Object Detection Overview

Naeemullah Khan
naeemullah.khan@kaust.edu.sa

KAUST Academy
King Abdullah University of Science and Technology

June 3, 2025

Table of Contents

Object Detection: Challenges

Object detection is a complex task in computer vision that involves identifying and localizing objects within images.

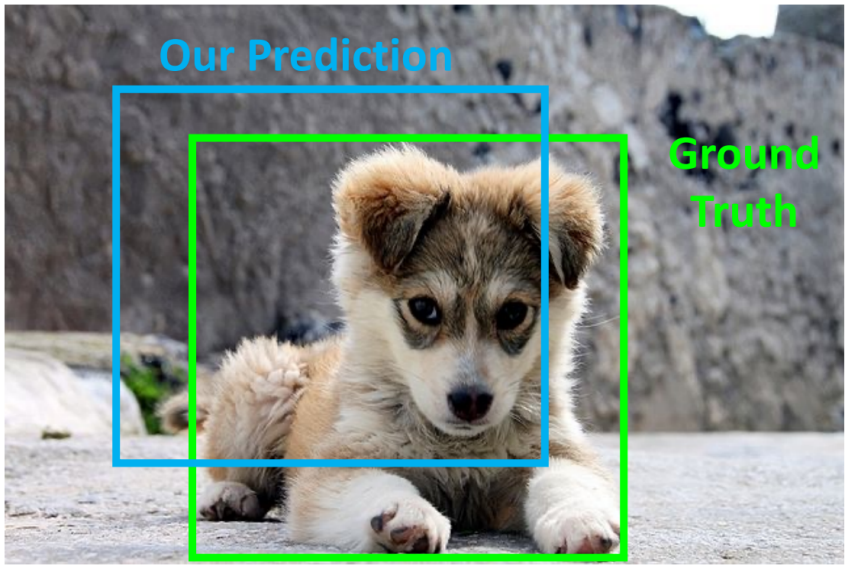
Unlike image classification, which assigns a single label to an entire image, object detection requires both identifying the objects present and determining their locations within the image.

This task presents several challenges that make it more difficult than classification:

- **Multiple outputs:** Need to output variable numbers of objects per image
- **Multiple types of output:** Need to predict "what" (category label) as well as "where" (bounding box)
- **Large images:** Classification works at 224x224; need higher resolution for detection, often $\sim 800 \times 600$

Comparing Boxes: Intersection over Union (IoU)

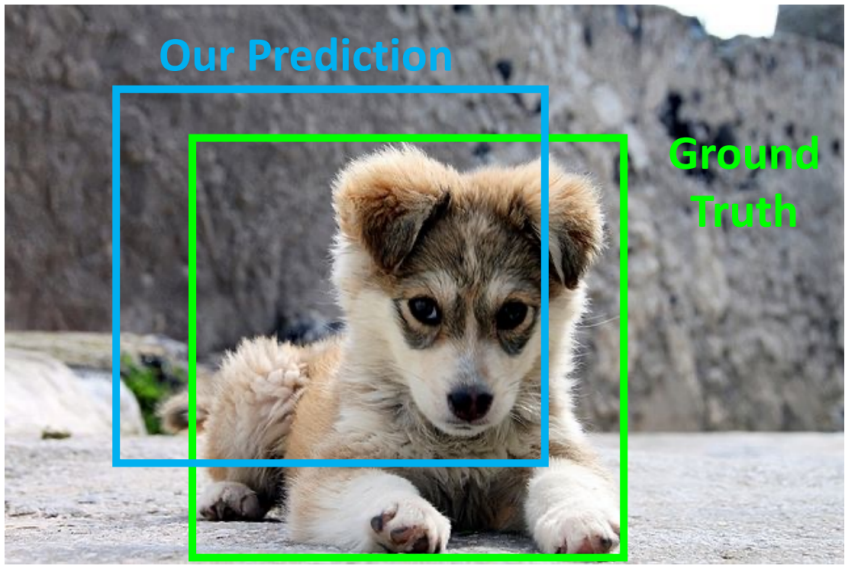
- How can we compare our prediction to the ground-truth box?



Comparing Boxes: Intersection over Union (IoU)

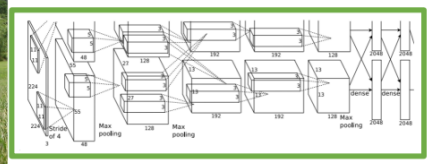
- How can we compare our prediction to the ground-truth box?
- **Intersection over Union** (IoU) (Also called "Jaccard similarity" or "Jaccard index"):

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



Detecting a Single Object

Often pretrained
on ImageNet
(Transfer learning)



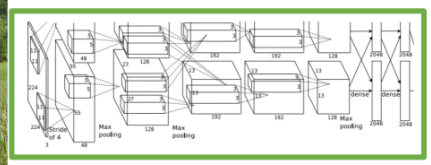
This image is CC0 public domain

Vector:
4096

Treat localization as a
regression problem!

Detecting a Single Object (cont.)

