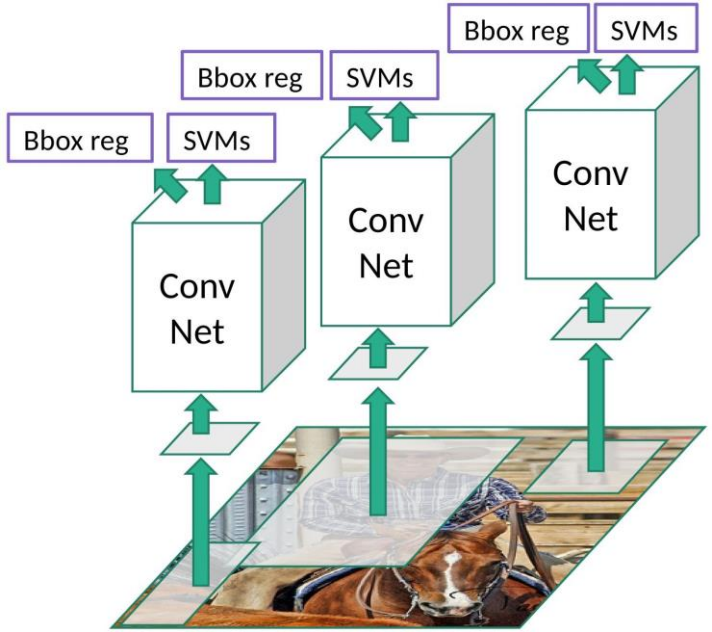
Warped image regions

Regions of Interest (RoI) from a proposal method (~2k)

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.

R-CNN: Problems

- Ad hoc training objectives
  - Fine-tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressions (least squares)
- Training is slow (84h), takes a lot of disk space
- Inference (detection) is slow
  - 47s / image with VGG16 [Simonyan & Zisserman. ICLR15]
  - Fixed by SPP-net [He et al. ECCV14]



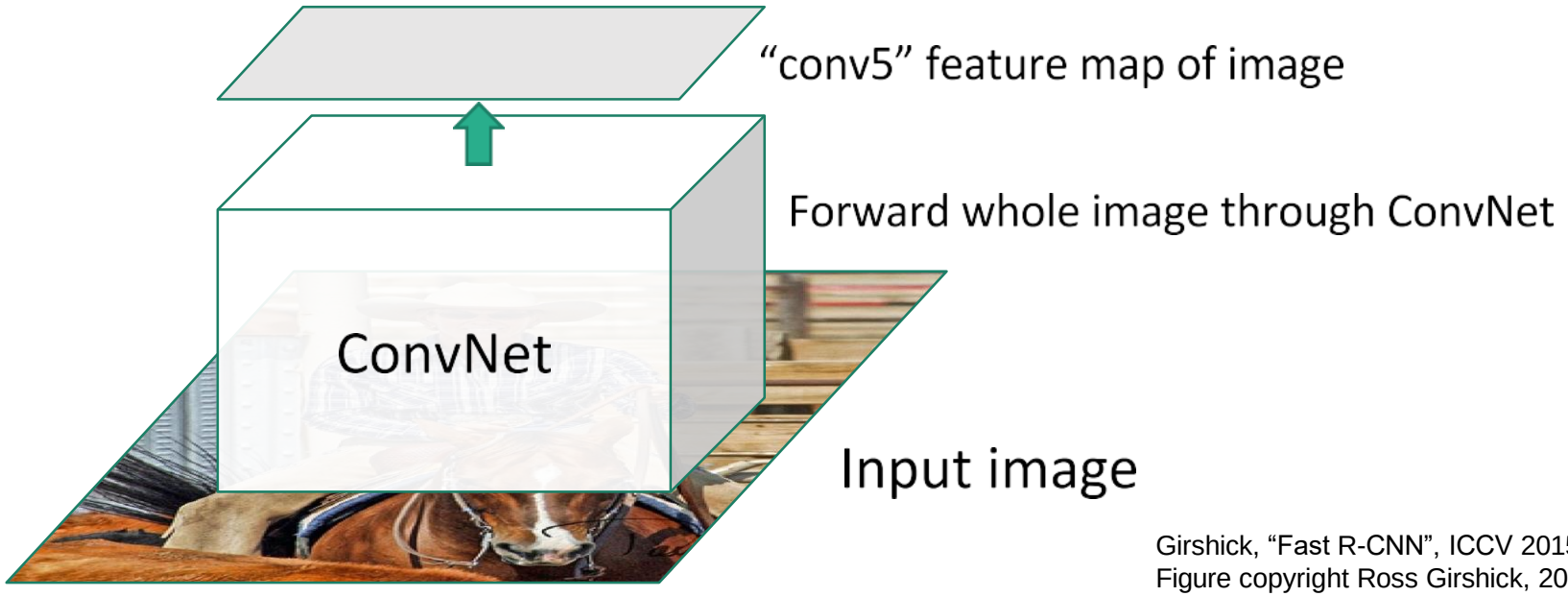
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.

Fast R-CNN

Input image

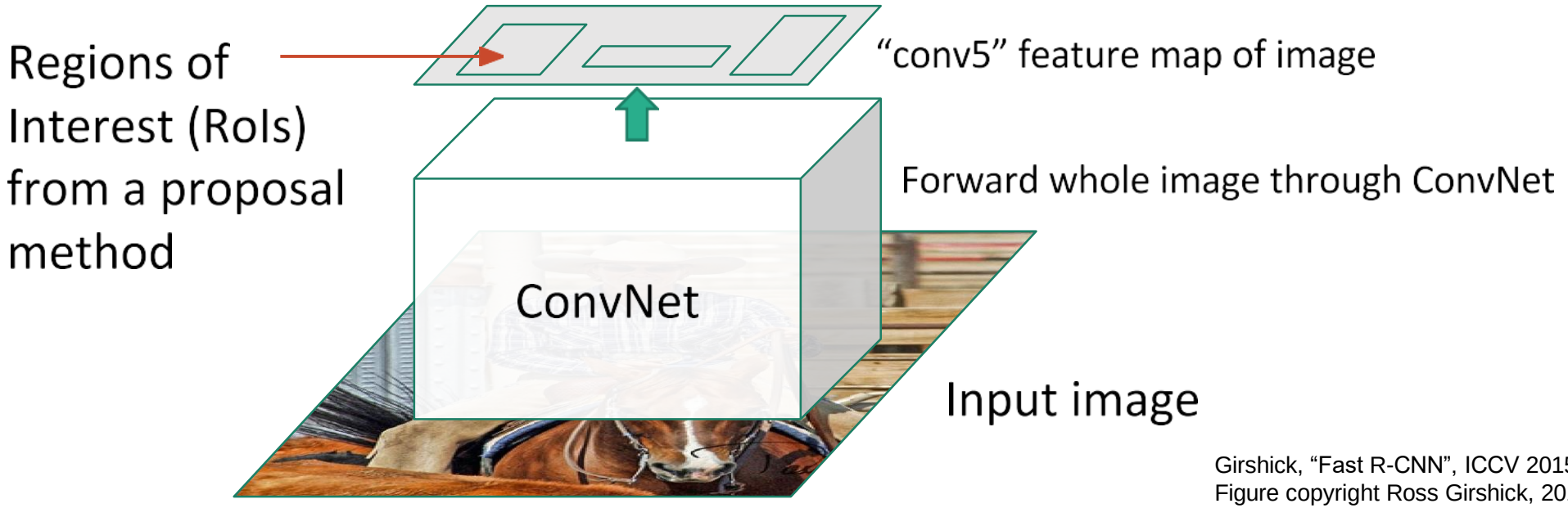
Girshick, “Fast R-CNN”, ICCV 2015.

Fast R-CNN



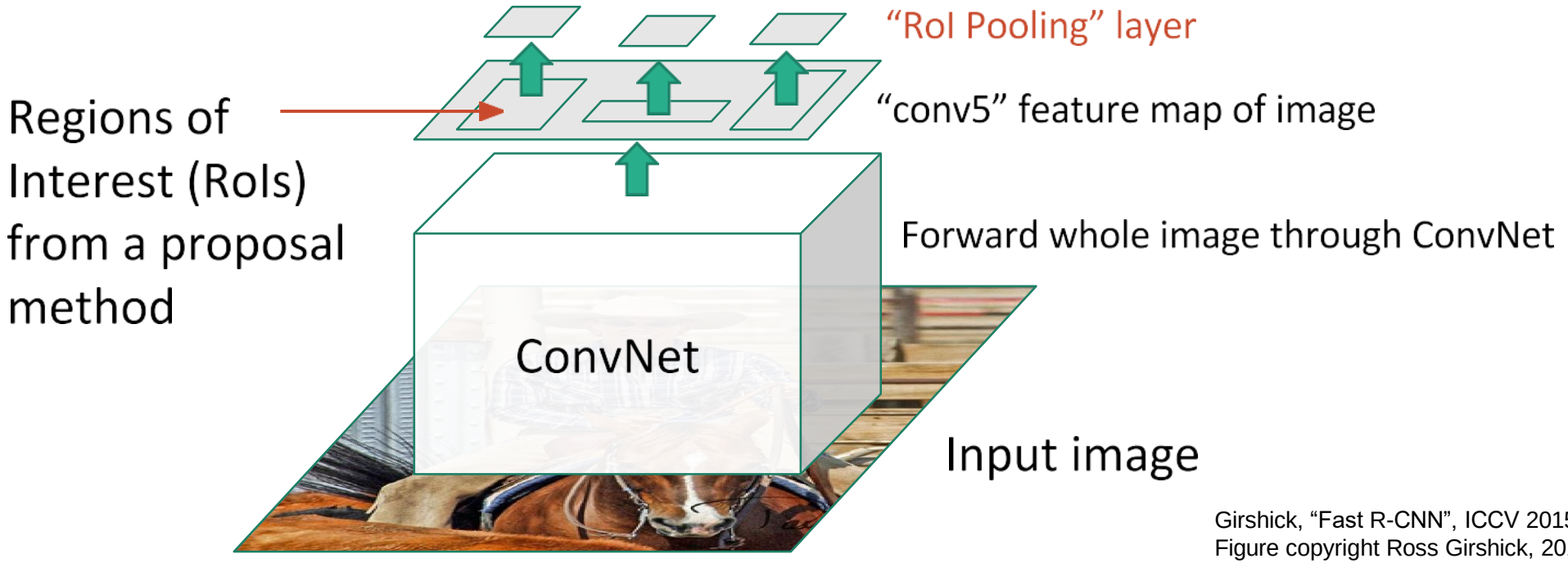
Girshick, “Fast R-CNN”, ICCV 2015.

Fast R-CNN



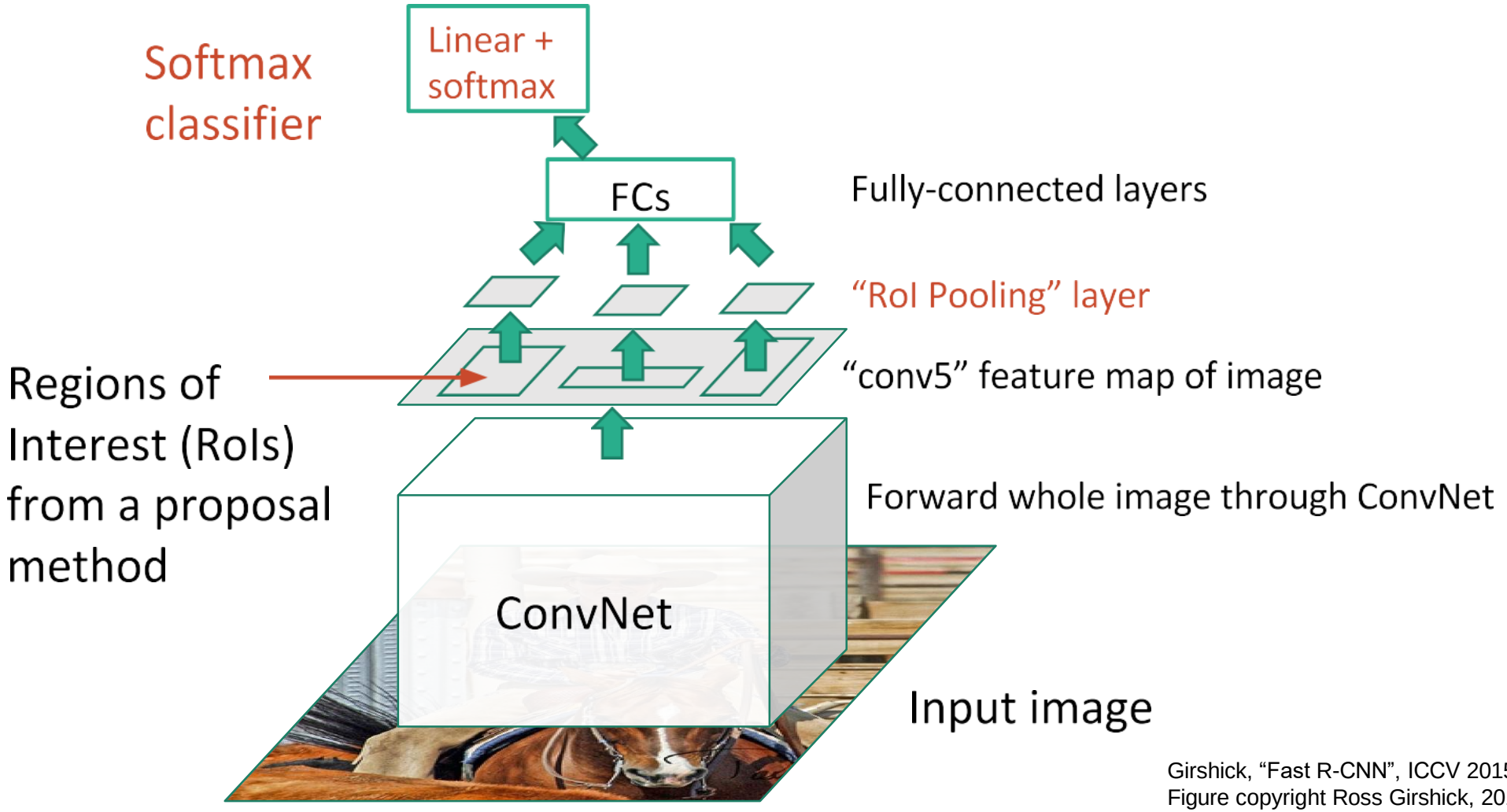
Girshick, "Fast R-CNN", ICCV 2015.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN



