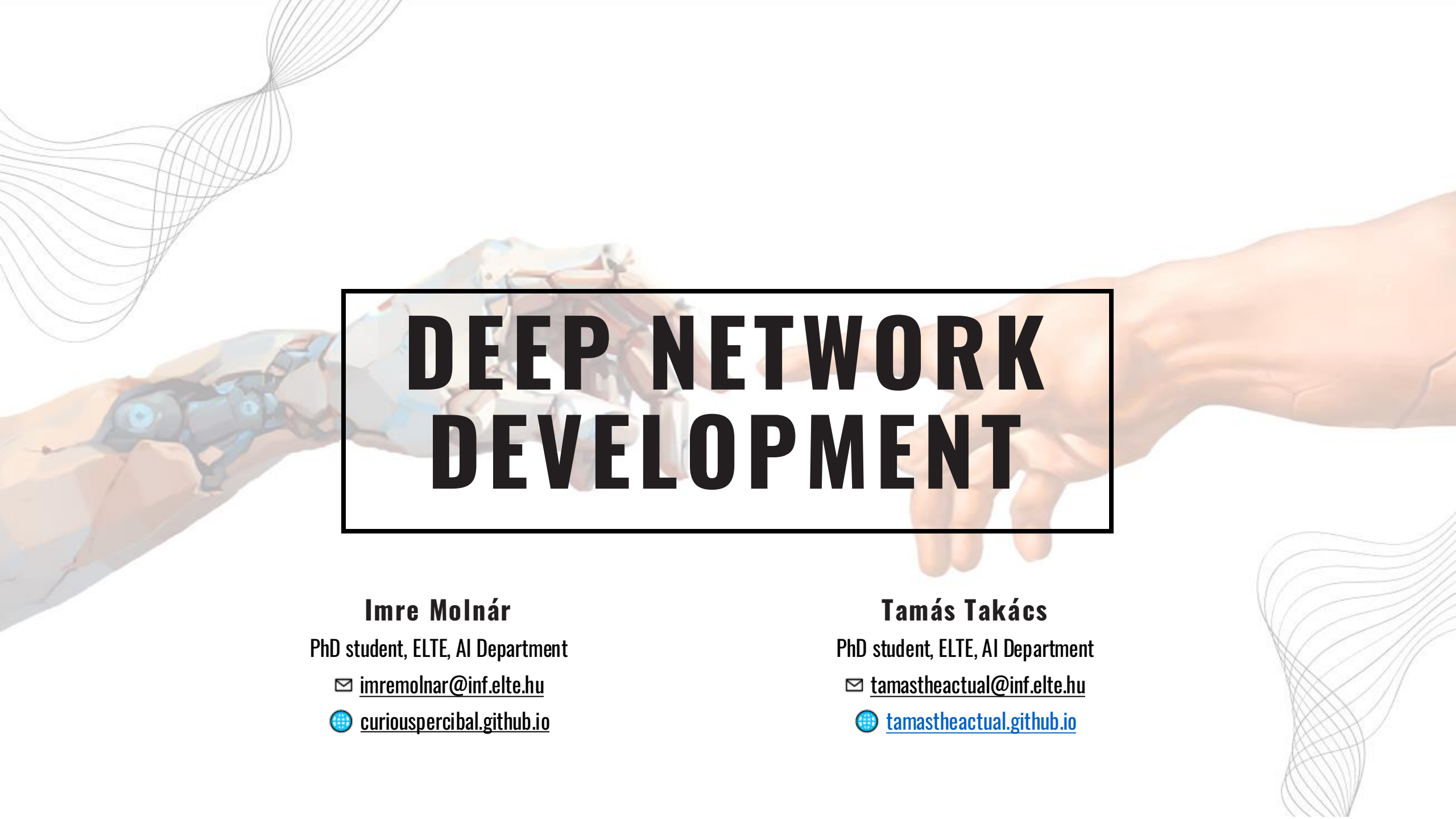


DEEP NETWORK DEVELOPMENT

Imre Molnár

PhD student, ELTE, AI Department

imremolnar@inf.elte.hu

curiouspercibal.github.io

Tamás Takács

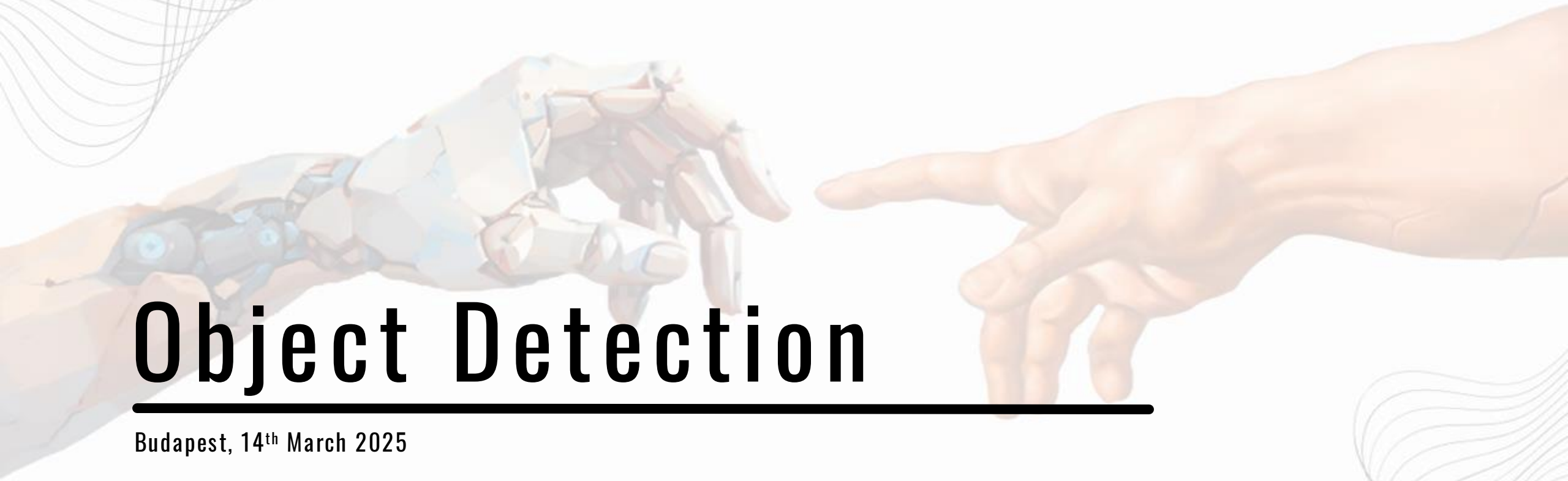
PhD student, ELTE, AI Department

tamastheactual@inf.elte.hu

tamastheactual.github.io

Deep Network Development

Lecture 5.



Object Detection

Budapest, 14th March 2025

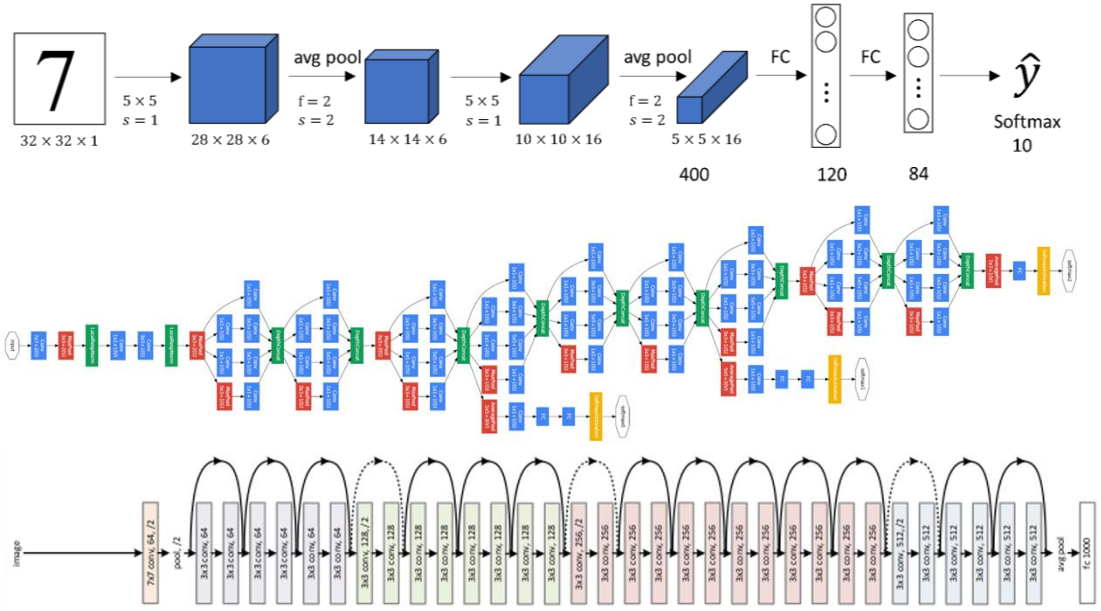
1 Two Stage Detectors

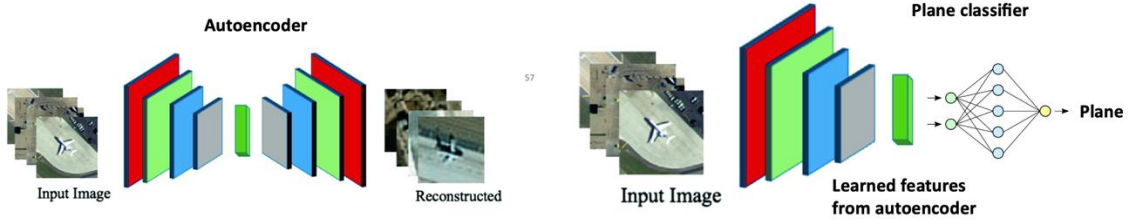
2 One Stage Detectors

3 Object Detection Metrics

Previously on Lecture 4

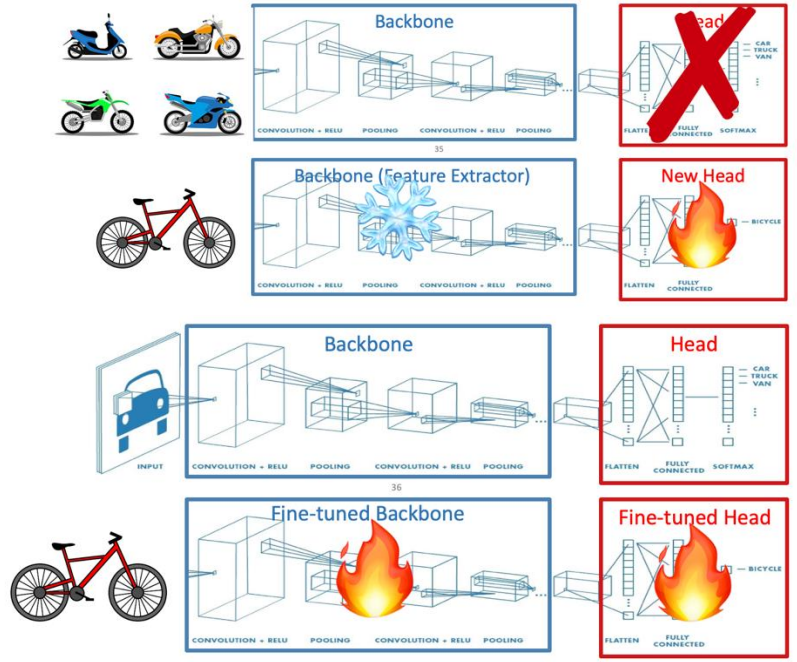
CNN architectures



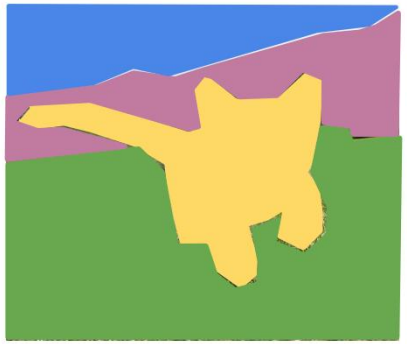
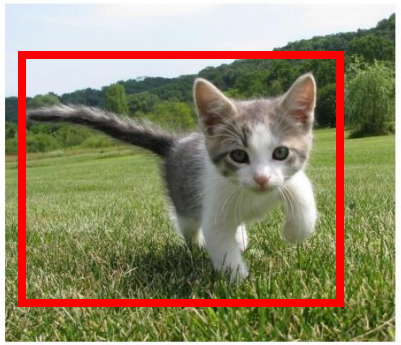
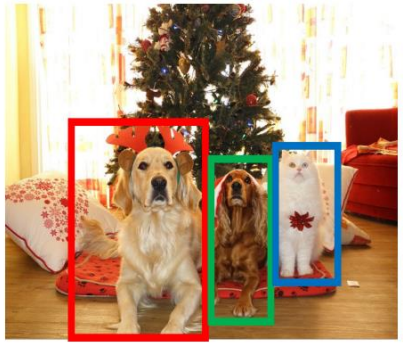


When to use Transfer Learning

- Task A and B have the same input x
- When you have a lot of data for the problem you are transferring from (A) and few data for the problem you are transferring to (B)
- Low level features from A could be helpful for learning B
- Faster training. Use pre-trained weights as initialization point whether than randomly initializing weights



Supervised Learning tasks

Classification	Semantic Segmentation	Classification + Localization	Object Detection	Instance Segmentation
				
CAT	GRASS, CAT, TREE, SKY	CAT	DOG, DOG, CAT	DOG, DOG, CAT
Single Object	No objects, just pixels	Single Object	Multiple Objects	Multiple Objects
				

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

What is Object Detection?

Classification
[cat, dog, car]

[0.9, 0.05, 0.05]

[0.7, 0.21, 0.09]

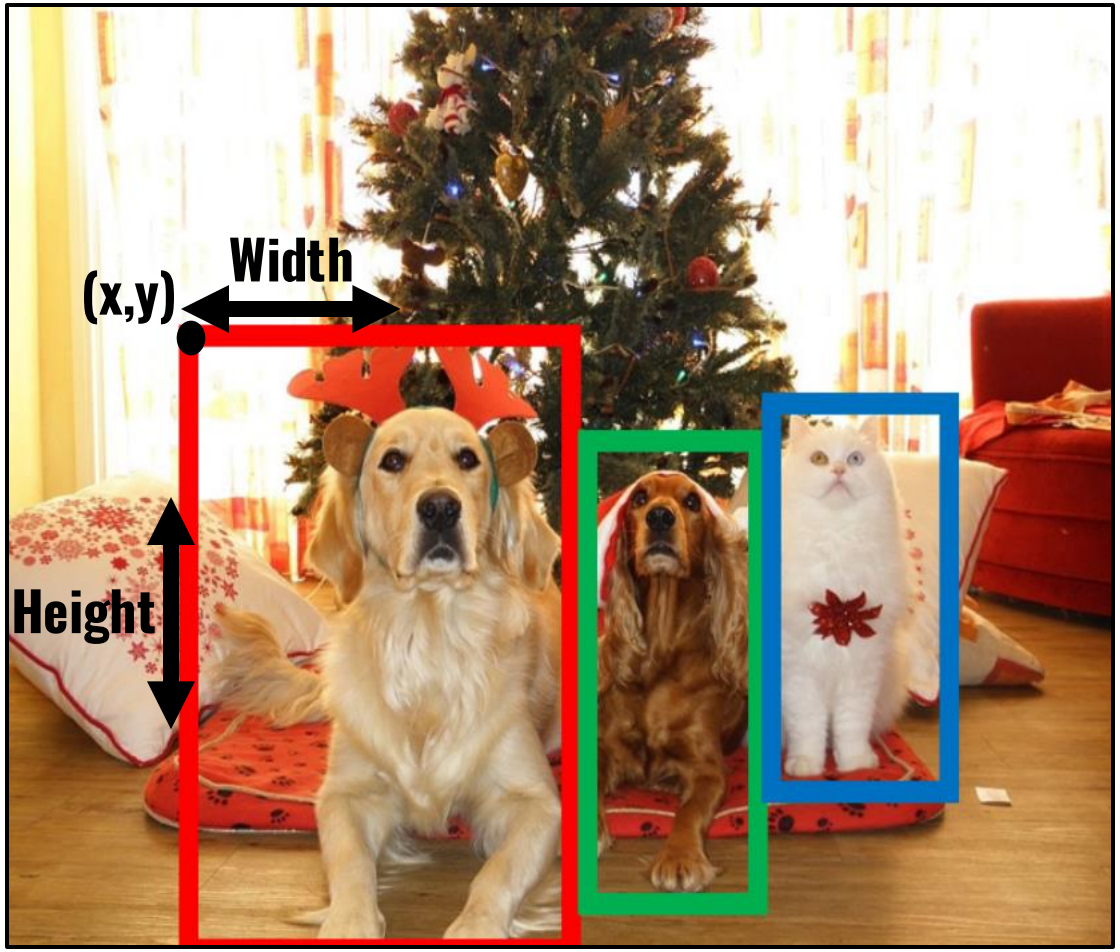
WHERE?

[0.48, 0.51, 0.01]

What is Object Detection?

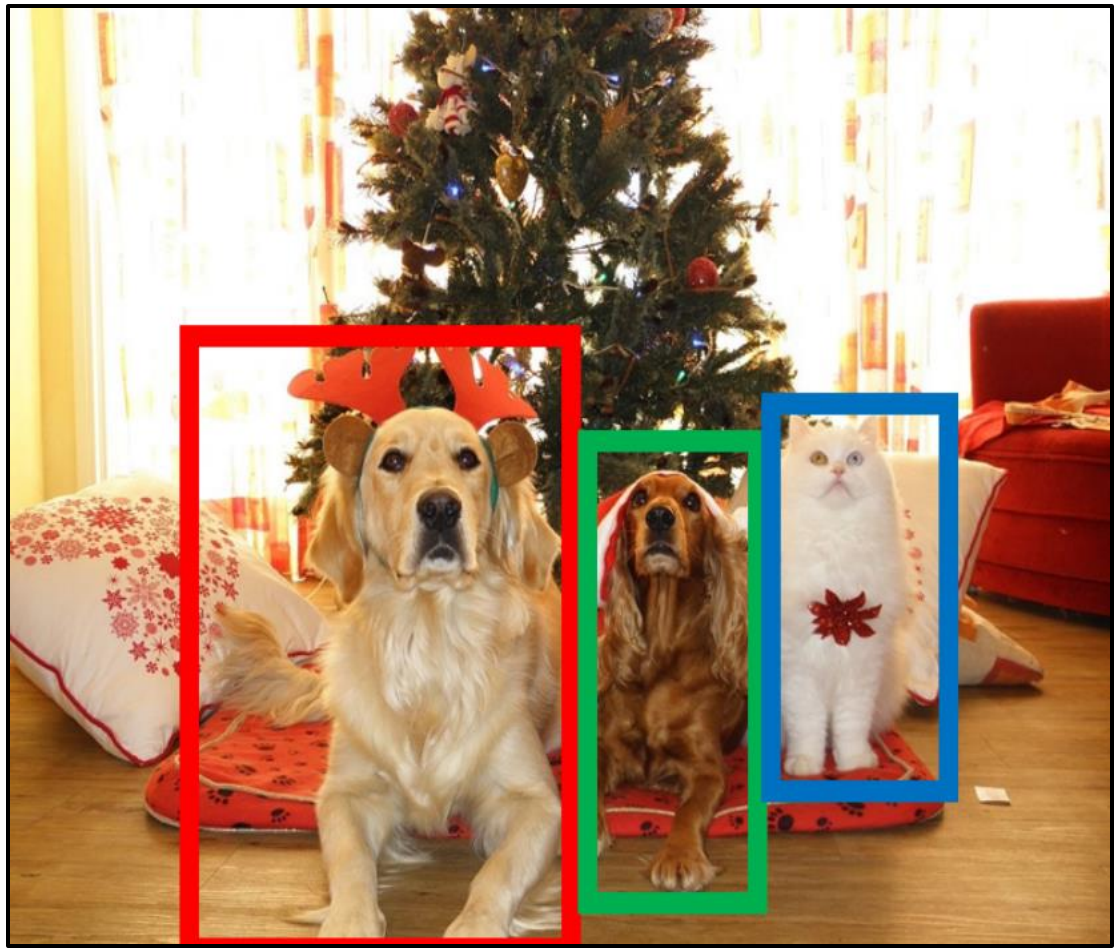
- **Supervised Learning Task**
- **Input**: RGB image
- **Output**: A set of detected objects;

For each object predict:
- **Category label** (from fixed, known set of categories)
- **Bounding box** (four numbers: x, y, width, height)

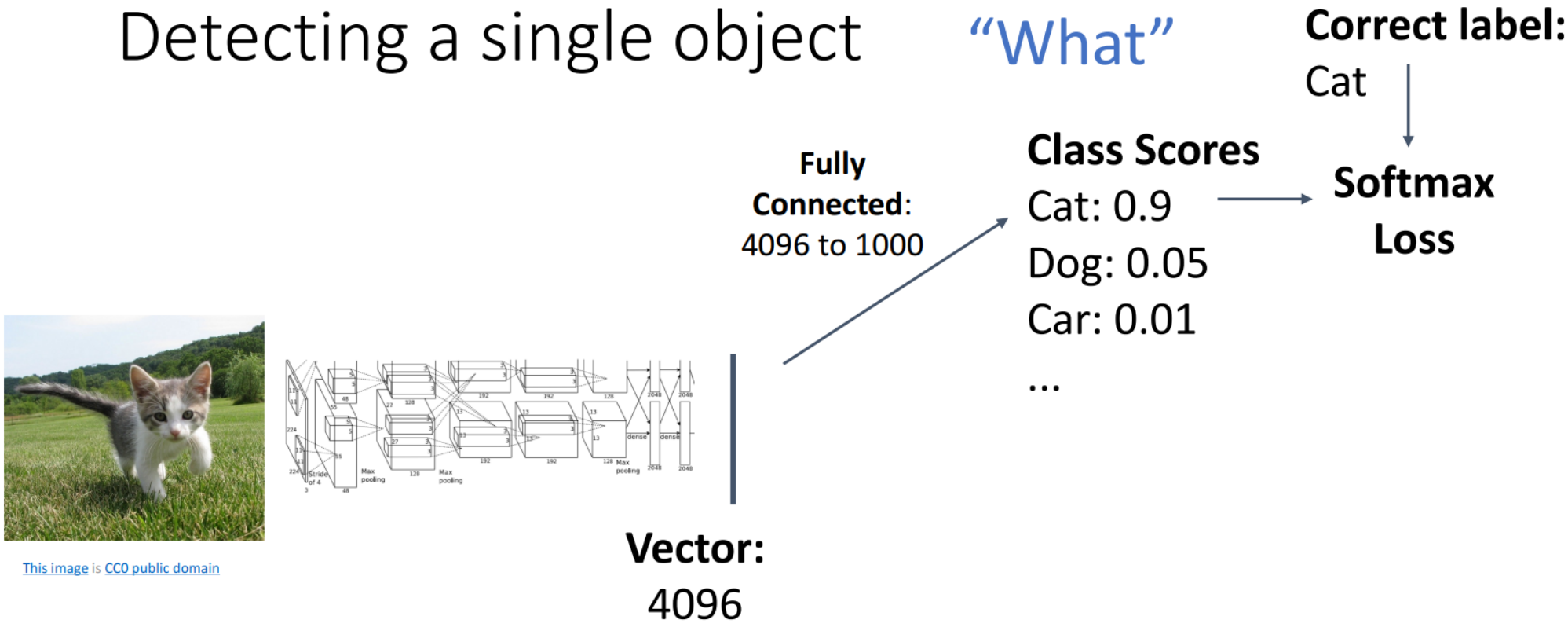


What is Object Detection?

- **Multiple outputs**: Need to output variable numbers of objects per image
- **Multiple types of outputs**: Need to predict “what” (category label) as well as “where” (bounding box)
- **Large images**: Classification works at 224x224; need higher resolution for detection, often ~800x600

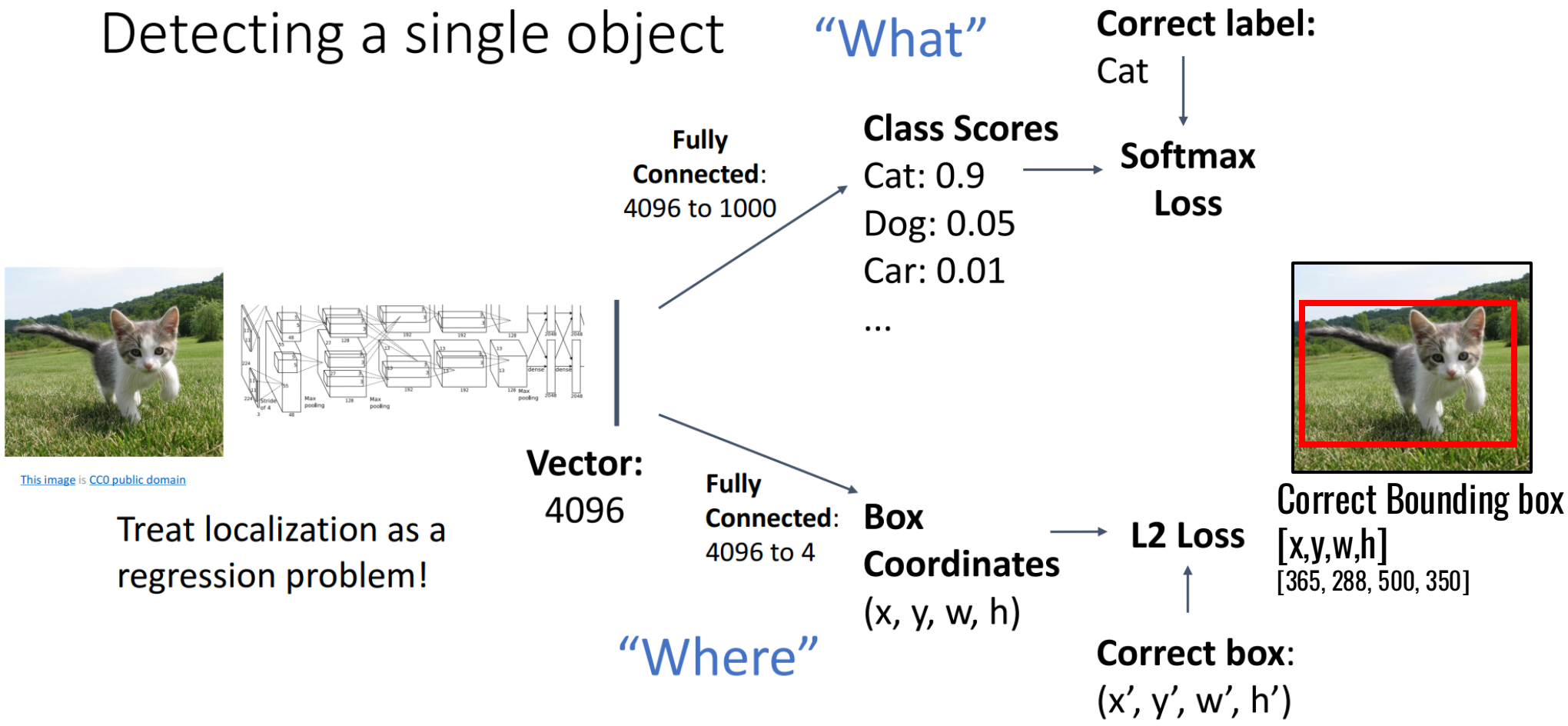


Classification + Localization (Object Detection for a single object)



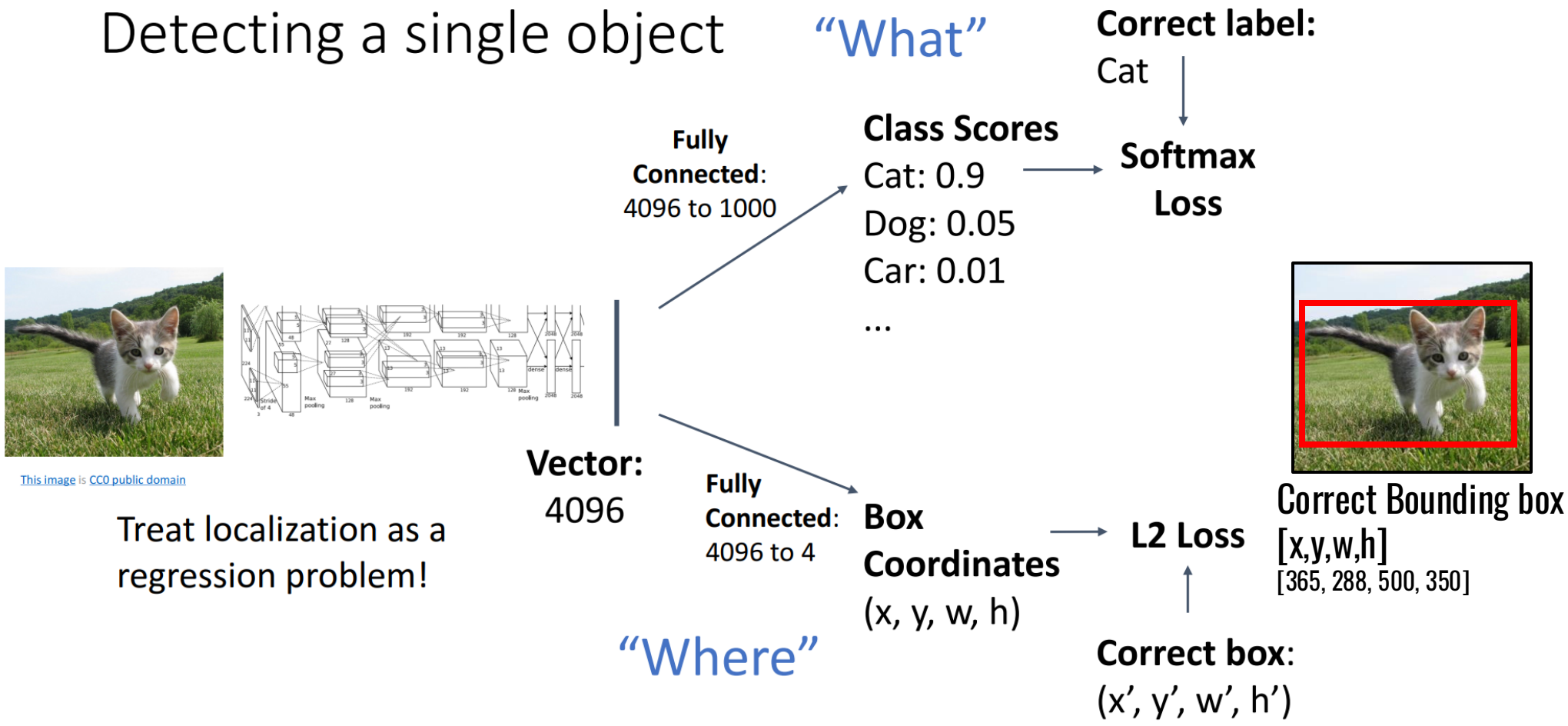
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Classification + Localization (Object Detection for a single object)



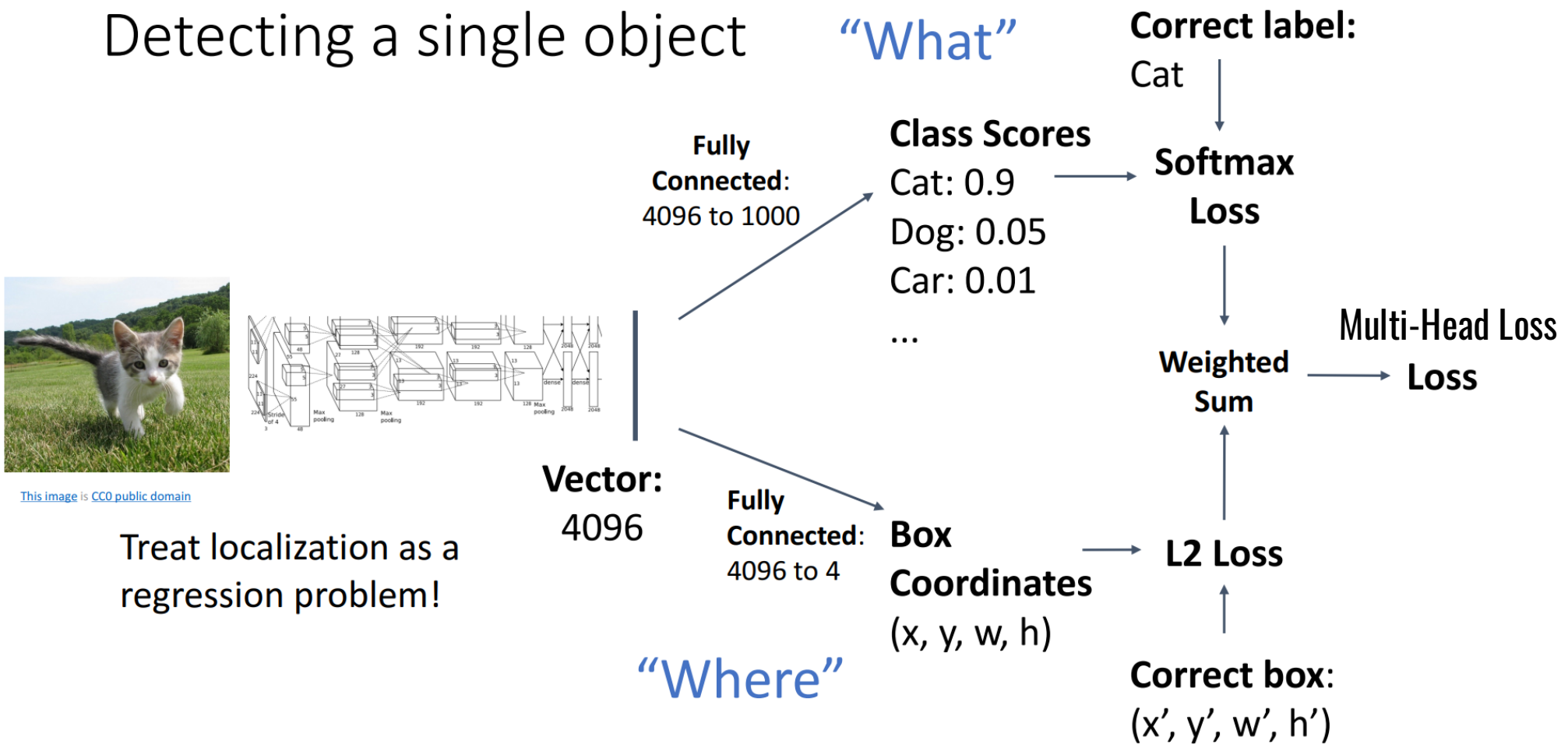
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Classification + Localization (Object Detection for a single object)



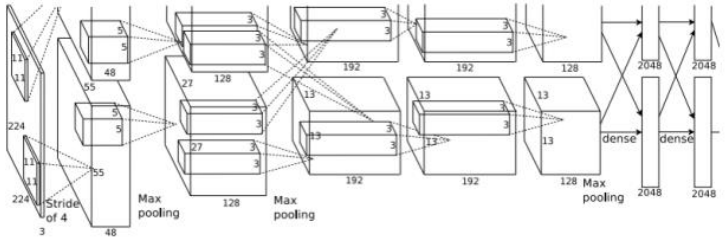
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Classification + Localization (Object Detection for a single object)



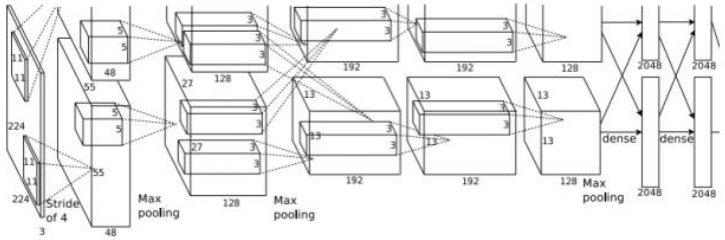
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Classification + Localization (Object Detection for a single object)



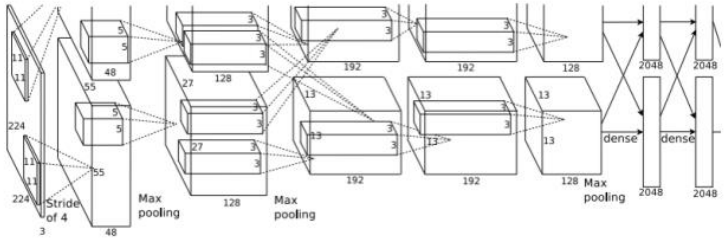
CAT: (x, y, w, h)

4 numbers



DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)

16 numbers



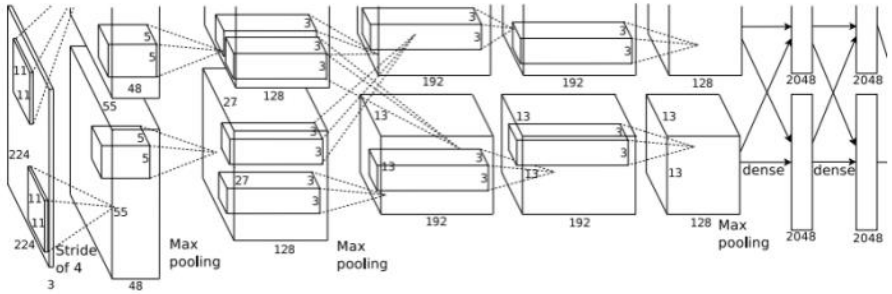
DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....