This image is CC0 public domain



Often pretrained on ImageNet (Transfer learning)

Fully Connected: 4096 to 1000

Vector: 4096

"What"

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

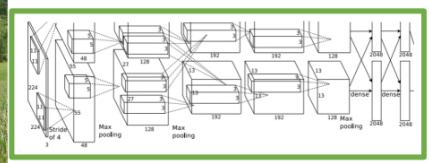
Correct label: Cat

Softmax Loss

Treat localization as a regression problem!

Detecting a Single Object (cont.)

This image is CC0 public domain



Often pretrained on ImageNet (Transfer learning)

Treat localization as a regression problem!

Vector: 4096

Fully Connected: 4096 to 1000

Fully Connected: 4096 to 4

"What"

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

Softmax Loss

Correct label: Cat

"Where"

Box Coordinates (x, y, w, h)

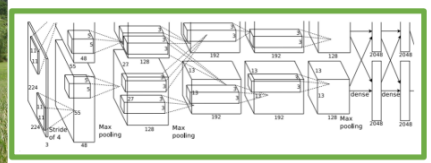
L2 Loss

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

Detecting a Single Object (cont.)

This image is CC0 public domain

Often pretrained on ImageNet (Transfer learning)



Treat localization as a regression problem!

Vector: 4096

Fully Connected: 4096 to 1000

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

Fully Connected: 4096 to 4

Box Coordinates (x, y, w, h)

"What"

"Where"

Correct label: Cat

Softmax Loss

Weighted Sum → **Loss**

Multitask Loss

$$L = L_{cls} + \lambda L_{reg}$$

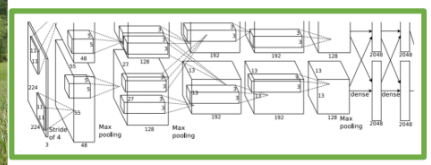
L2 Loss

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

Detecting a Single Object (cont.)

This image is CC0 public domain

Often pretrained on ImageNet (Transfer learning)



Vector: 4096

Fully Connected: 4096 to 1000

"What"

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

Correct label: Cat

Softmax Loss

Multitask Loss

Weighted Sum → Loss

$L = L_{cls} + \lambda L_{reg}$

Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

L2 Loss

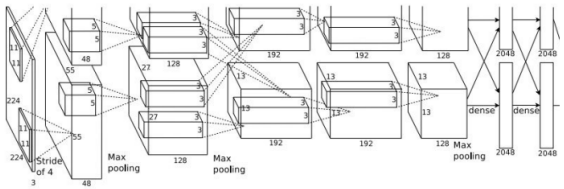
Correct box:
(x', y', w', h')

"Where"

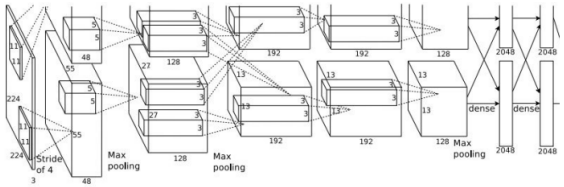
Treat localization as a regression problem!

Problem: Images can have more than one object!

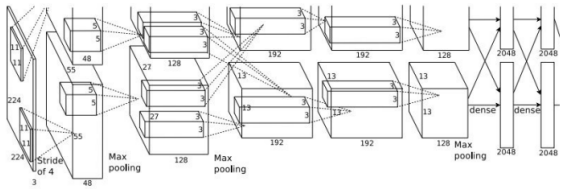
Multiple Objects



CAT: (x, y, w, h)



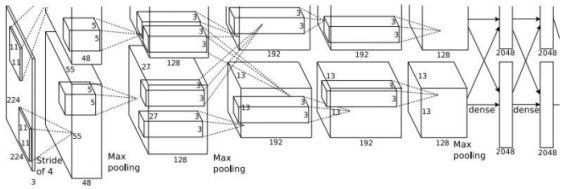
DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)



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

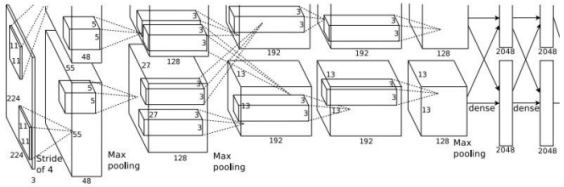
Multiple Objects (cont.)

Each image needs a different number of outputs!



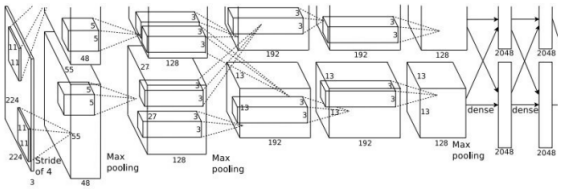
CAT: (x, y, w, h)

4 numbers



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

12 numbers

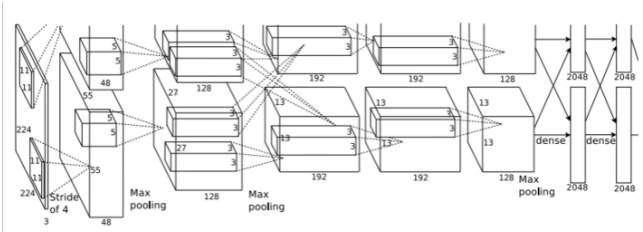


DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....