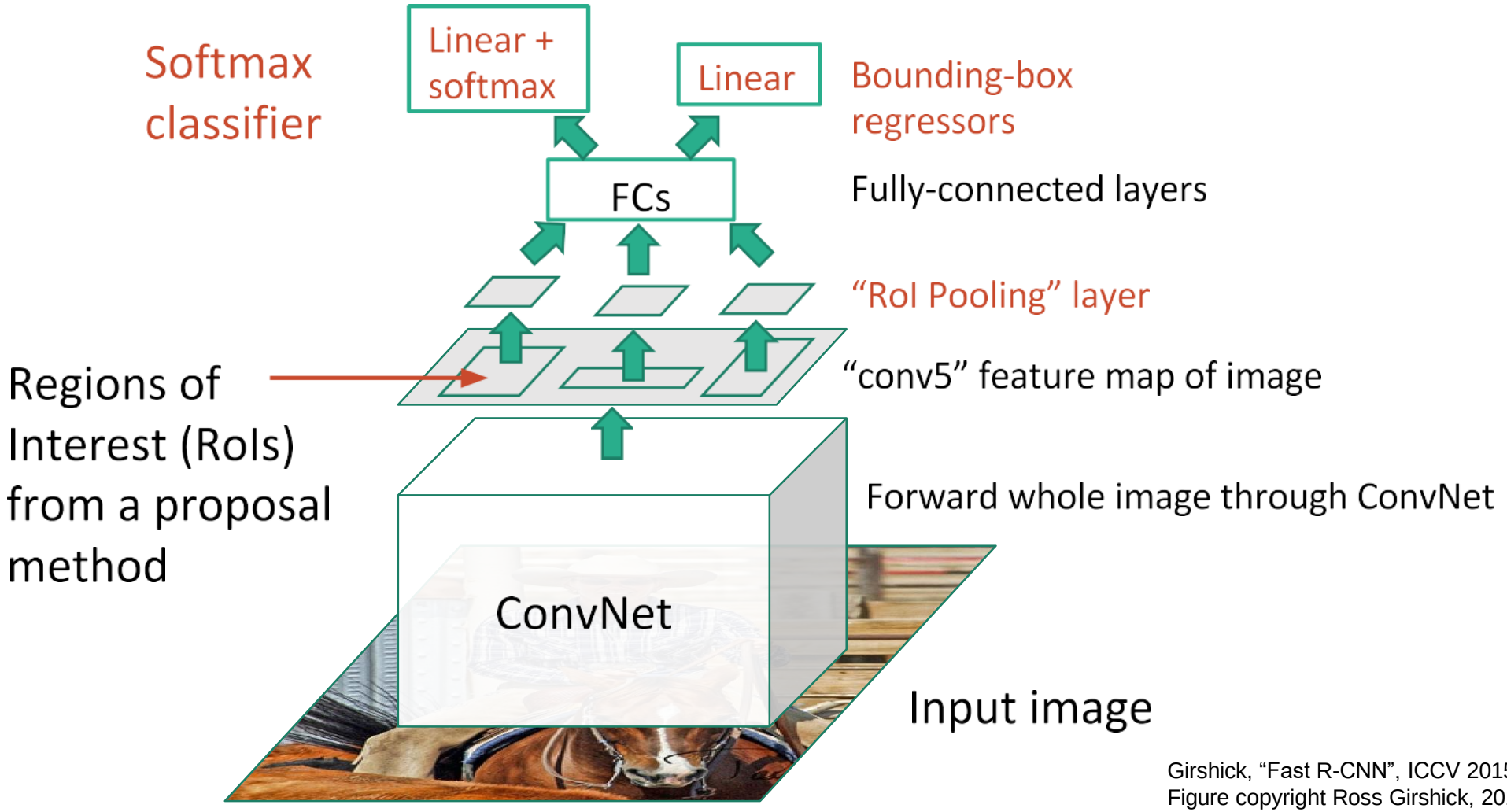
Girshick, "Fast R-CNN", ICCV 2015.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN



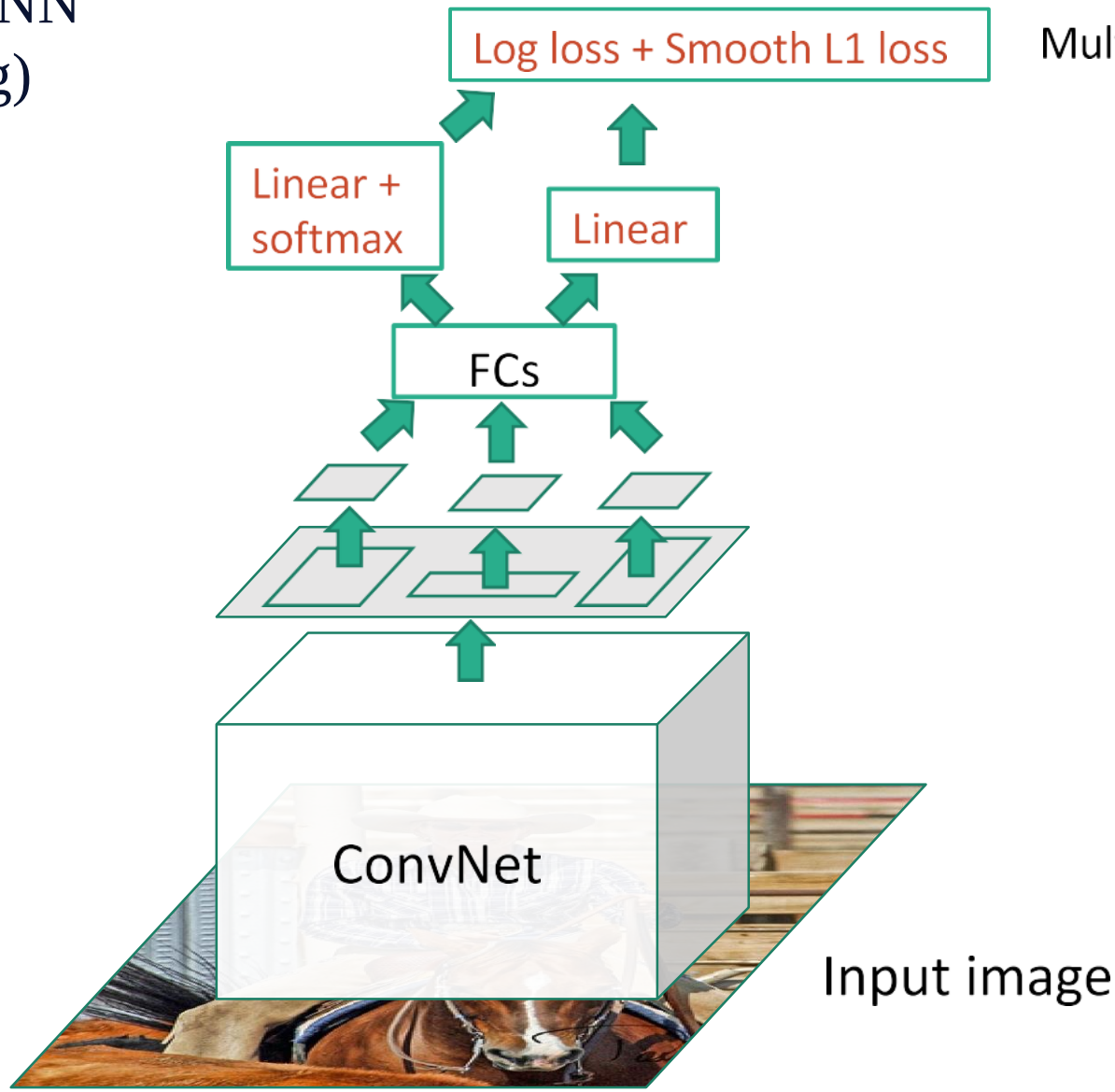
Girshick, "Fast R-CNN", ICCV 2015.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN



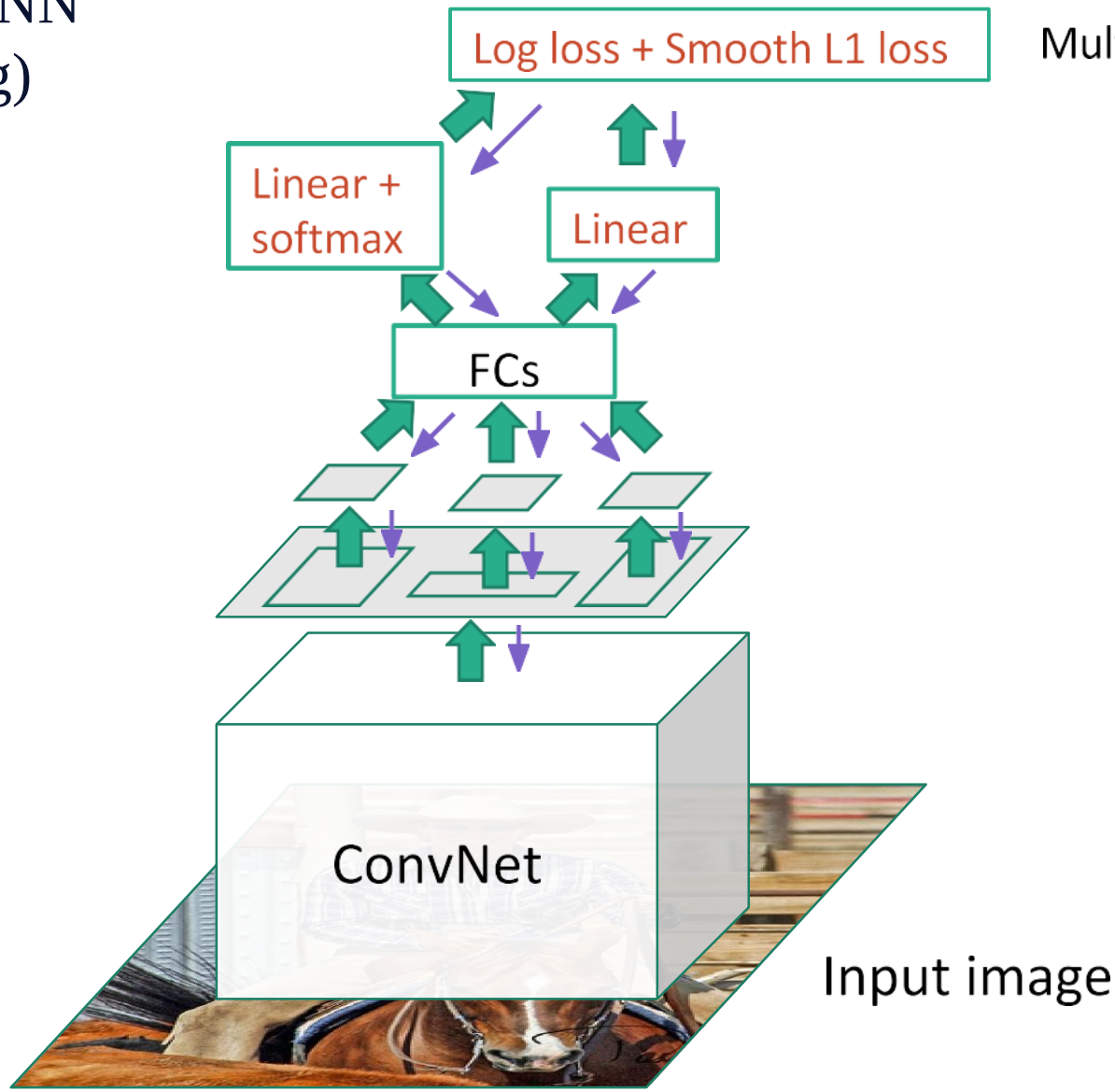
Girshick, "Fast R-CNN", ICCV 2015.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN (Training)