Many numbers!

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Sliding Window (Naïve approach)

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

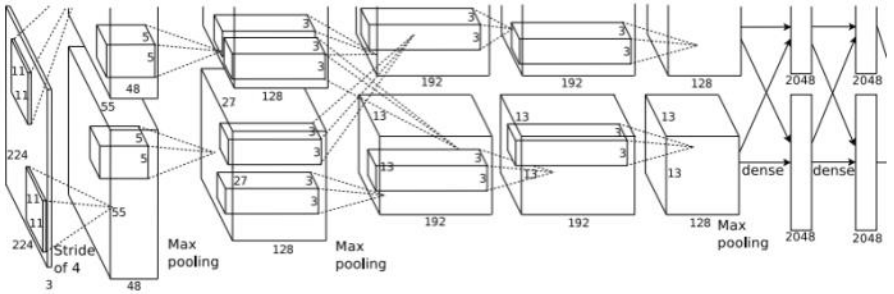


Dog? NO
Cat? NO
Background? YES

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Sliding Window (Naïve approach)

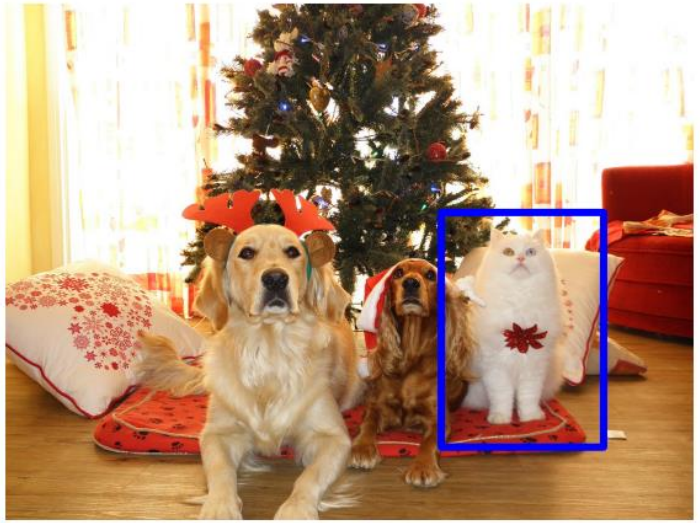
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Sliding Window (Naïve approach)



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

800 x 600 image has ~58M boxes! No way we can evaluate them all

Question: How many possible boxes are there in an image of size H x W?

Consider a box of size h x w:
Possible x positions: W – w + 1
Possible y positions: H – h + 1
Possible positions:
(W – w + 1) * (H – h + 1)

Total possible boxes:

$$\sum_{h=1}^{H}\sum_{w=1}^{W}(W-w+1)(H-h+1)$$

$$=\frac{H(H+1)}{2}\frac{W(W+1)}{2}$$

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

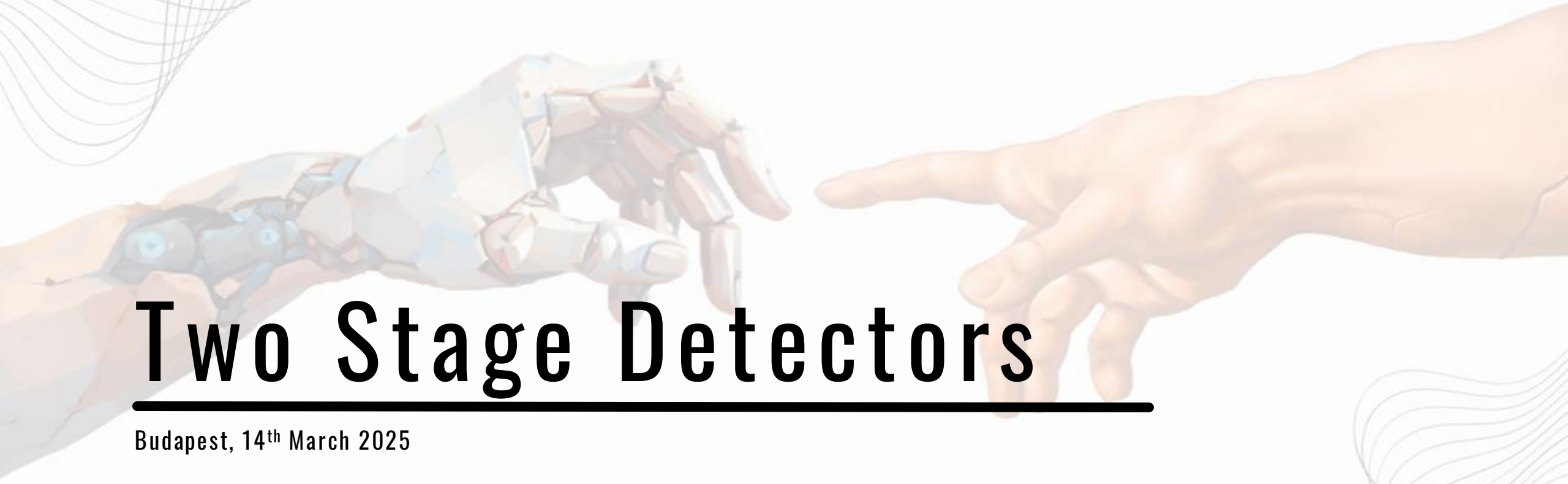
Object Detection for multiple objects

Region proposal based (Two Stage Detectors)

- R-CNN
- Fast R-CNN
- Faster R-CNN

Deep Network Development

Lecture 5.



Two Stage Detectors

Budapest, 14th March 2025

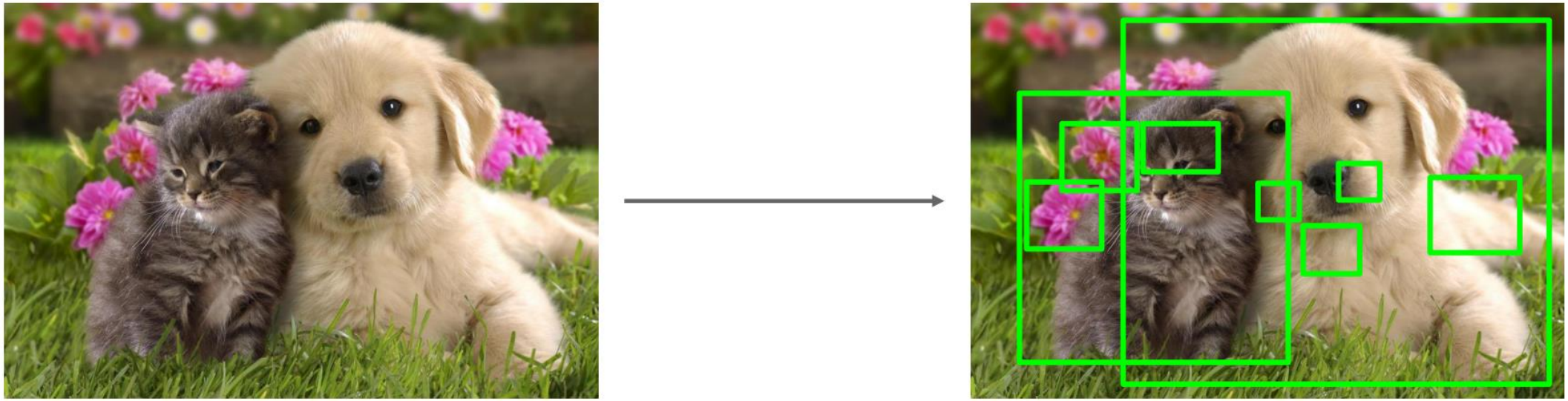
1 Two Stage Detectors

2 One Stage Detectors

3 Object Detection Metrics

Region Proposal Detectors (Two Stage Detectors)

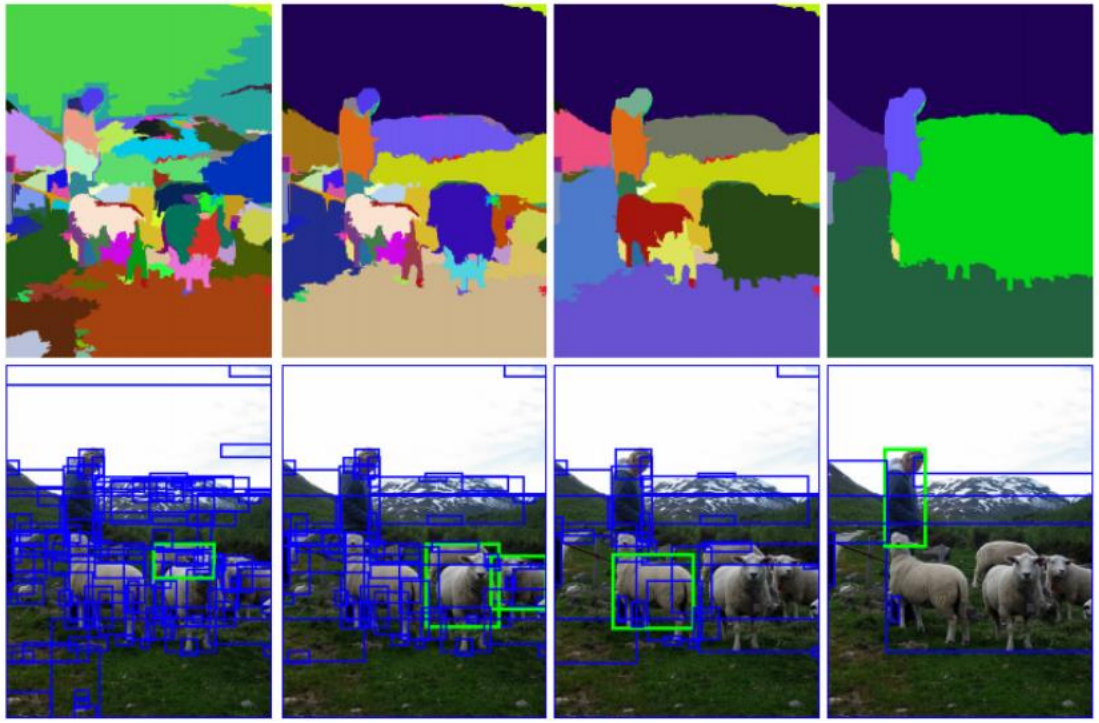
- Find a small set of boxes that are likely to cover all objects
- Often based on heuristics: e.g. look for “blob-like” image regions
- Relatively fast to run; e.g. Selective Search gives 2000 region proposals in a few seconds on CPU



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

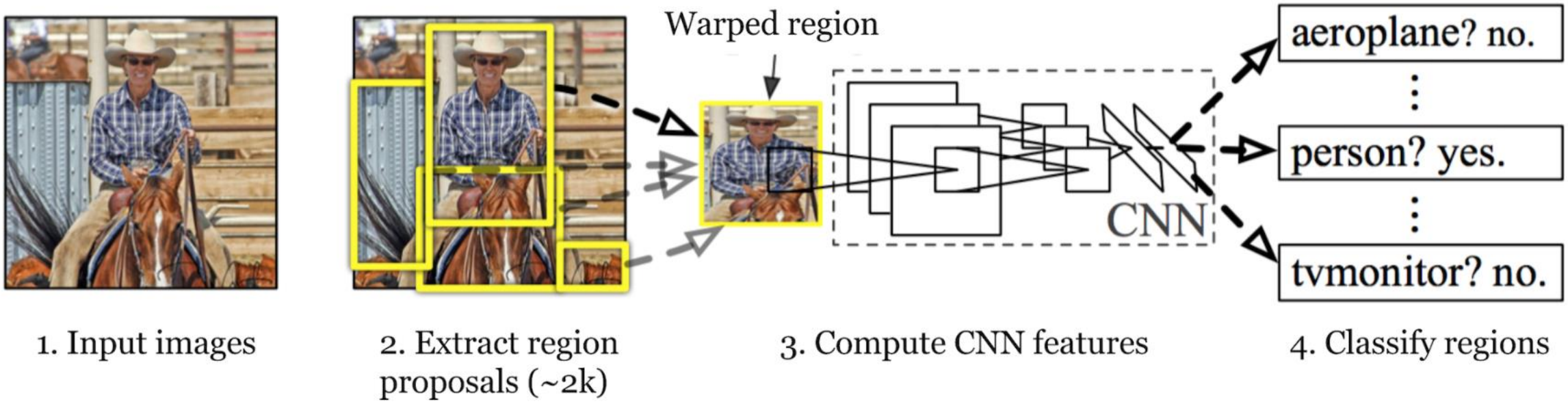
Region Proposal Detectors (Two Stage Detectors)

- Selective Search



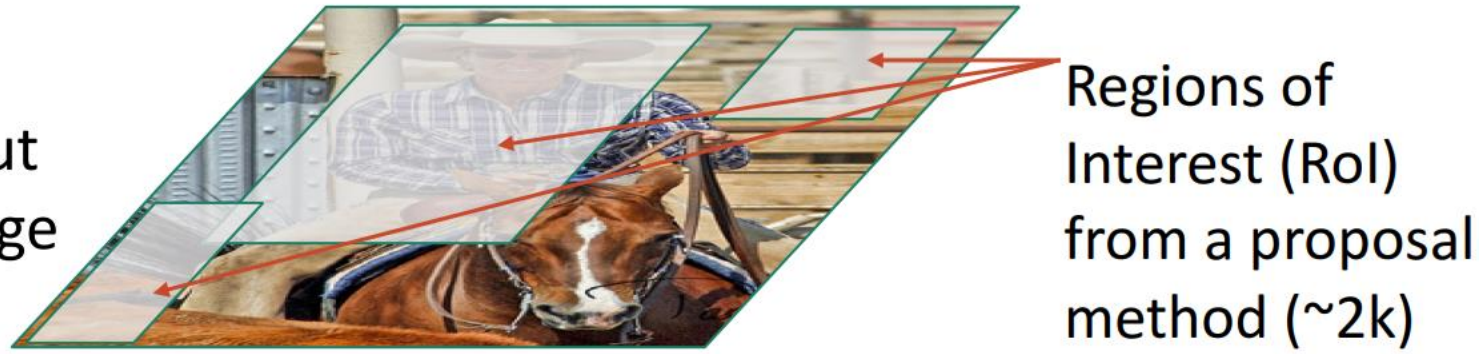
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

R-CNN: Region-Based Convolutional Neural Network [1]



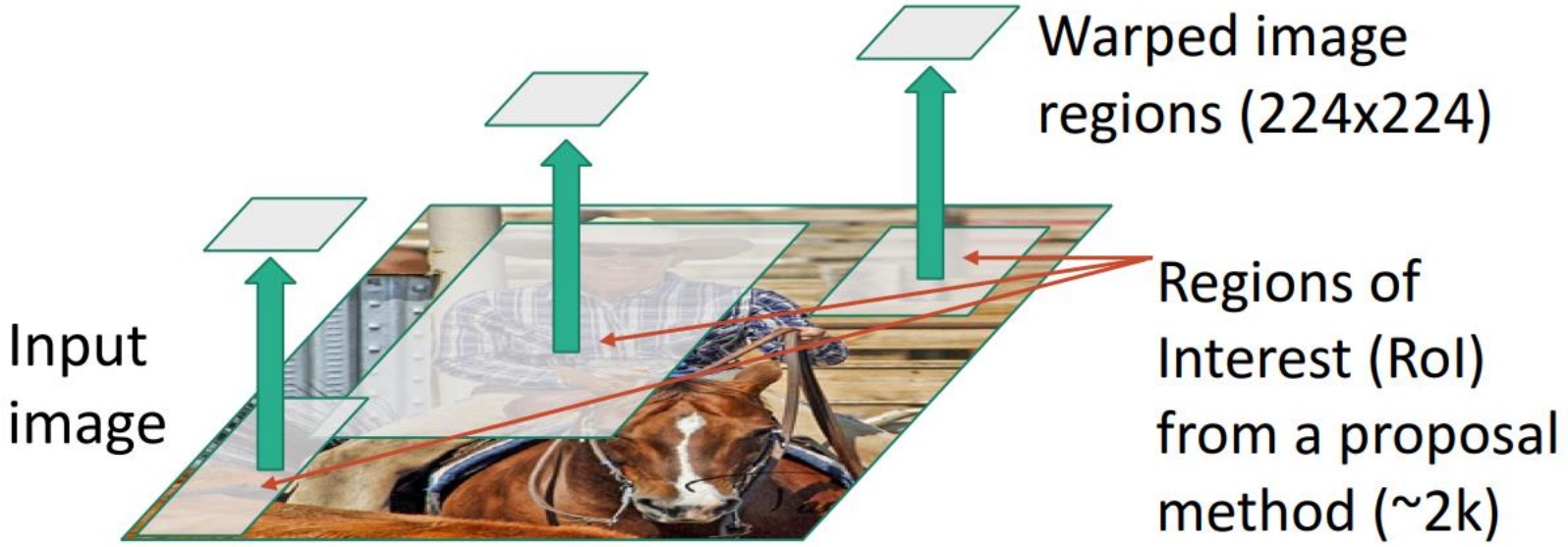
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/
[1] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. arXiv [Cs.CV]. Retrieved from http://arxiv.org/abs/1311.2524

R-CNN: Region-Based Convolutional Neural Network



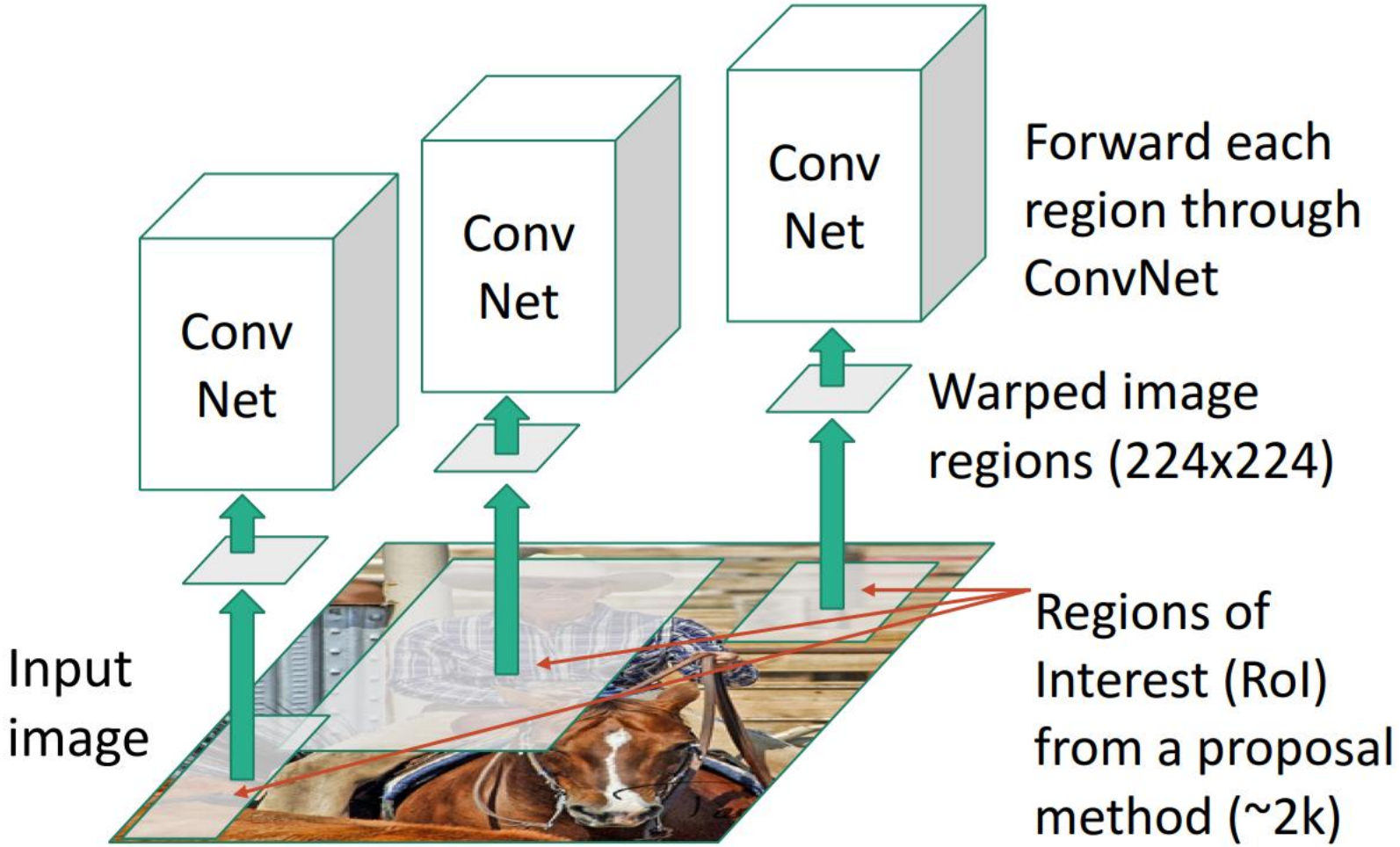
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

R-CNN: Region-Based Convolutional Neural Network



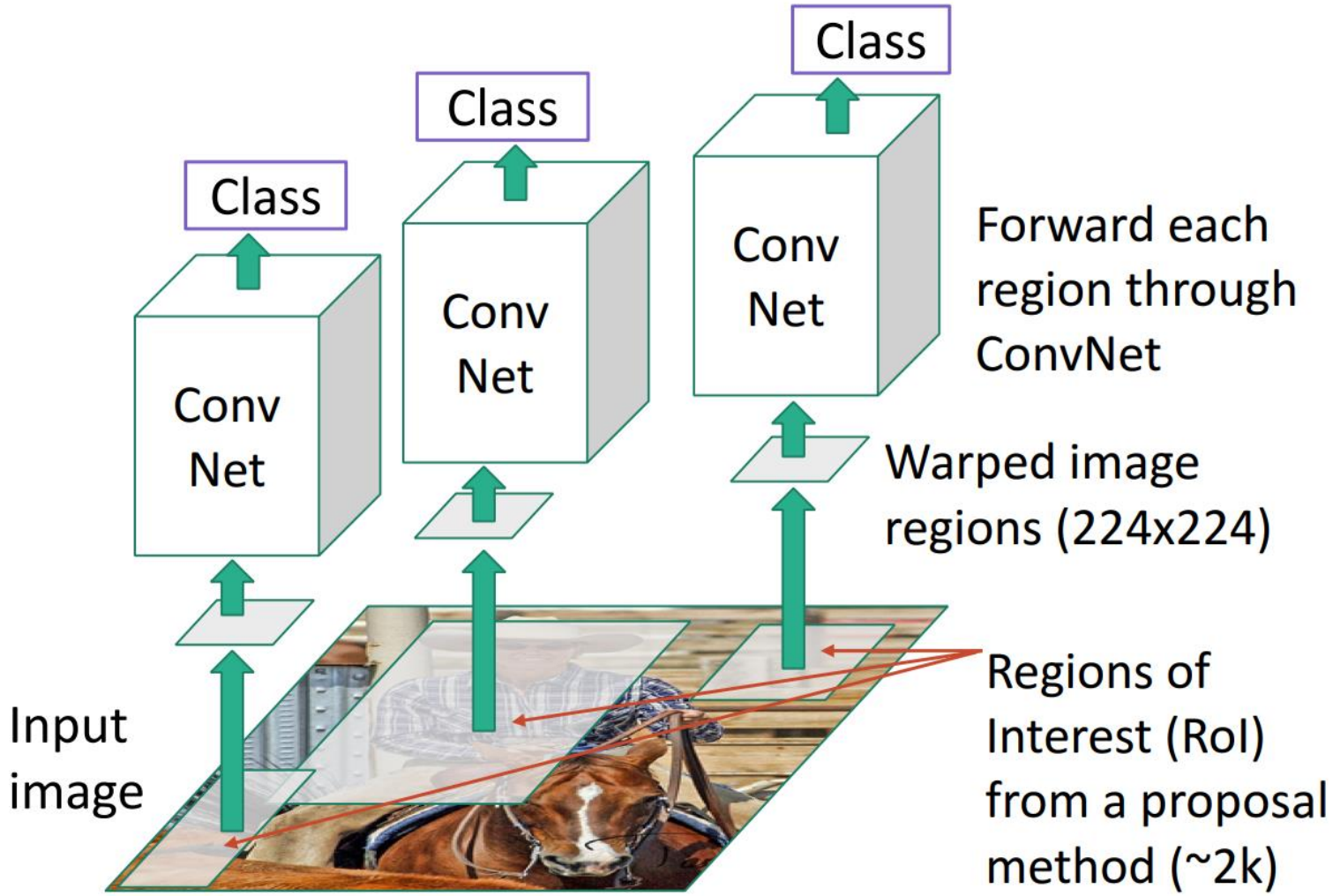
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

R-CNN: Region-Based Convolutional Neural Network



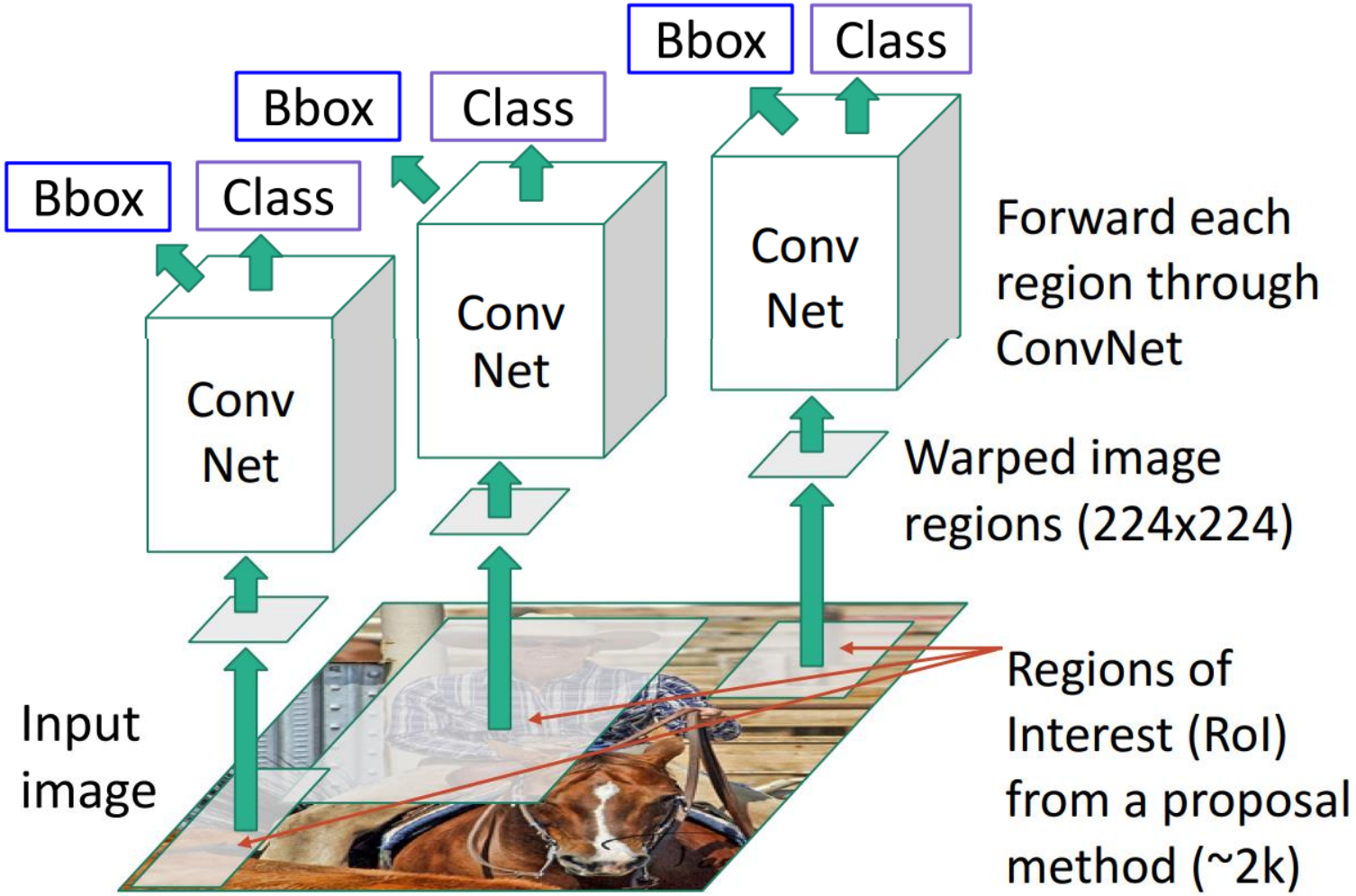
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

R-CNN: Region-Based Convolutional Neural Network



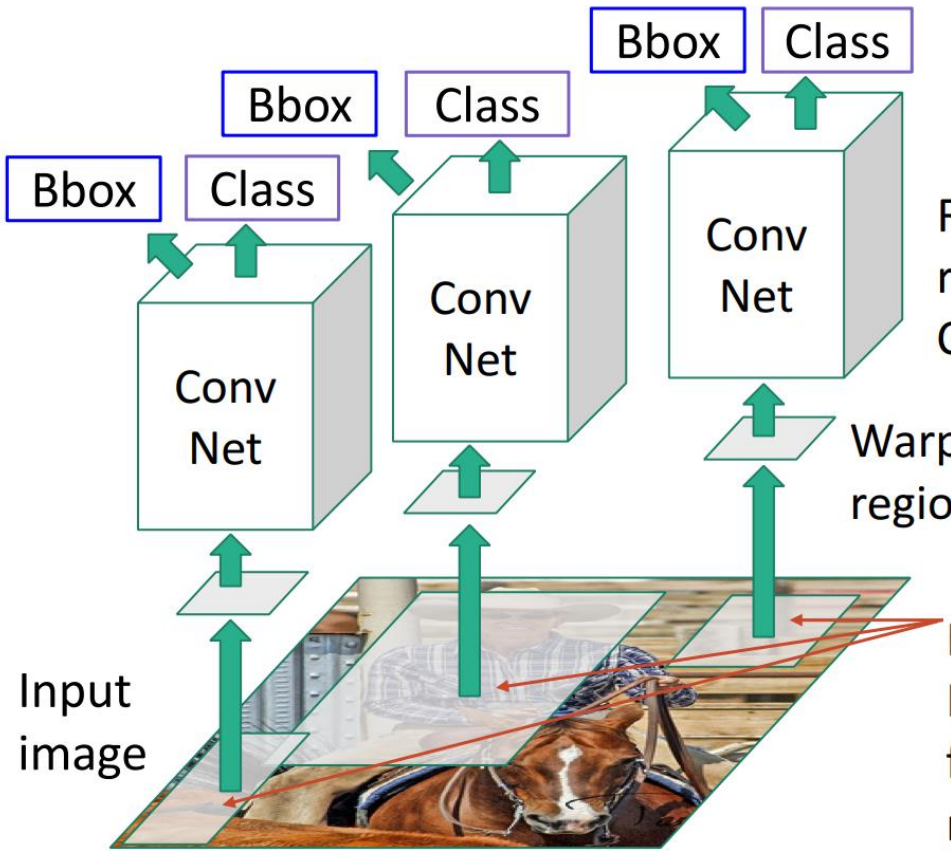
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

R-CNN: Region-Based Convolutional Neural Network



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

R-CNN: Region-Based Convolutional Neural Network



Forward each region through ConvNet

Classify each region

Bounding box regression:
Predict "transform" to correct the RoI: 4 numbers (t_x, t_y, t_h, t_w)

Region proposal: (p_x, p_y, p_h, p_w)
Transform: (t_x, t_y, t_h, t_w)
Output box: (b_x, b_y, b_h, b_w)

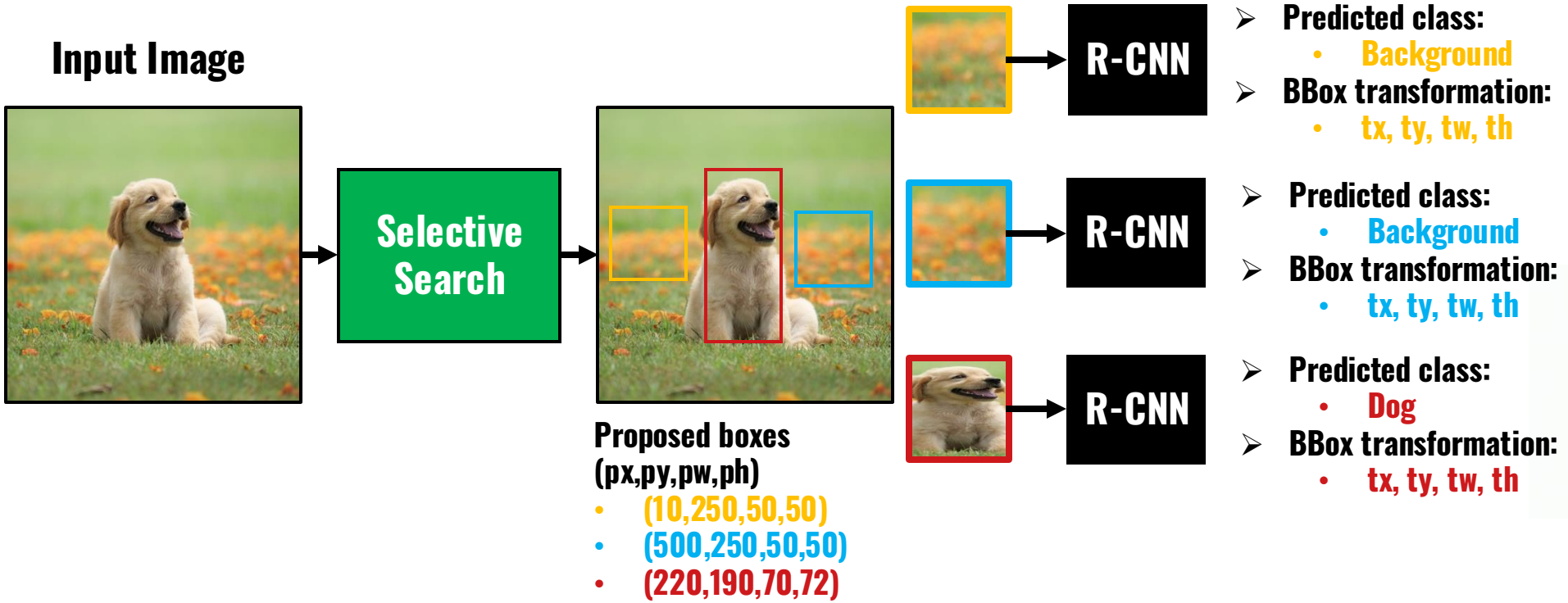
Translate relative to box size:
$b_x = p_x + p_w t_x \qquad b_y = p_y + p_h t_y$

Log-space scale transform:
$b_w = p_w \exp(t_w) \qquad b_h = p_h \exp(t_h)$

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

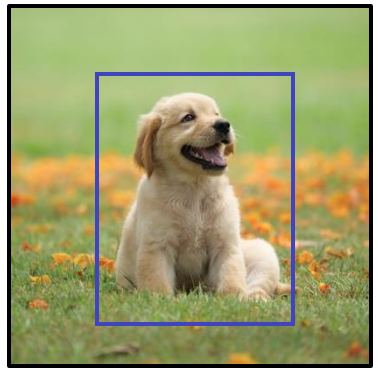
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

R-CNN: Region-Based Convolutional Neural Network



R-CNN: Region-Based Convolutional Neural Network

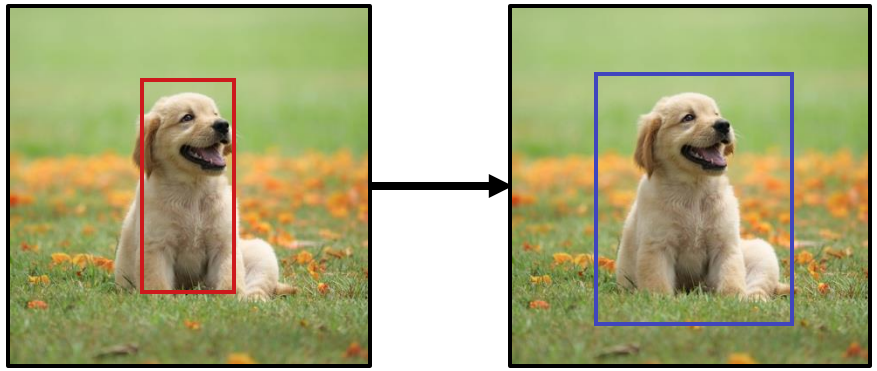
Ground Truth:
- Class: Dog
- Bounding Box (x,y,w,h) = (200,200,120,80)