Many numbers!

Detecting Multiple Objects: Sliding Window

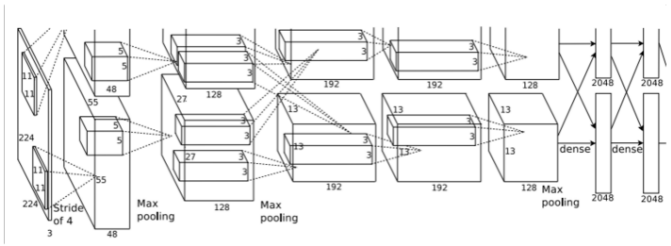
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? NO
Background? YES

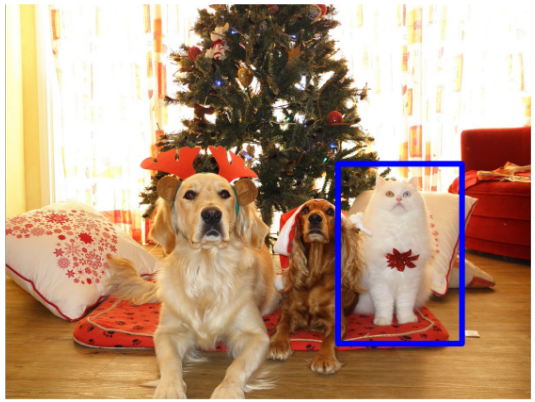
Detecting Multiple Objects: Sliding Window (cont.)

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

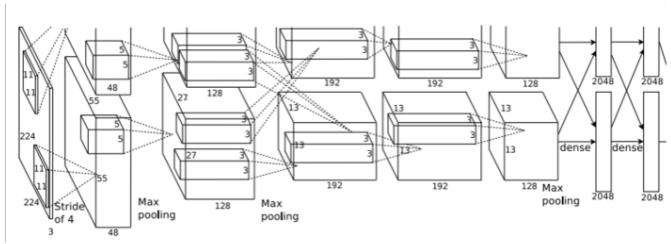


Dog? YES
Cat? NO
Background? NO

Detecting Multiple Objects: Sliding Window (cont.)

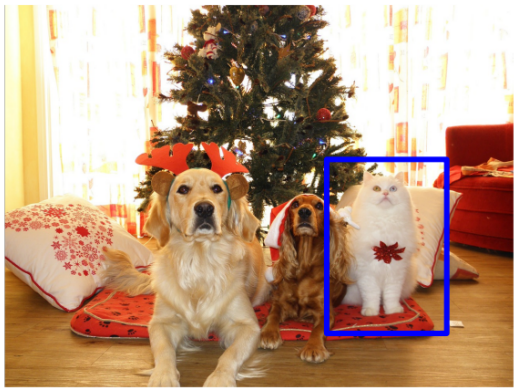


Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? YES
Background? NO

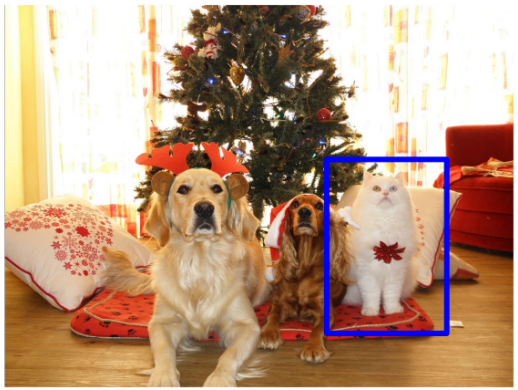
Detecting Multiple Objects: Sliding Window (cont.)



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

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

Detecting Multiple Objects: Sliding Window (cont.)

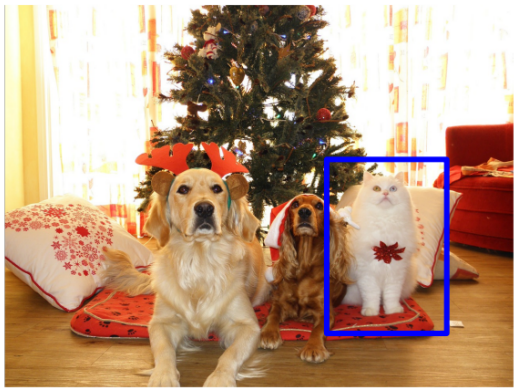


Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

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)

Detecting Multiple Objects: Sliding Window (cont.)



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

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

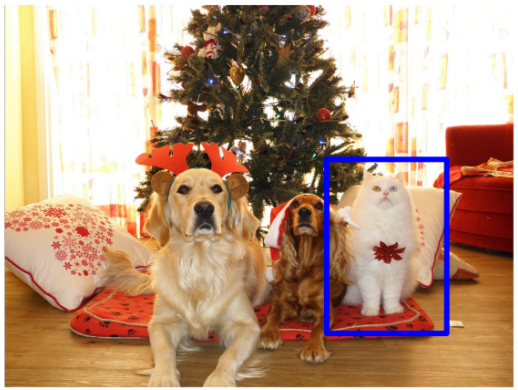
Consider a box of size h x w:
Possible x positions: W – w + 1
Possible y positions: H – h + 1
Possible positions:
(W – w + 1) * (H – h + 1)

Total possible boxes:

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

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

Detecting Multiple Objects: Sliding Window (cont.)