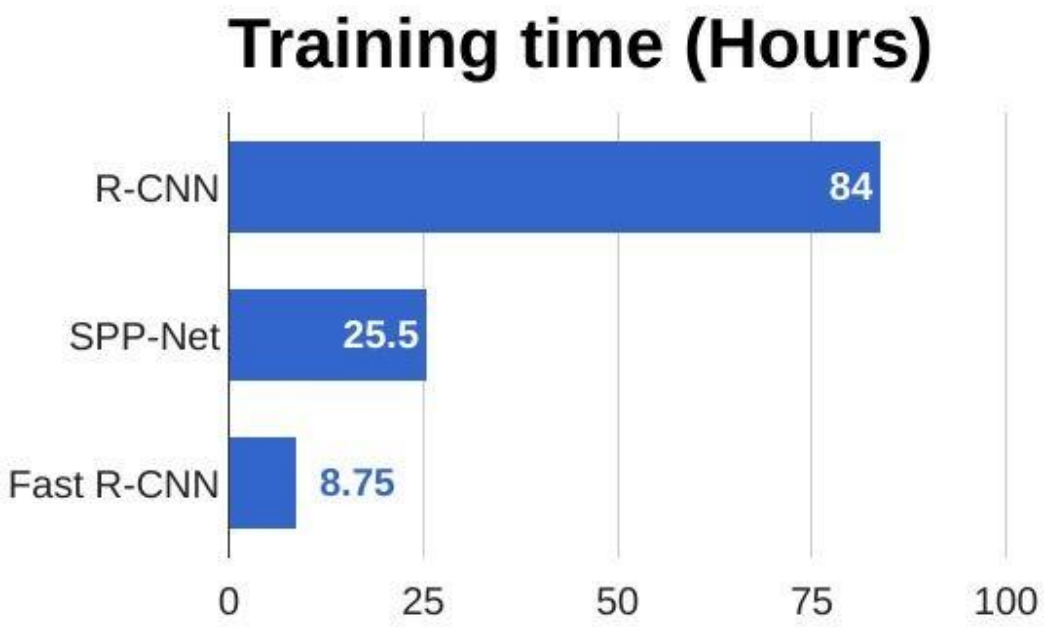
Girshick, “Fast R-CNN”, ICCV 2015.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

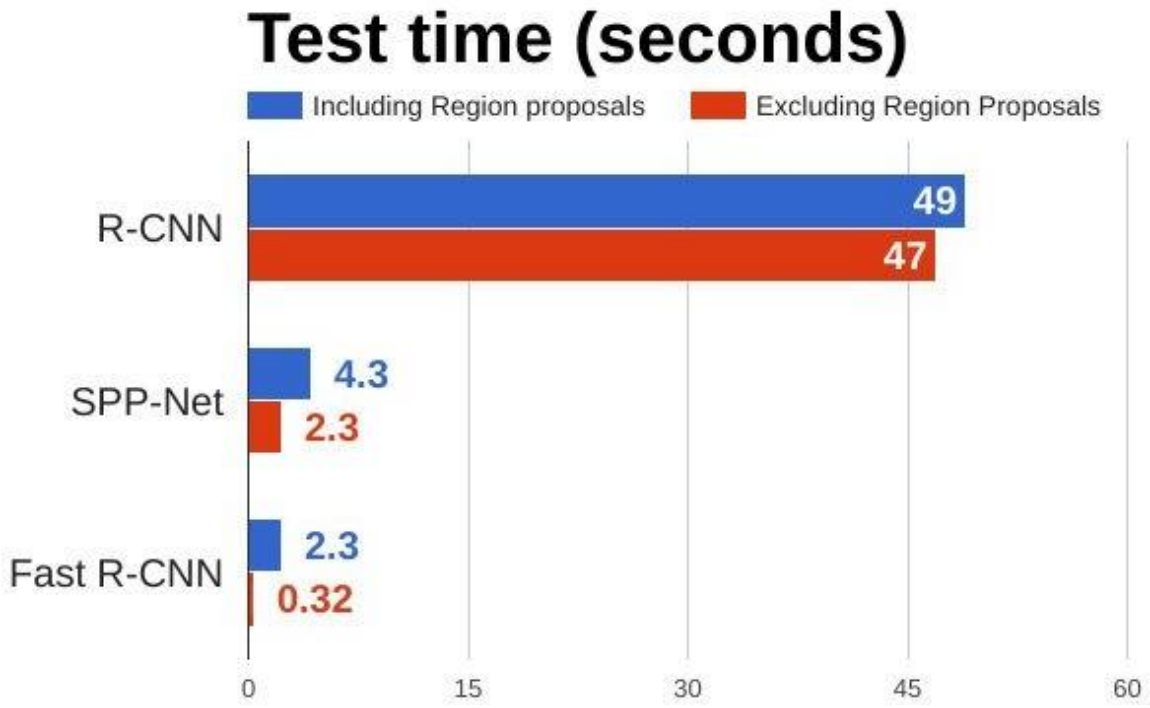
Fast R-CNN (Training)



Girshick, “Fast R-CNN”, ICCV 2015.

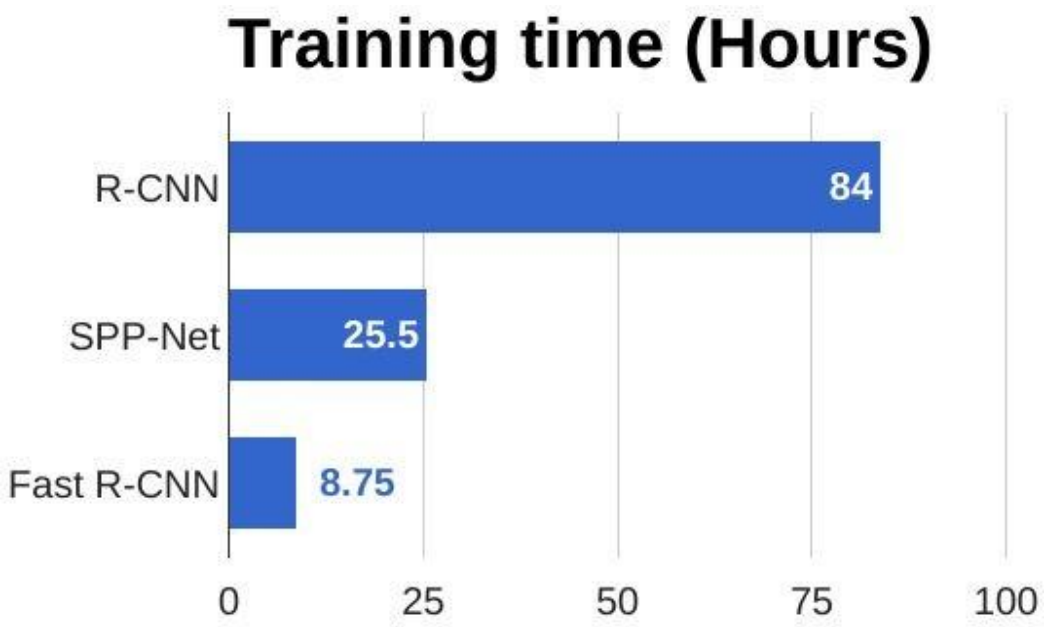
R-CNN vs SPP vs Fast R-CNN

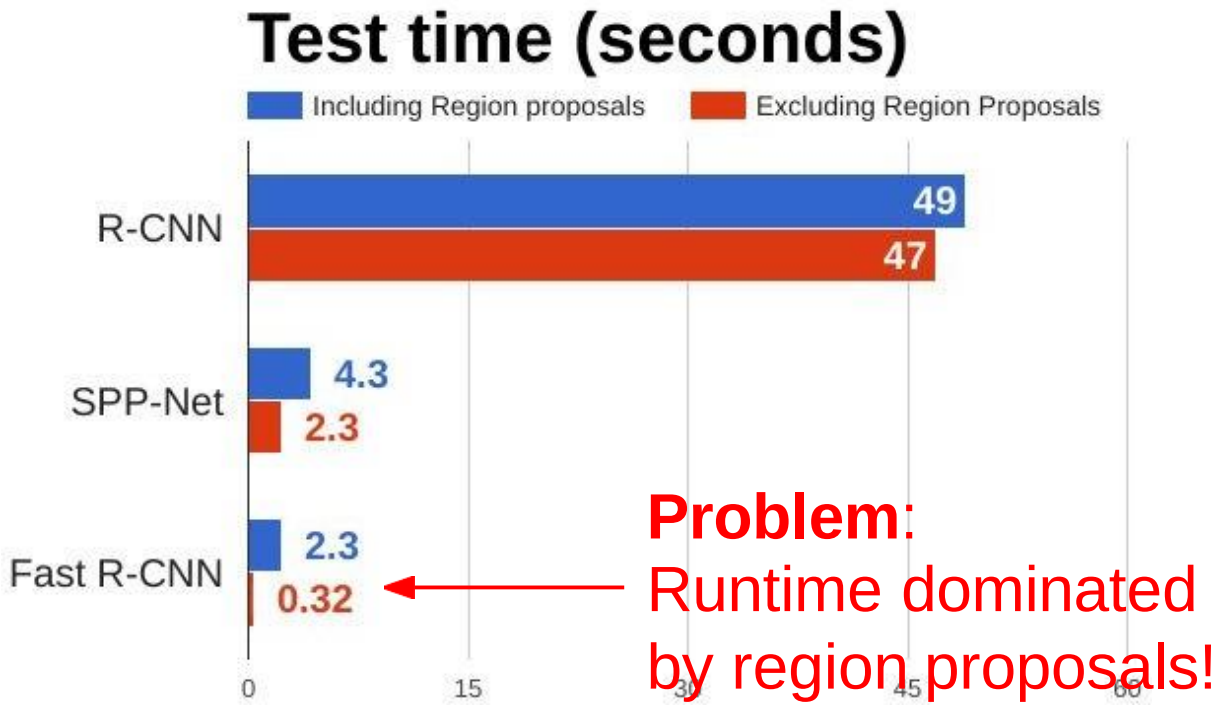




Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
He et al, “Spatial pyramid pooling in deep convolutional networks for visual recognition”, ECCV 2014
Girshick, “Fast R-CNN”, ICCV 2015

R-CNN vs SPP vs Fast R-CNN





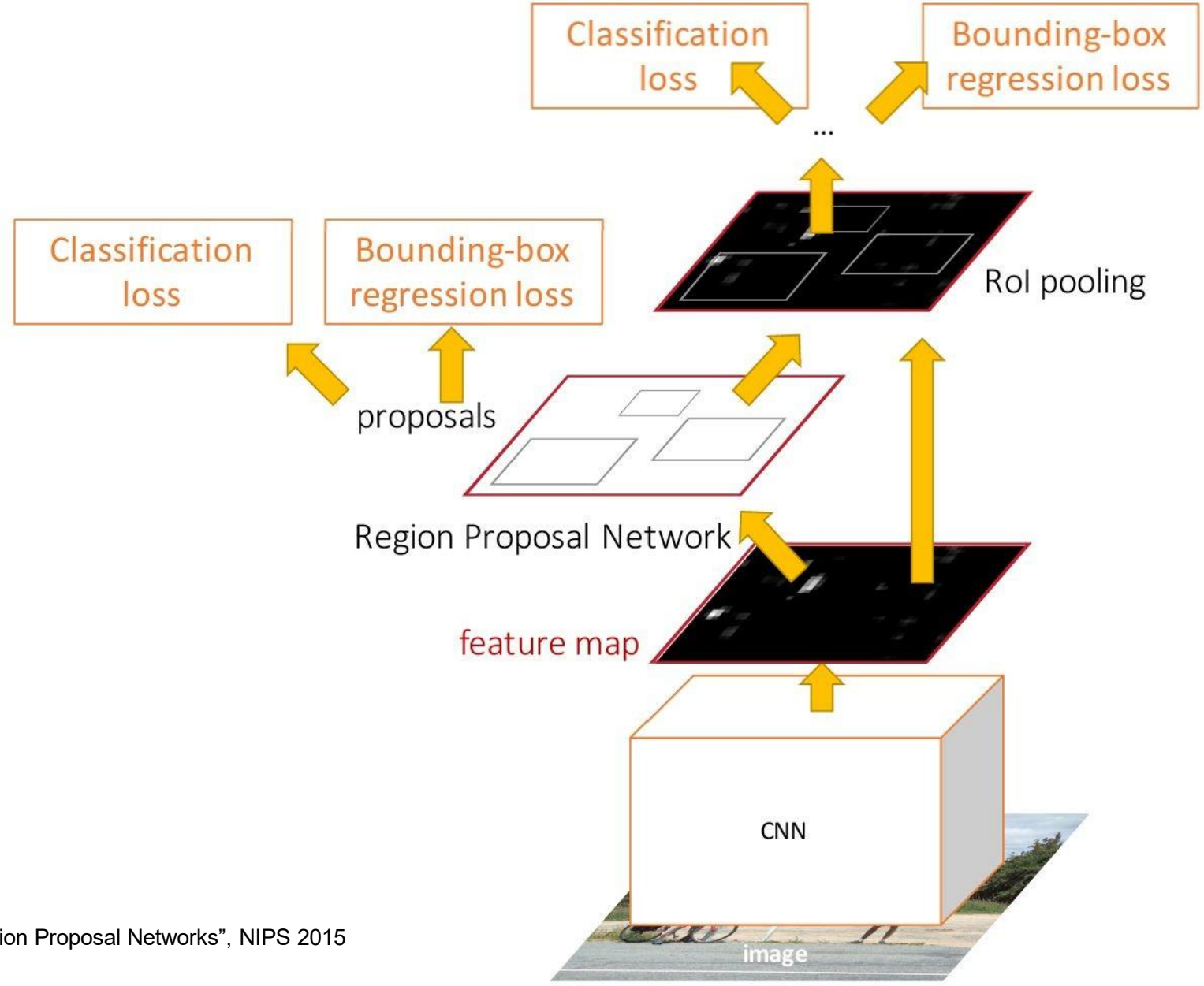
Problem:
Runtime dominated by region proposals!