How to transform the proposed box into the correct bounding box?
We use the predicted transformation values (tx,ty,tw,th)

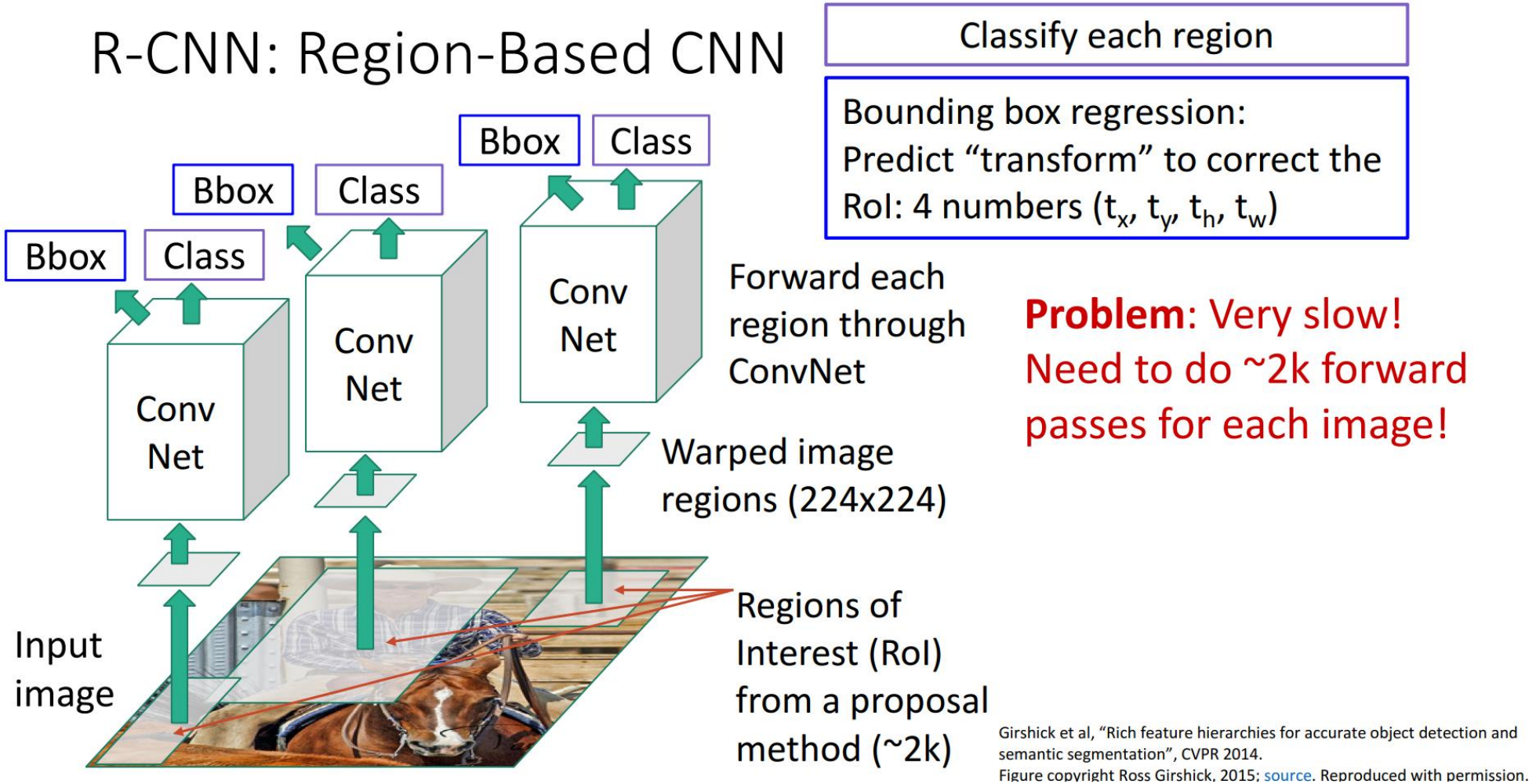
For example:

x = px + tx



Bounding box regression:
Predict "transform" to correct the RoI: 4 numbers (t_x, t_y, t_h, t_w)

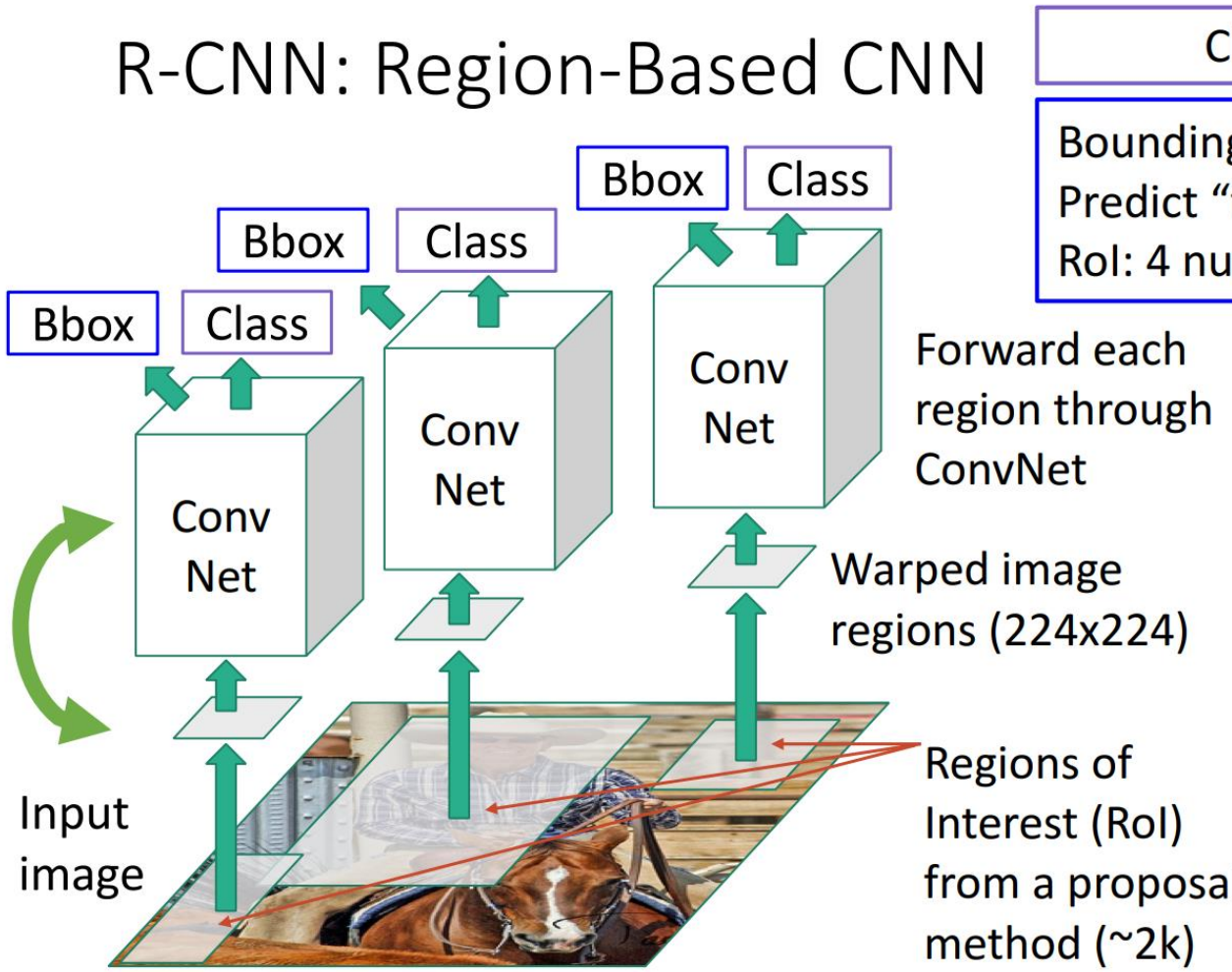
R-CNN: Region-Based Convolutional Neural Network



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

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

R-CNN: Region-Based Convolutional Neural Network



Classify each region

Bounding box regression:
Predict "transform" to correct the RoI: 4 numbers (t_x, t_y, t_h, t_w)

Forward each region through ConvNet

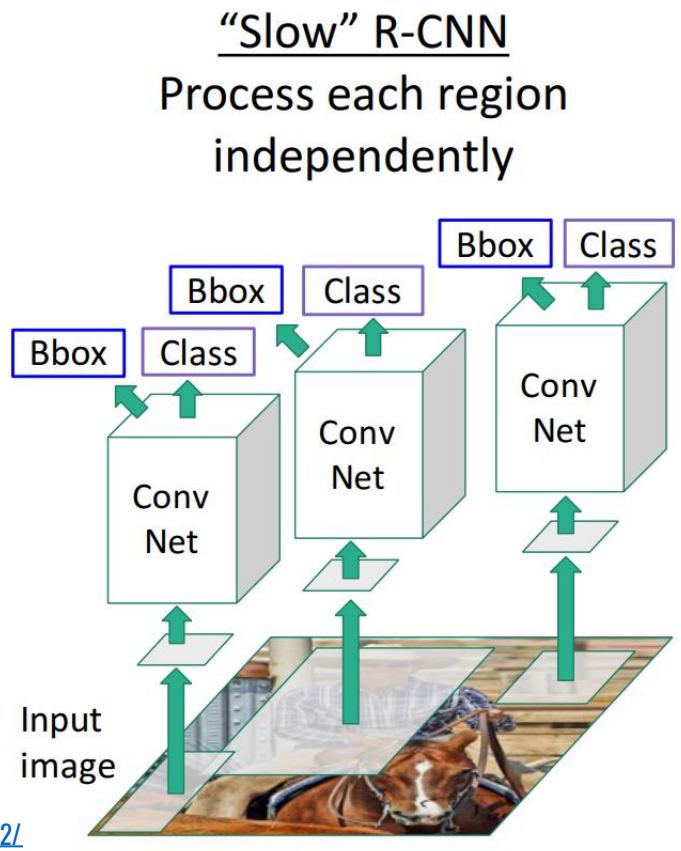
Problem: Very slow!
Need to do ~2k forward passes for each image!

Solution: Run CNN *before* warping!

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

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

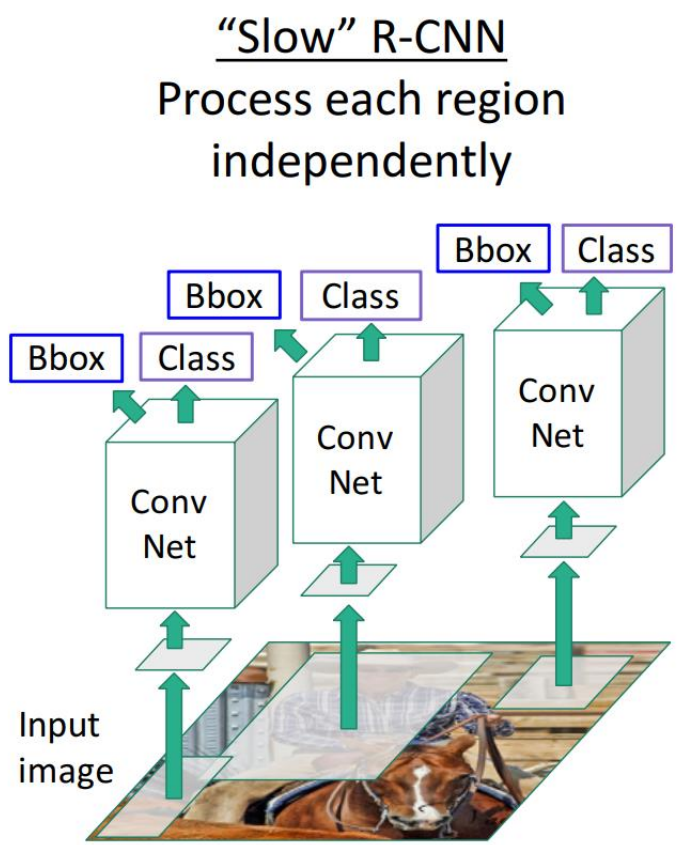
Fast R-CNN: Region-Based Convolutional Neural Network [2]



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

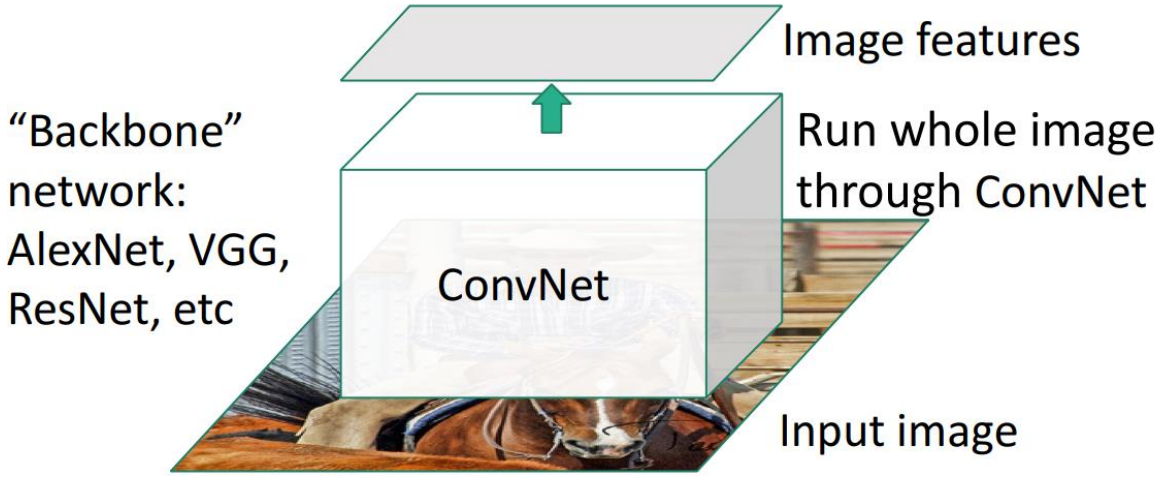
[2] Girshick, R. (2015). Fast R-CNN. arXiv [Cs.CV]. Retrieved from http://arxiv.org/abs/1504.08083

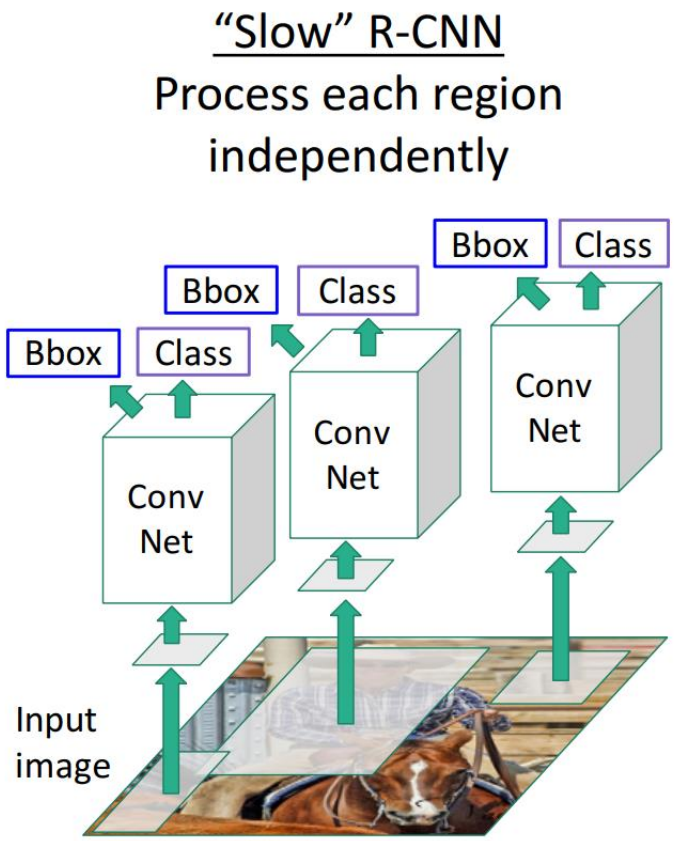
Fast R-CNN: Region-Based Convolutional Neural Network



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

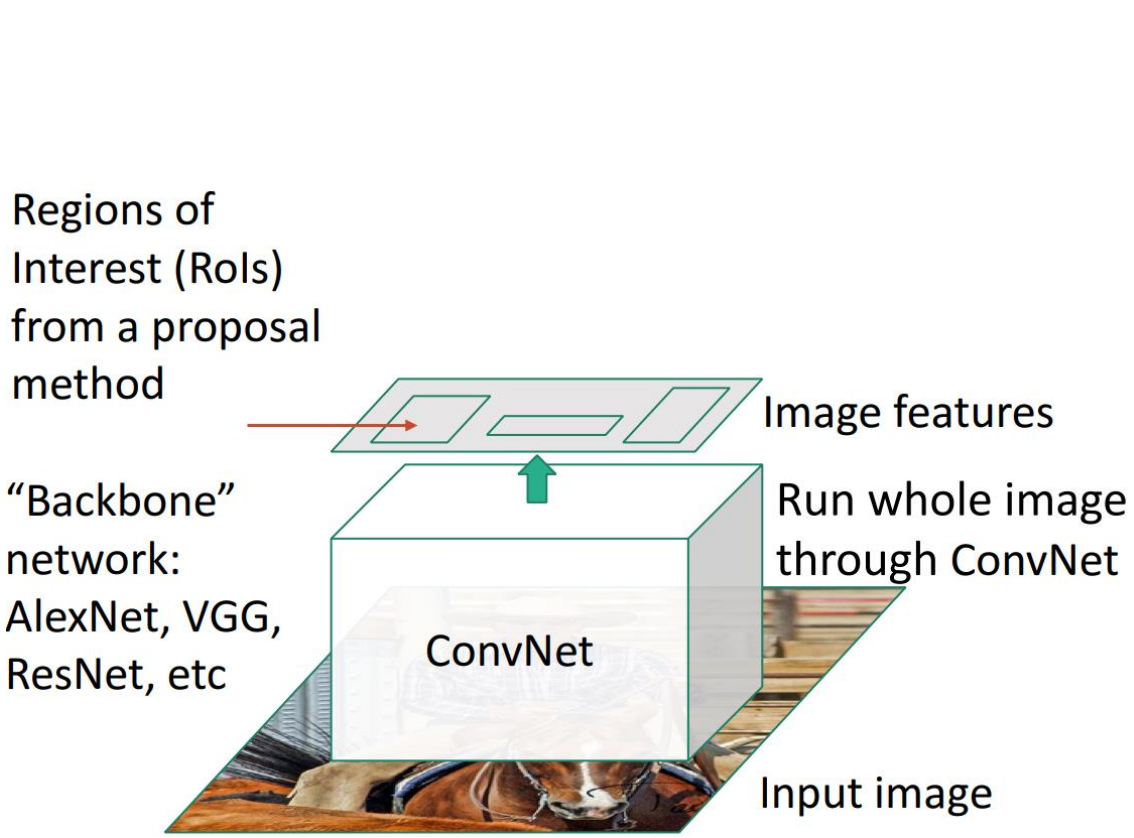
Fast R-CNN: Region-Based Convolutional Neural Network

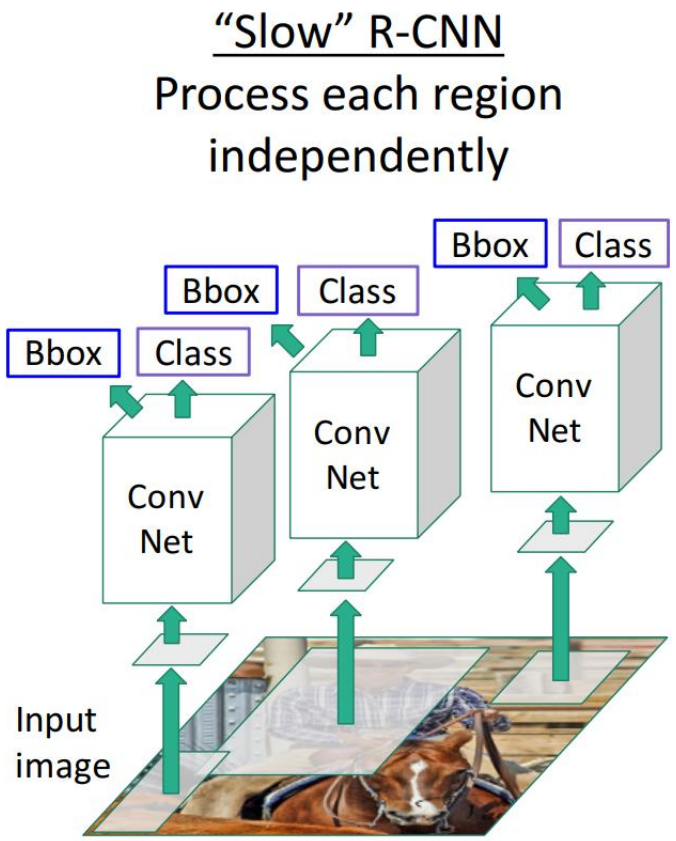




Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Fast R-CNN: Region-Based Convolutional Neural Network





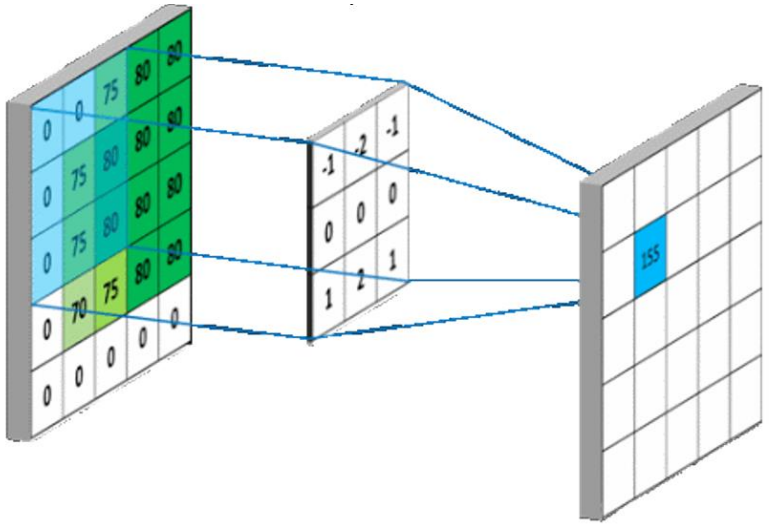
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

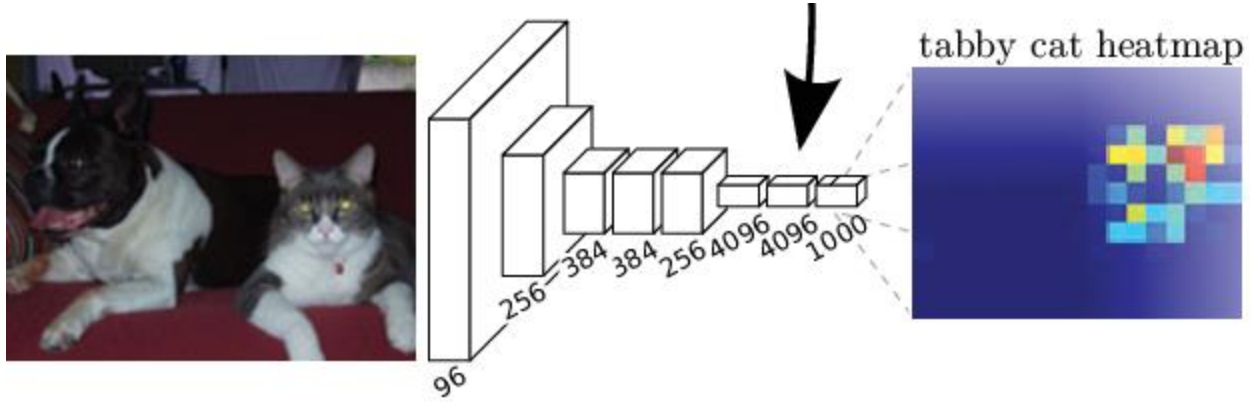
Quick ROI projection to feature map insight

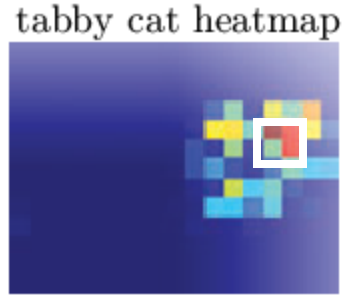
Convolutional Layer

- **Filter / Kernel size = f x f**

Convolution
Input: 5x5
Filter: 3x3
Output: 3x3

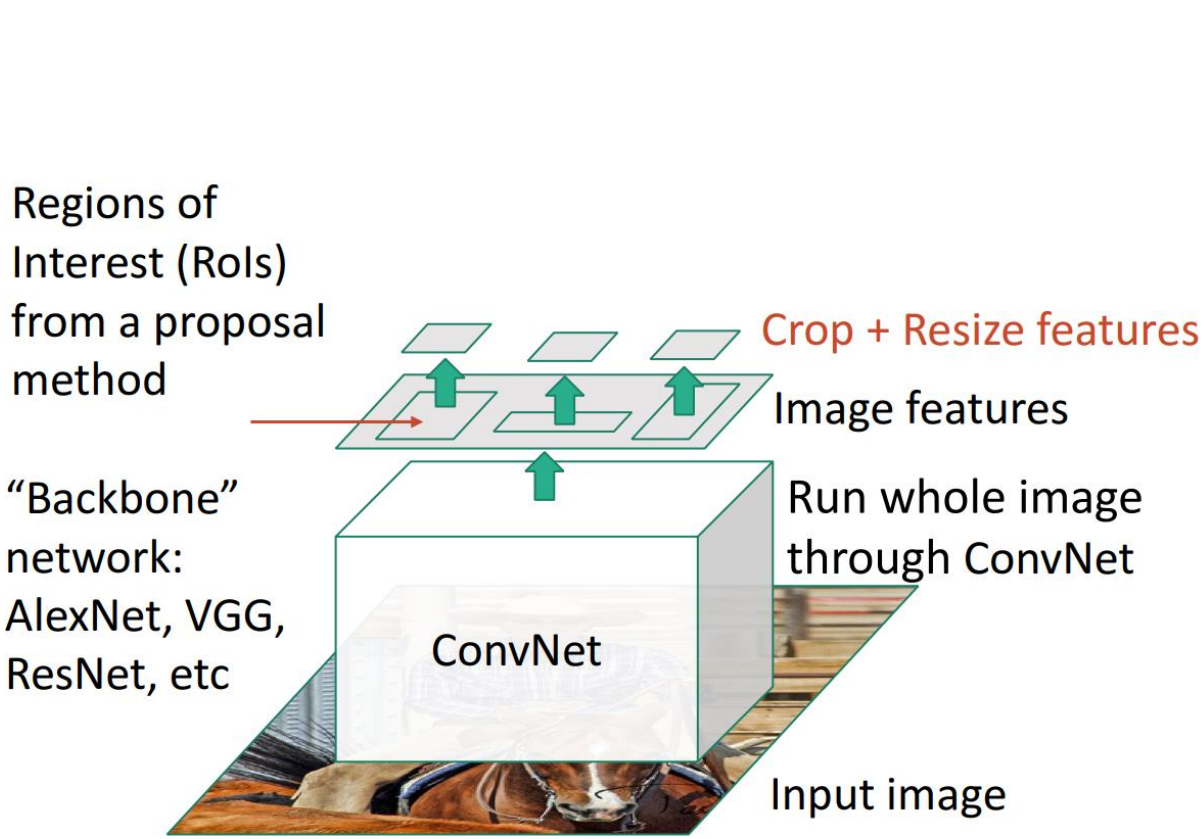


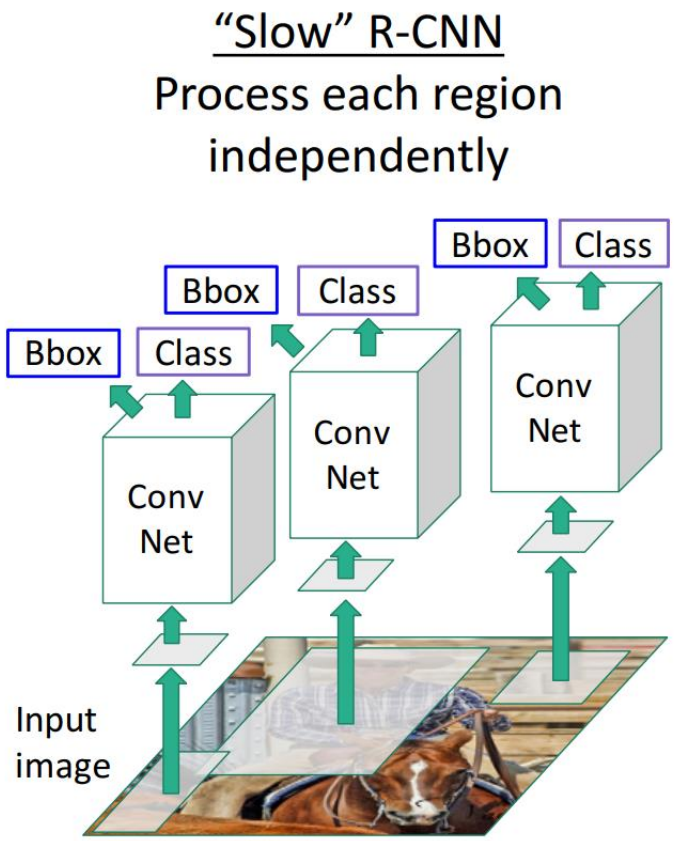




In practice it is harder, there are several tricks. For example: ROI Pool and ROI Align

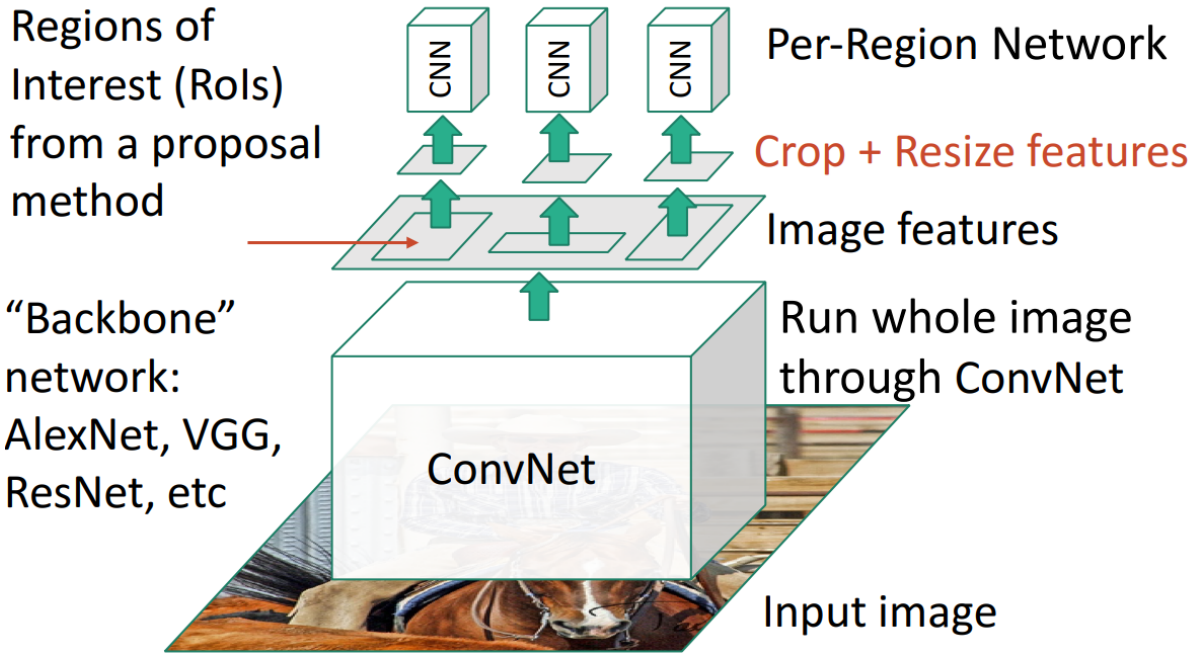
Fast R-CNN: Region-Based Convolutional Neural Network

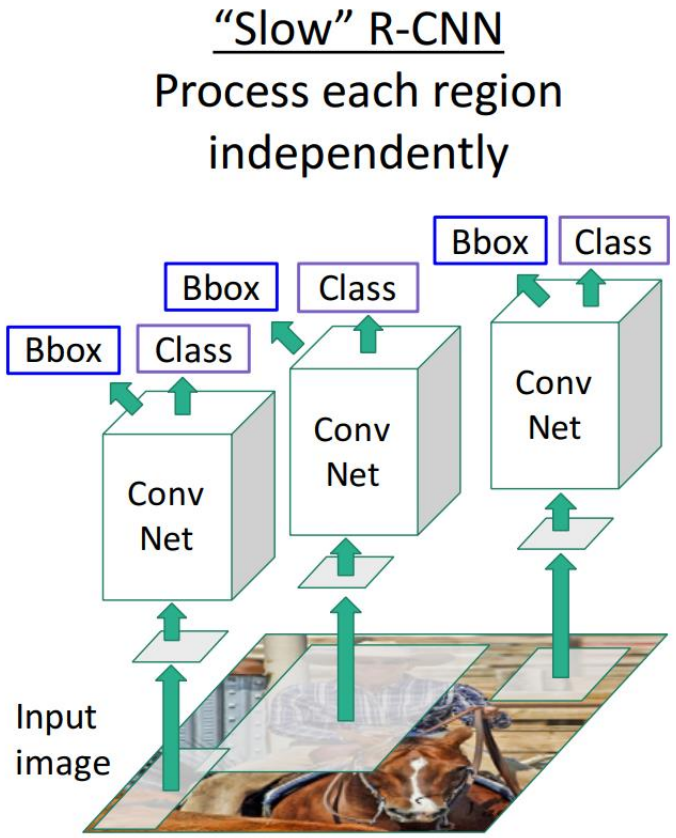




Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

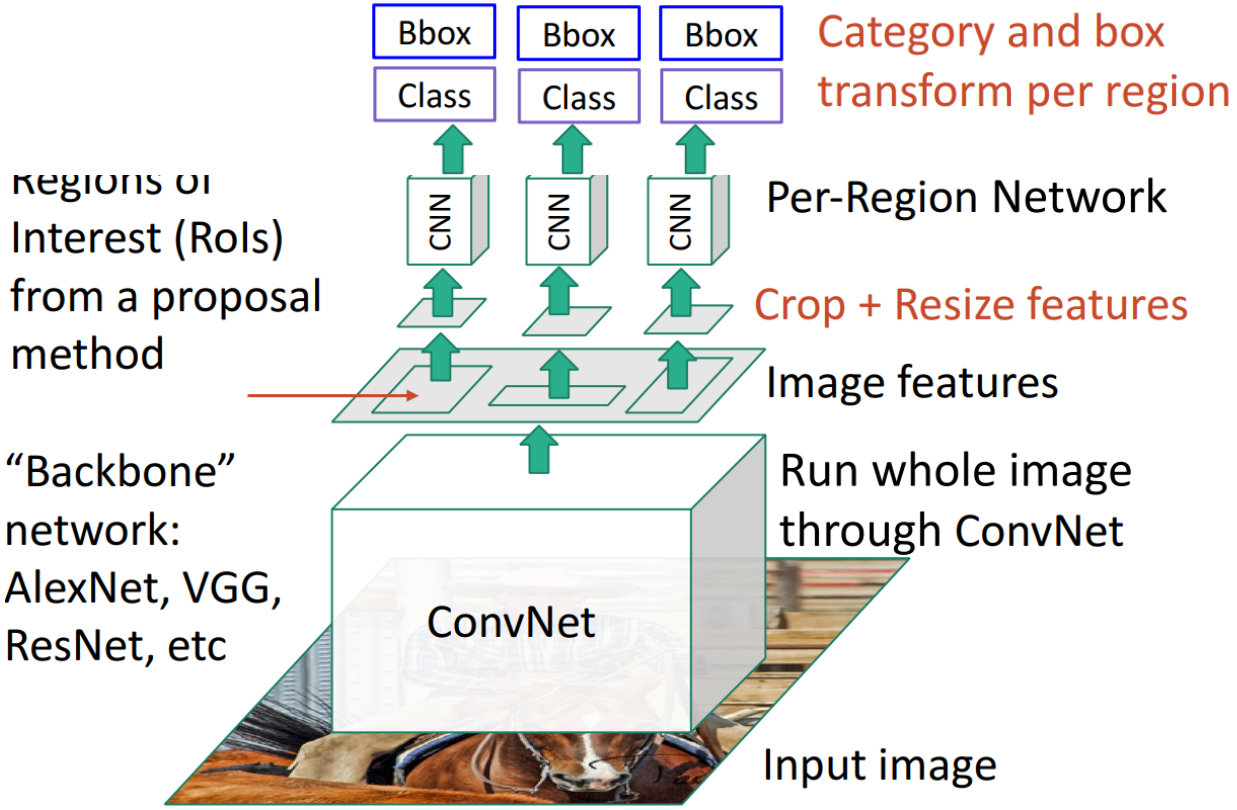
Fast R-CNN: Region-Based Convolutional Neural Network