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:

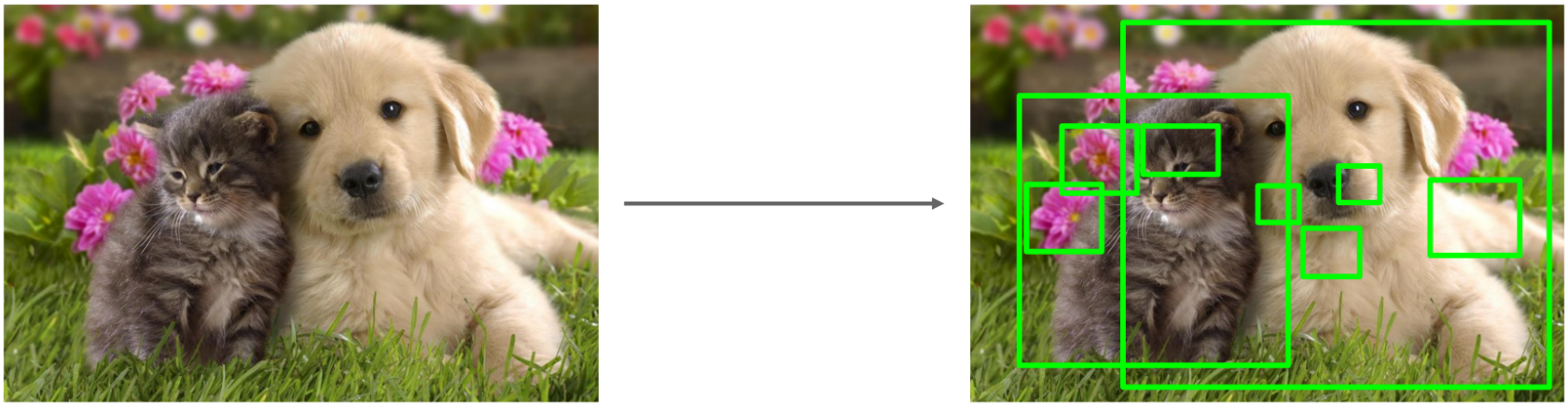
$$\sum_{h=1}^{H}\sum_{w=1}^{W}(W-w+1)(H-h+1)$$

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

Detecting Multiple Objects: Sliding Window (cont.)

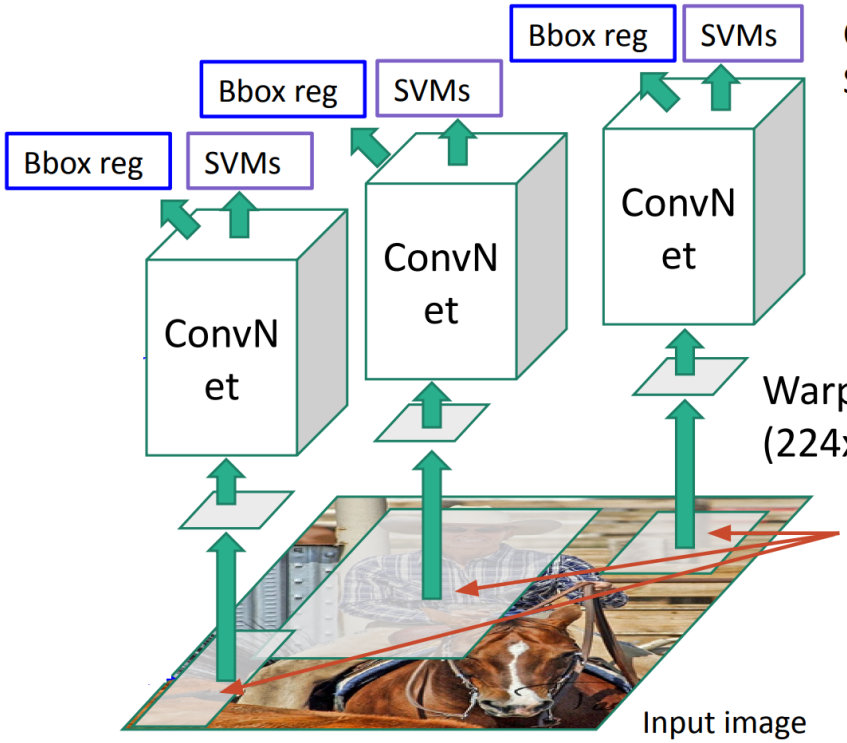
Region Proposal

- 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



R-CNN

Predict “corrections” to the RoI: 4 numbers: (dx, dy, dw, dh)