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
He et al, “Spatial pyramid pooling in deep convolutional networks for visual recognition”, ECCV 2014
Girshick, “Fast R-CNN”, ICCV 2015

Faster R-CNN

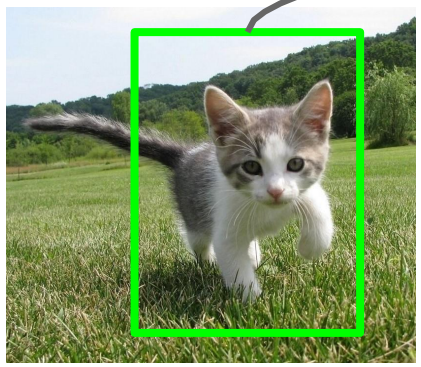
- Make CNN do proposals!
- Insert Region Proposal Network (RPN) to predict proposals from features

- Jointly train with 4 losses:
  - RPN classify object / not object
  - RPN regress box coordinates
  - Final classification score (object classes)
  - Final box coordinates



Ren et al, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission

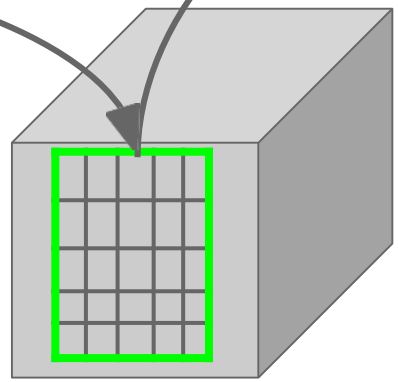
Faster R-CNN: RoI Pooling



Hi-res input image:
3 x 640 x 480
with region
proposal

Project anchor
onto features

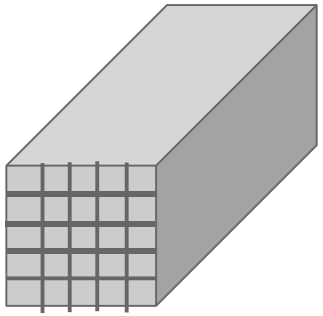
CNN



Hi-res conv features:
512 x 20 x 15;

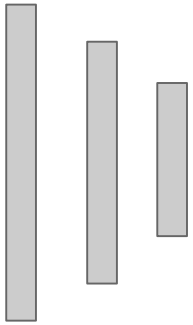
Projected region
proposal is e.g.
512 x 18 x 8
(varies per proposal)

Divide projected
proposal into 7x7
grid, max-pool
within each cell



RoI conv features:
512 x 7 x 7
for region proposal

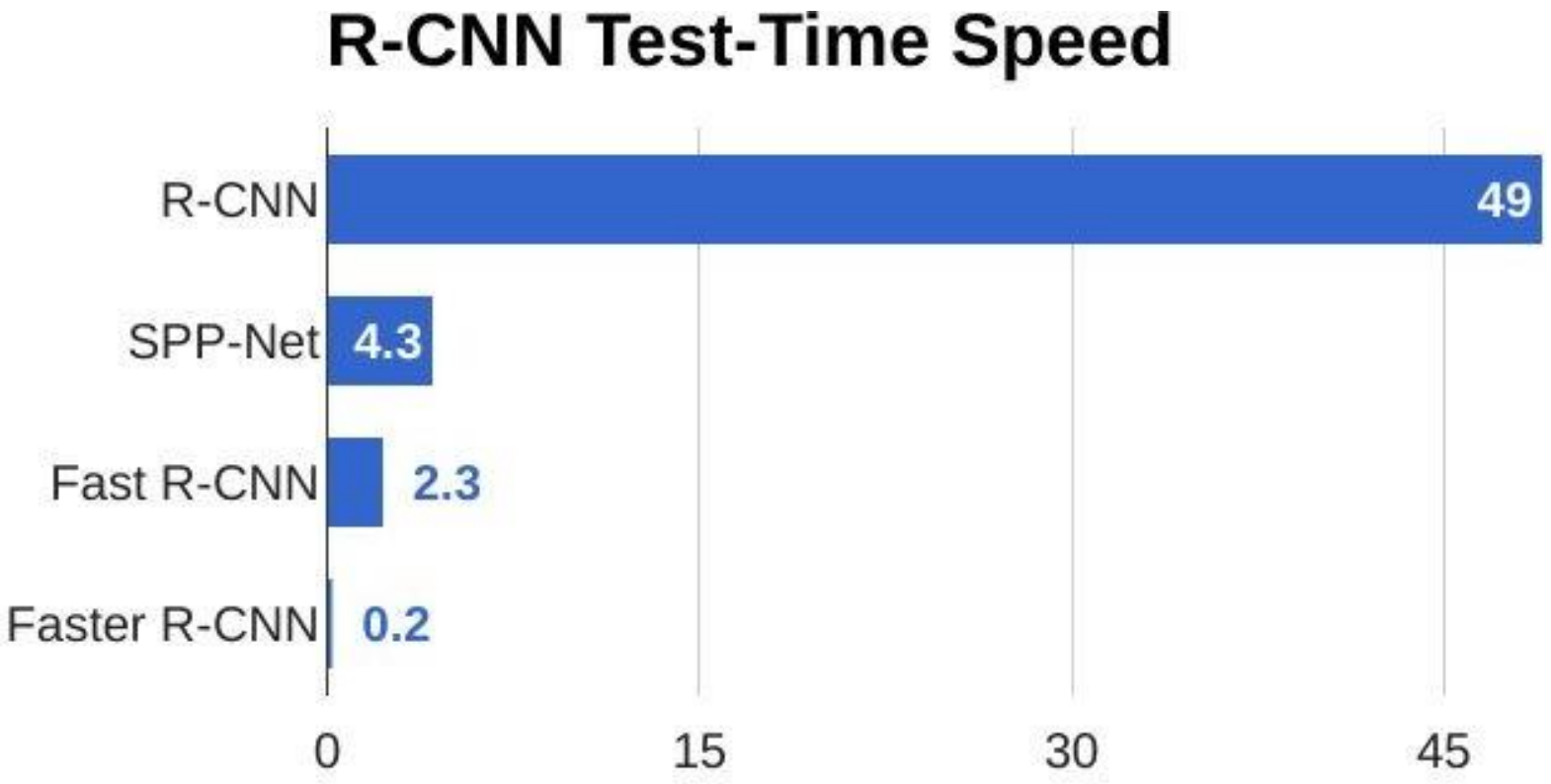
Fully-connected
layers



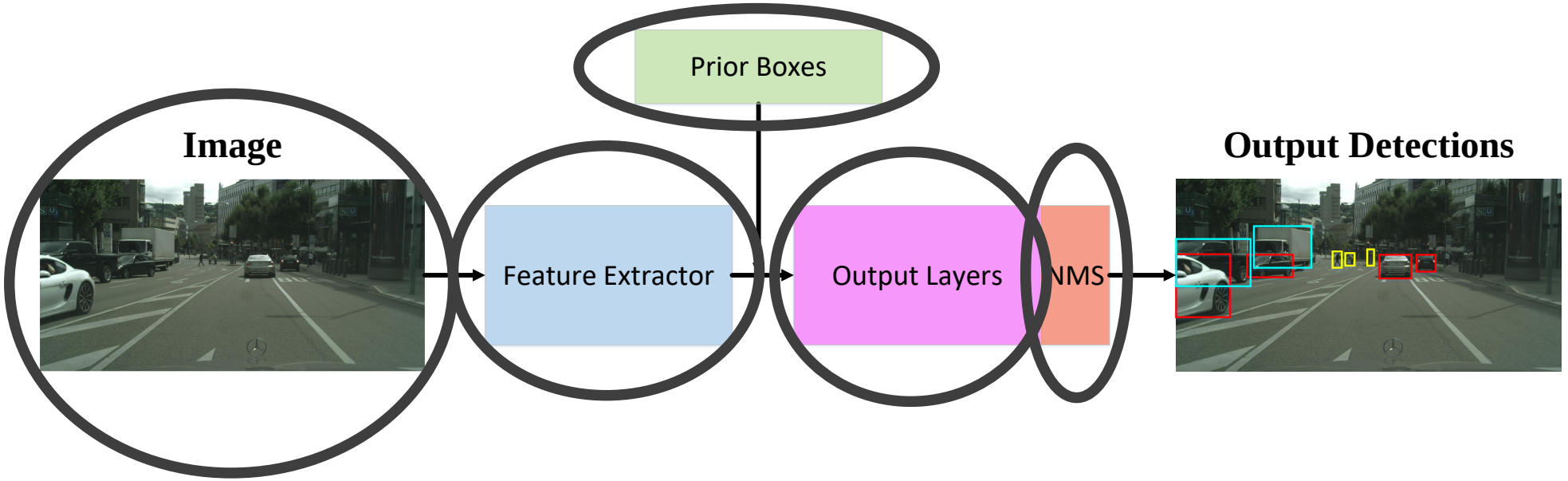
Fully-connected layers expect
low-res conv features:
512 x 7 x 7

Girshick, “Fast R-CNN”, ICCV 2015.

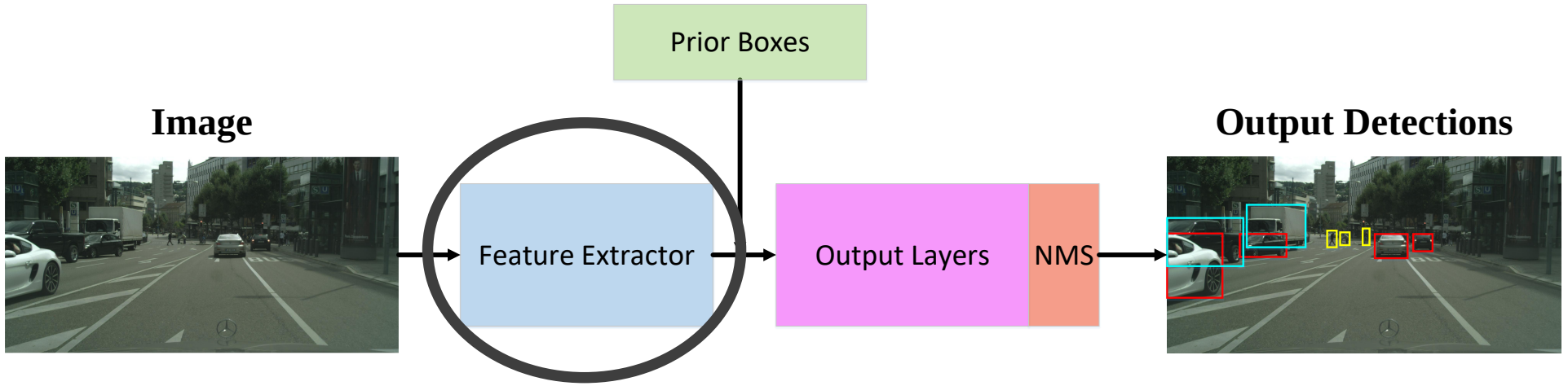
Faster R-CNN:


R-CNN Test-Time Speed
R-CNN
SPP-Net
Fast R-CNN
Faster R-CNN
49
4.3
2.3
0.2
0
15
30
45

Region Proposal Networks For 2D Object Detection



The Feature Extractor

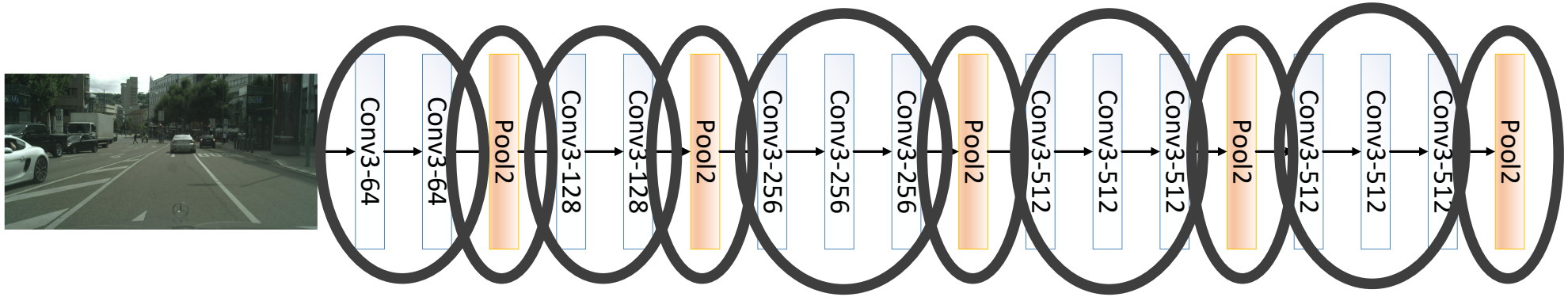


The Feature Extractor

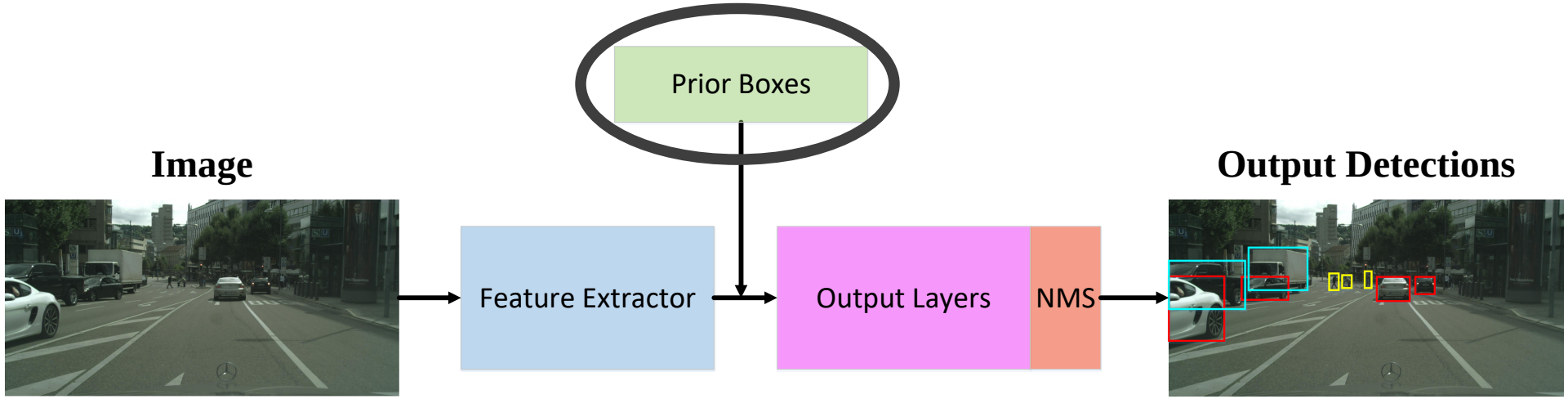
- Feature extractors are the most computationally expensive component of the 2D object detector
- The output of feature extractors usually has much **lower width and height** than those of the input image, but much **greater depth**
- Very active area of research, with new extractors proposed on regular basis
- **Most common extractors are**: VGG, ResNet, and Inception