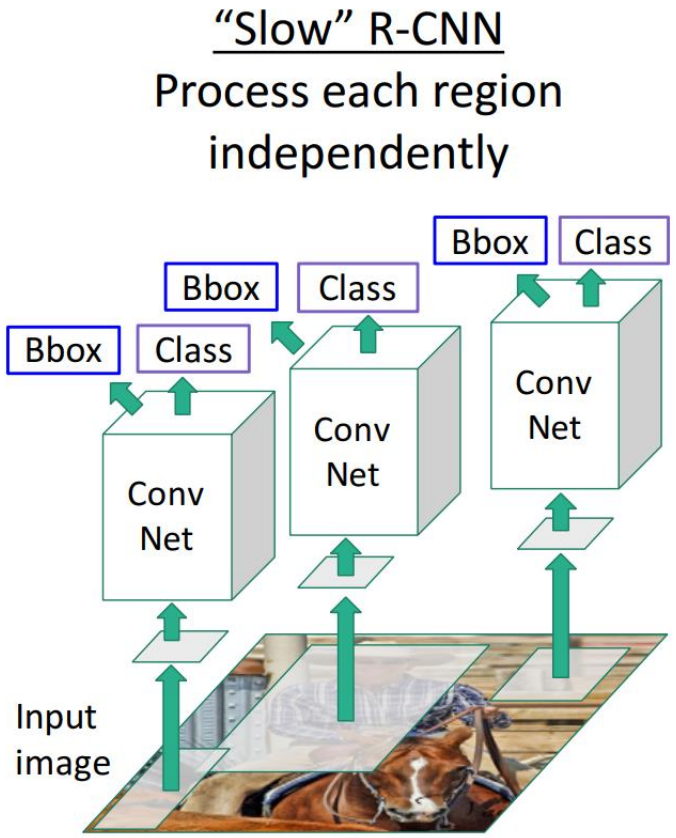




Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

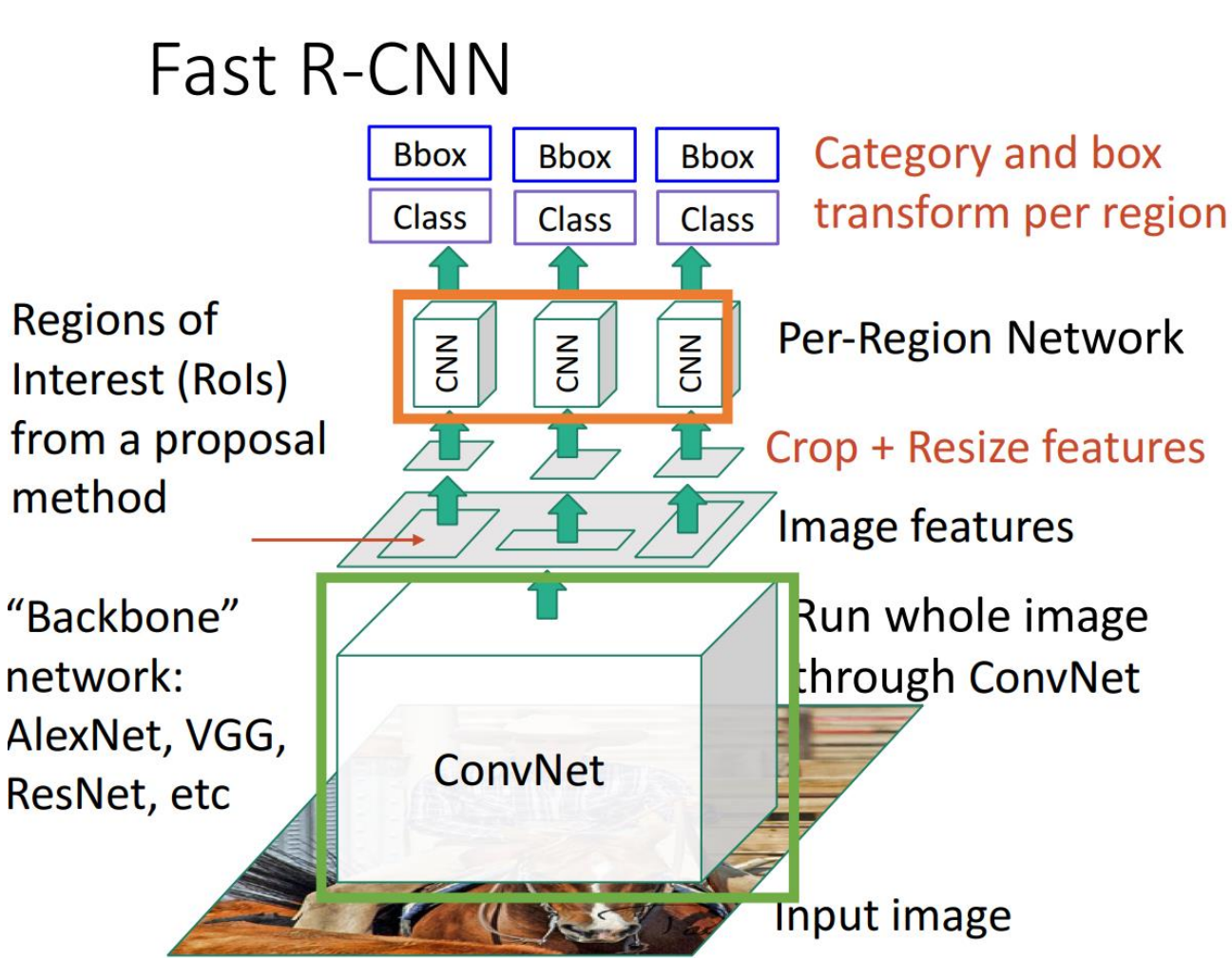
Fast R-CNN: Region-Based Convolutional Neural Network

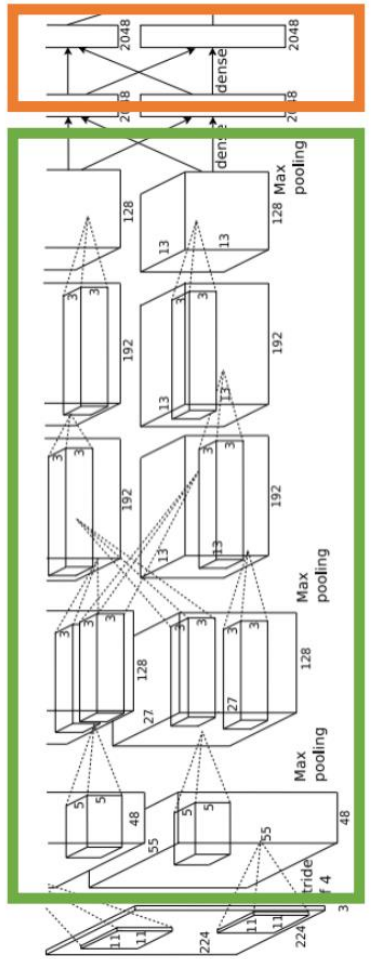




Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Fast R-CNN: Region-Based Convolutional Neural Network

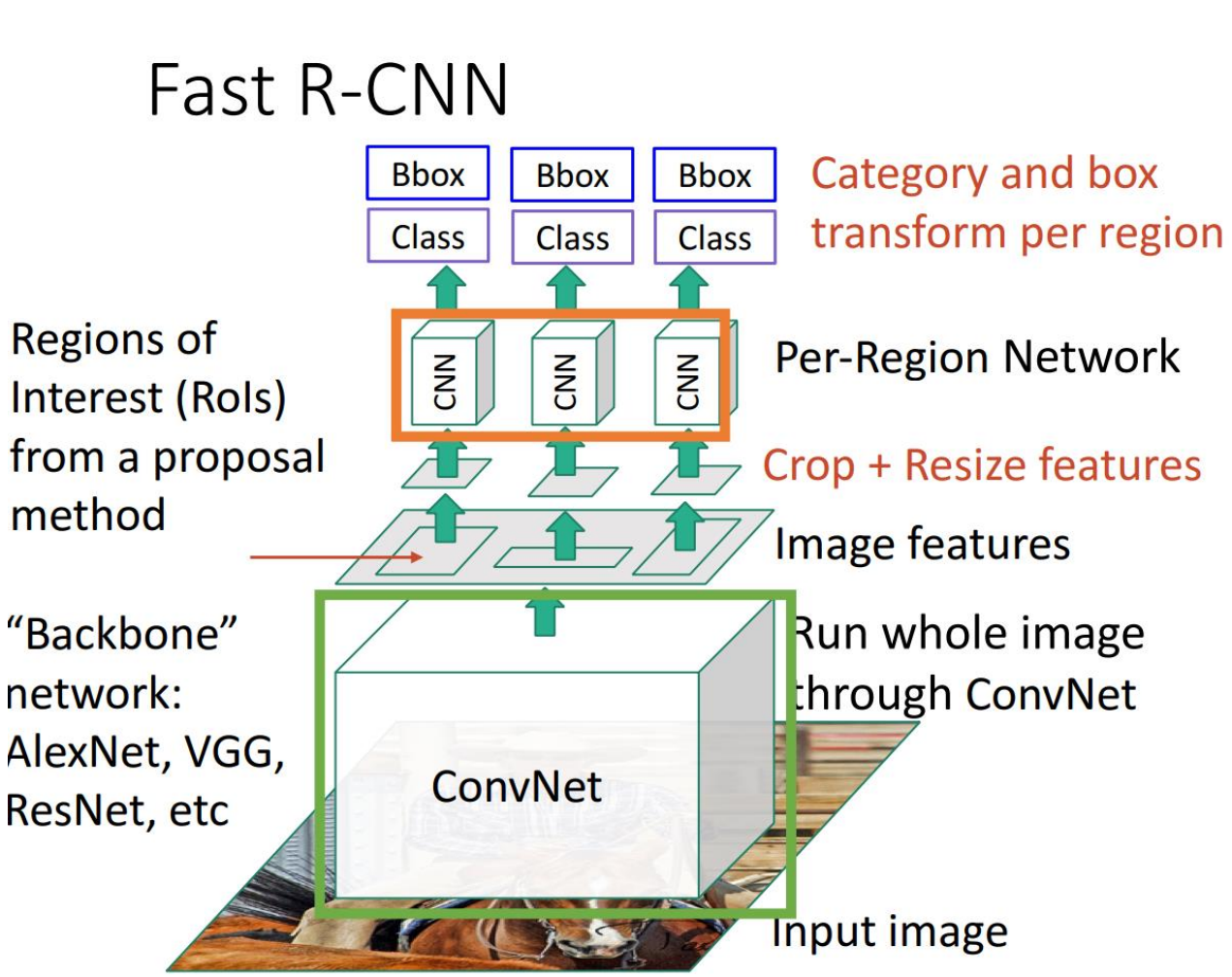


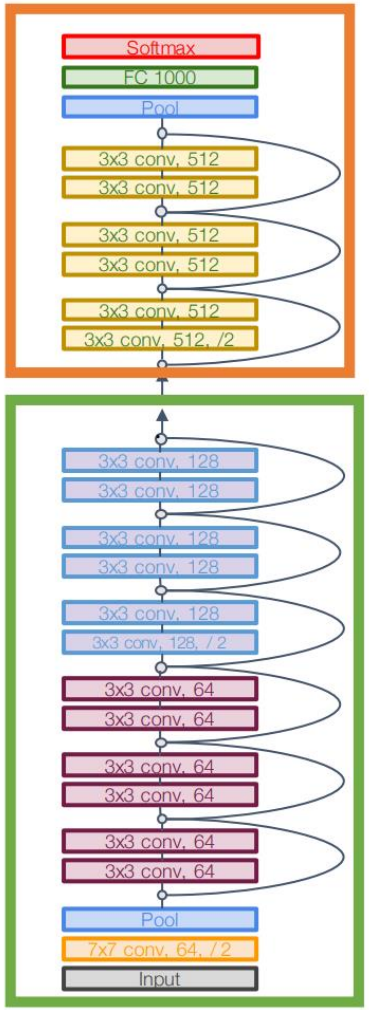


Example:
When using AlexNet for detection, five conv layers are used for backbone and two FC layers are used for per-region network

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Fast R-CNN: Region-Based Convolutional Neural Network

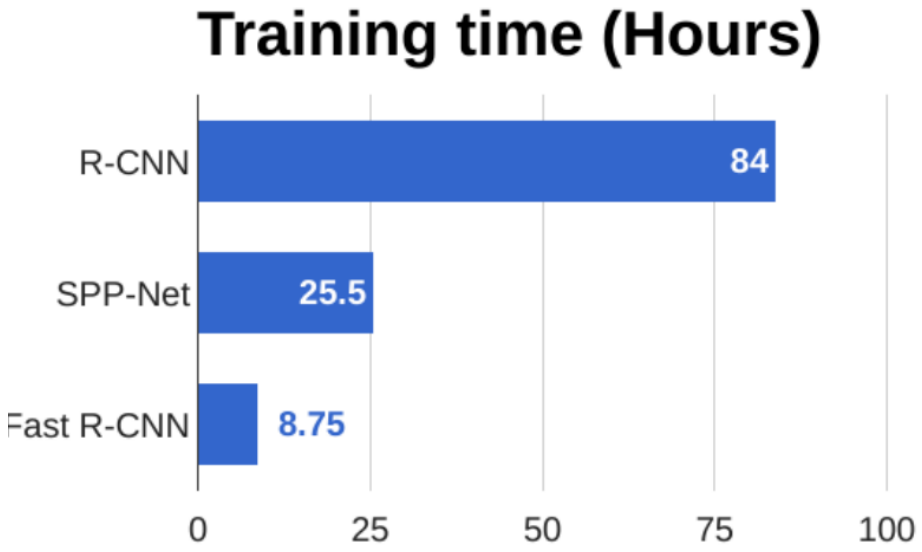


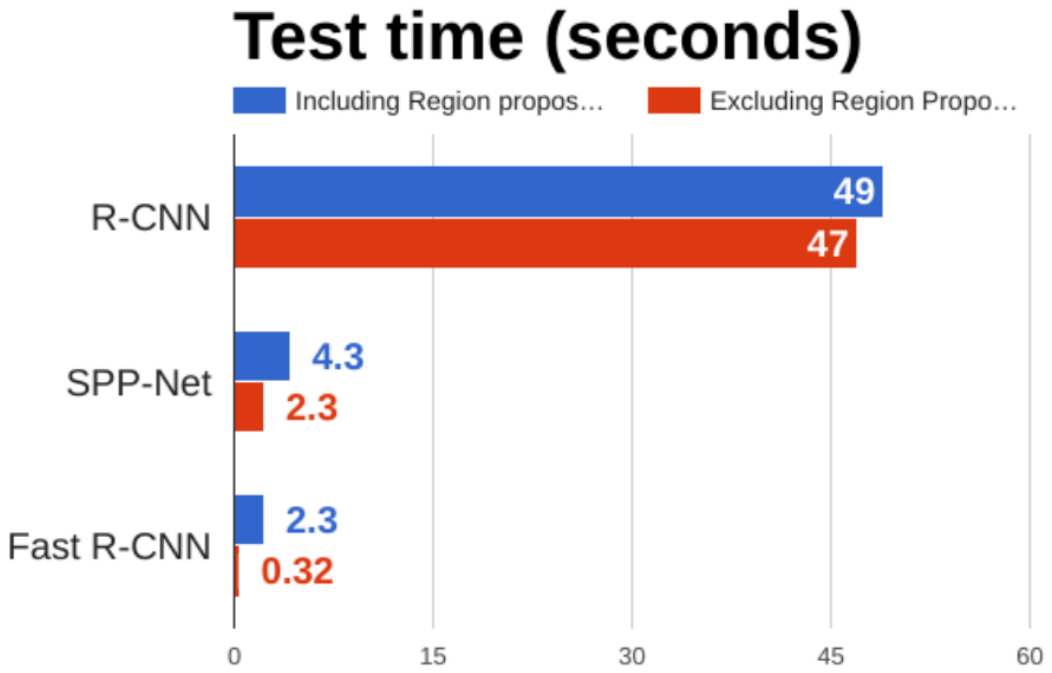


Example:
For ResNet, last stage is used as per-region network; the rest of the network is used as backbone

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

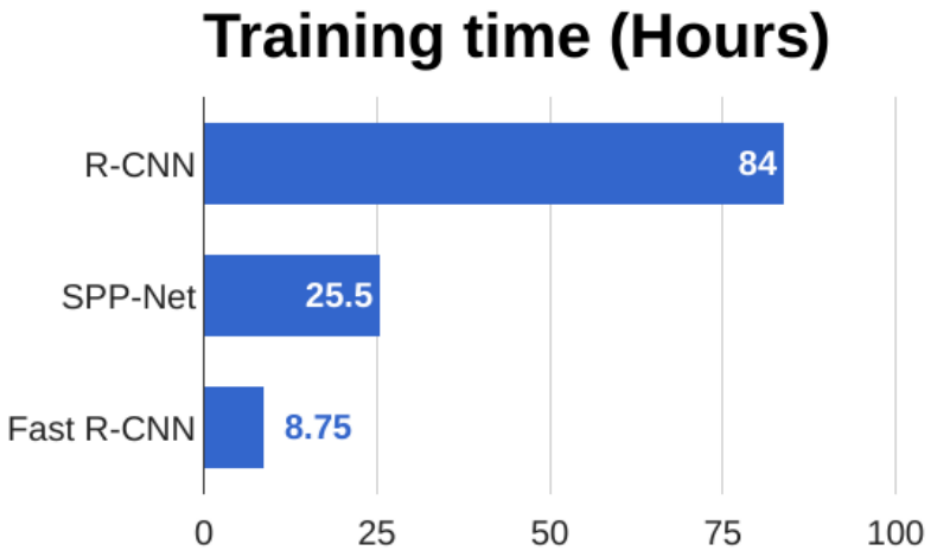
Fast R-CNN vs “Slow” R-CNN

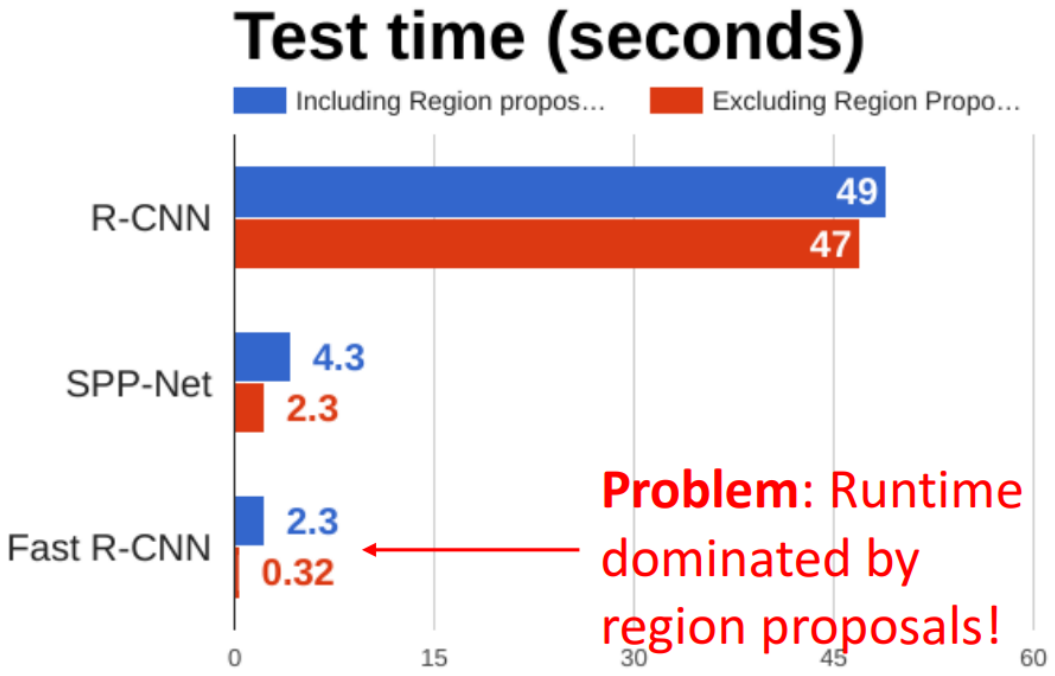




Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Fast R-CNN vs “Slow” R-CNN

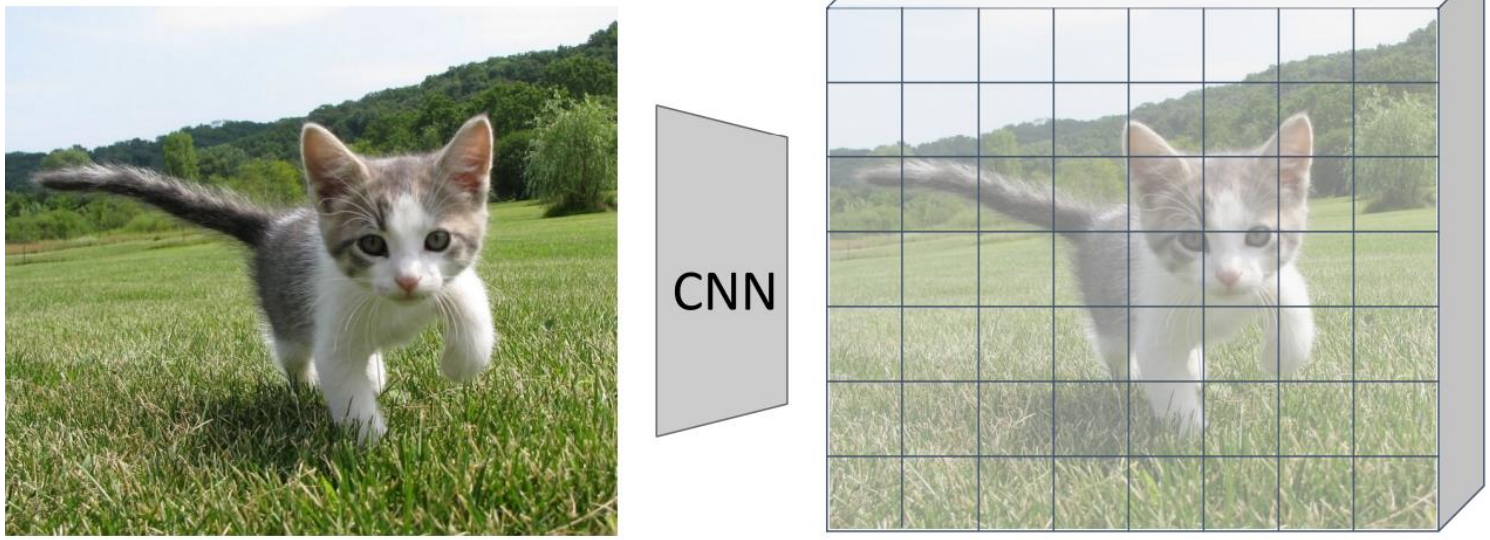




Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Region Proposal Network (RPN)

Run backbone CNN to get features aligned to input image



Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 20 x 15)

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Region Proposal Network (RPN)

Run backbone CNN to get features aligned to input image

Input Image
(e.g. 3 x 640 x 480)

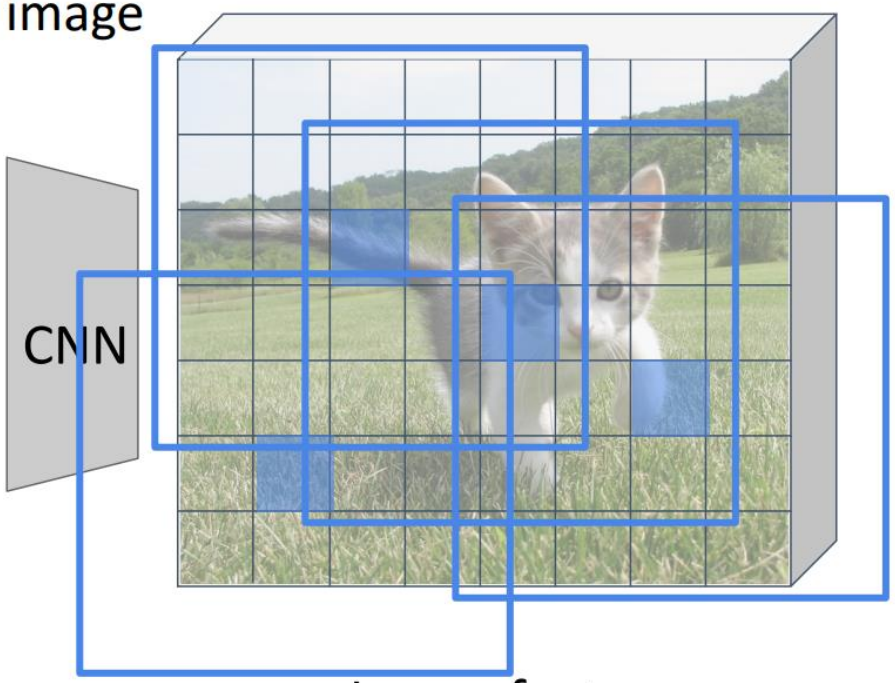


Image features
(e.g. 512 x 20 x 15)

Imagine an **anchor box** of fixed size at each point in the feature map

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Region Proposal Network (RPN)

Run backbone CNN to get features aligned to input image

Imagine an **anchor box** of fixed size at each point in the feature map

Input Image
(e.g. 3 x 640 x 480)

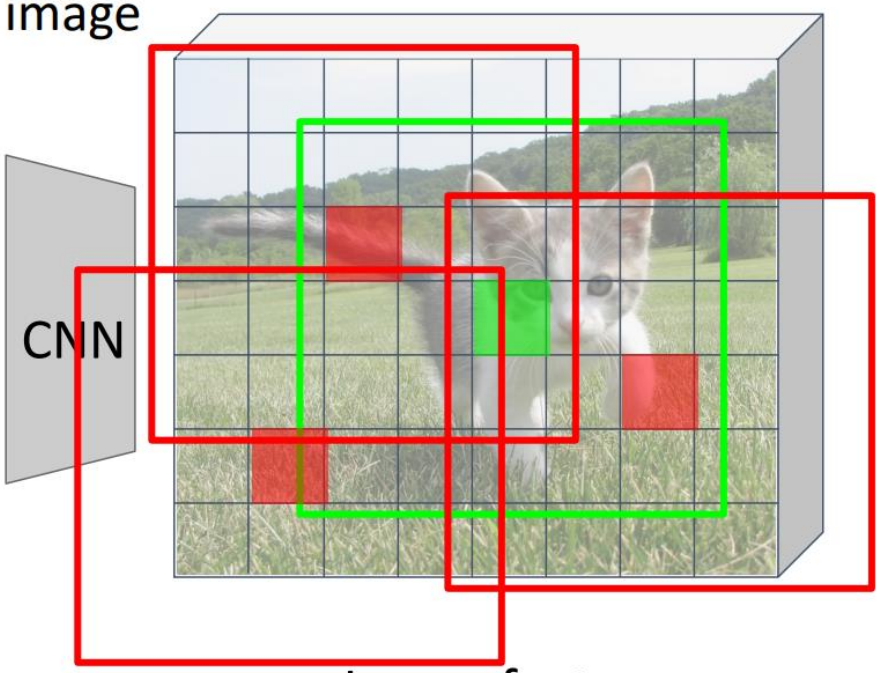


Image features
(e.g. 512 x 20 x 15)

Anchor is an object?
1 x 20 x 15

At each point, predict whether the corresponding anchor contains an object (per-cell logistic regression, predict scores with conv layer)

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Region Proposal Network (RPN)

Run backbone CNN to get features aligned to input image

Input Image
(e.g. 3 x 640 x 480)

CNN

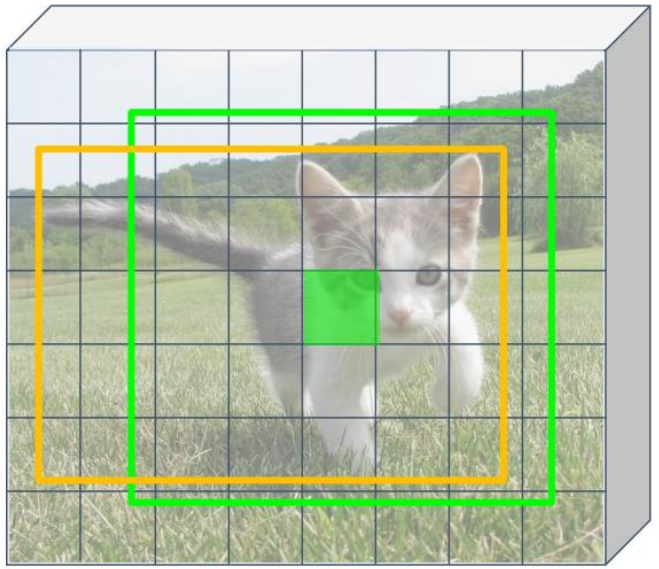


Image features
(e.g. 512 x 20 x 15)

Imagine an **anchor box** of fixed size at each point in the feature map

Conv

Anchor is an object?
1 x 20 x 15

Box transforms
4 x 20 x 15

For positive boxes, also predict a box transform to regress from **anchor box** to **object box**

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Region Proposal Network (RPN)

Run backbone CNN to get features aligned to input image

Input Image
(e.g. 3 x 640 x 480)

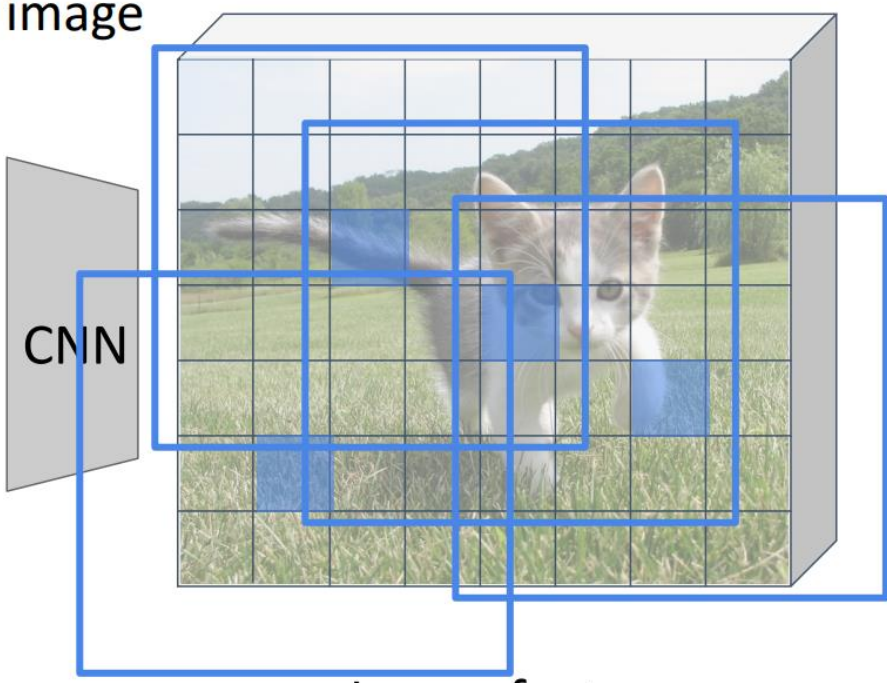
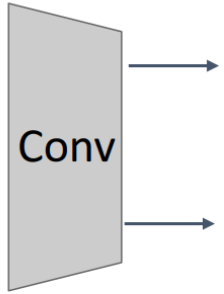


Image features
(e.g. 512 x 20 x 15)

Problem: Anchor box may have the wrong size / shape
Solution: Use **K** different anchor boxes at each point!