Classify regions with SVMs

Forward each region through ConvNet

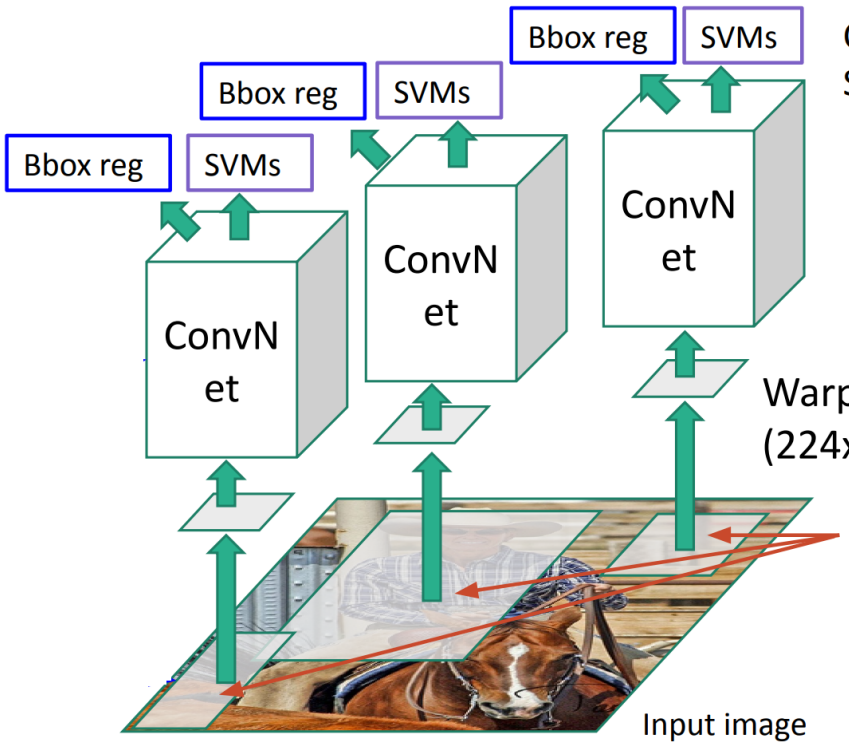
Warped image regions (224x224 pixels)

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

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 (cont.)

Predict "corrections" to the RoI: 4 numbers: (dx, dy, dw, dh)



Classify regions with SVMs

Forward each region through ConvNet

Warped image regions (224x224 pixels)

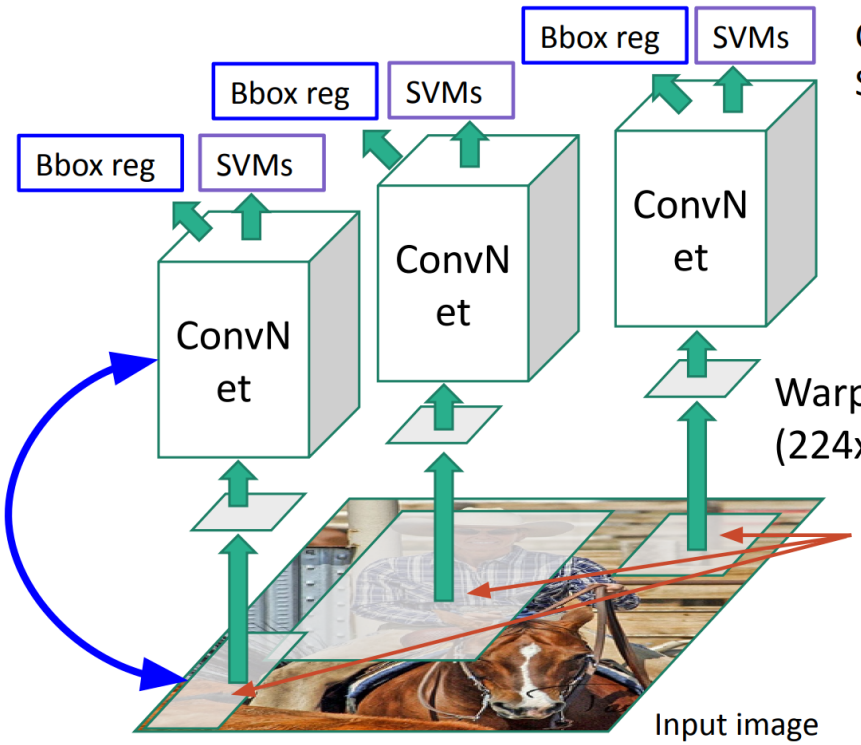
Regions of Interest (RoI) from a proposal method (~2k)

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

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 (cont.)