VGG Feature Extractor

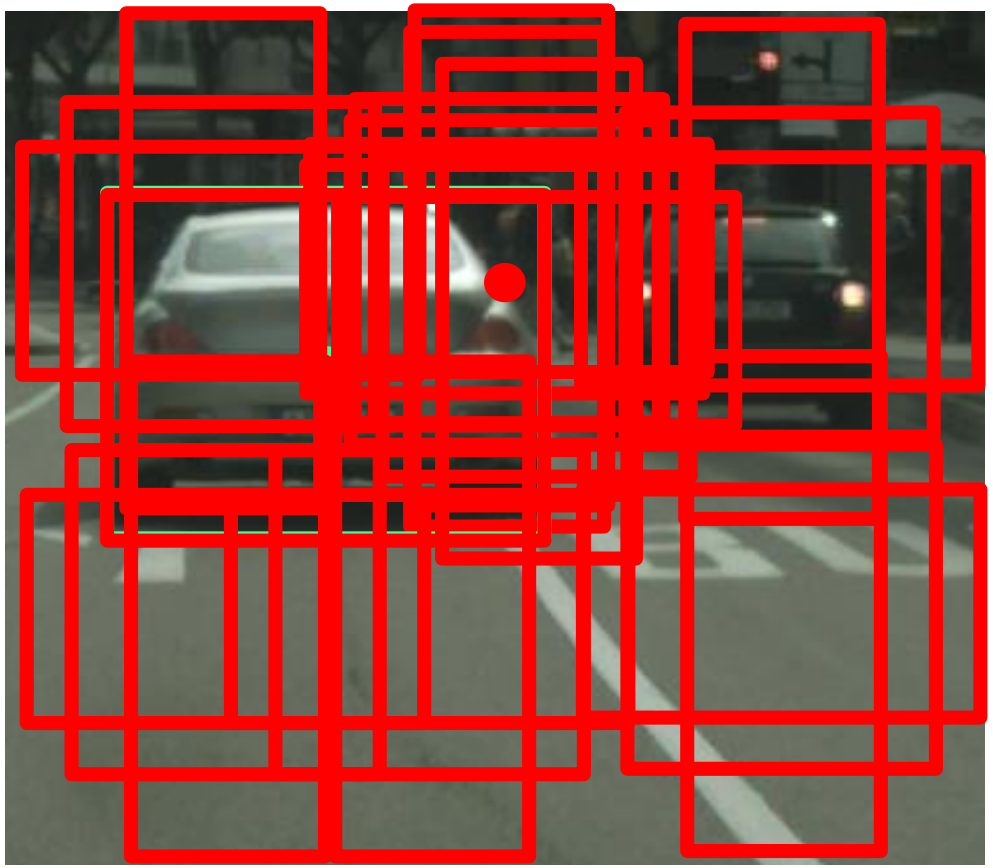
- Alternating convolutional and pooling layers
- All convolutional layers are of size 3x3xK, with stride 1 and 1 zero-padding
- All pooling layers use the **max** function, and are of size 2x2, with stride 2 and no padding