Anchor is an object?
K x 20 x 15

Box transforms
4K x 20 x 15

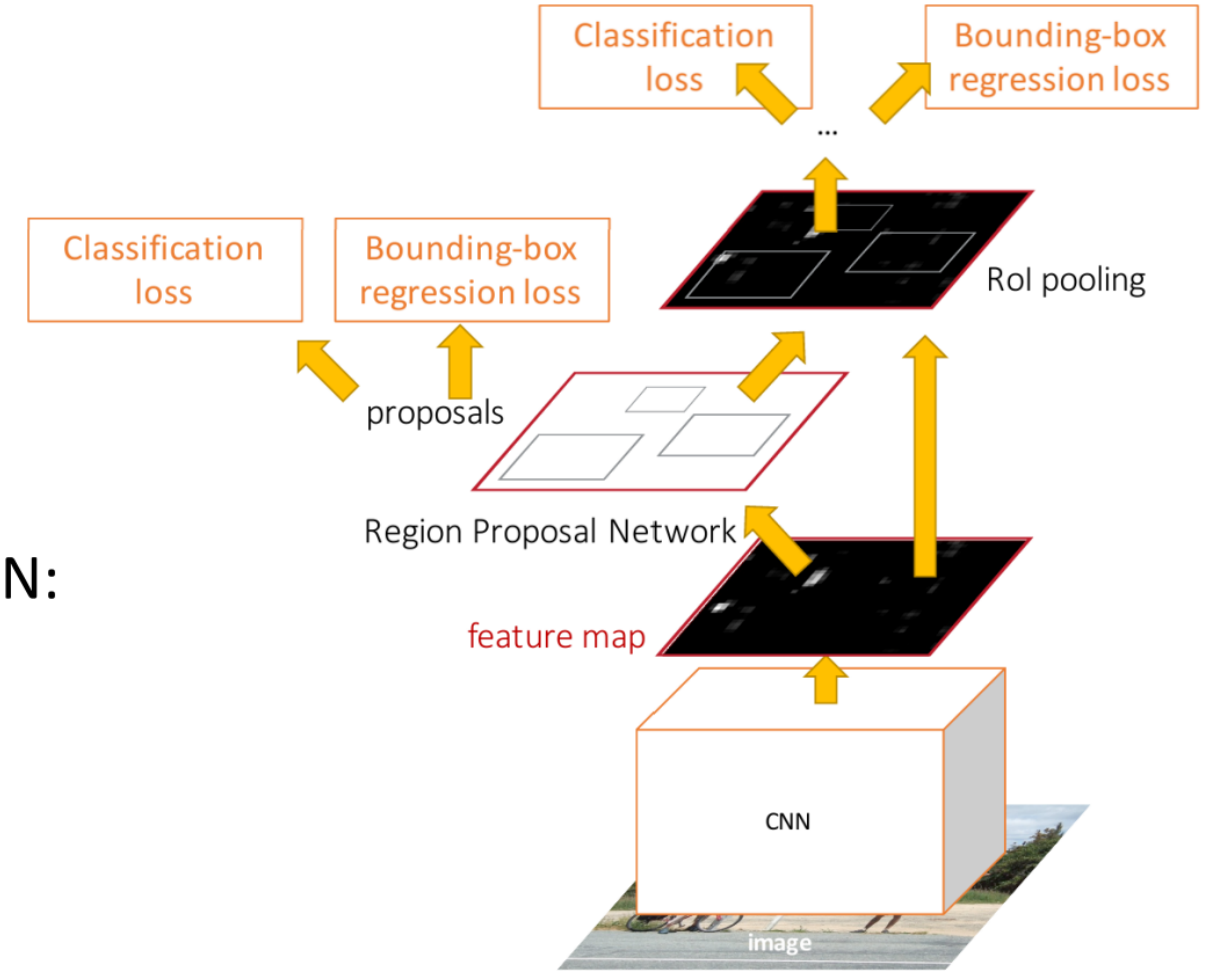
At test time: sort all K*20*15 boxes by their score, and take the top ~300 as our region proposals

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Faster R-CNN: Learnable Region Proposals [3]

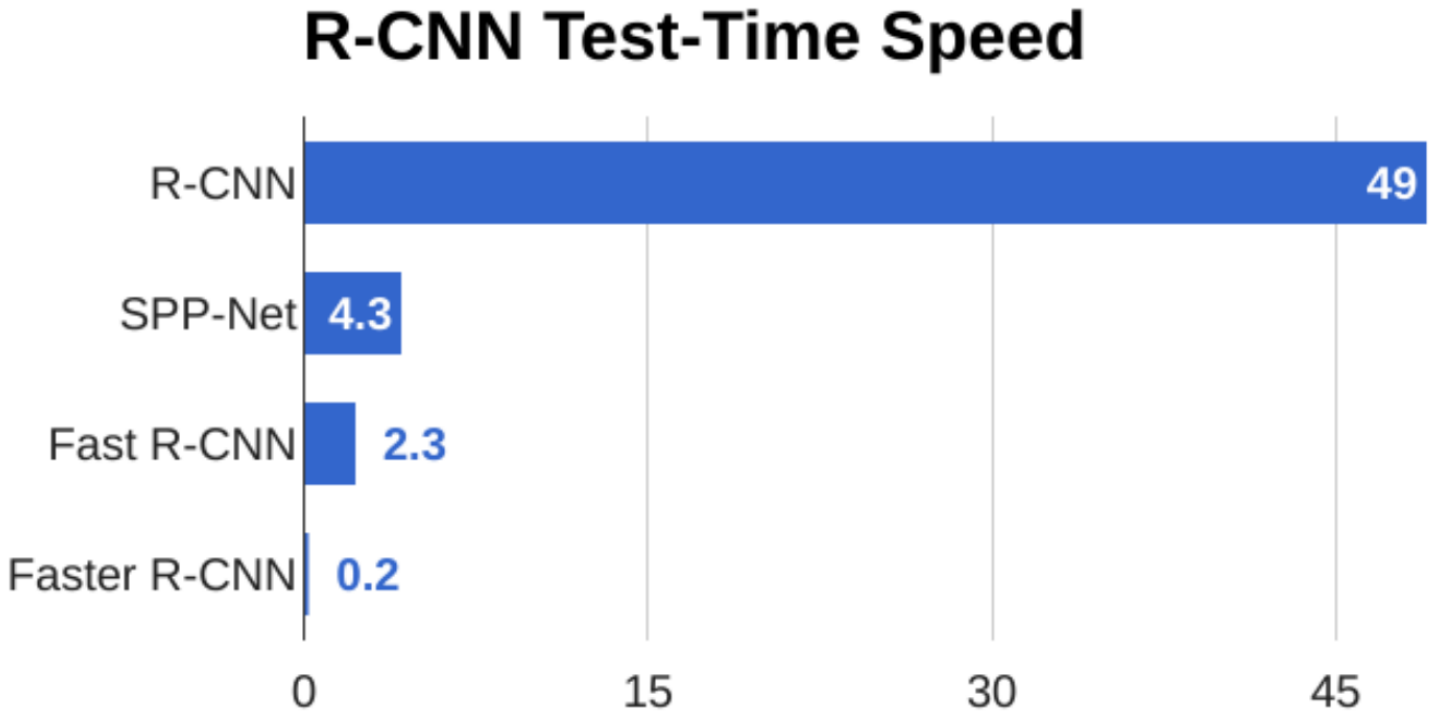
Insert **Region Proposal Network (RPN)** to predict proposals from features

Otherwise same as Fast R-CNN: Crop features for each proposal, classify each one



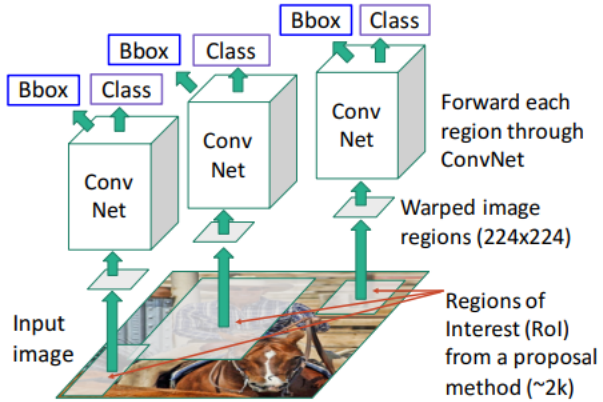
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/
[3] Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. arXiv [Cs.CV]. Retrieved from http://arxiv.org/abs/1506.01497

Faster R-CNN: Learnable Region Proposals [3]

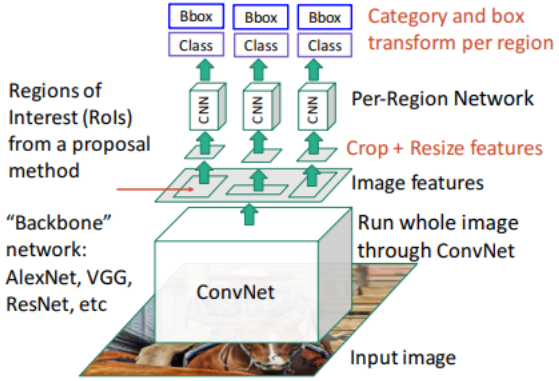


R-CNN Family Comparison

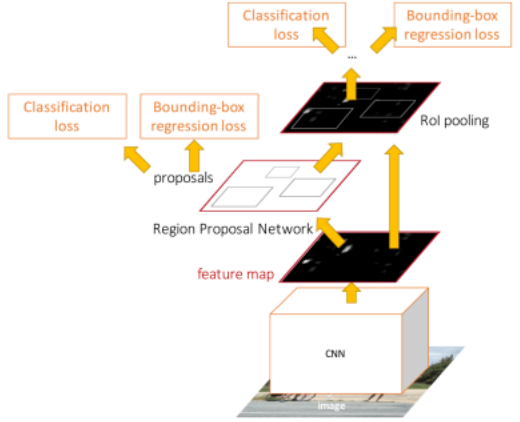
"Slow" R-CNN: Run CNN independently for each region



Fast R-CNN: Apply differentiable cropping to shared image features



Faster R-CNN: Compute proposals with CNN



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

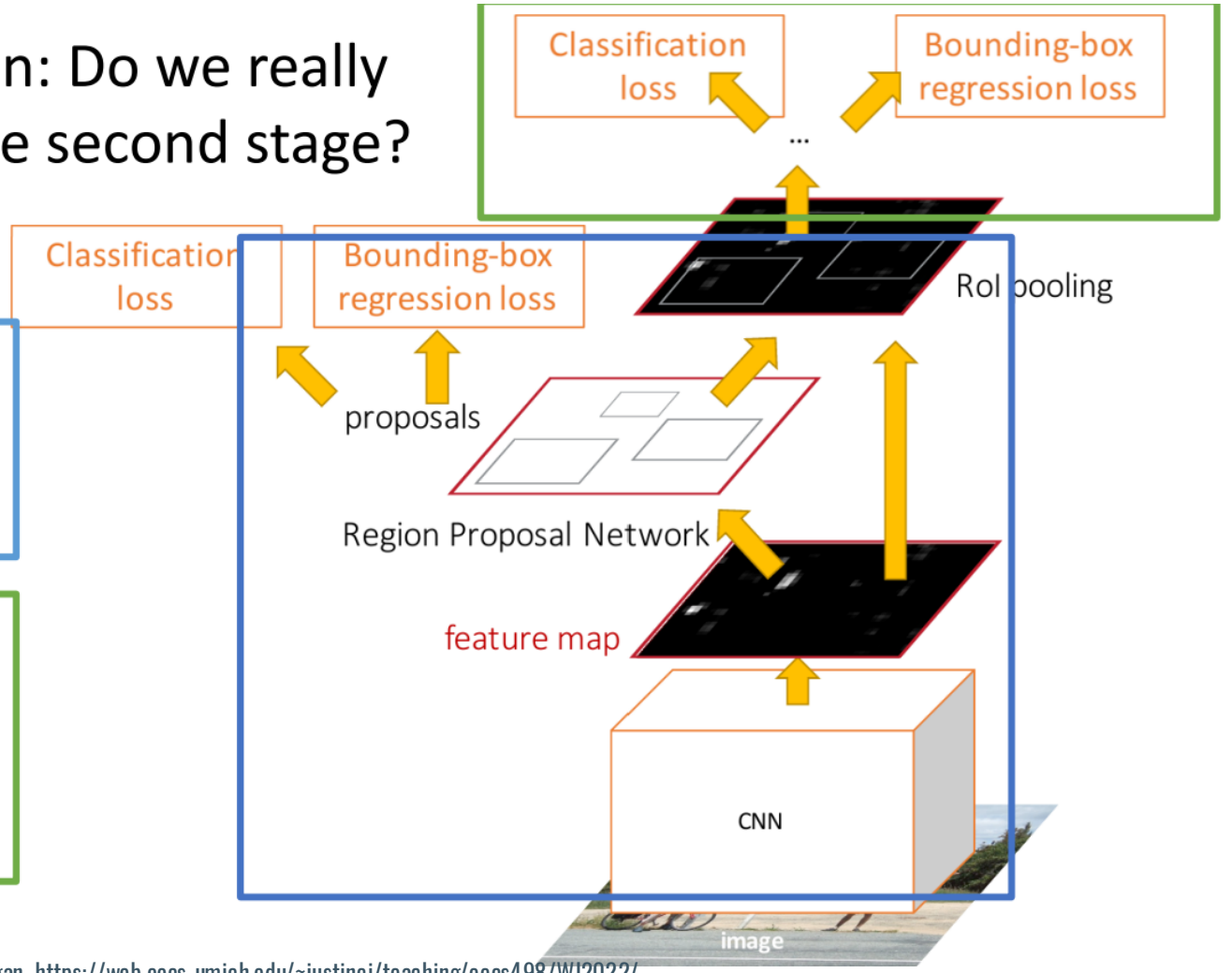
R-CNN Family

Question: Do we really need the second stage?

Faster R-CNN is a
Two-stage object detector

First stage: Run once per image
- Backbone network
- Region proposal network

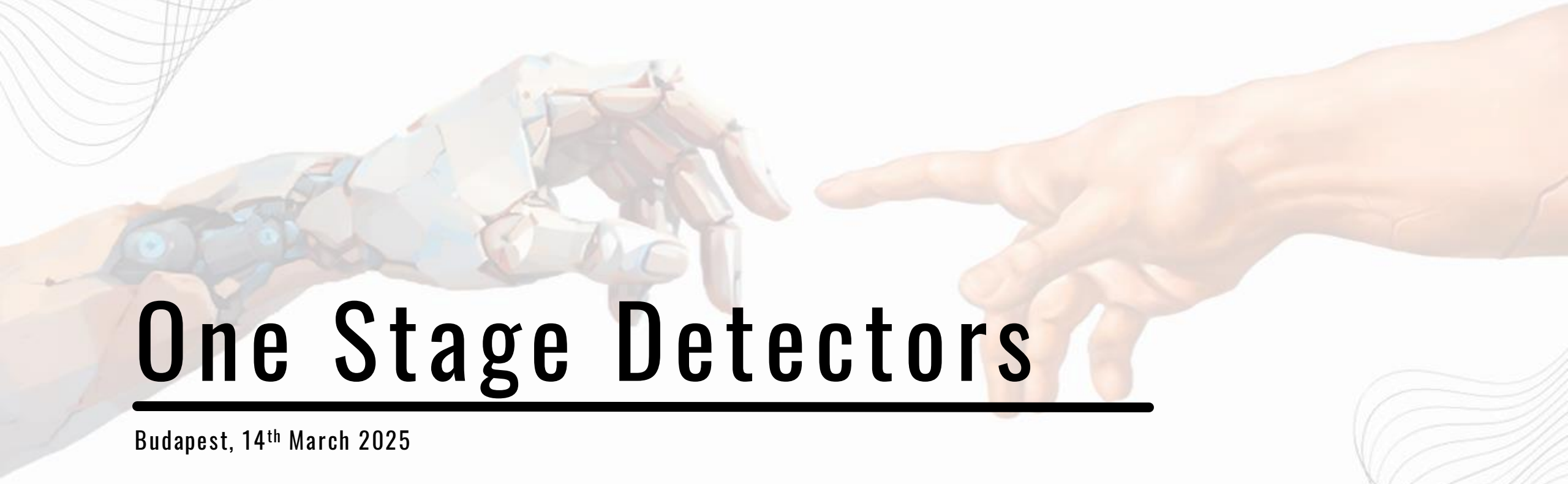
Second stage: Run once per region
- Crop features: RoI pool / align
- Predict object class
- Prediction bbox offset



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Deep Network Development

Lecture 5.



One Stage Detectors

Budapest, 14th March 2025

1 Two Stage Detectors

2 One Stage Detectors

3 Object Detection Metrics

Problems with Faster R-CNN

Run backbone CNN to get features aligned to input image

Input Image
(e.g. 3 x 640 x 480)

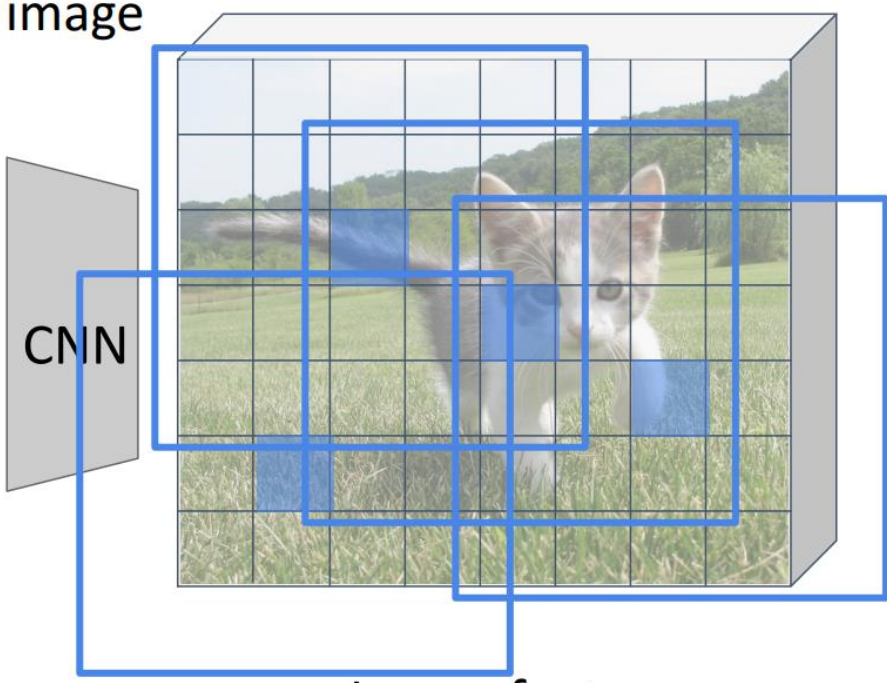
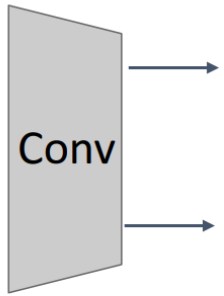


Image features
(e.g. 512 x 20 x 15)

Problem: Anchor box may have the wrong size / shape
Solution: Use **K** different anchor boxes at each point!



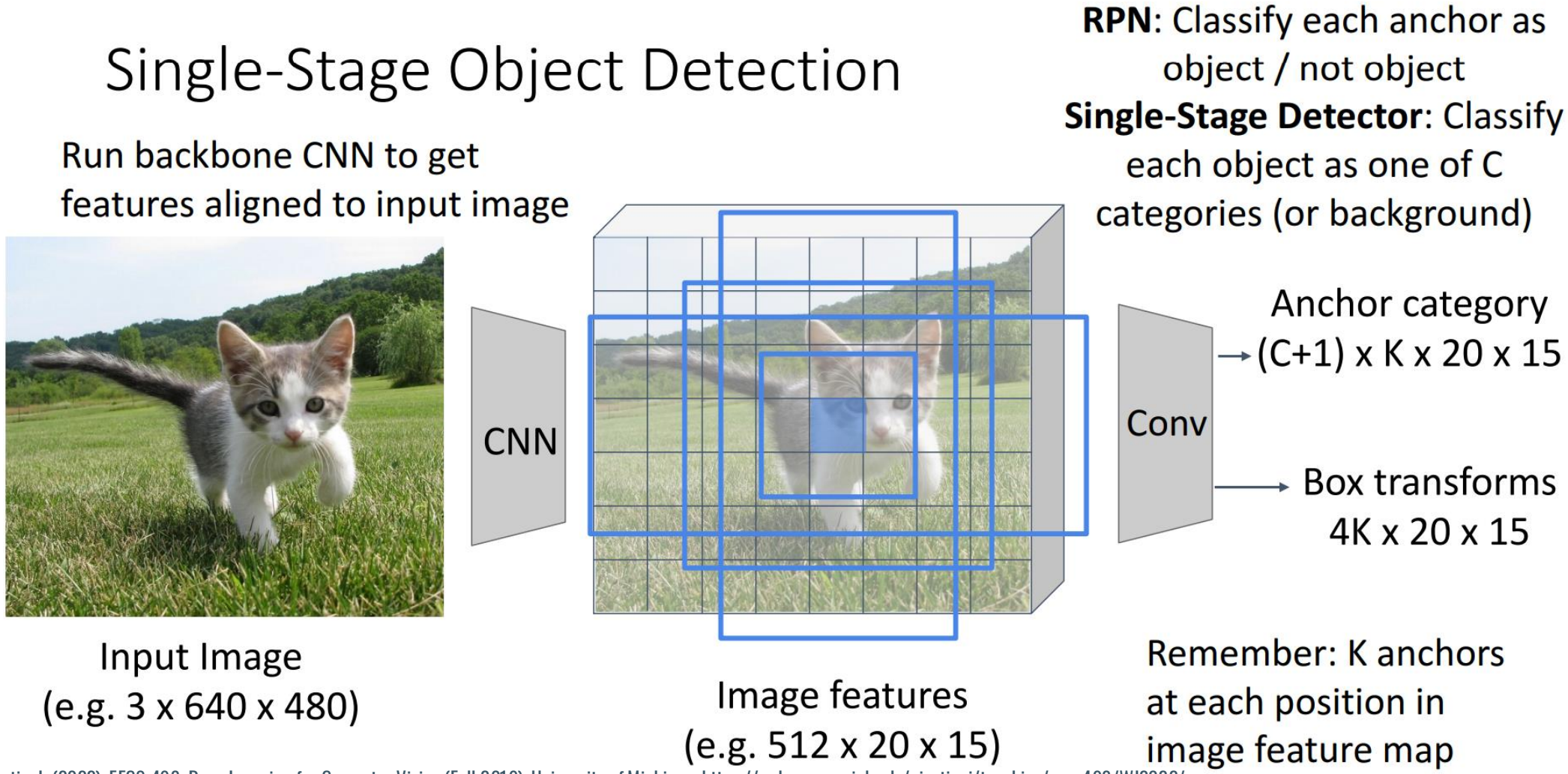
Anchor is an object?
K x 20 x 15

Box transforms
4K x 20 x 15

At test time: sort all K*20*15 boxes by their score, and take the top ~300 as our region proposals

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Problems with Faster R-CNN

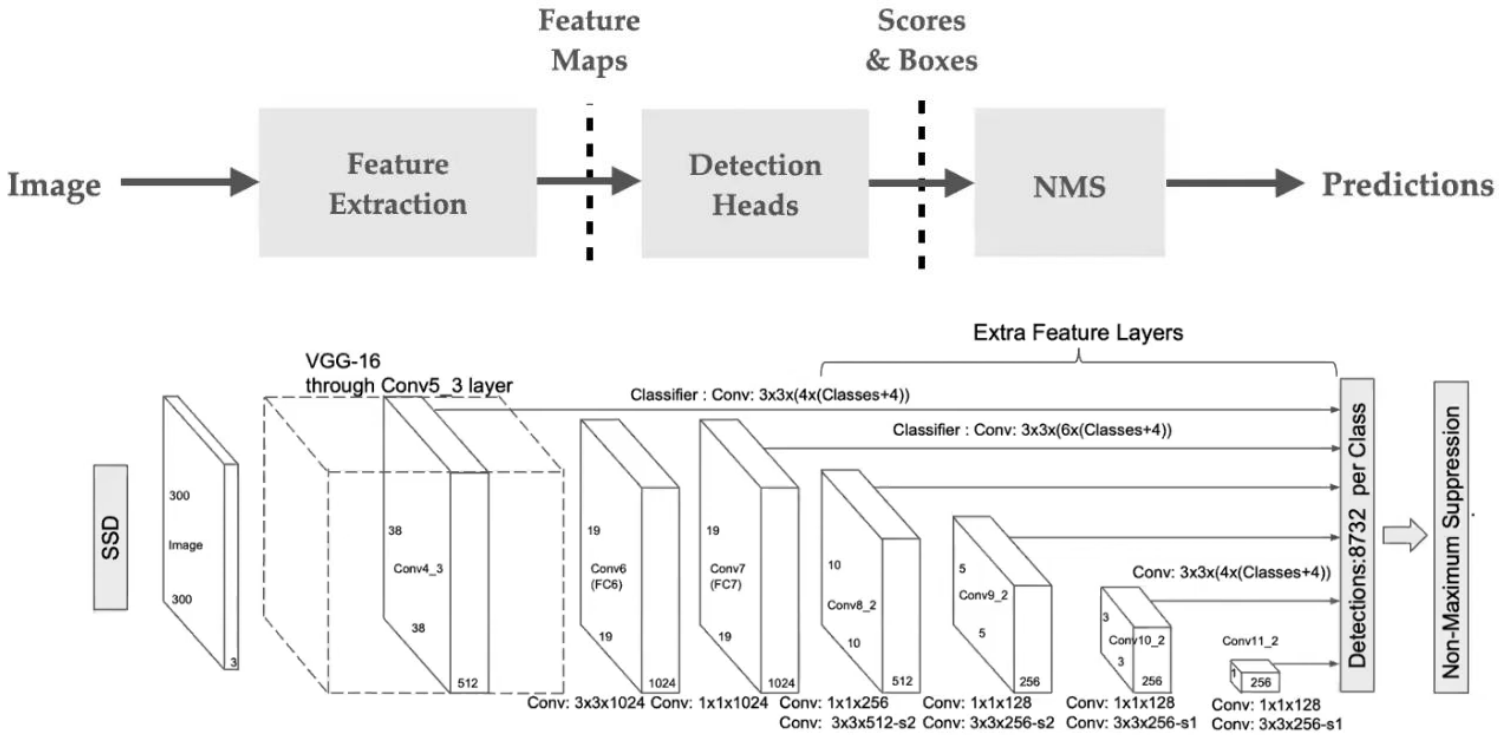


Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Single-Shot Detectors (SSD) [4]

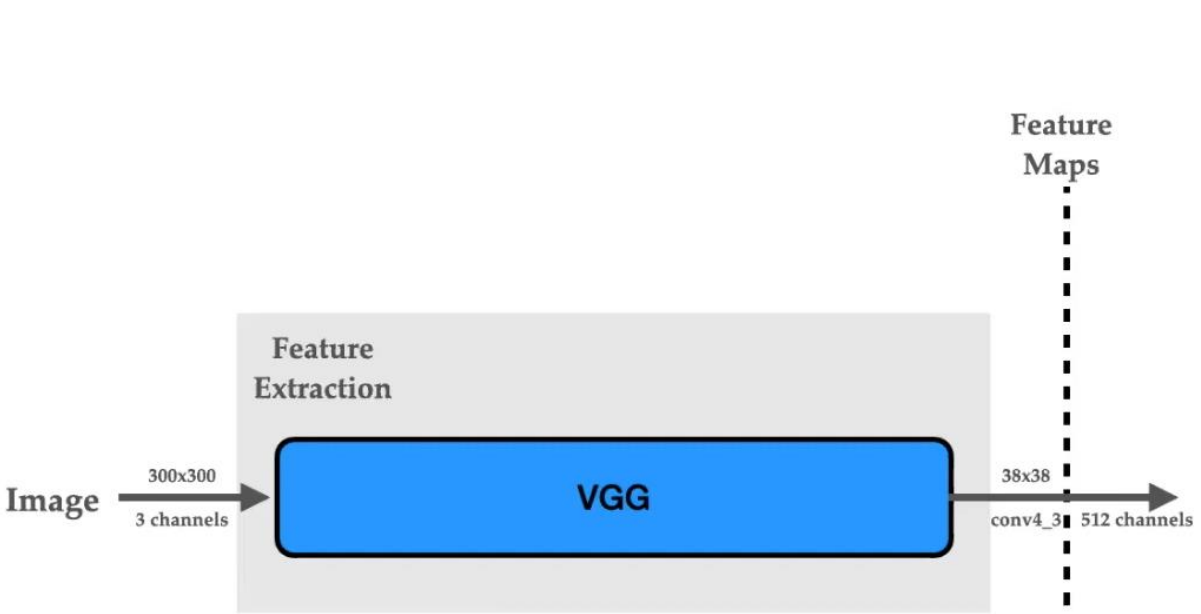
Key Ideas:

1. Multiple Layers → handle different scales
2. Different filters predict boxes of different shapes/sizes



[4] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In Computer Vision – ECCV 2016 (pp. 21–37). doi:10.1007/978-3-319-46448-0_2

Single-Shot Detectors (SSD) [4]

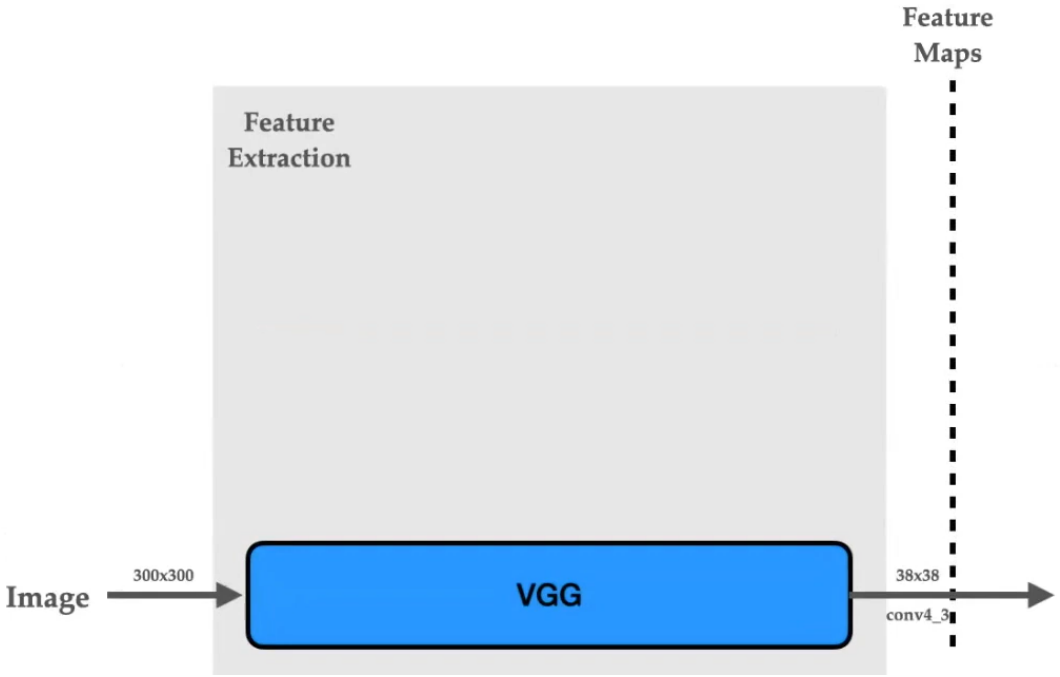


VGG

conv3-64
conv3-64
conv3-128
conv3-128
conv3-256
conv3-256
conv3-256
conv3-256
conv3-512
conv3-512
conv3-512
conv3-512
conv3-512
conv3-512
conv3-512
conv3-512
maxpool
FC-4096
FC-4096
FC-1000
soft-max

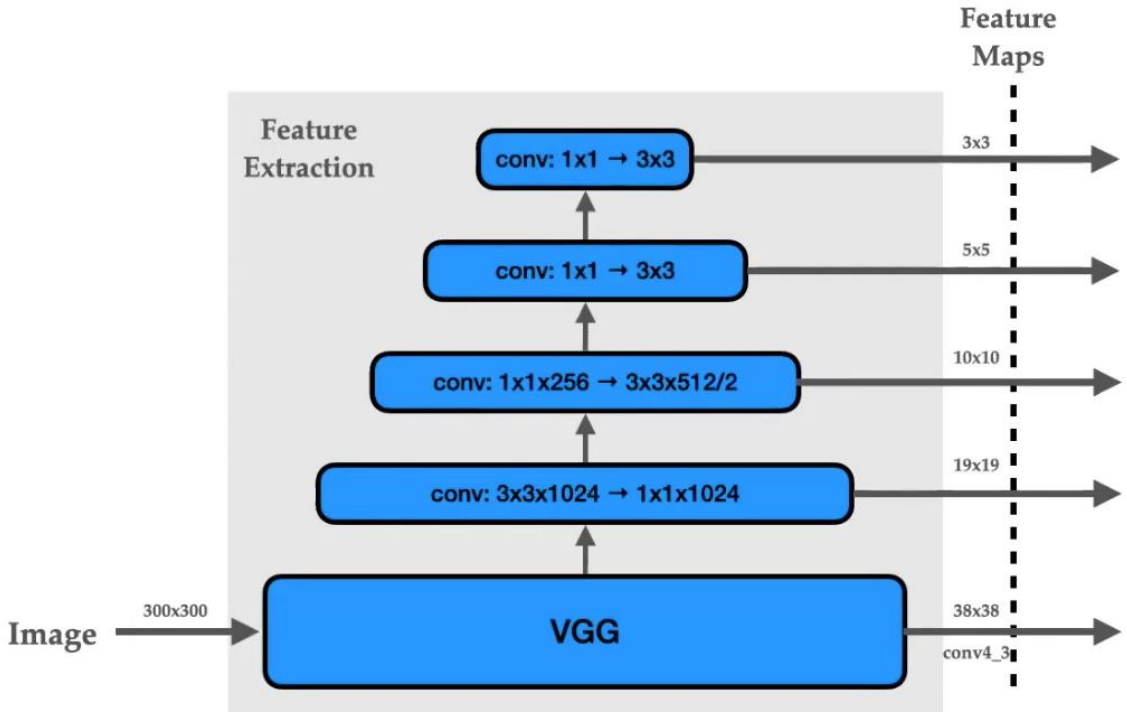
[4] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In Computer Vision – ECCV 2016 (pp. 21–37). doi:10.1007/978-3-319-46448-0_2

Single-Shot Detectors (SSD) [4]



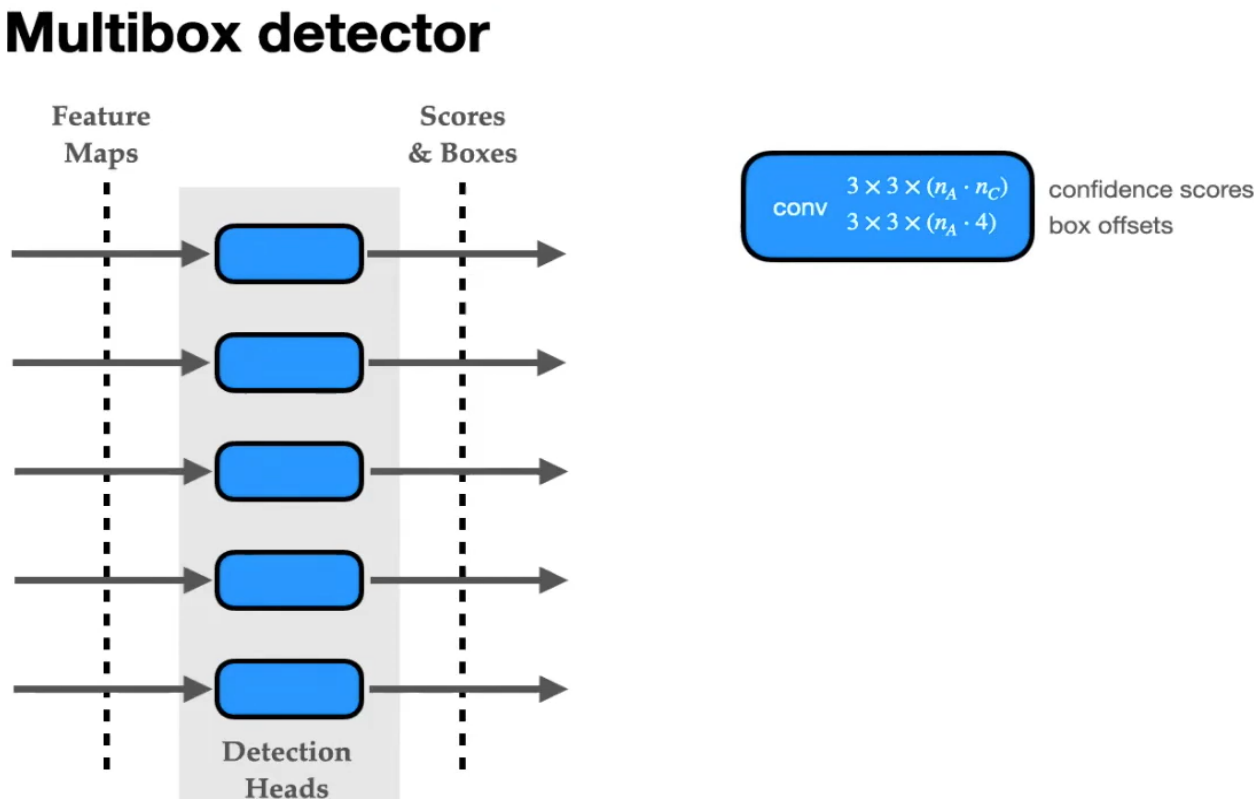
[4] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In Computer Vision – ECCV 2016 (pp. 21–37). doi:10.1007/978-3-319-46448-0_2

Single-Shot Detectors (SSD) [4]



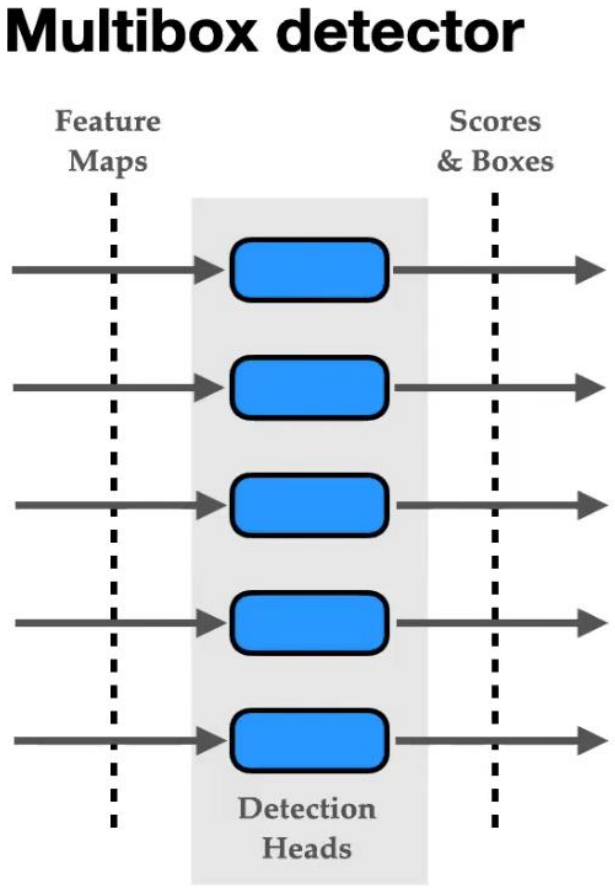
[4] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In Computer Vision – ECCV 2016 (pp. 21–37). doi:10.1007/978-3-319-46448-0_2

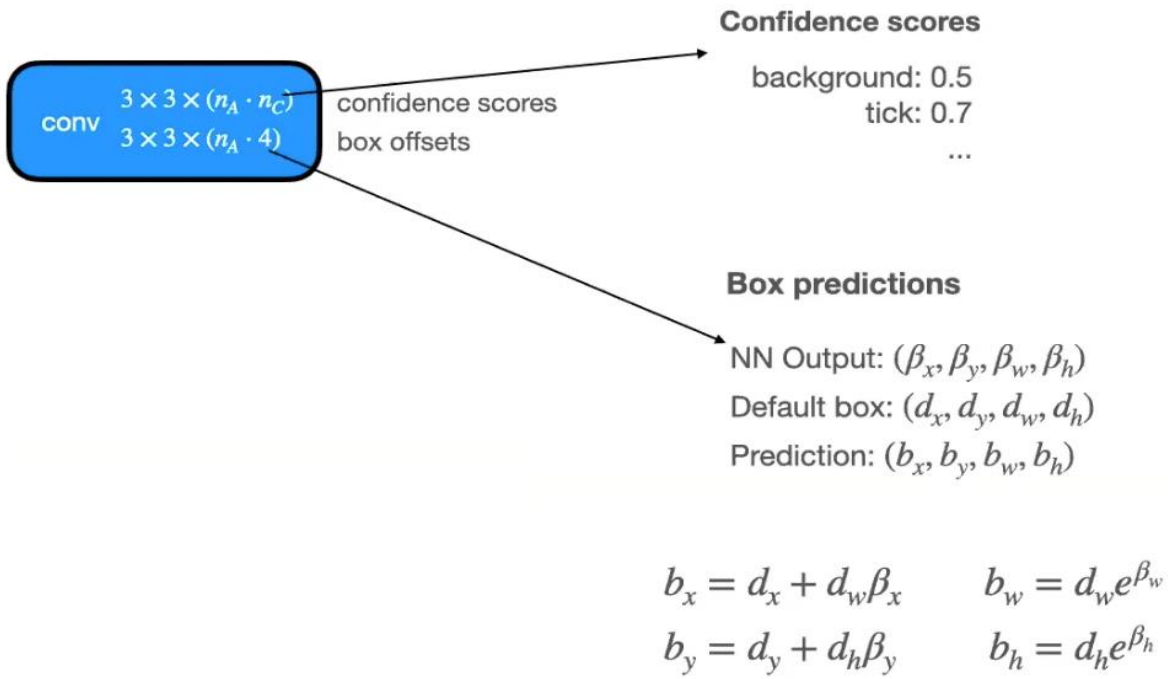
Single-Shot Detectors (SSD) [4]