Predict "corrections" to the RoI: 4 numbers: (dx, dy, dw, dh)



Classify regions with SVMs

Forward each region through ConvNet

Warped image regions (224x224 pixels)

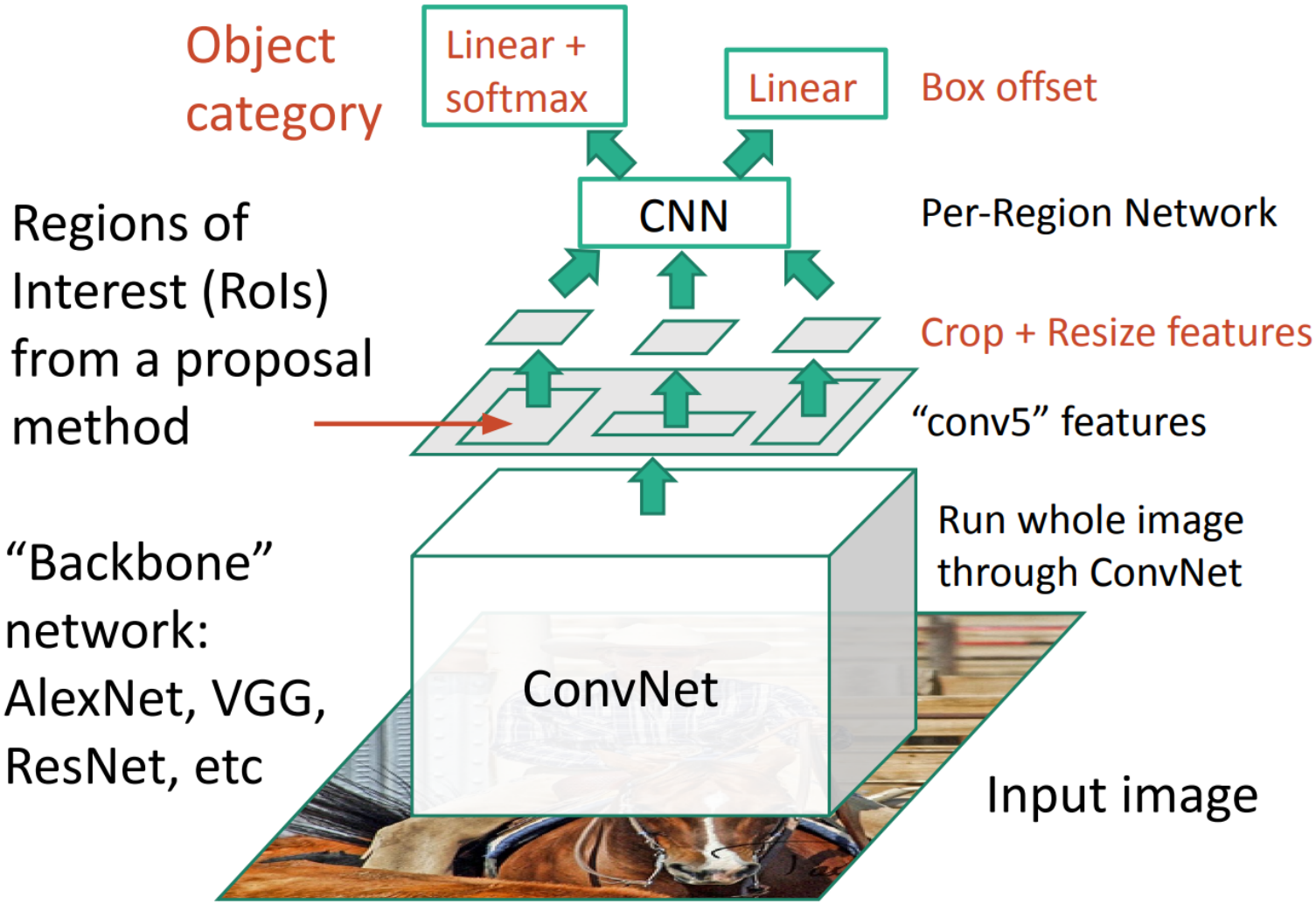
Regions of Interest (RoI) from a proposal method (~2k)

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

Idea: Pass the image through convnet before cropping! Crop the conv feature instead!

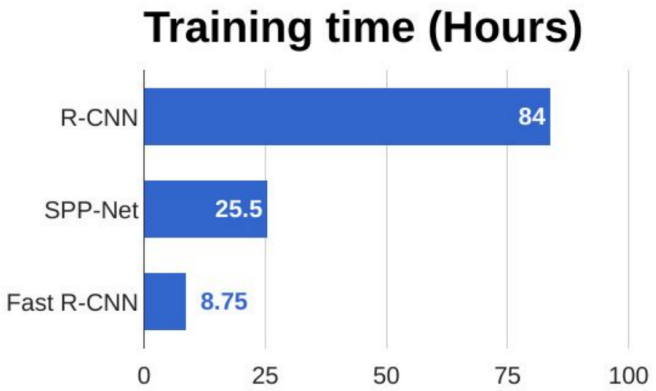
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