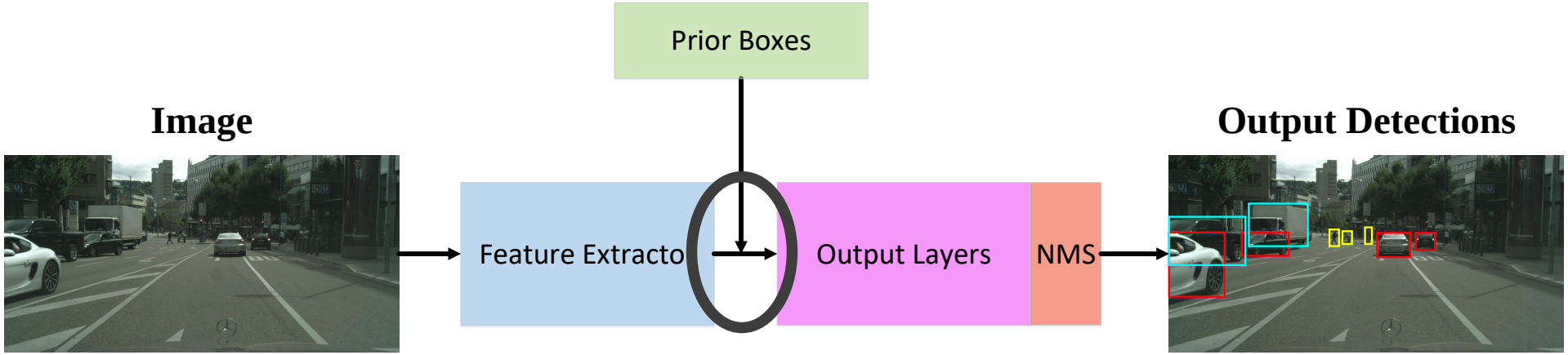
Prior/Anchor Bounding Boxes



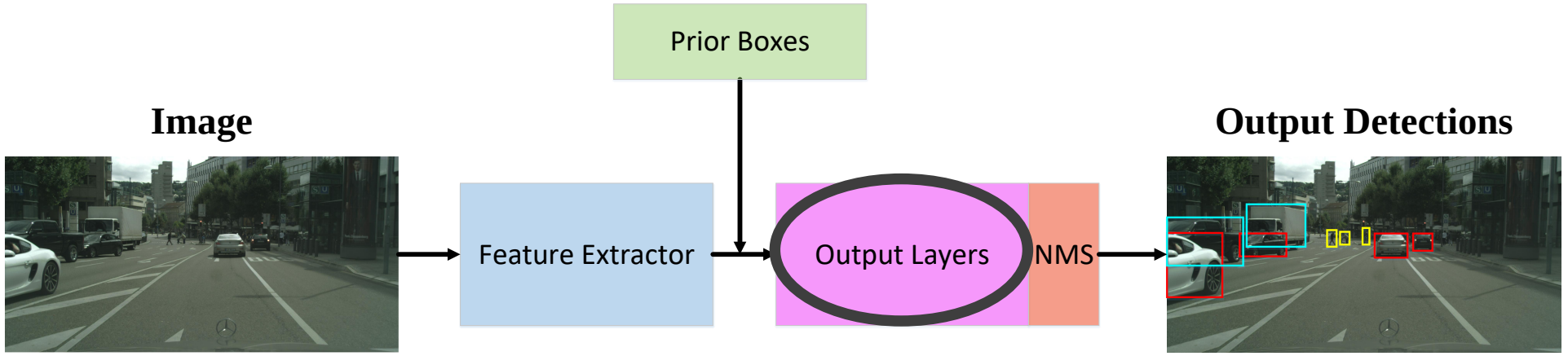
Prior/Anchor Bounding Boxes



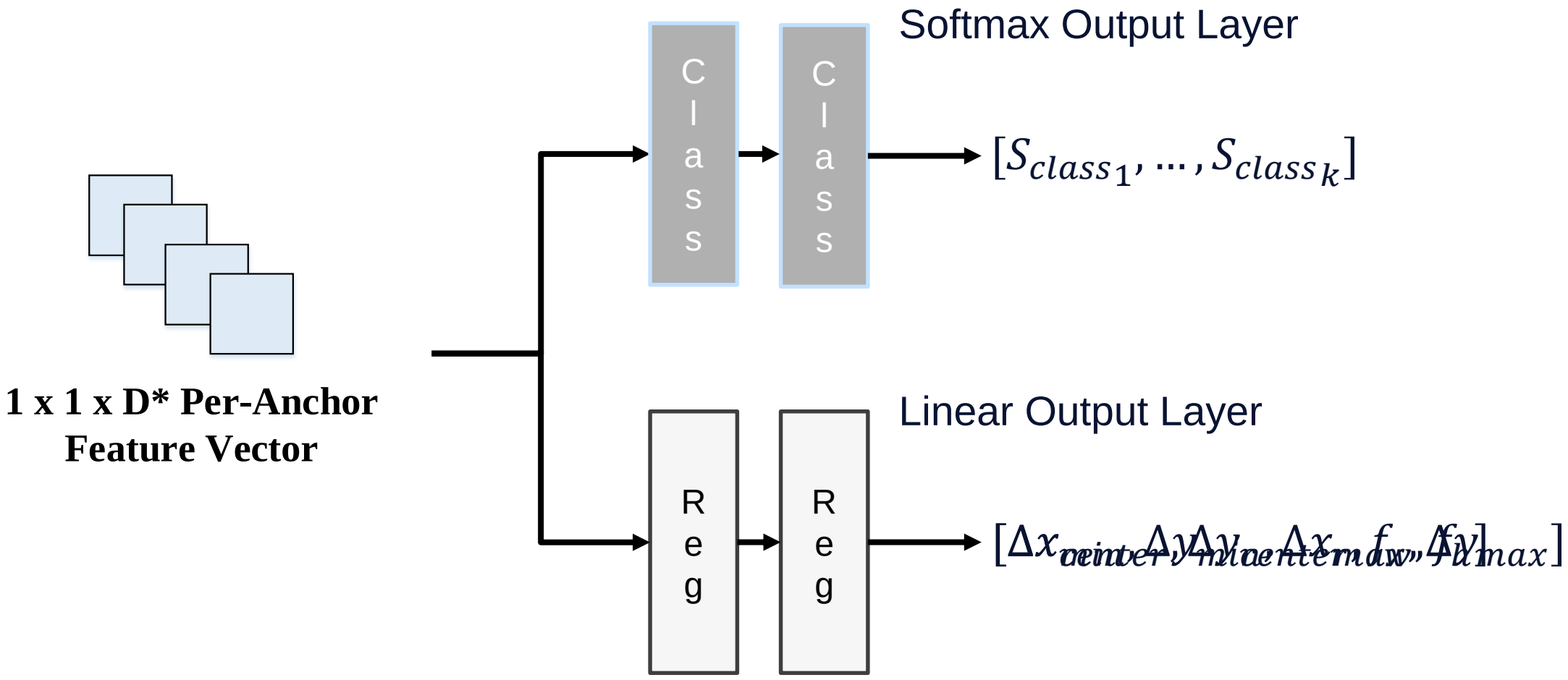
Prior/Anchor Bounding Boxes



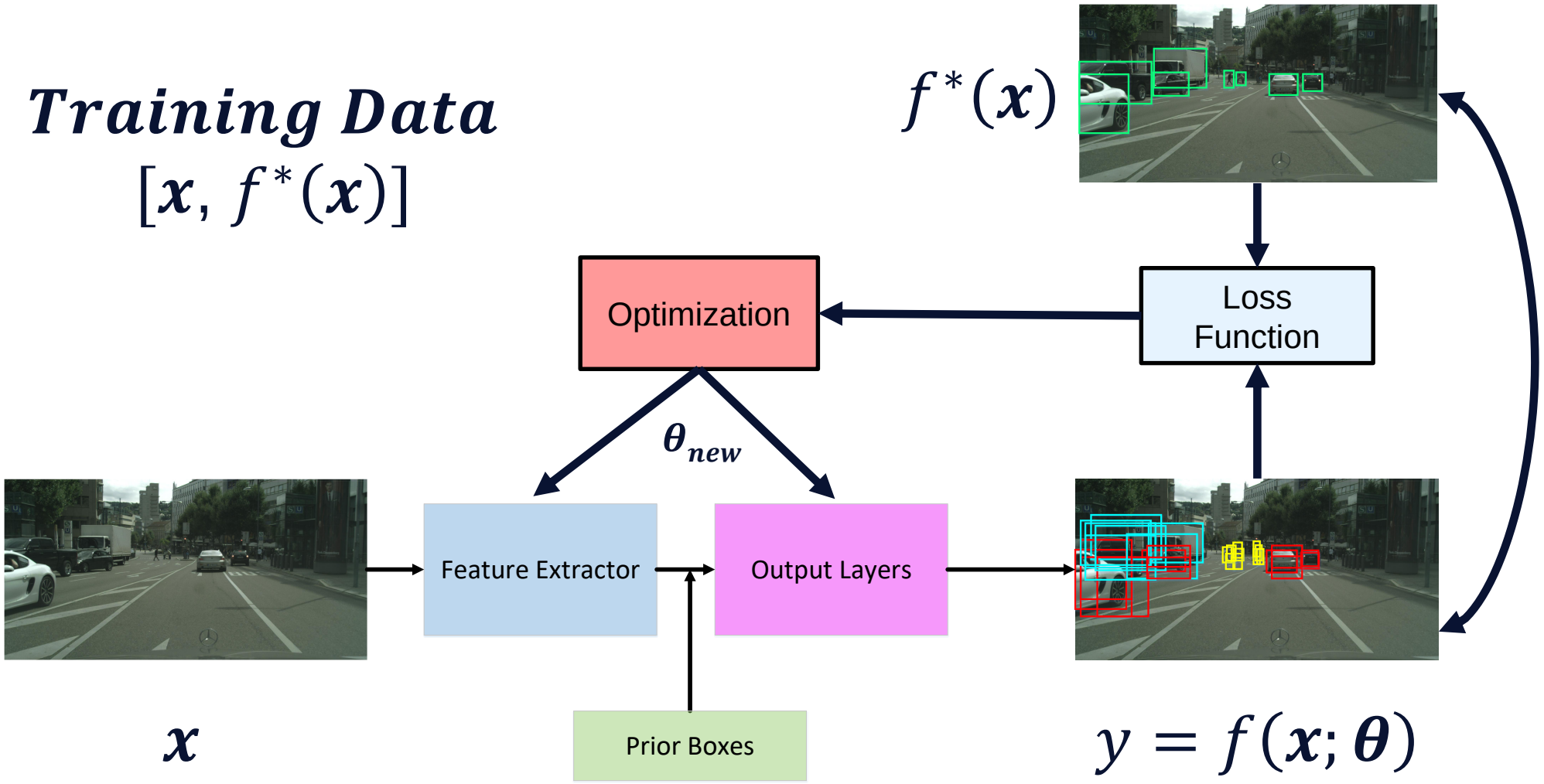
Output Layers



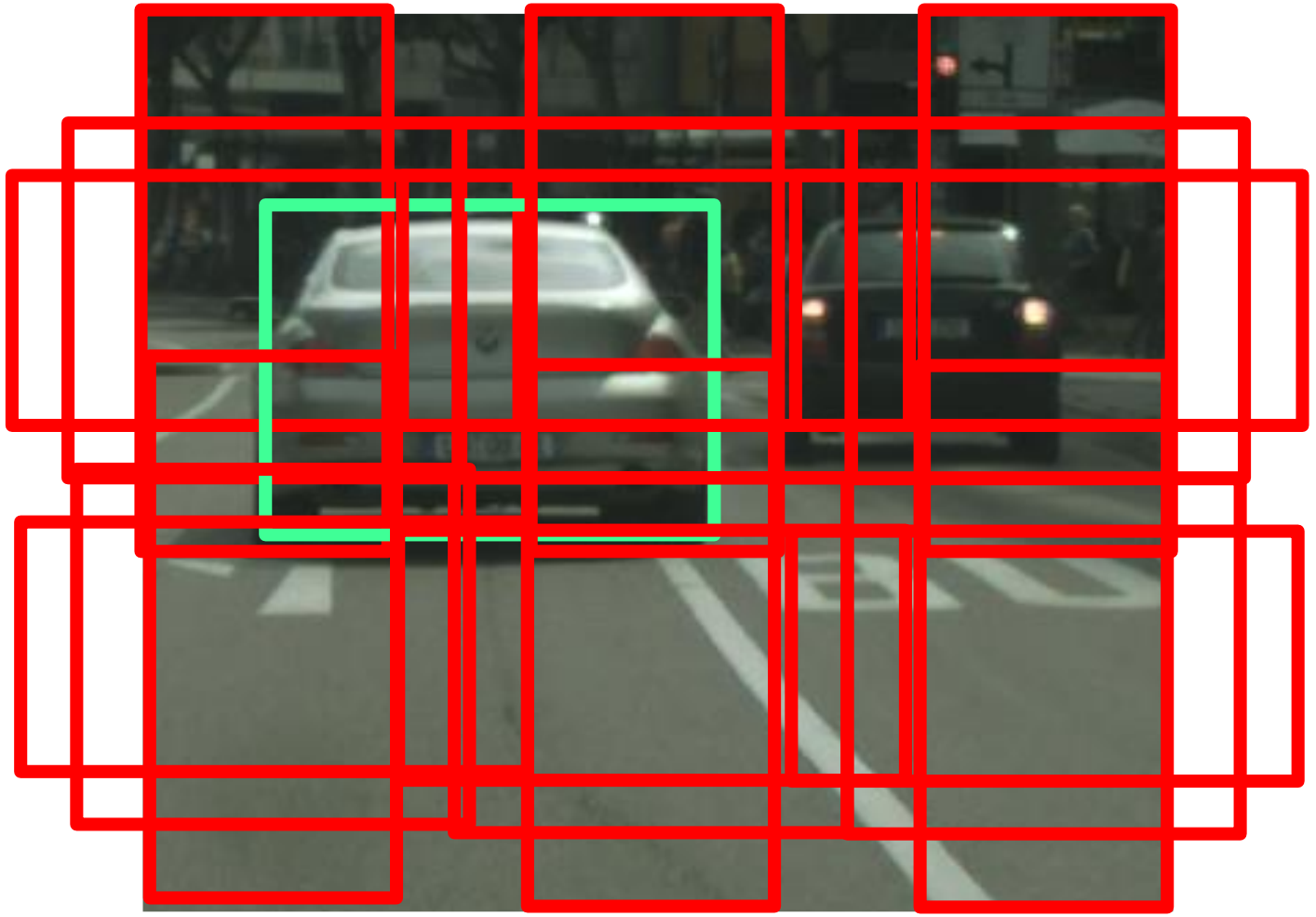
Classification VS Regression Heads



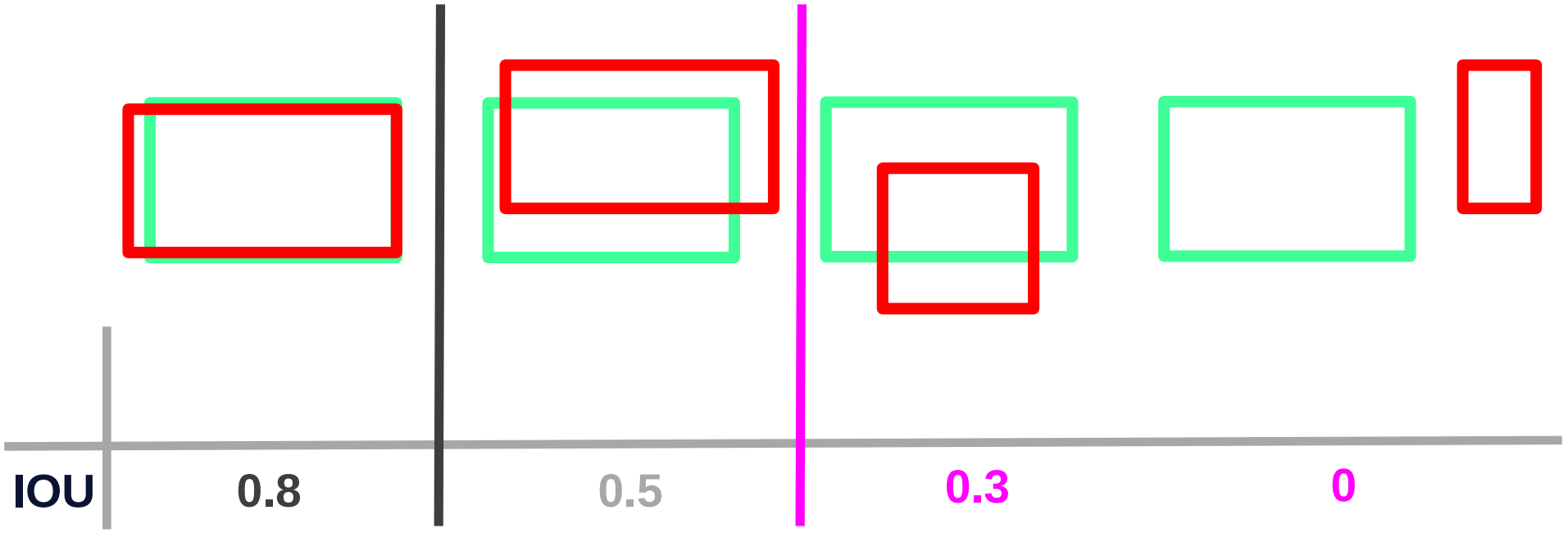
2D Object Detector Training



MiniBatch Selection



MiniBatch Selection



- Negative Member Threshold: < 0.4
- Positive Member Threshold: > 0.6

Minibatch Selection

- Negative anchors target:
  - **Classification:** Background
  - **Regression:** None
- Positive anchors target:
  - **Classification:** Category of the ground truth bounding box
  - **Regression:** Align box parameters with highest IOU ground truth bounding box

Minibatch Selection

- **Problem:** Majority of anchors are negatives - results in neural network that will label all detections as background
- **Solution:** Sample a chosen **minibatch size**, with 3:1 ratio of negative to positive anchors to minimize bias towards the negative class
- Choose negatives with **highest classification loss** (online hard negative mining) to be included in the minibatch

- Example: minibatch size is 64→ 48 hardest negatives and 16 positives

Classification Loss

$$L_{cls} = \frac{1}{N_{total}} \sum_{i} H(s_i^*, s_i)$$

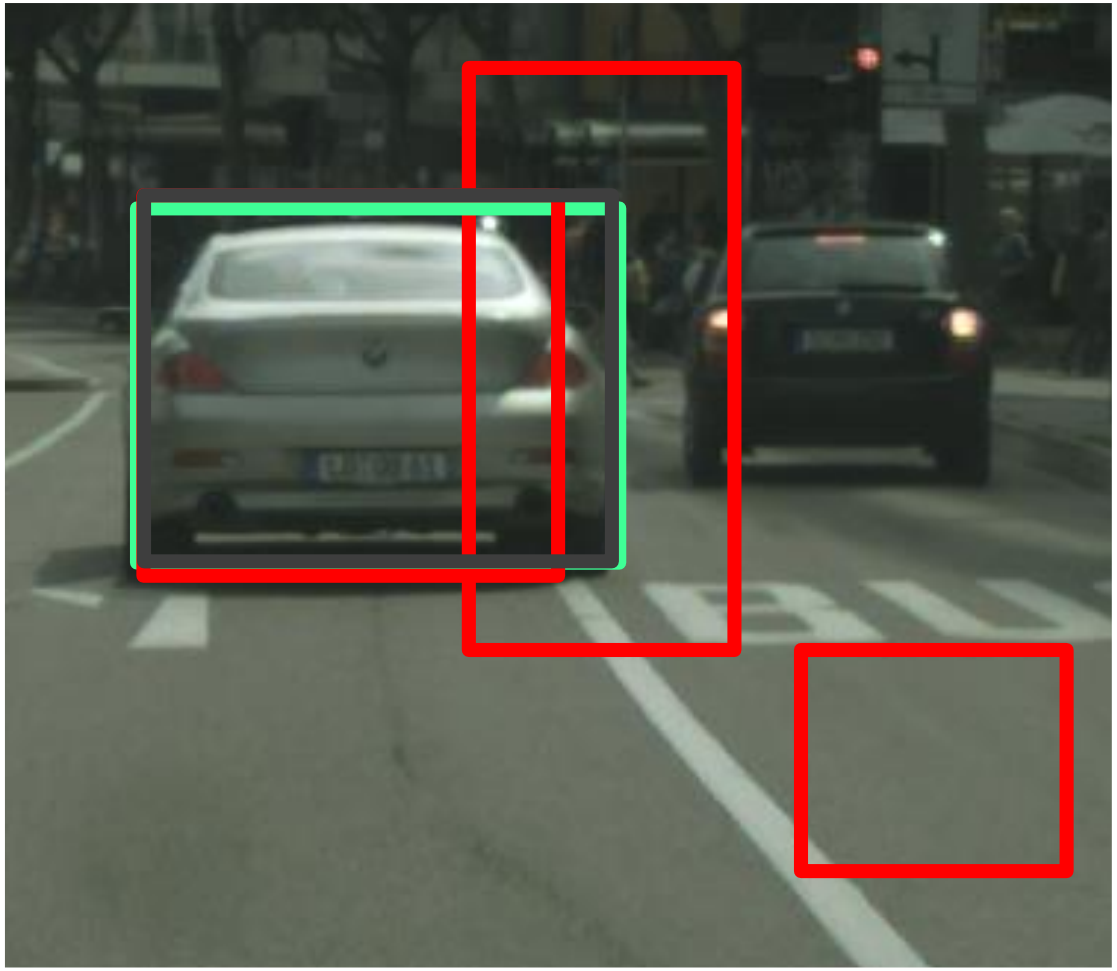
- N_{total} is the size of our minibatch
- s_i is the output of the neural network
- s_i^* is the anchor classification target:
  - **Background** if anchor is negative
  - **Ground truth box class** if anchor is positive