[4] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In Computer Vision – ECCV 2016 (pp. 21–37). doi:10.1007/978-3-319-46448-0_2

Single-Shot Detectors (SSD) [4]

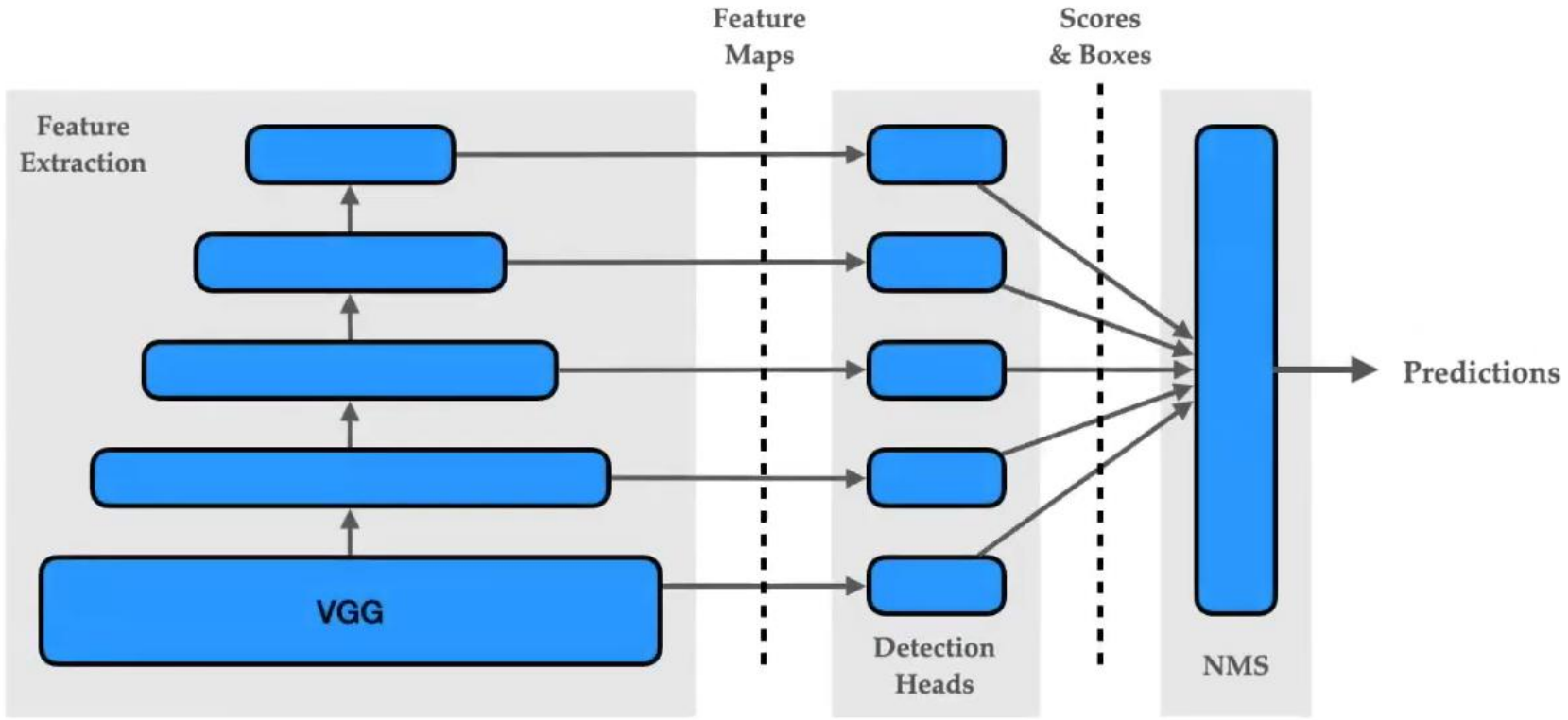




$$b_x = d_x + d_w \beta_x \qquad b_w = d_w e^{\beta_w}$$
$$b_y = d_y + d_h \beta_y \qquad b_h = d_h e^{\beta_h}$$

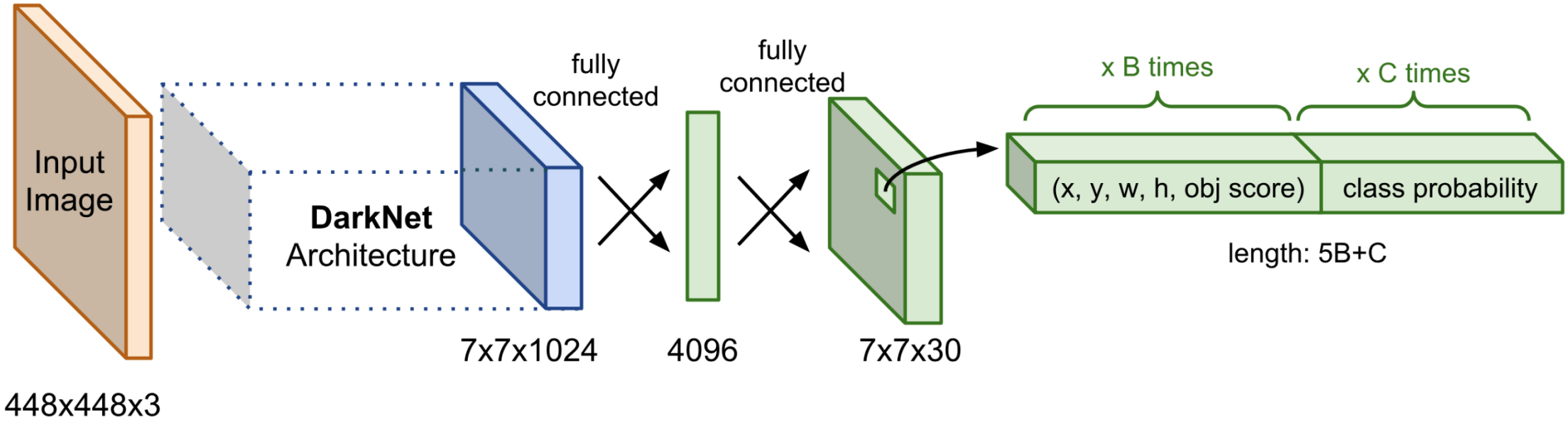
[4] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In Computer Vision – ECCV 2016 (pp. 21–37). doi:10.1007/978-3-319-46448-0_2

Single-Shot Detectors (SSD) [4]



[4] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In Computer Vision – ECCV 2016 (pp. 21–37). doi:10.1007/978-3-319-46448-0_2

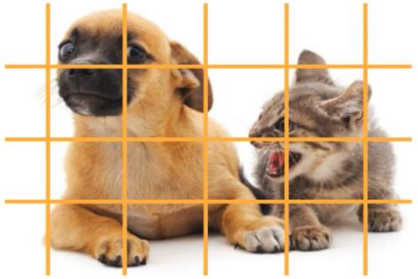
You Only Look Once (YOLO) [5]



[5] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. arXiv [Cs.CV]. Retrieved from http://arxiv.org/abs/1506.02640

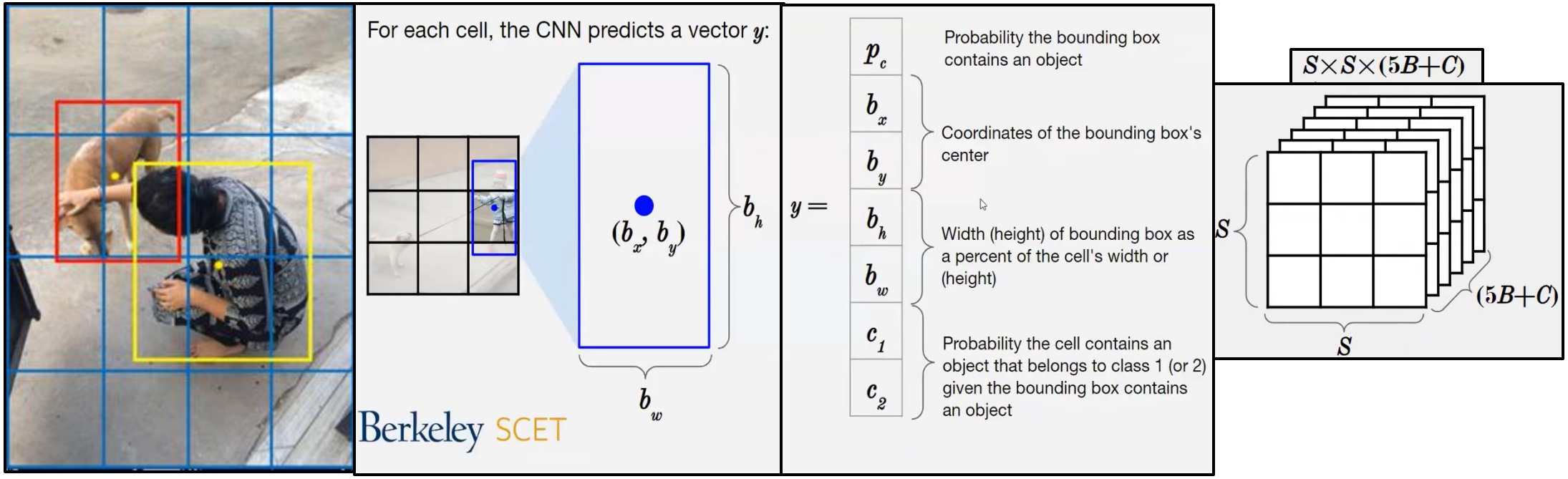
You Only Look Once (YOLO) [5]

- Anchor boxes are highly overlapped in SSD
- YOLO cuts the input image uniformly into S x S anchor boxes
- Each anchor box predicts B bounding boxes
- V2 and V3 add more improvements



[5] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. arXiv [Cs.CV]. Retrieved from http://arxiv.org/abs/1506.02640

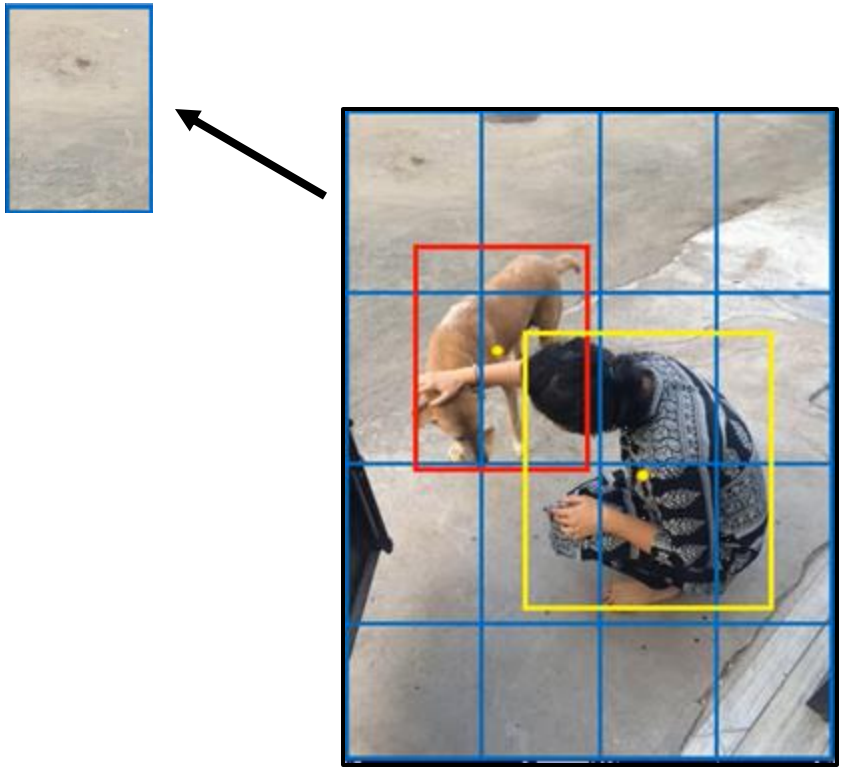
You Only Look Once (YOLO) [5]



[5] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. arXiv [Cs.CV]. Retrieved from http://arxiv.org/abs/1506.02640

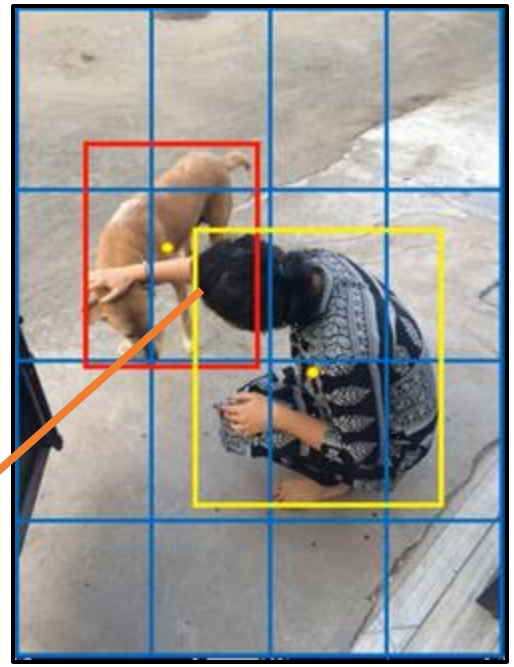
You Only Look Once (YOLO) - Format

$$\begin{pmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{pmatrix} \begin{pmatrix} 0 \\ - \\ - \\ - \\ - \\ - \\ - \end{pmatrix}$$



You Only Look Once (YOLO) - Format

$$\begin{pmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{pmatrix} \quad \begin{pmatrix} 0 \\ - \\ - \\ - \\ - \\ - \\ - \end{pmatrix}$$



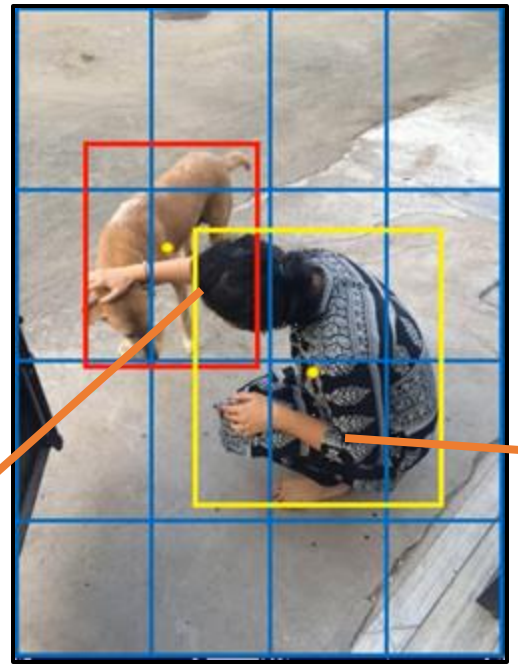
$$\begin{pmatrix} 1 \\ 0.05 \\ 0.3 \\ 2 \\ 1.3 \\ 1 \\ 0 \end{pmatrix}$$

(0,0)

(1,1)

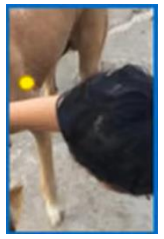
You Only Look Once (YOLO) - Format

$$\begin{pmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{pmatrix} \quad \begin{pmatrix} 0 \\ - \\ - \\ - \\ - \\ - \\ - \end{pmatrix}$$

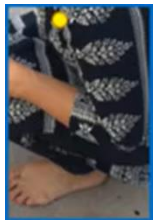


$$\begin{pmatrix} 1 \\ 0.05 \\ 0.3 \\ 2 \\ 1.3 \\ 1 \\ 0 \end{pmatrix}$$

(0,0)

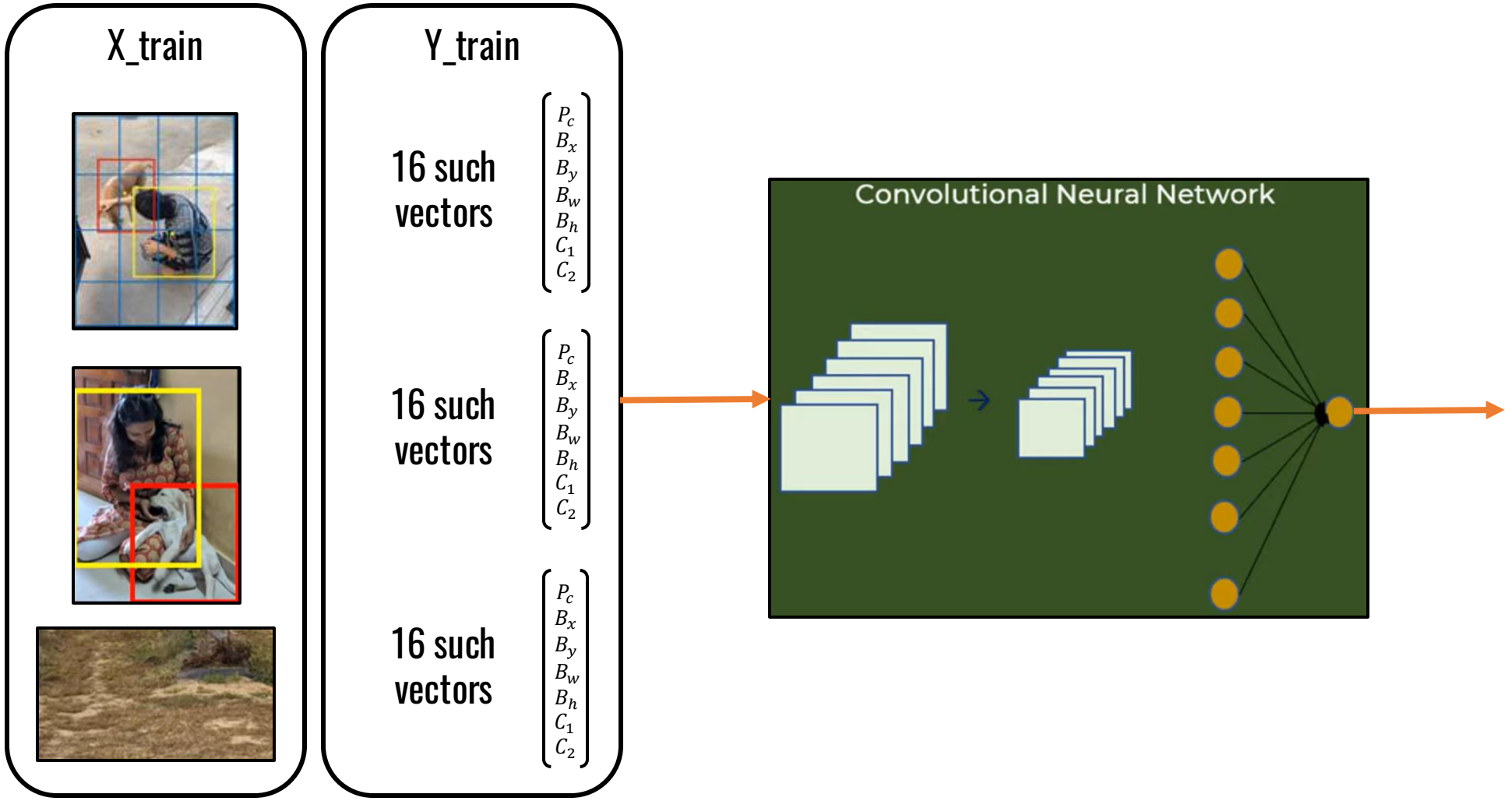


(1,1)



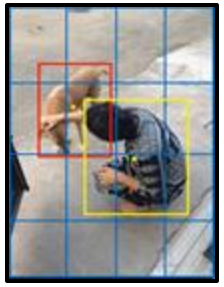
$$\begin{pmatrix} 1 \\ 0.32 \\ 0.02 \\ 3 \\ 2 \\ 0 \\ 1 \end{pmatrix}$$

You Only Look Once (YOLO) - Training



You Only Look Once (YOLO) - Prediction

X_train



Y_train

16 such vectors

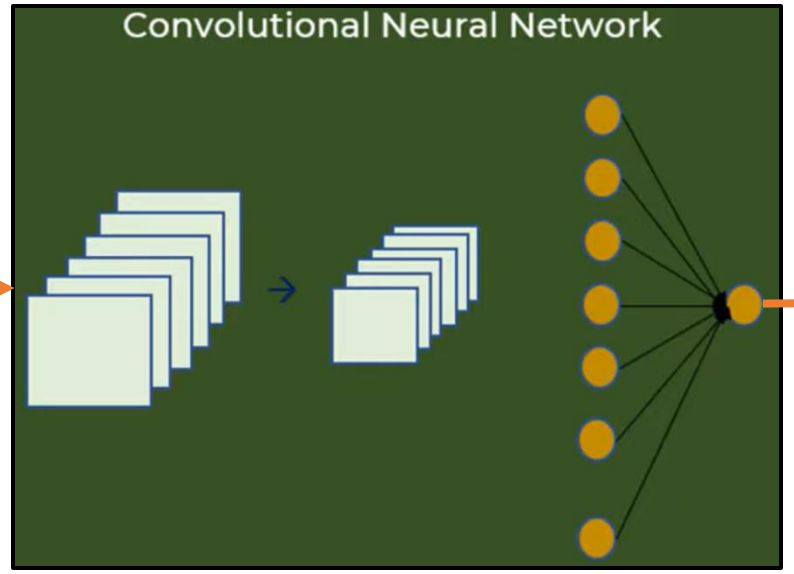
$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix}$$

16 such vectors

$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix}$$

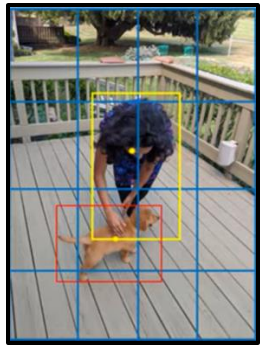
16 such vectors

$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix}$$



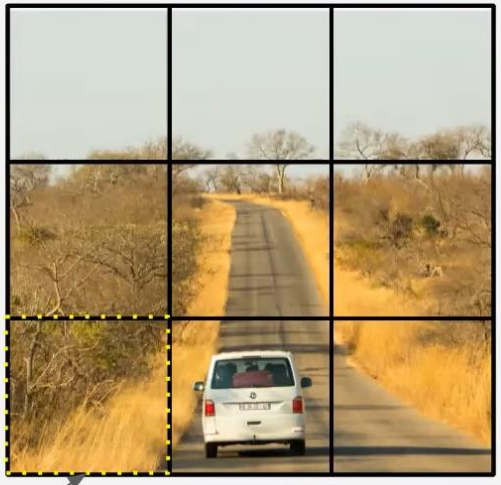
16 such vectors

$$\begin{bmatrix} P_c \\ B_x \\ B_y \\ B_w \\ B_h \\ C_1 \\ C_2 \end{bmatrix}$$



You Only Look Once (YOLO)

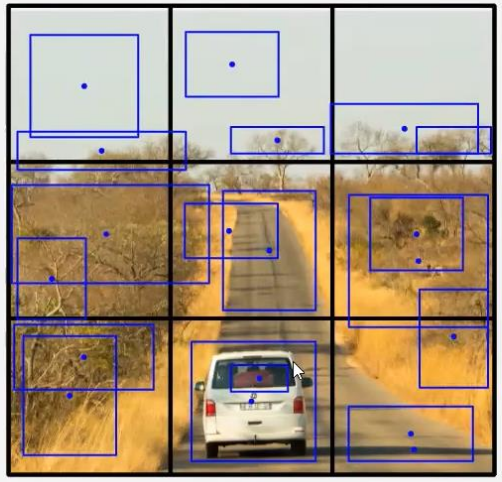
1. **Divide the image into cells with an S x S grid.**



$S = 3$

Cell

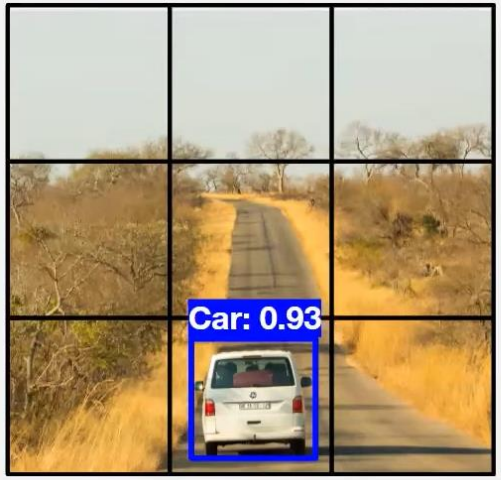
2. **Each cell predicts B bounding boxes.**



$B = 2$

A cell is responsible for detecting an object if the object's bounding box falls within the cell. (Notice that each cell has 2 blue dots.)

3. **Return bounding boxes above confidence threshold.**



All other bounding boxes have a confidence probability less than the threshold (say 0.90) so they are suppressed.

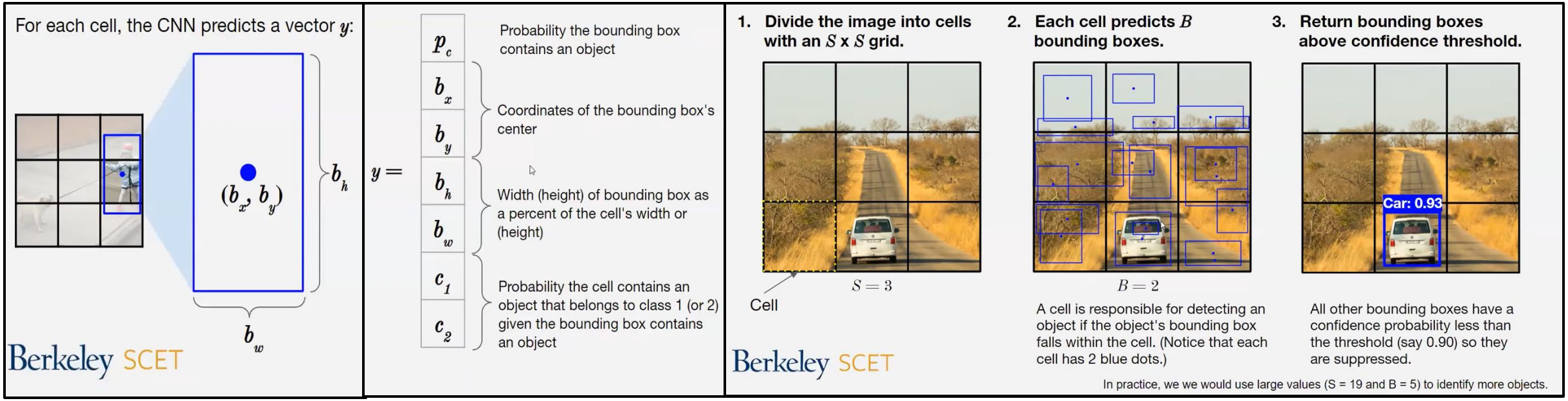
In practice, we we would use large values (S = 19 and B = 5) to identify more objects.

Berkeley SCET

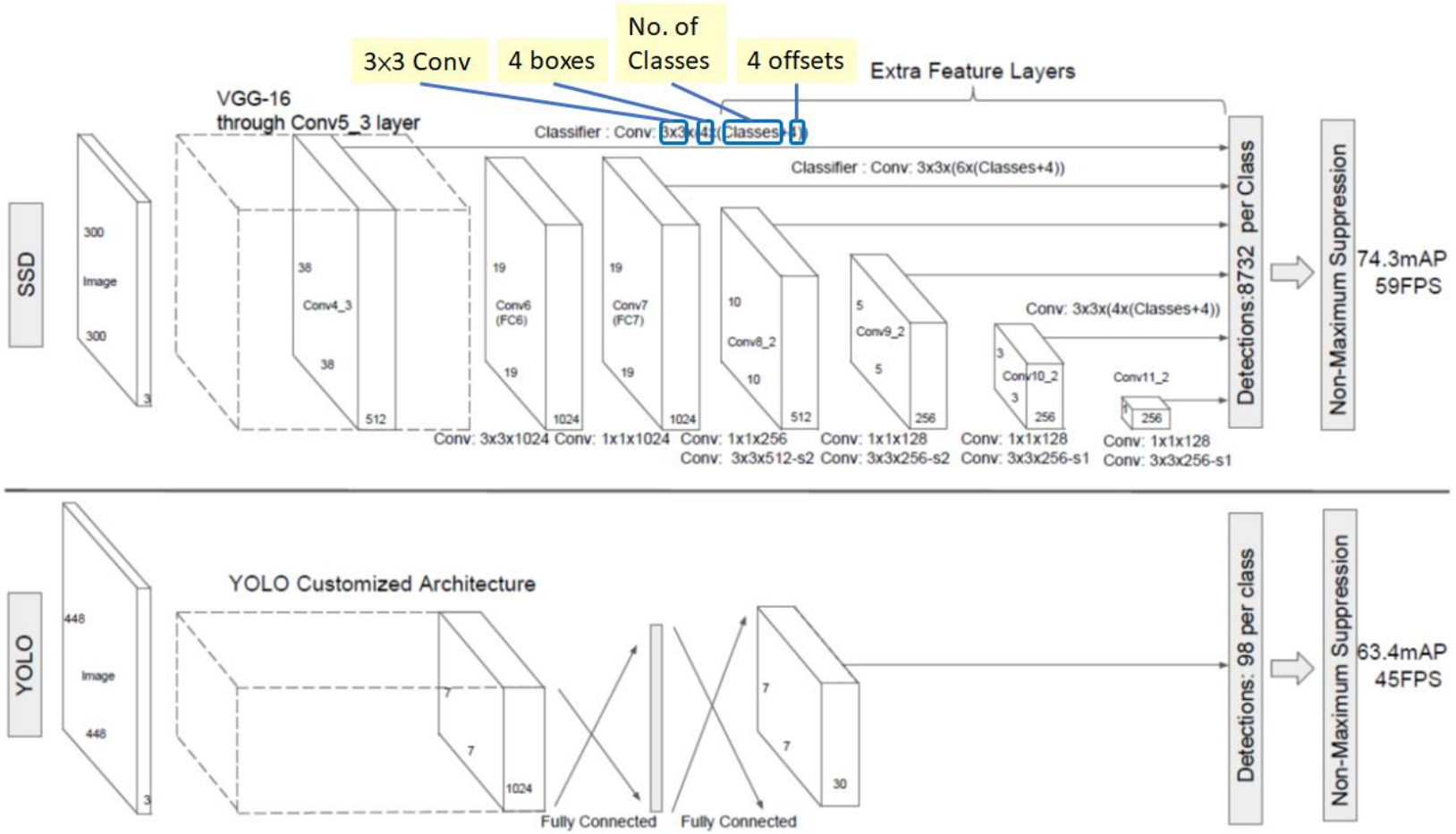
You Only Look Once (YOLO)

Please check the following links for a detailed explanation of YOLO versions:

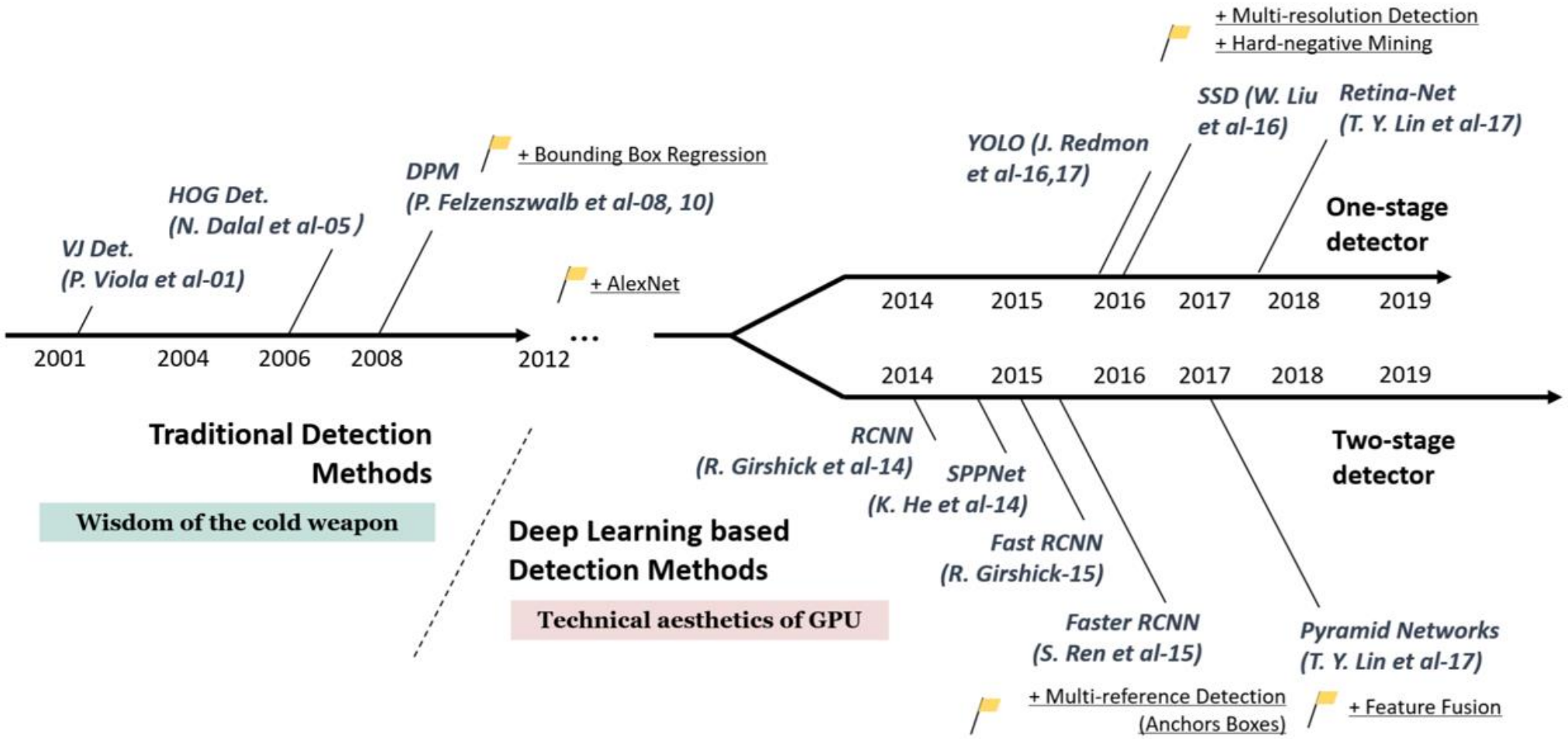
- https://www.v7labs.com/blog/yolo-object-detection
- https://www.datacamp.com/blog/yolo-object-detection-explained
- Video comparison



SSD vs YOLO

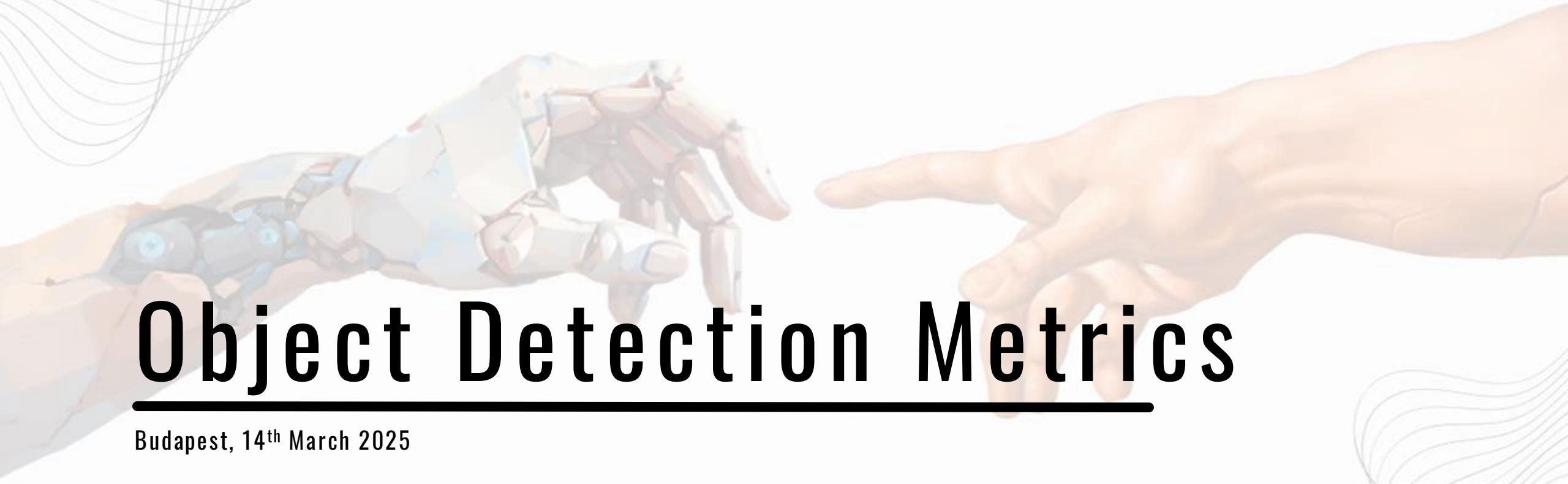

3x3 Conv
4 boxes
No. of Classes
4 offsets
Extra Feature Layers
VGG-16
through Conv5_3 layer
Classifier : Conv: 3x3x(4x(Classes+4))
Classifier : Conv: 3x3x(6x(Classes+4))
Conv: 3x3x(4x(Classes+4))
SSD
300
Image
300
3
38
Conv4_3
38
512
19
Conv6
(FC6)
19
1024
19
Conv7
(FC7)
19
1024
10
Conv8_2
10
512
5
Conv9_2
5
256
3
Conv10_2
3
256
Conv11_2
1
256
Detections:8732 per Class
Non-Maximum Suppression
74.3mAP
59FPS
Conv: 3x3x1024
Conv: 1x1x1024
Conv: 1x1x256
Conv: 3x3x512-s2
Conv: 1x1x128
Conv: 3x3x256-s2
Conv: 1x1x128
Conv: 3x3x256-s1
Conv: 1x1x128
Conv: 3x3x256-s1
YOLO
448
Image
448
3
YOLO Customized Architecture
7
7
1024
Fully Connected
Fully Connected
7
7
30
Detections: 98 per class
Non-Maximum Suppression
63.4mAP
45FPS

Other Methods



Deep Network Development

Lecture 5.