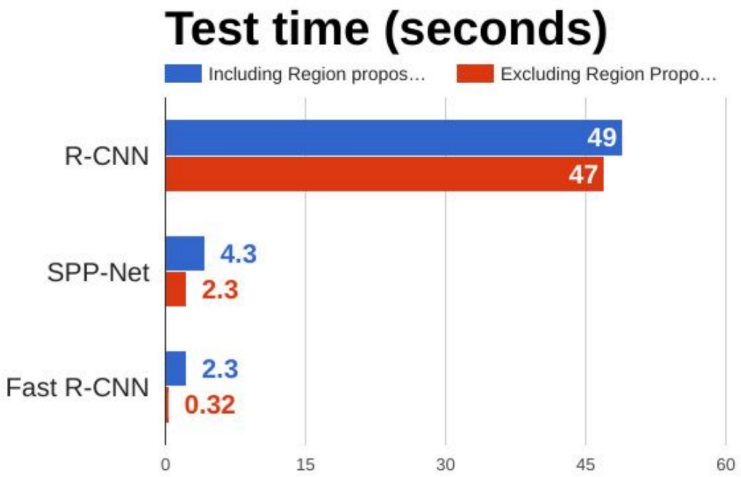
Fast R-CNN



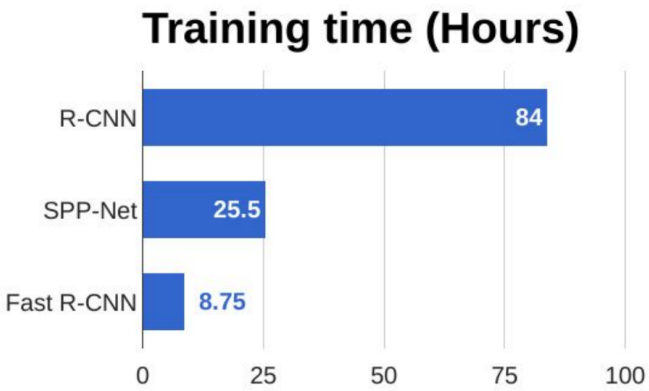
Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

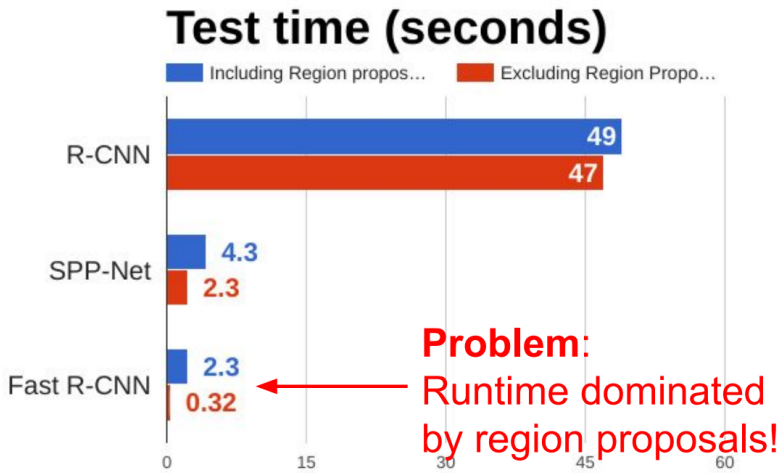
Fast R-CNN (cont.)





Fast R-CNN (cont.)