Regression Loss

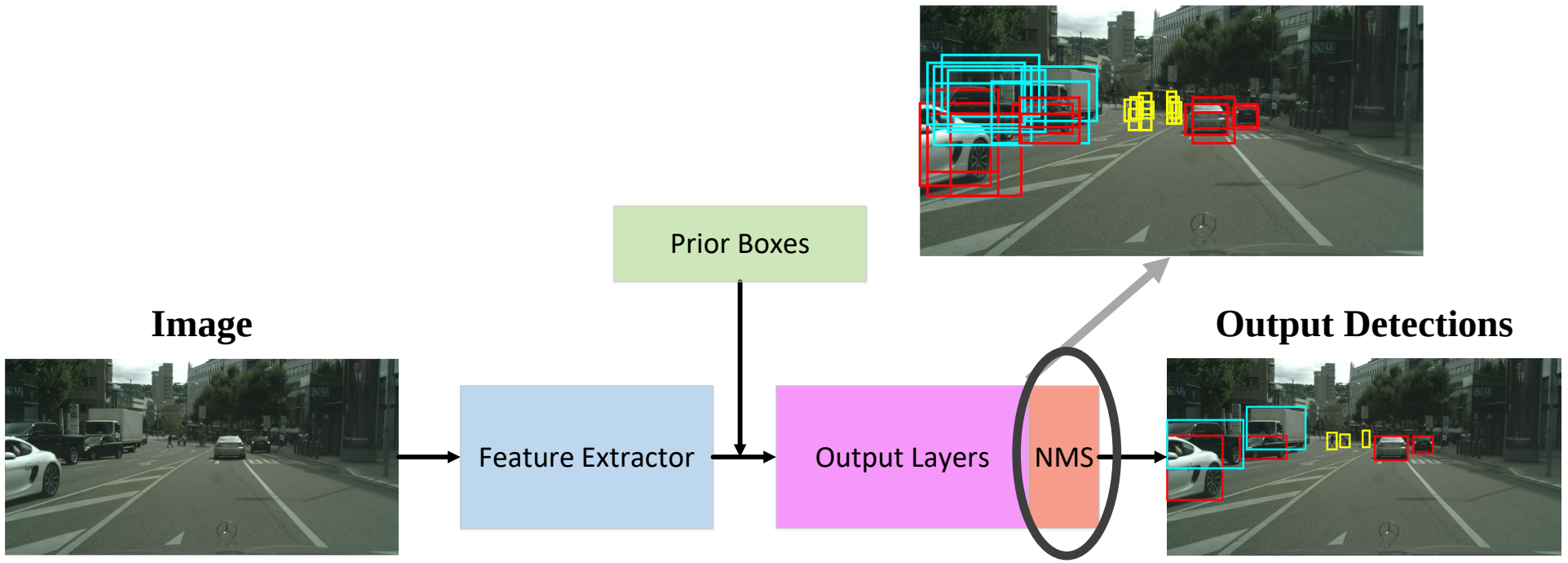
$$L_{reg} = \frac{1}{N_p} \sum_i p_i L_2(b_i^*, b_i)$$

- p_i is 0 if anchor is negative and 1 if anchor is positive
- N_p is the number of positive anchors in the minibatch
- b_i^* is the ground truth bounding box
- b_i is the estimated bounding box, applying the regressed residuals to the anchor box parameters

Visual Representation Of Training



Inference Time

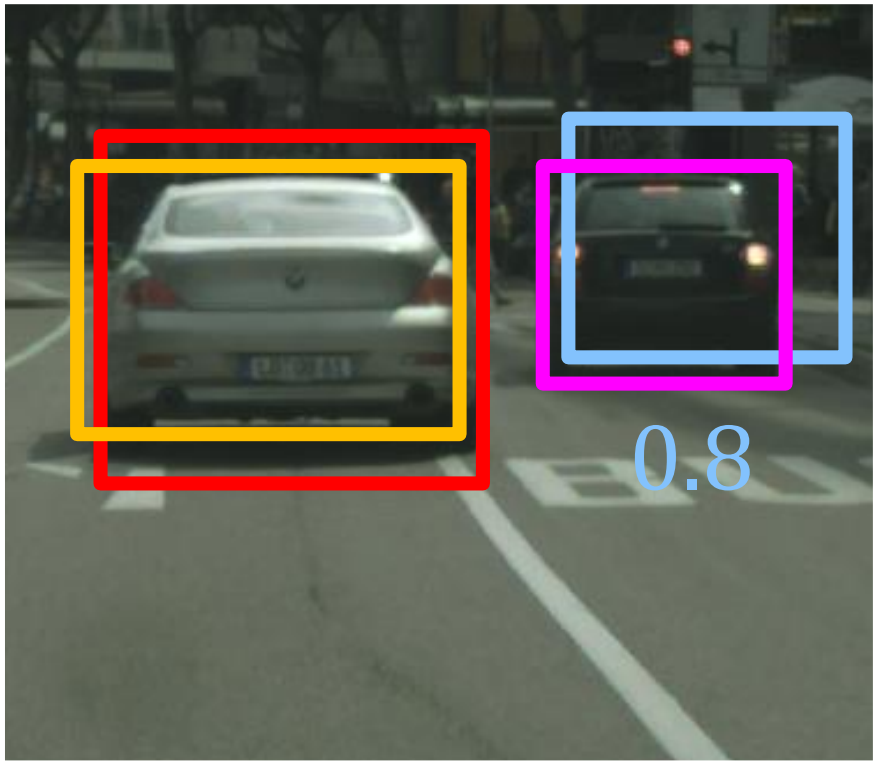


Non-Maximum Suppression

Input:
$B = \{B_1 ... B_n\} | B_i = (x_i, y_i, w_i, h_i, s_i) \forall i \in [1, n]$
IOU threshold η
begin
    $\bar{B} = Sort(B, s, \downarrow)$
    $D = \emptyset$
    for $b \in \bar{B}$ *and* $\bar{B}$ *not* $\emptyset$ **do**
        $b_{max} = b$
        $\bar{B} \leftarrow \bar{B} \backslash b_{max}$
        $D \leftarrow D \cup b_{max}$
        for $b_i \in \bar{B} \backslash b_{max}$ **do**
            if $IOU(b_{max}, b_i) \geq \eta$ **then**
                $\bar{B} \leftarrow \bar{B} \backslash b_i$
            end
        end
    end
    Output: D
end

Non-Maximum Suppression

$\geq \quad \eta = 0.7$



$$B = \{B_1, B_2, B_3, B_4\}$$
$$\overline{B} = \{B_1, B_2, B_3, B_4\}$$
$$S = \{0.98, 0.94, 0.6, 0.45\}$$
$$D = \{\ \}$$

$$b_{max} = \{B_1\}$$

Non-Maximum Suppression

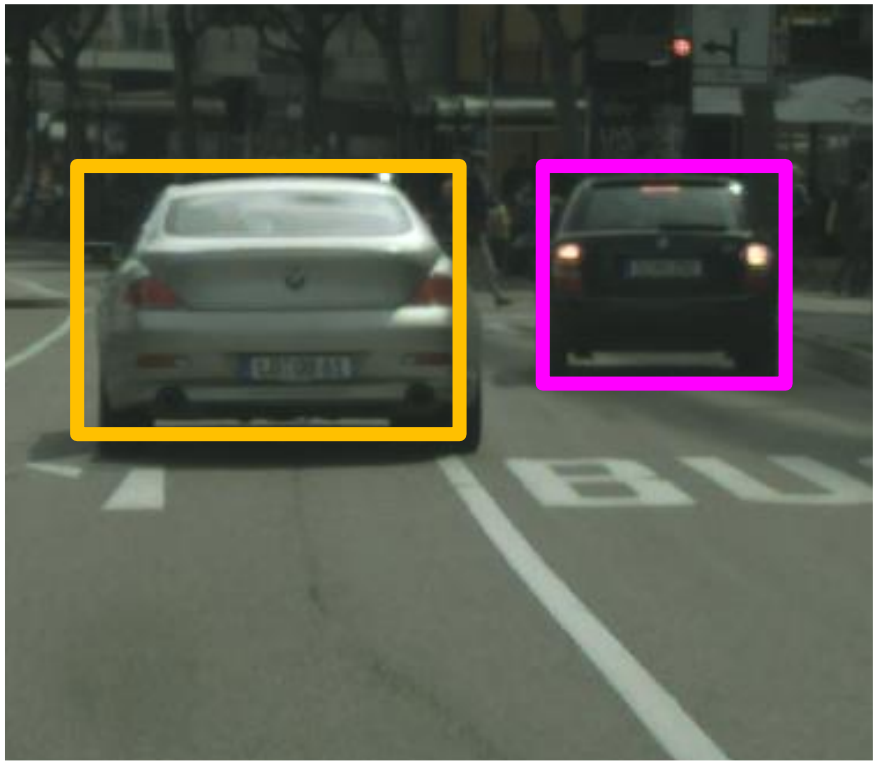
$\geq \eta = 0.7$

$$B = \{B_1, B_2, B_3, B_4\}$$
$$\overline{B} = \{B_2, B_4\}$$
$$S = \{0.94, 0.45\}$$
$$D = \{B_1\}$$

$$b_{max} = \{B_2\}$$

Non-Maximum Suppression

$\eta = 0.7$



$\boldsymbol{B} = \{B_1, B_2, B_3, B_4\}$

$\overline{\boldsymbol{B}} = \{\}$

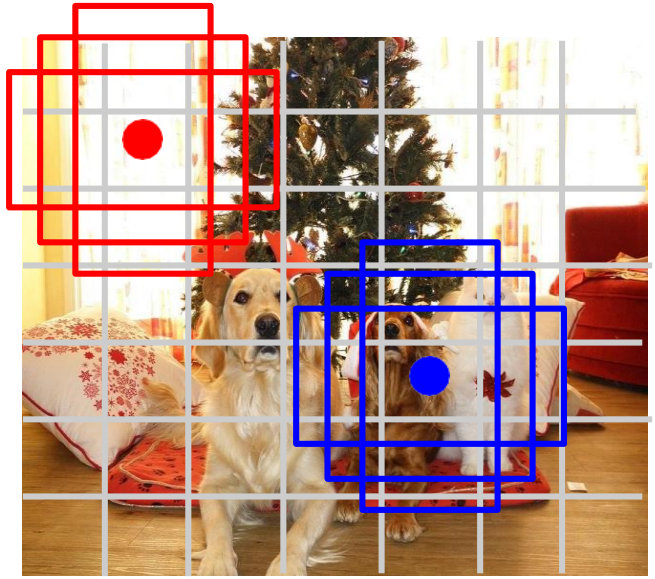
$\boldsymbol{S} = \{\}$

$\boldsymbol{D} = \{B_1, B_2\}$

$b_{max} = \{B_2\}$

Detection without Proposals: YOLO / SSD

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
Here B = 3

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)

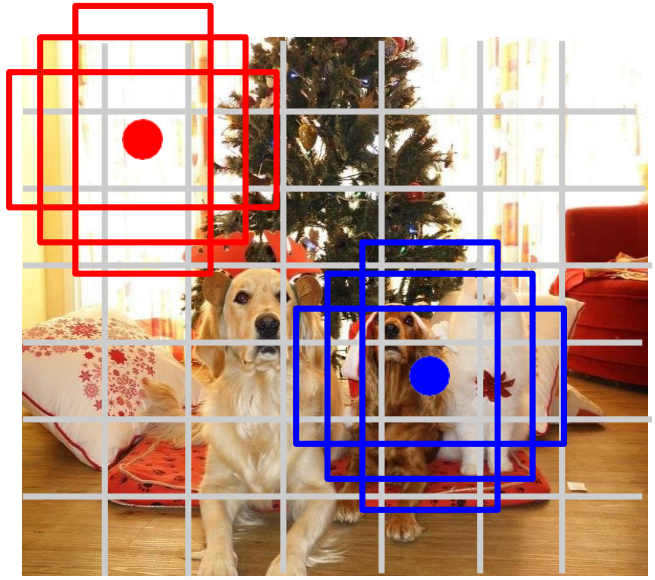
Output:
7 x 7 x (5 * B + C)

Redmon et al, “You Only Look Once: Unified, Real-Time Object Detection”, CVPR 2016
Liu et al, “SSD: Single-Shot MultiBox Detector”, ECCV 2016

Detection without Proposals: YOLO / SSD

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
Here B = 3

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)

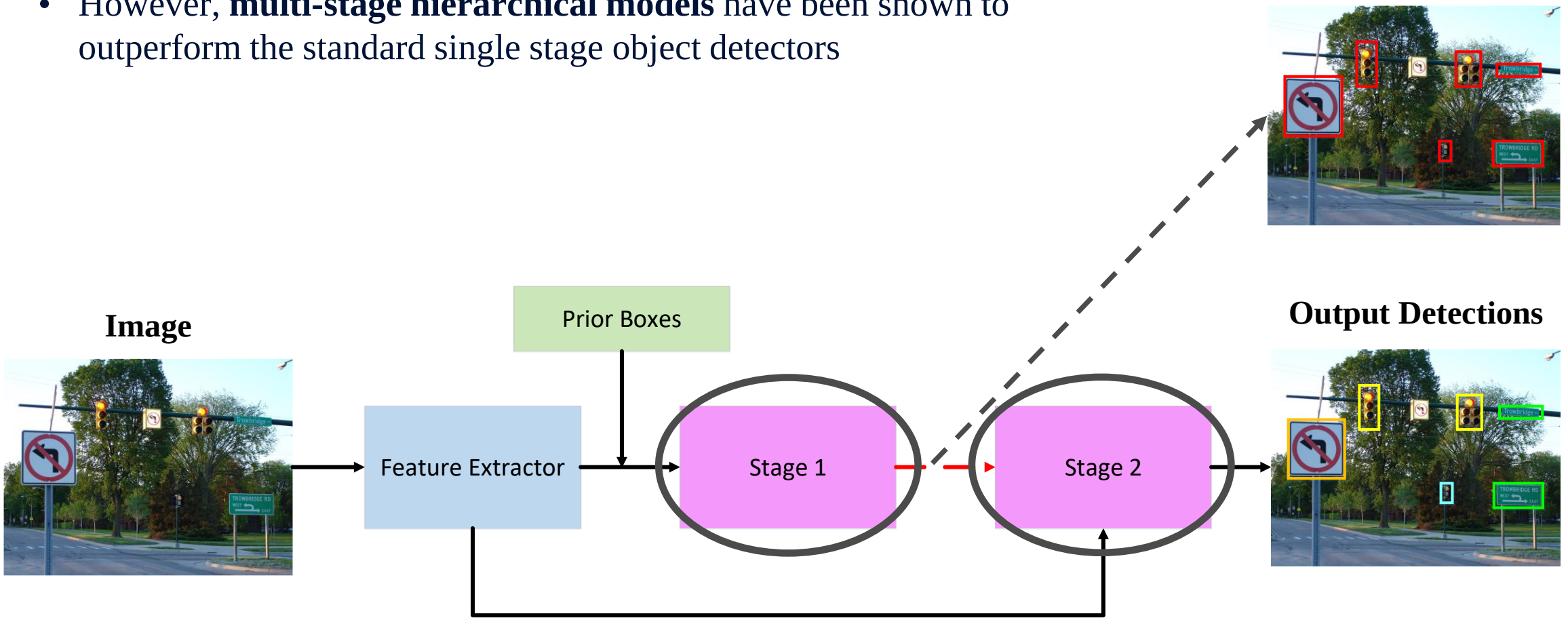
Redmon et al, “You Only Look Once: Unified, Real-Time Object Detection”, CVPR 2016
Liu et al, “SSD: Single-Shot MultiBox Detector”, ECCV 2016

Traffic Sign and Traffic Signal Detection

- Traffic signs and signals appear smaller in size compared to cars, two-wheelers, and pedestrians.
- Traffic signs are highly variable with many classes to be trained on.
- Traffic signals have different states that are required to be detected.
- In addition, traffic signals change state as the car drives!