Object Detection Metrics

Budapest, 14th March 2025

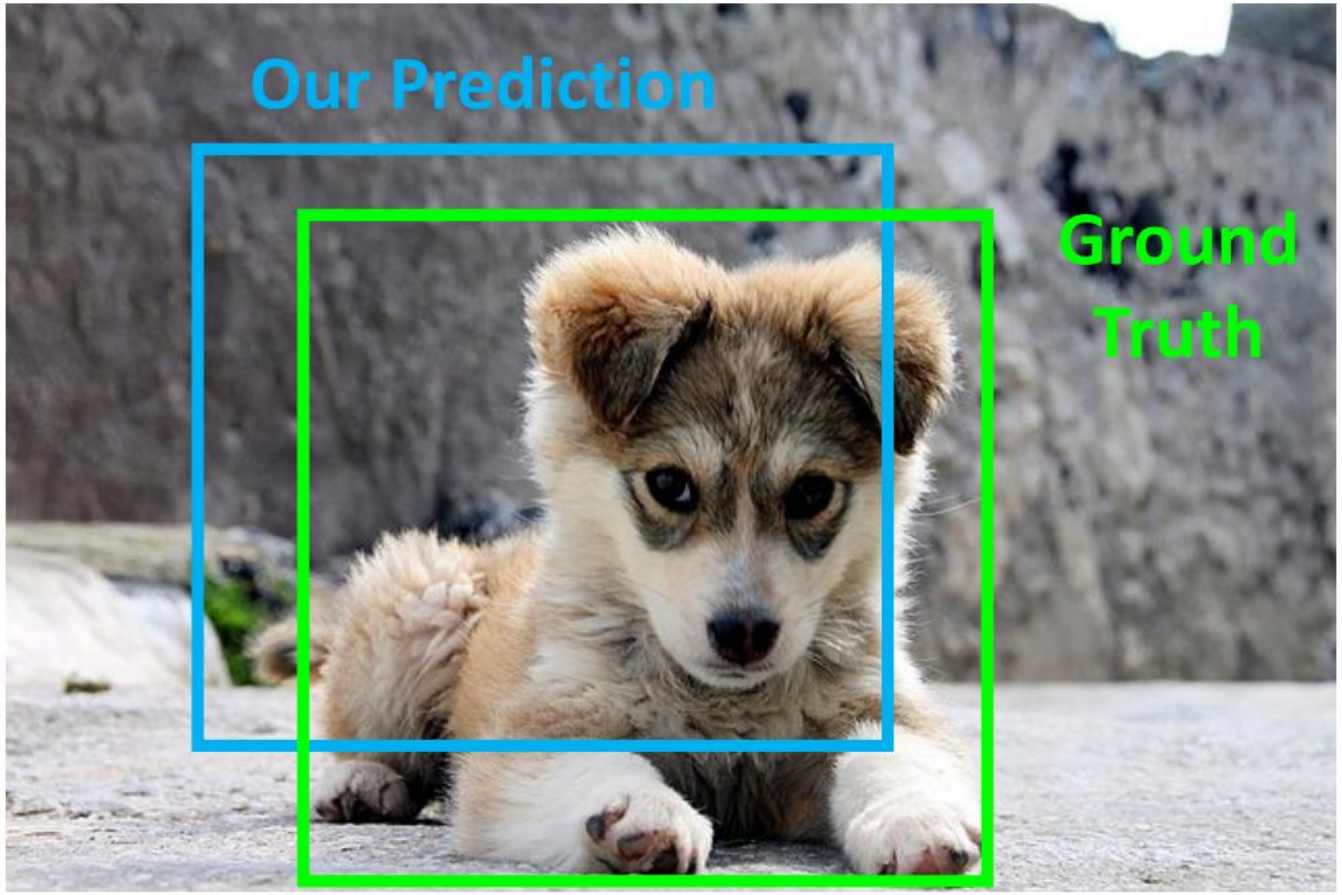
1 Two Stage Detectors

2 One Stage Detectors

3 Object Detection Metrics

Comparing boxes: Intersection over Union (IoU)

How can we compare our prediction to the ground-truth box?



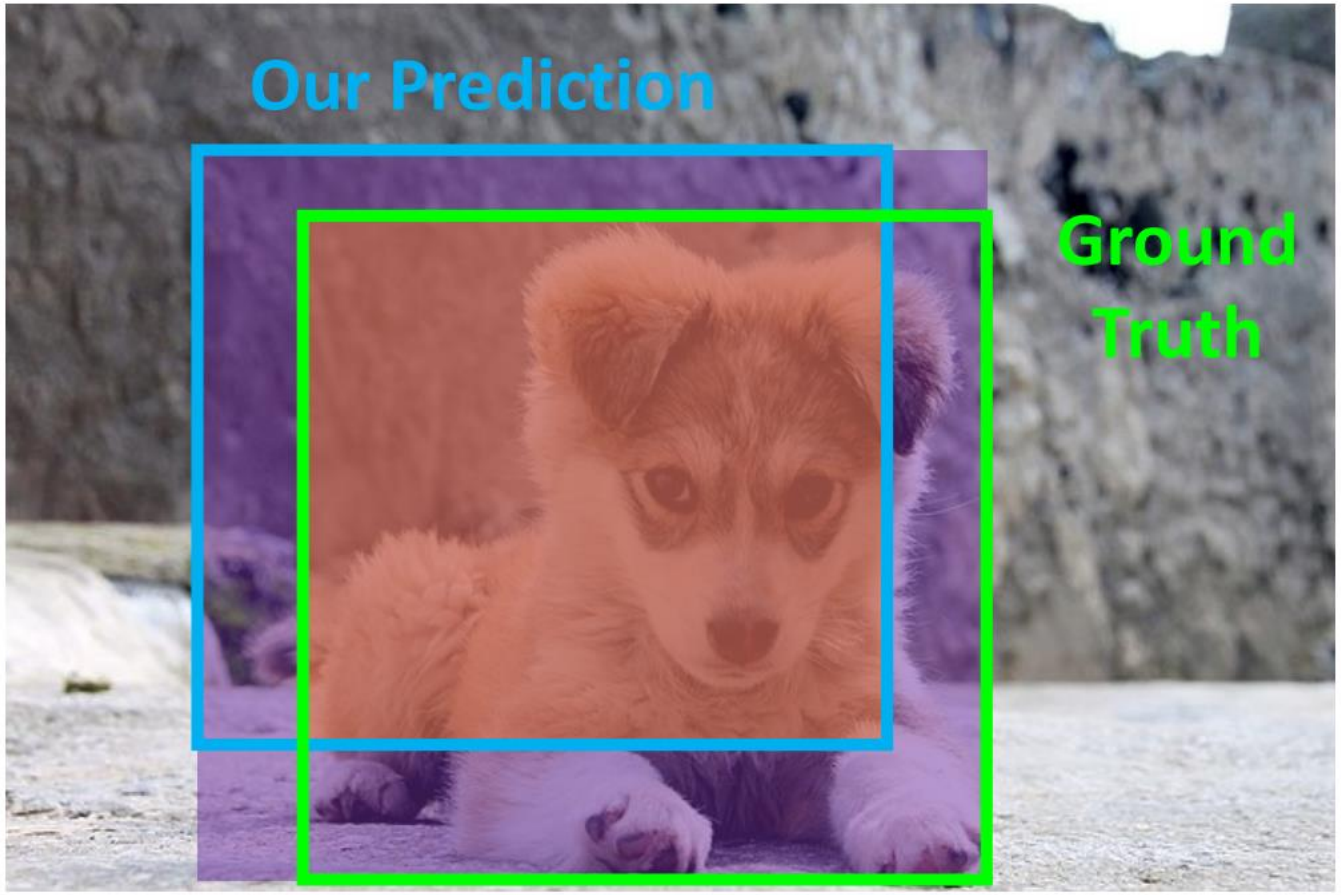
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Comparing boxes: Intersection over Union (IoU)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU)
(Also called “Jaccard similarity” or “Jaccard index”):

$$\frac{Area\ of\ Intersection}{Area\ of\ Union}$$



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

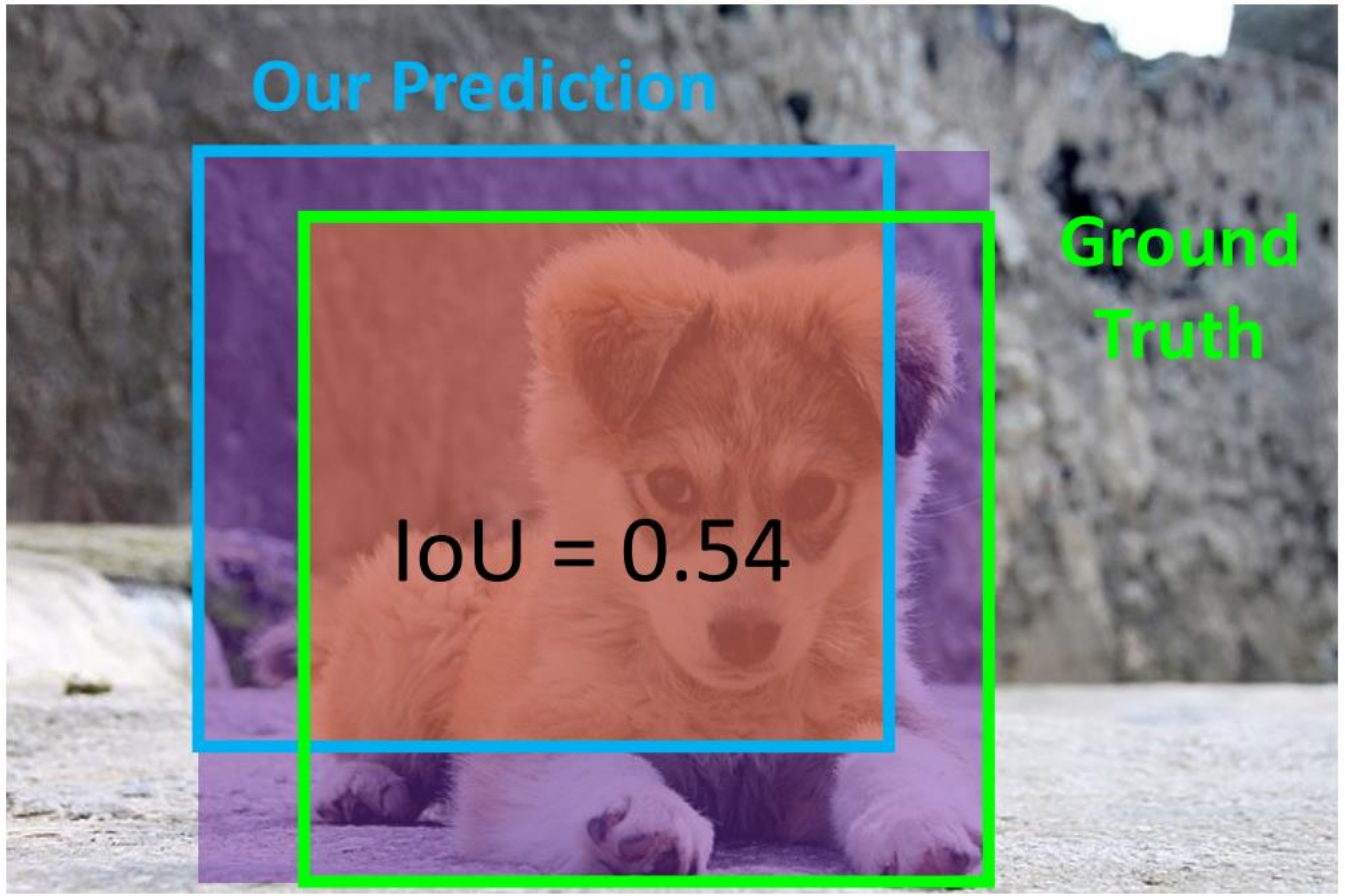
Comparing boxes: Intersection over Union (IoU)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU) (Also called "Jaccard similarity" or "Jaccard index"):

$$\frac{Area\ of\ Intersection}{Area\ of\ Union}$$

IoU > 0.5 is "decent"



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Comparing boxes: Intersection over Union (IoU)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU) (Also called "Jaccard similarity" or "Jaccard index"):

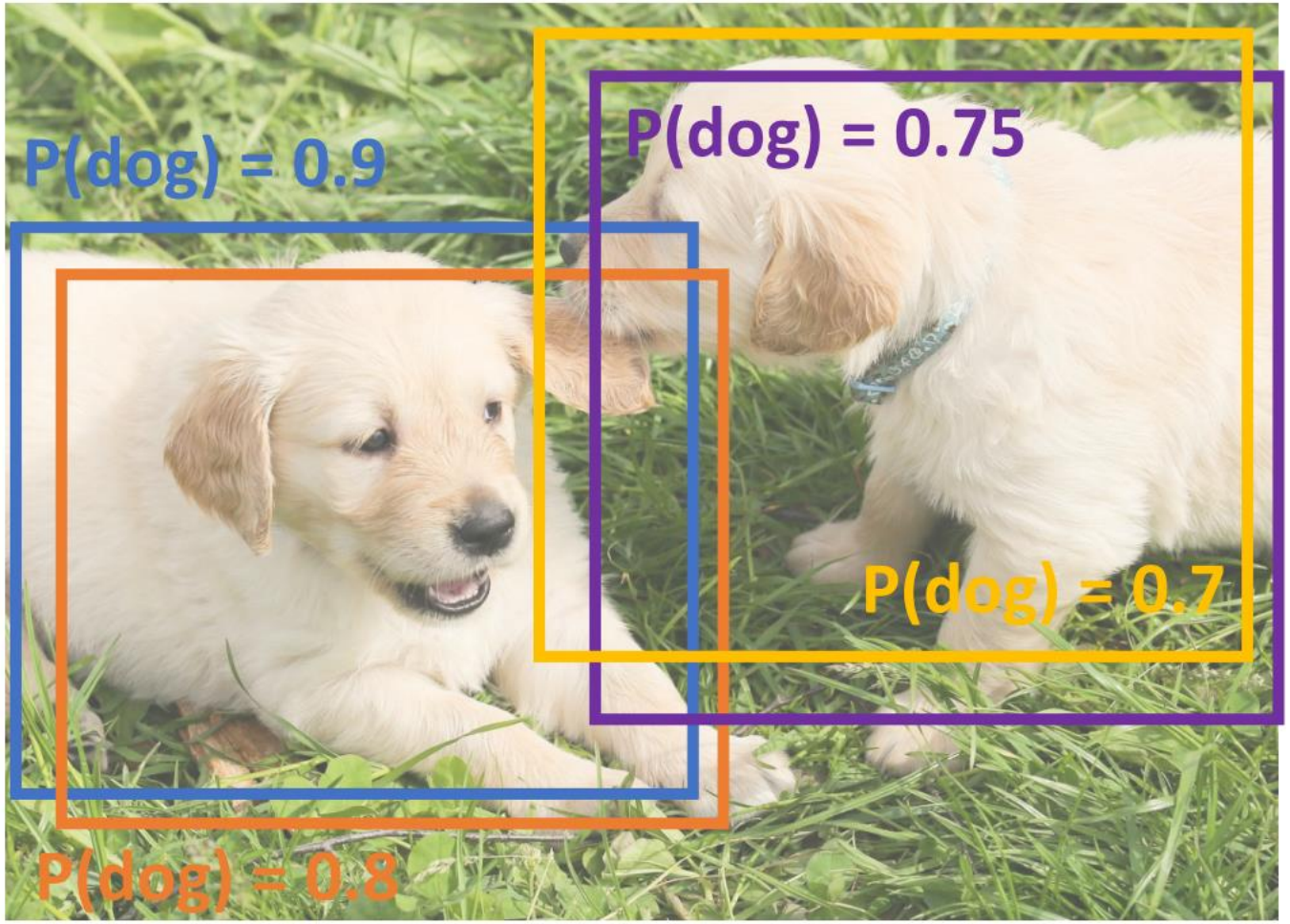
$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$

IoU > 0.5 is "decent",
IoU > 0.7 is "pretty good",
IoU > 0.9 is "almost perfect"

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Overlapping boxes

Problem: Object detectors often output many overlapping detections:



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

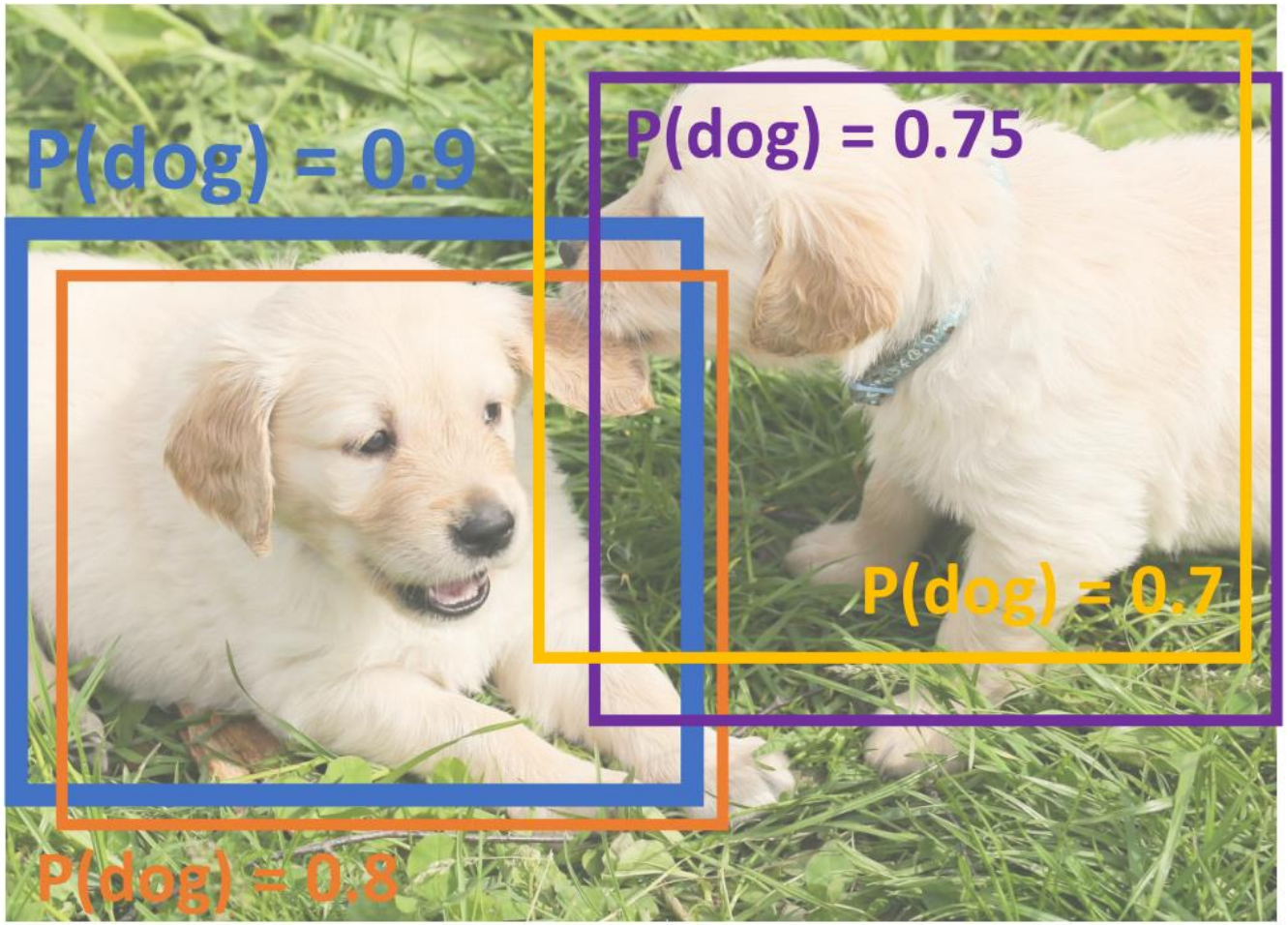
Overlapping boxes

Problem: Object detectors often output many overlapping detections:

Solution: Post-process raw detections using **Non-Max Suppression (NMS)**

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1

IoU(,) = **0.78**
IoU(,) = 0.05
IoU(,) = 0.07



Puppy image is CC0 Public Domain

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

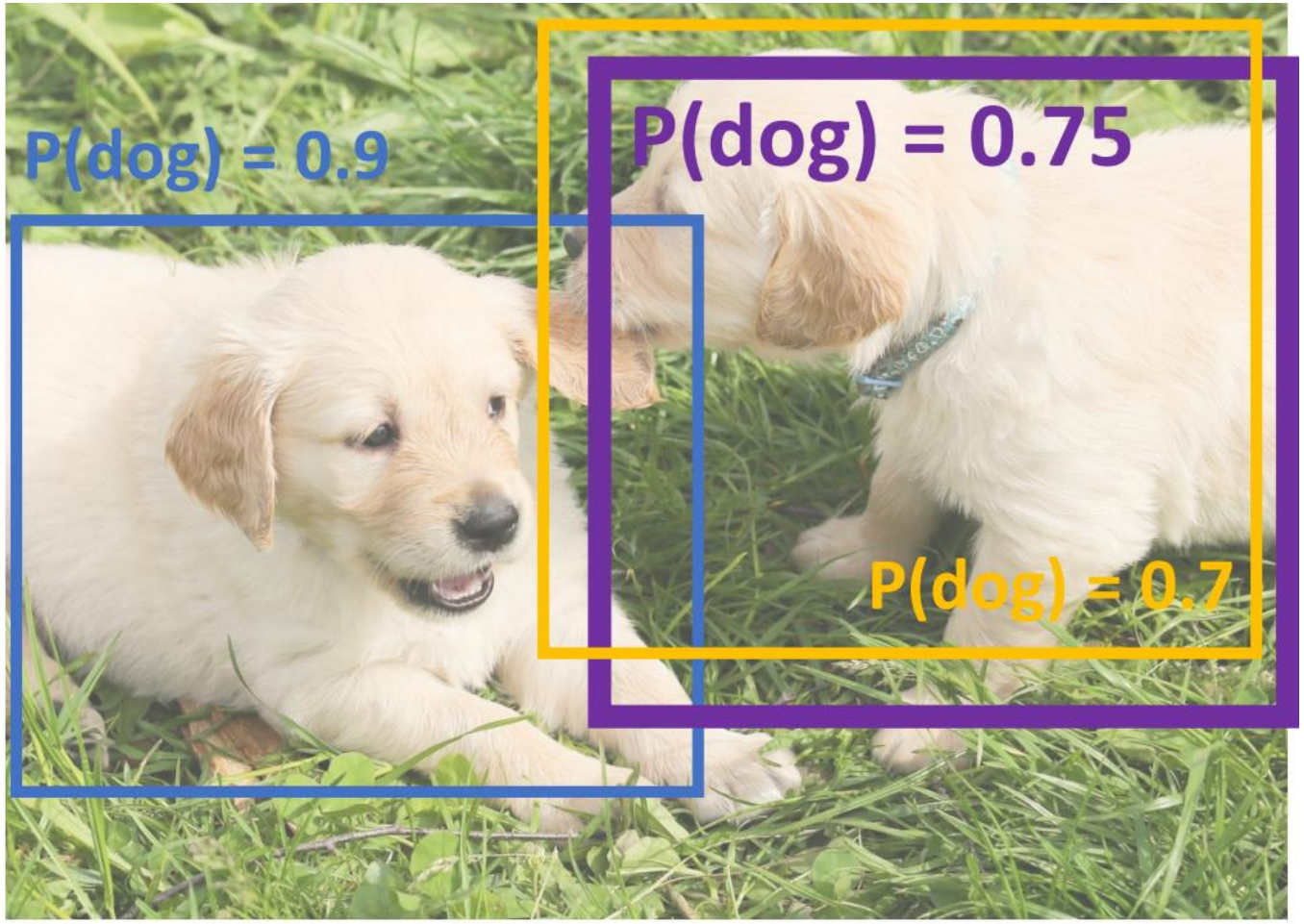
Overlapping boxes

Problem: Object detectors often output many overlapping detections:

Solution: Post-process raw detections using **Non-Max Suppression (NMS)**

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1

IoU(,) = **0.74**



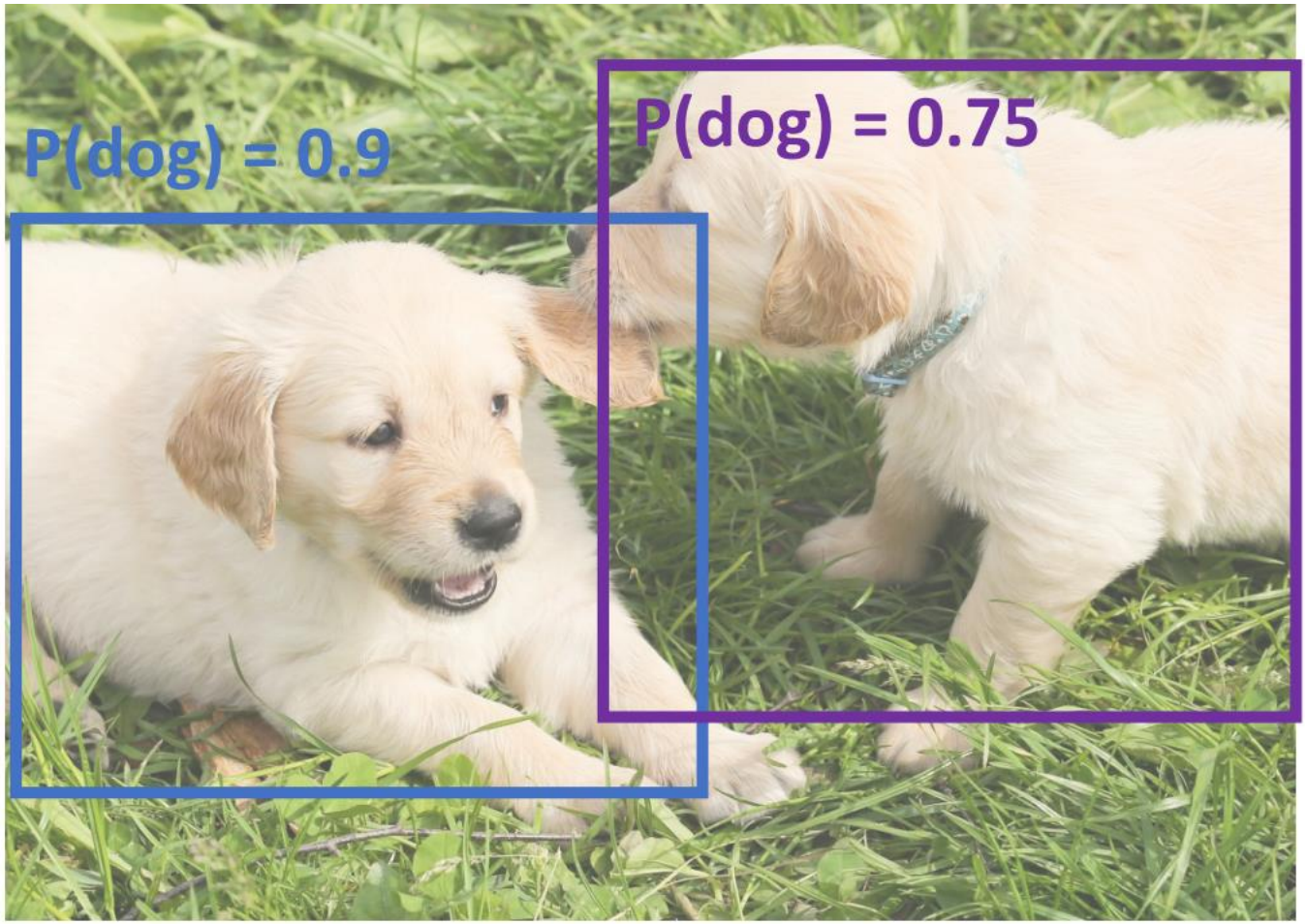
Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Overlapping boxes

Problem: Object detectors often output many overlapping detections:

Solution: Post-process raw detections using **Non-Max Suppression (NMS)**

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

NMS Limitations

Problem: Object detectors often output many overlapping detections:

Solution: Post-process raw detections using **Non-Max Suppression (NMS)**

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1

Problem: NMS may eliminate ”good” boxes when objects are highly overlapping... no good solution =(

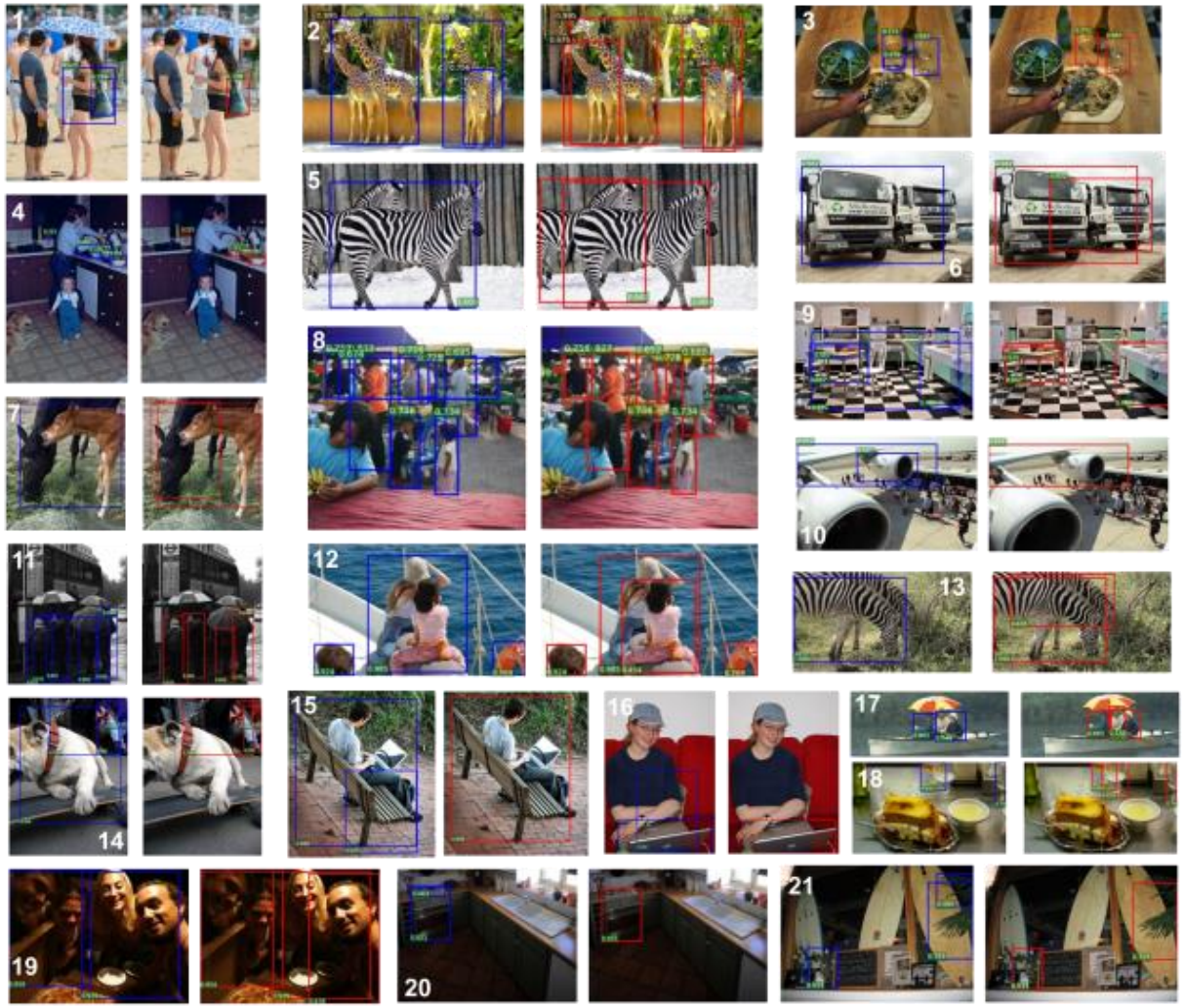
Crowd image is free for commercial use under the Pixabay license

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

NMS Limitations

There are a few other methods that try to improve detections:

- **Soft-NMS:** Traditional NMS is a binary operation that discards all but the highest-scoring bounding box. Soft-NMS, on the other hand, assigns lower scores to overlapping boxes rather than completely removing them. This results in smoother score degradation for close-by objects, reducing the chance of removing valid detections.



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

NMS Limitations

There are a few other methods that try to improve detections:

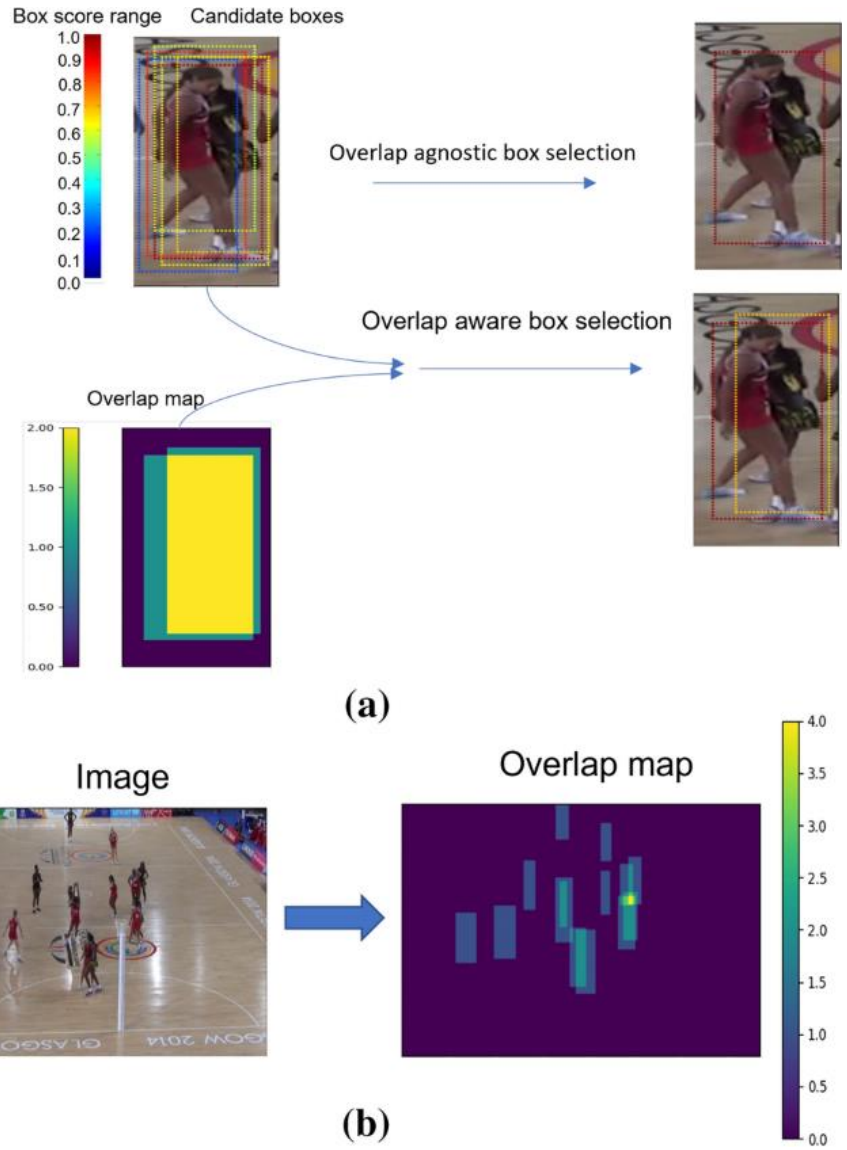
- **IoU Threshold Adaptation:** Instead of using a fixed IoU (Intersection over Union) threshold for NMS, you can dynamically adjust the threshold based on the object's characteristics. For instance, you might use a higher IoU threshold for large objects and a lower IoU threshold for smaller objects.