Traffic sign and signal detection

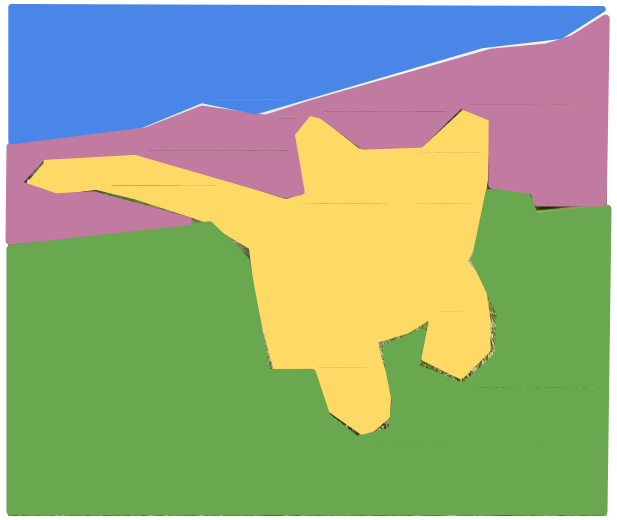
- 2D object detectors can be used to perform traffic sign and traffic signal detection without any modifications
- However, **multi-stage hierarchical models** have been shown to outperform the standard single stage object detectors



2D Object Detection Examples

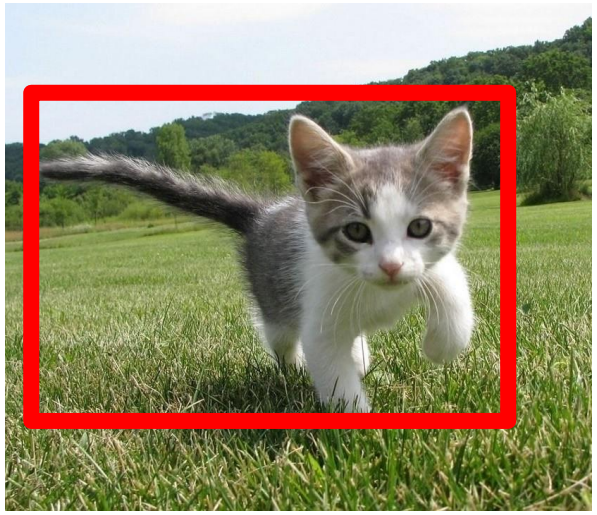
- Yolo/SSD/Faster-RCNN in the wild

Instance Segmentation



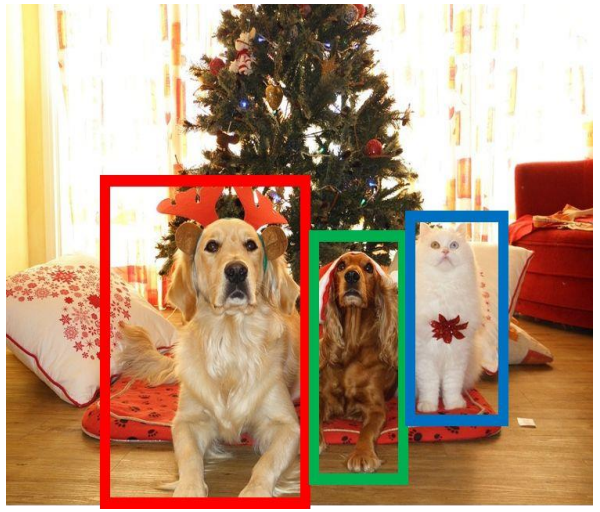
GRASS, CAT, TREE, SKY

No objects, just pixels

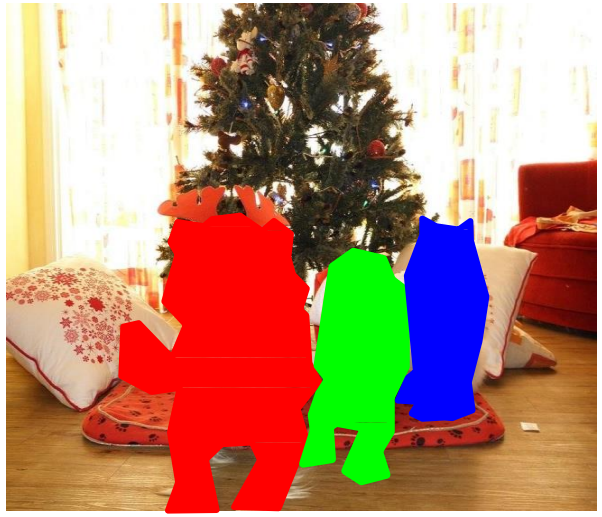


CAT

Single Object



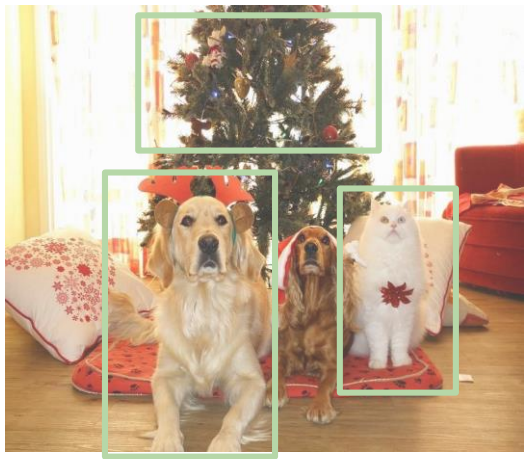
DOG, DOG, CAT



DOG, DOG, CAT

Multiple Object

Mask R-CNN



CNN

RoI Align

Classification Scores: C
Box coordinates (per class): 4 * C

Conv

Conv

256 x 14 x 14

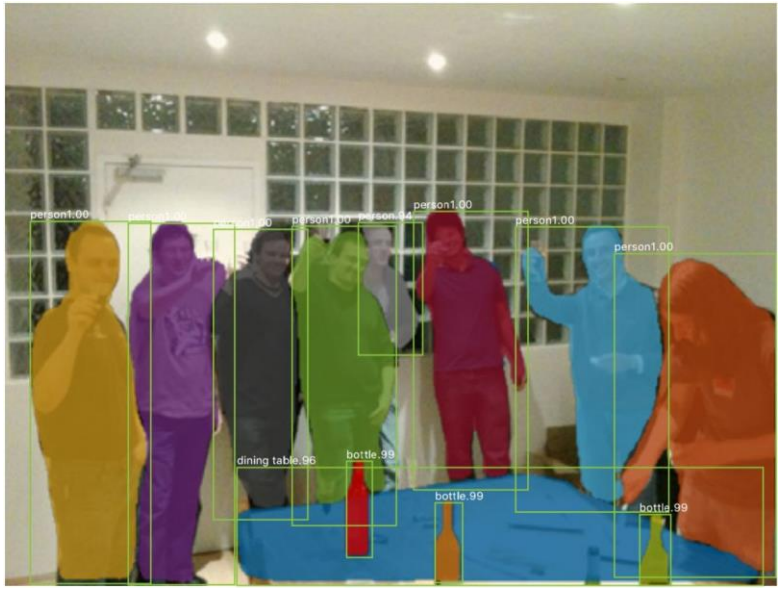
256 x 14 x 14

Predict a mask for each of C classes

C x 14 x 14

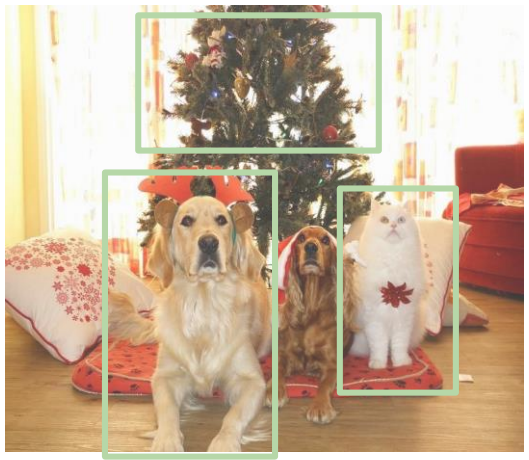
He et al, “Mask R-CNN”, arXiv 2017

Mask R-CNN: Very Good Results!



He et al, “Mask R-CNN”, arXiv 2017

Mask R-CNN Also does pose



CNN

RoI Align

256 x 14 x 14

Conv

256 x 14 x 14

Conv

Classification Scores: C

Box coordinates (per class): 4 * C

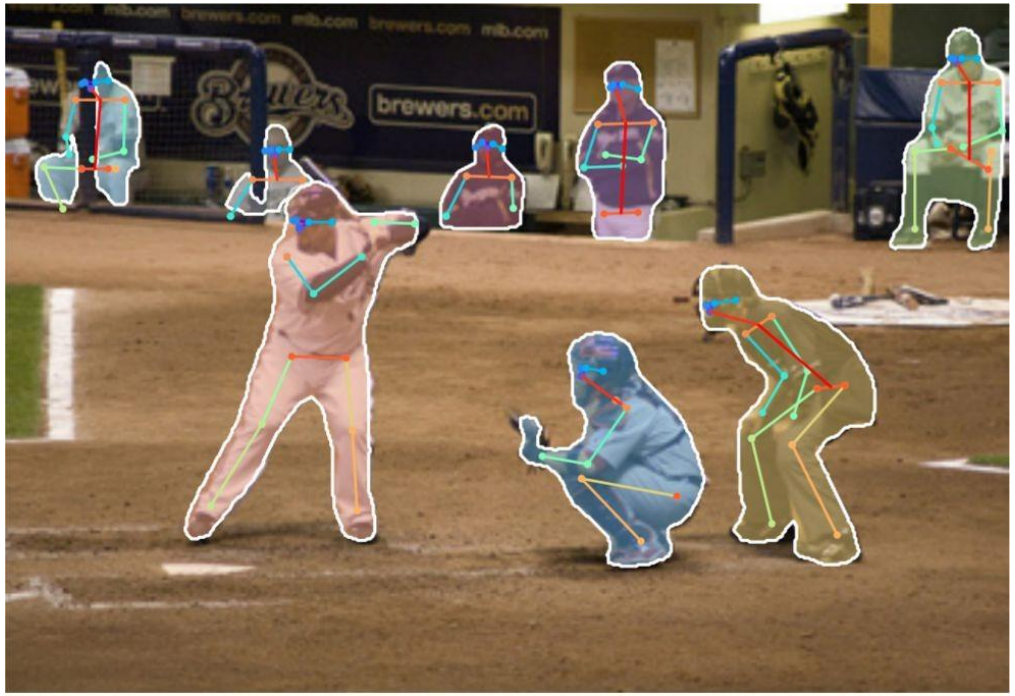
Joint coordinates

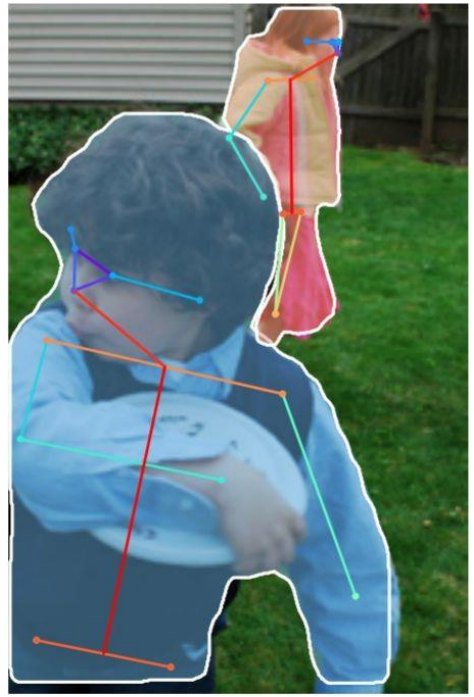
Predict a mask for each of C classes

C x 14 x 14

He et al, “Mask R-CNN”, arXiv 2017

Mask R-CNN Also does pose





He et al, “Mask R-CNN”, arXiv 2017