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

NMS Limitations

More methods:

- Selection of object detections using overlap map predictions (https://link.springer.com/article/10.1007/s00521-022-07469-x)
- https://www.sciencedirect.com/science/article/pii/S2214914721001914
- https://arxiv.org/pdf/2207.00865.pdf
- And more...
- Open research field (make your contribution)



Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Precision = $\frac{\text{True Positive}}{\text{Actual Results}}$ or $\frac{\text{True Positive}}{\text{True Positive + False Positive}}$

Recall = $\frac{\text{True Positive}}{\text{Predicted Results}}$ or $\frac{\text{True Positive}}{\text{True Positive + False Negative}}$

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

Example with detecting Traffic lights
Positive: Traffic light
Negative: Background (non traffic light)

TRUE POSITIVE

FALSE POSITIVE

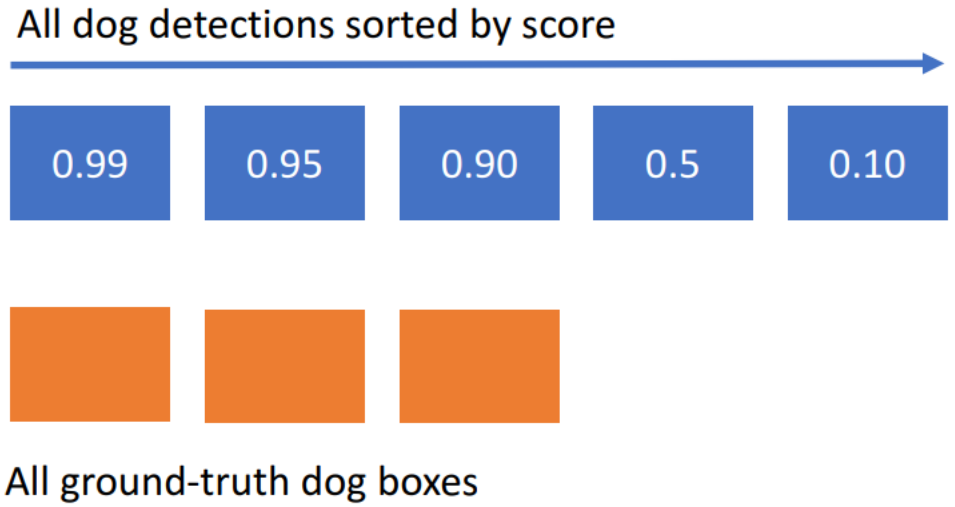
FALSE NEGATIVE

TRUE NEGATIVE

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Evaluating Object Detectors: Mean Average Precision (mAP)

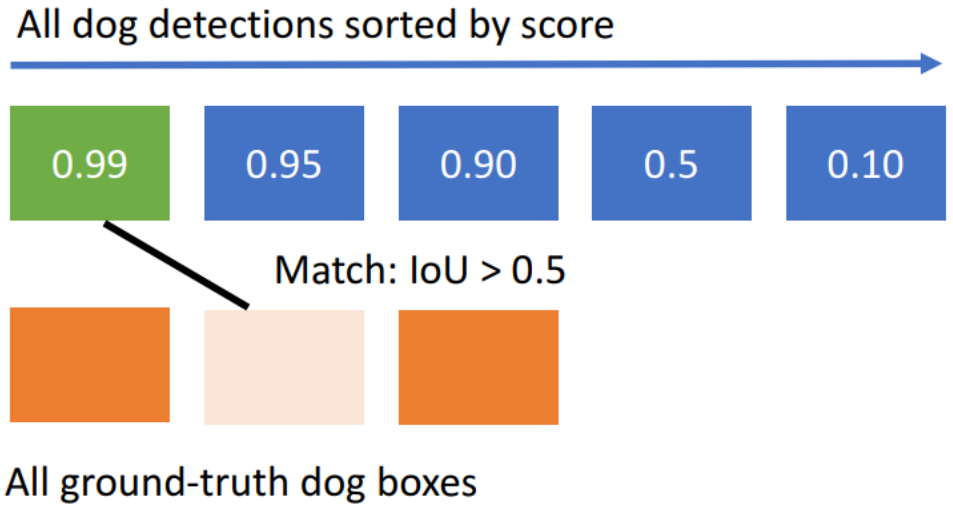
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Evaluating Object Detectors: Mean Average Precision (mAP)

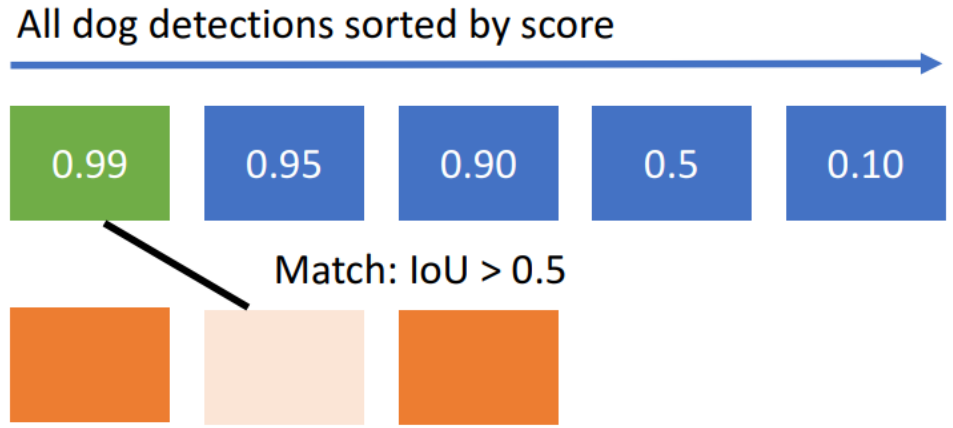
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Evaluating Object Detectors: Mean Average Precision (mAP)

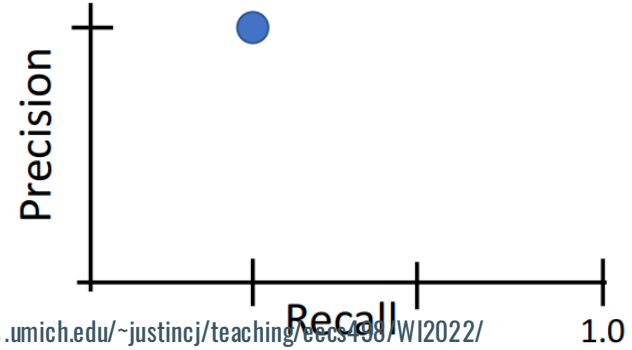
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR Curve



All ground-truth dog boxes

Precision = 1/1 = 1.0

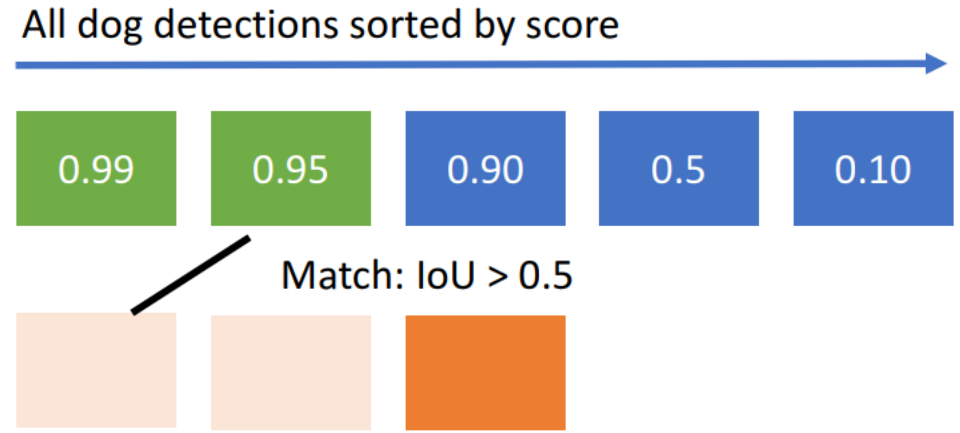
Recall = 1/3 = 0.33



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

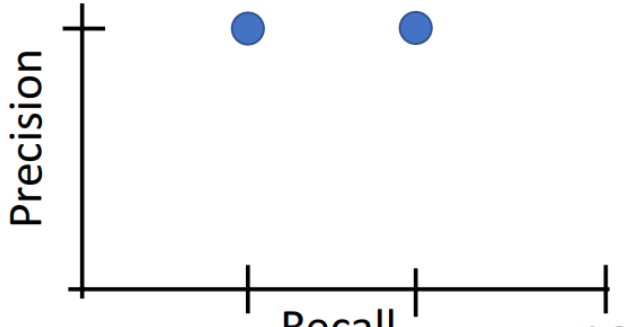
Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR Curve



Precision = 2/2 = 1.0

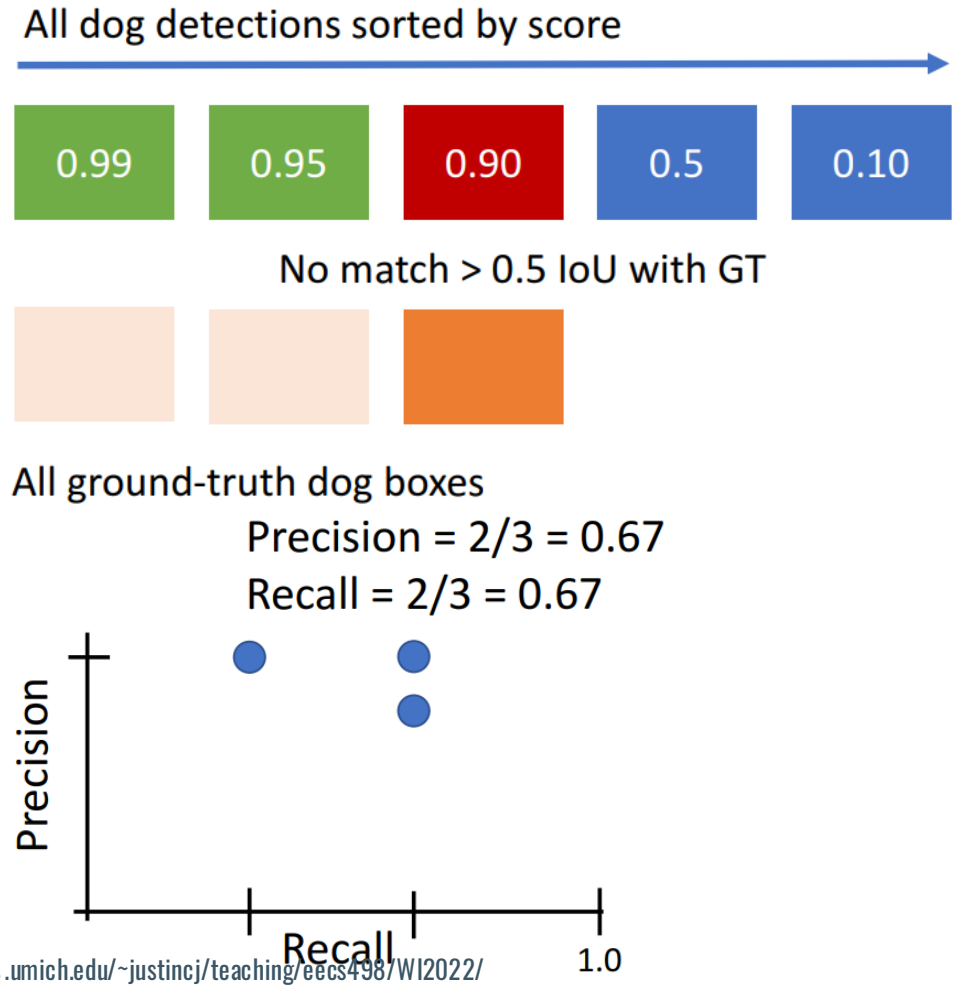
Recall = 2/3 = 0.67



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Evaluating Object Detectors: Mean Average Precision (mAP)

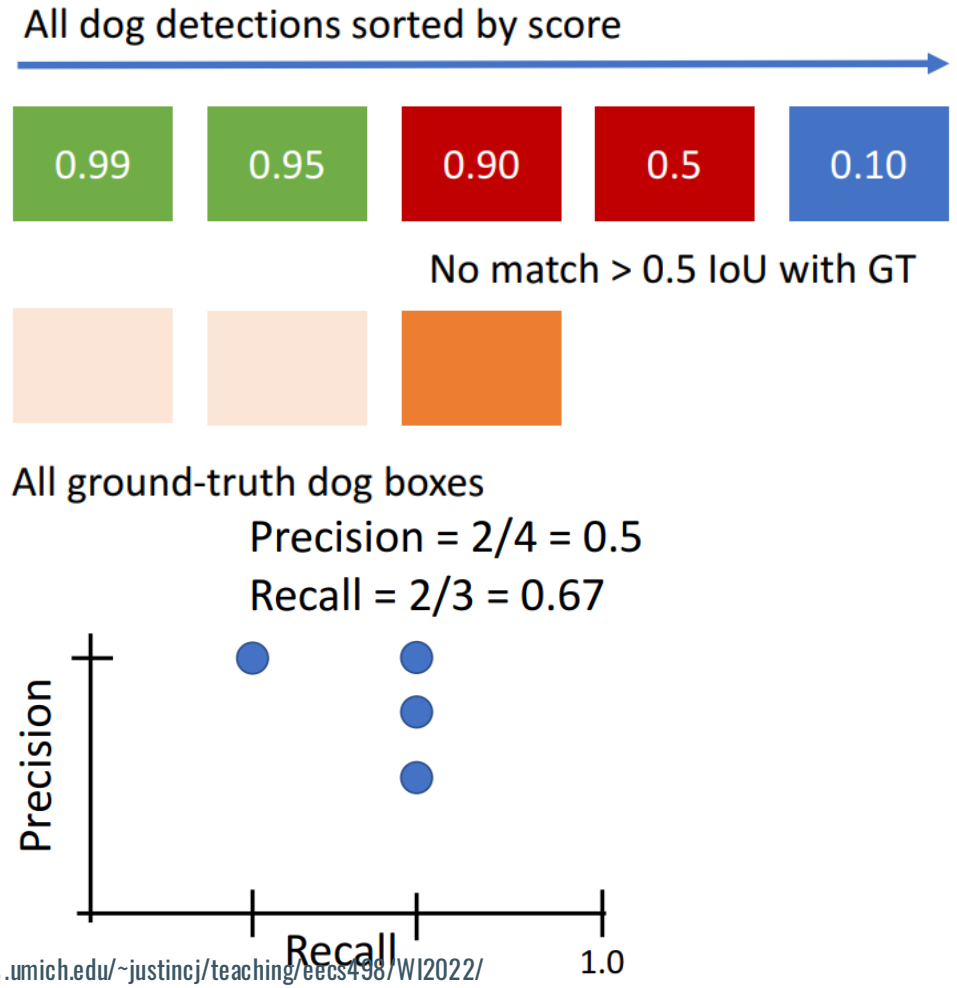
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR Curve



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Evaluating Object Detectors: Mean Average Precision (mAP)

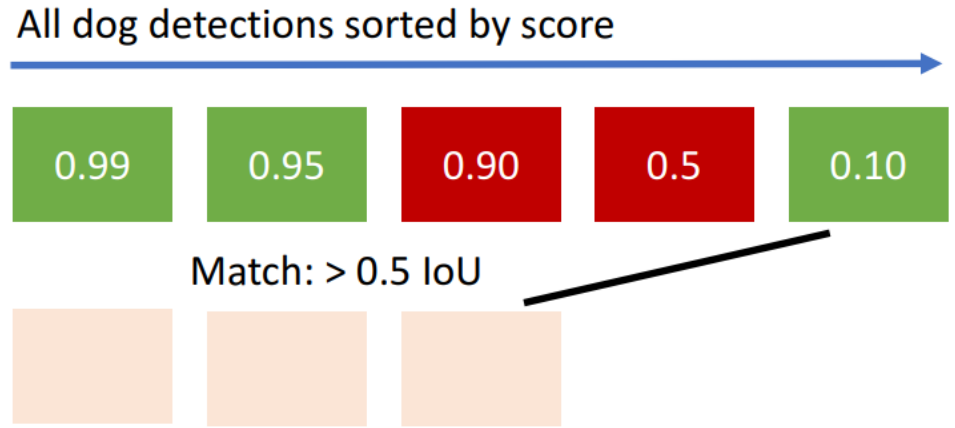
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR Curve

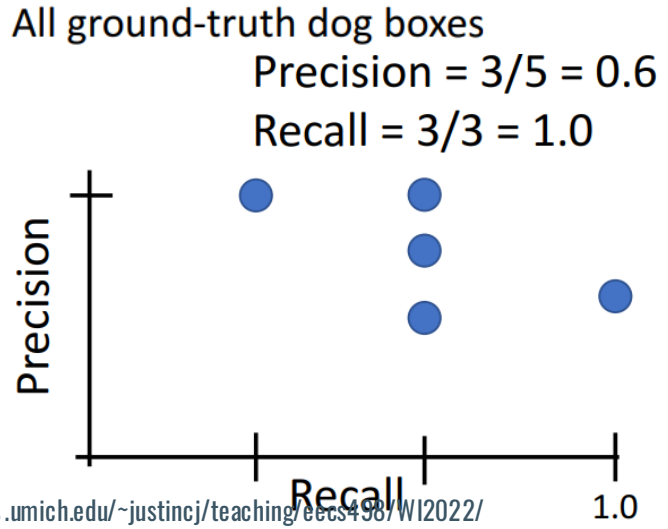


Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR Curve

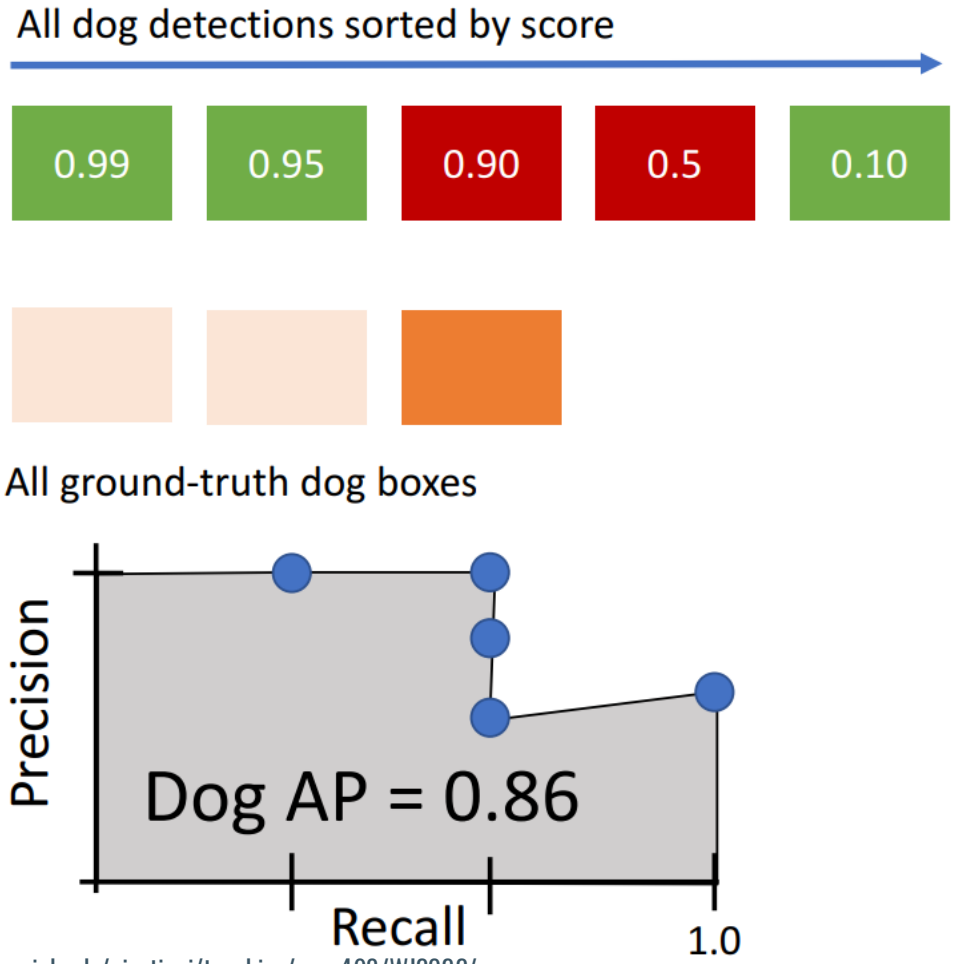




Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR Curve



Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR Curve
    2. Average Precision (AP) = area under PR curve
3. Mean Average Precision (mAP) = average of AP for each category

Car AP = 0.65

Cat AP = 0.80

Dog AP = 0.86

mAP@0.5 = 0.77

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Evaluating Object Detectors: Mean Average Precision (mAP)

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
    1. For each detection (highest score to lowest score)
        1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
        2. Otherwise mark it as negative
        3. Plot a point on PR Curve
    2. Average Precision (AP) = area under PR curve
3. Mean Average Precision (mAP) = average of AP for each category
4. For "COCO mAP": Compute mAP@thresh for each IoU threshold (0.5, 0.55, 0.6, …, 0.95) and take average

mAP@0.5 = 0.77

mAP@0.55 = 0.71

mAP@0.60 = 0.65

…

mAP@0.95 = 0.2

COCO mAP = 0.4

Example: Justin, J. (2022). EECS 498: Deep Learning for Computer Vision (Fall 2019). University of Michigan. https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

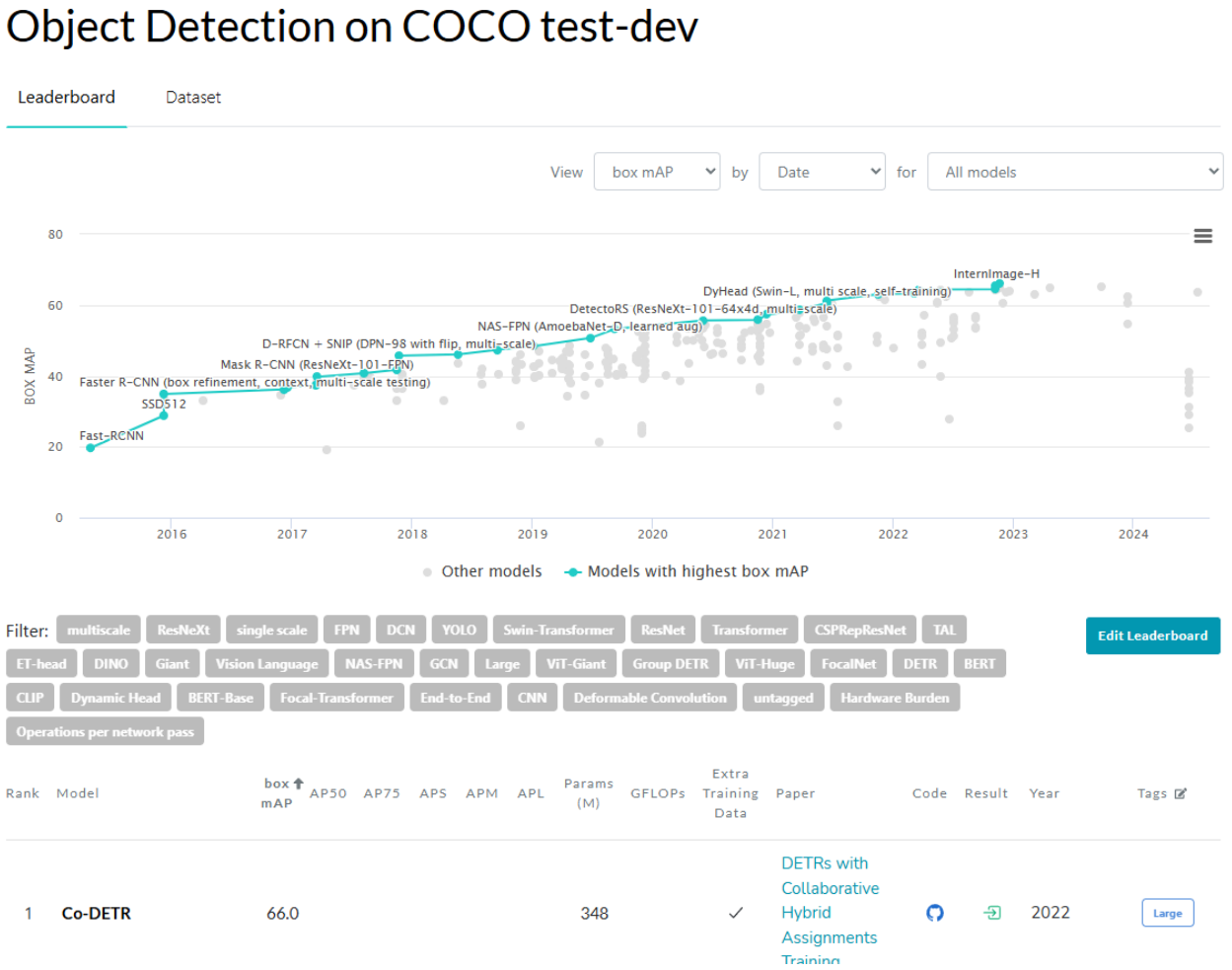
Understanding object detection paper results

Hardware independent metrics:

- **mAP (Mean Average Precision):** Overall average precision across classes and IoU thresholds, indicating detection quality.
- **AP50:** Average precision at an IoU threshold of 0.50, a lenient measure of detection accuracy.
- **AP75:** Average precision at an IoU threshold of 0.75, a stricter measure of accuracy.
- **APS/APM/APL:** Average precision for small, medium, and large objects, assessing model performance across object sizes.
- **Params (M):** Number of model parameters, in millions, indicating model size.
- **GFLOPs:** Computational complexity, showing the number of floating-point operations needed for a forward pass.

Hardware dependent metrics:

- Latency, Inference speed

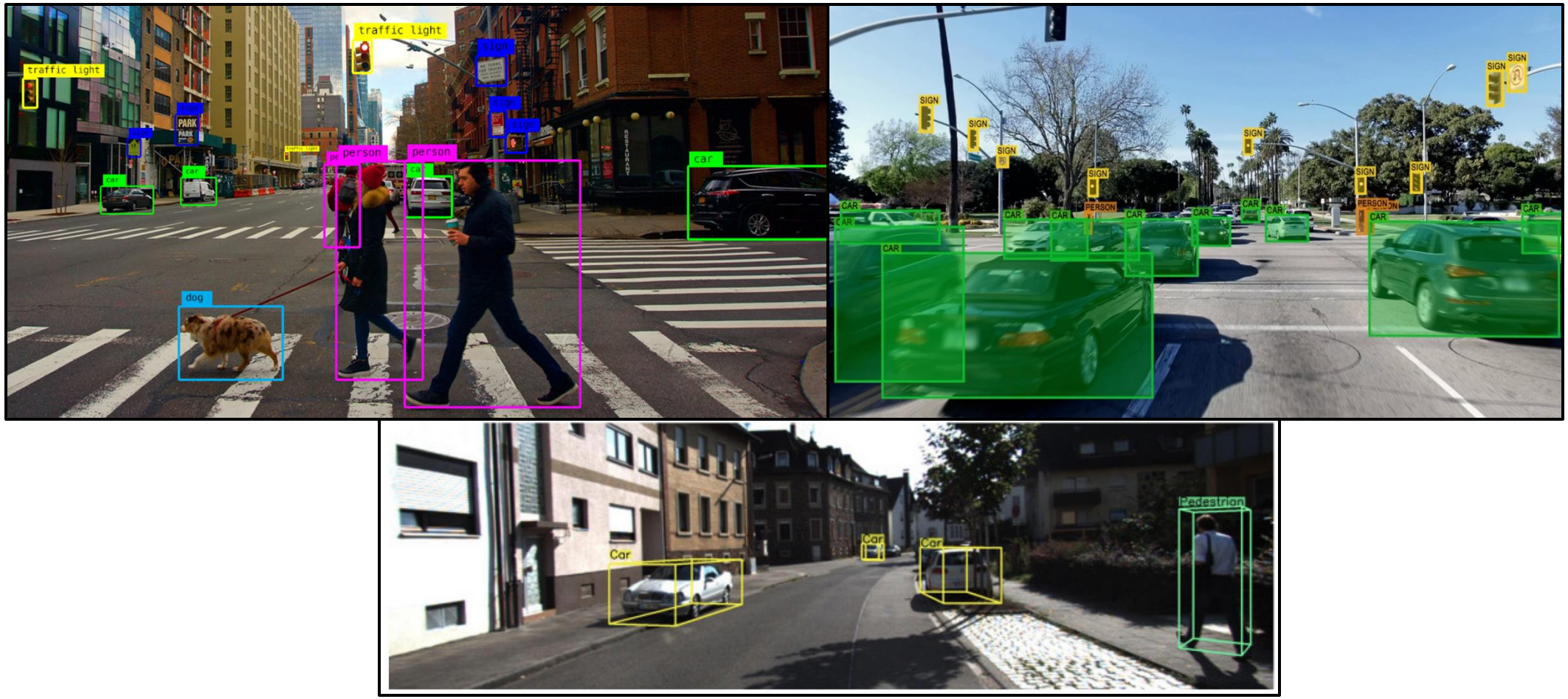


COCO test-dev Benchmark (Object Detection) | Papers With Code

Understanding object detection paper results

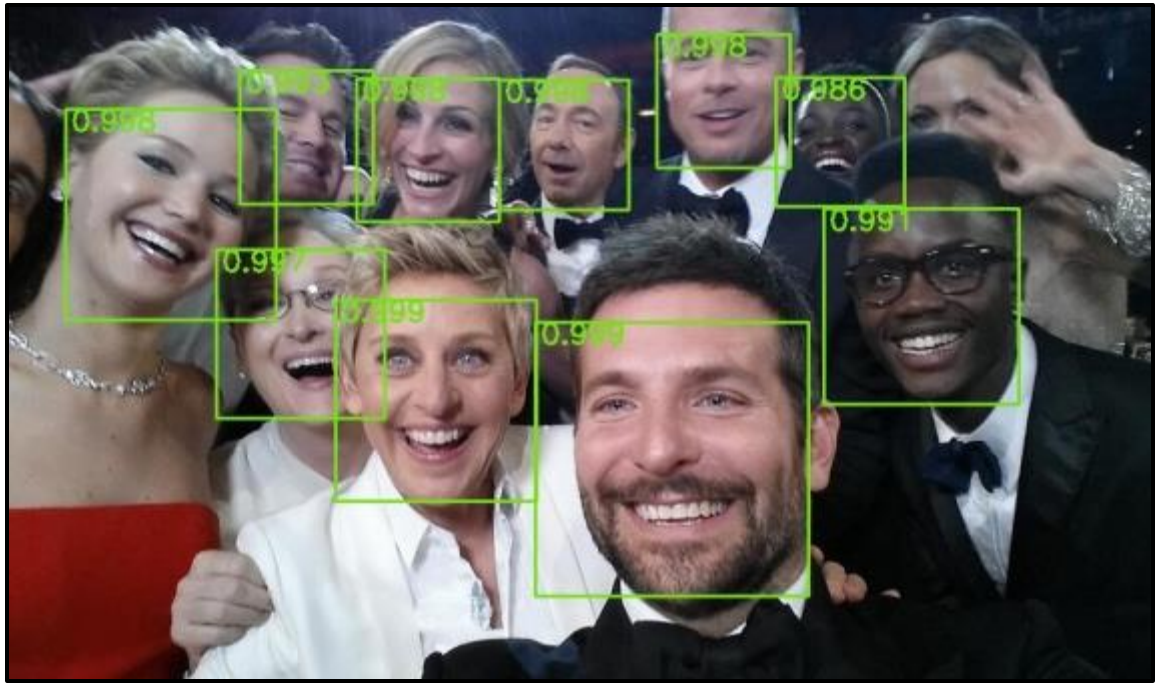
- **mAP (Mean Average Precision):** This is the average precision across different classes and IoU (Intersection over Union) thresholds. It's a comprehensive metric for measuring the performance of object detection models. The higher the mAP, the better the model's precision and recall performance over all categories and thresholds.
- **AP50 (Average Precision at IoU=0.50):** This metric evaluates the average precision when the IoU threshold is set to 0.50. This means that, for a detection to be considered correct, the predicted bounding box must overlap the ground truth by at least 50%. It is often considered a relatively lenient metric.
- **AP75 (Average Precision at IoU=0.75):** Like AP50.
- **APS (Average Precision for Small Objects):** This metric calculates the average precision for detecting small objects. Smaller objects can be more difficult to detect accurately, so this metric specifically tracks how well the model handles smaller object sizes.
- **APM/APL:** Average precision for medium and large objects, assessing model performance across object sizes.
- **Params (M) (Model Parameters in Millions):** This metric shows the number of parameters in the model, typically in millions (M). More parameters often mean a larger, more complex model, but not necessarily better performance.
- **GFLOPs (Giga Floating Point Operations):** This refers to the number of floating-point operations required to make a single forward pass through the model, usually measured in gigaflops (GFLOPs). It is an indicator of the model's computational complexity and can be used to gauge the efficiency of the model.

Autonomous Driving



Inventory Scan

Face Detection



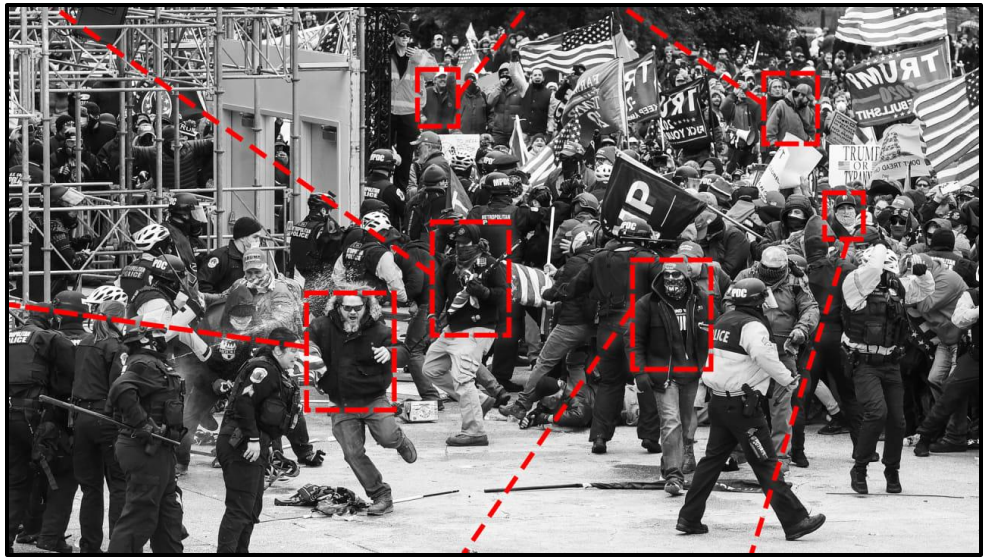
+ Face Recognition (endless applications)

Attendance check

Entry access to specific place

Security

...



Summary

- **Object Detection** is a supervised learning task
- Goal is to predict what and where are the objects in an image

- (Core) Region proposal methods:
  - **R-CNN:** uses selective search to find regions
  - **Fast R-CNN:** first extracts features and then uses selective search
  - **Faster R-CNN:** uses a learnable region proposal network

- One-stage methods like **YOLO** and **SSD**, remove the region proposal part used in two-stage methods like the R-CNN family

- Many useful applications

- Techniques like **IoU** and **NMS** improve the predictions of object detectors
- **mAP** is a metric to evaluate object detectors

Resources

Books:

- Courville, Goodfellow, Bengio: Deep Learning
  Freely available: https://www.deeplearningbook.org/
- Zhang, Aston and Lipton, Zachary C. and Li, Mu and Smola, Alexander J.: Dive into Deep Learning
  Freely available: https://d2l.ai/

Courses:

- Deep Learning specialization by Andrew NG
- https://www.coursera.org/specializations/deep-learning

Further Links + Resources

- https://www.youtube.com/watch?v=TB-fdISzpHQ
- https://www.youtube.com/watch?v=9AyMR4IhSWQ
- https://www.youtube.com/watch?v=ag3DLKsl2vk (for the explanation only)

That’s all for today!