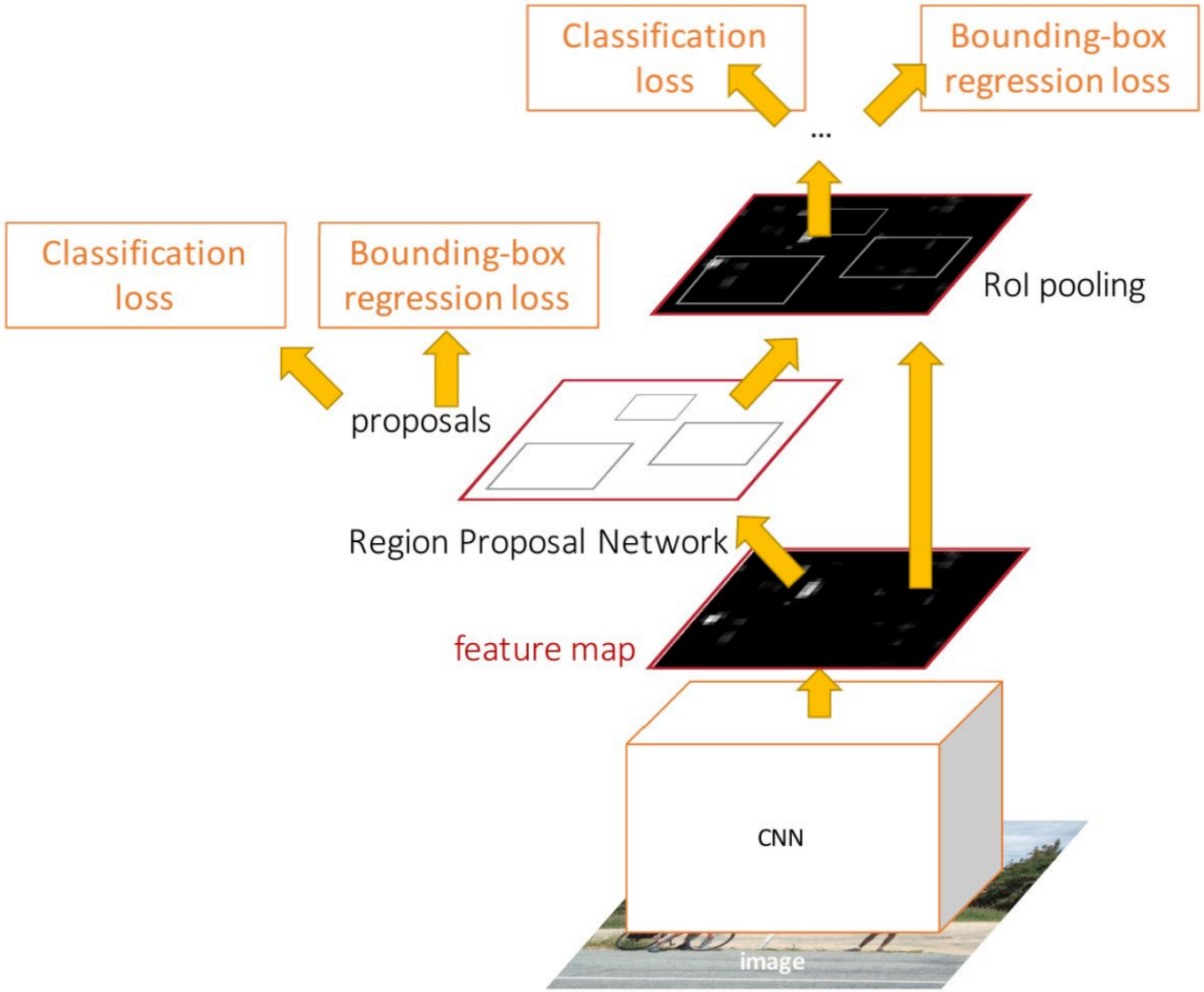
Faster R-CNN

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

Faster R-CNN

- Make CNN do proposals!
- Insert Region Proposal Network (RPN) to predict proposals from features
- Jointly train on 4 losses:
  - **RPN classification:** anchor box is object / not an object
  - **RPN regression:** predict transform from anchor box to proposal box
  - **Object classification:** classify proposals as background / object class
  - **Object regression:** predict transform from proposal box to object box

Faster R-CNN


Classification loss
Bounding-box regression loss
...
RoI pooling
Classification loss
Bounding-box regression loss
proposals
Region Proposal Network
feature map
CNN
image

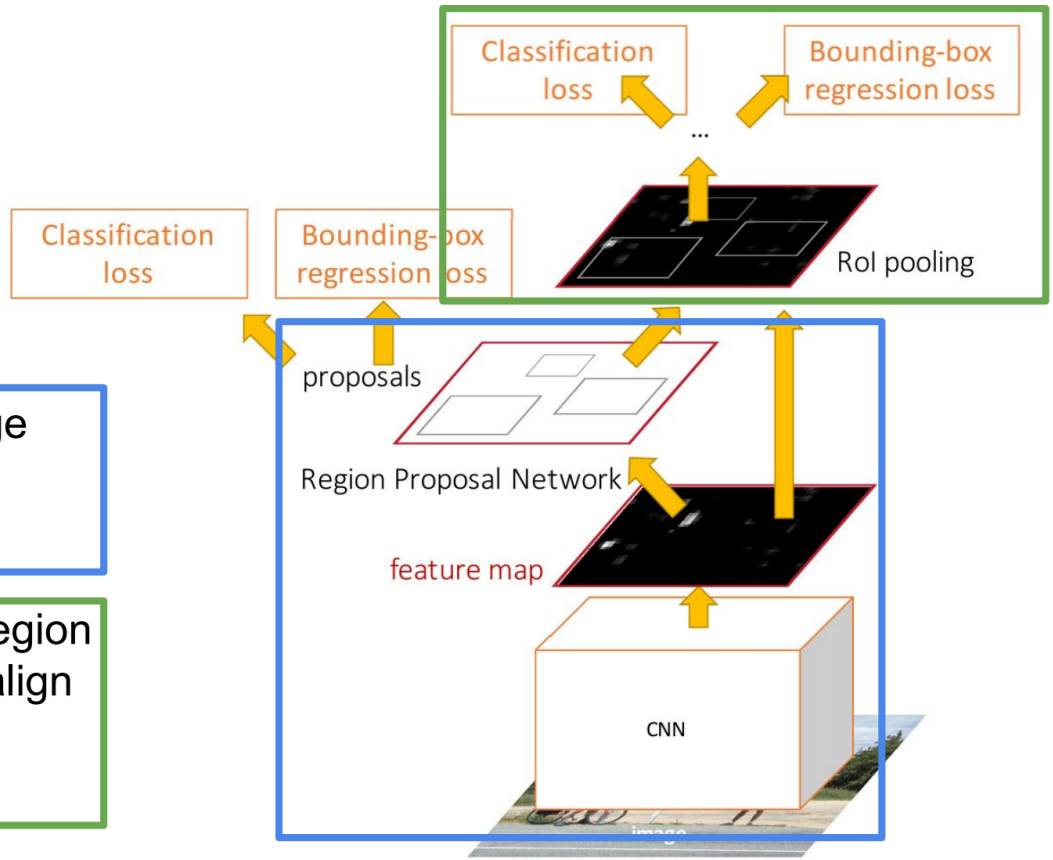
Faster R-CNN (cont.)

Faster R-CNN:
Make CNN do proposals!

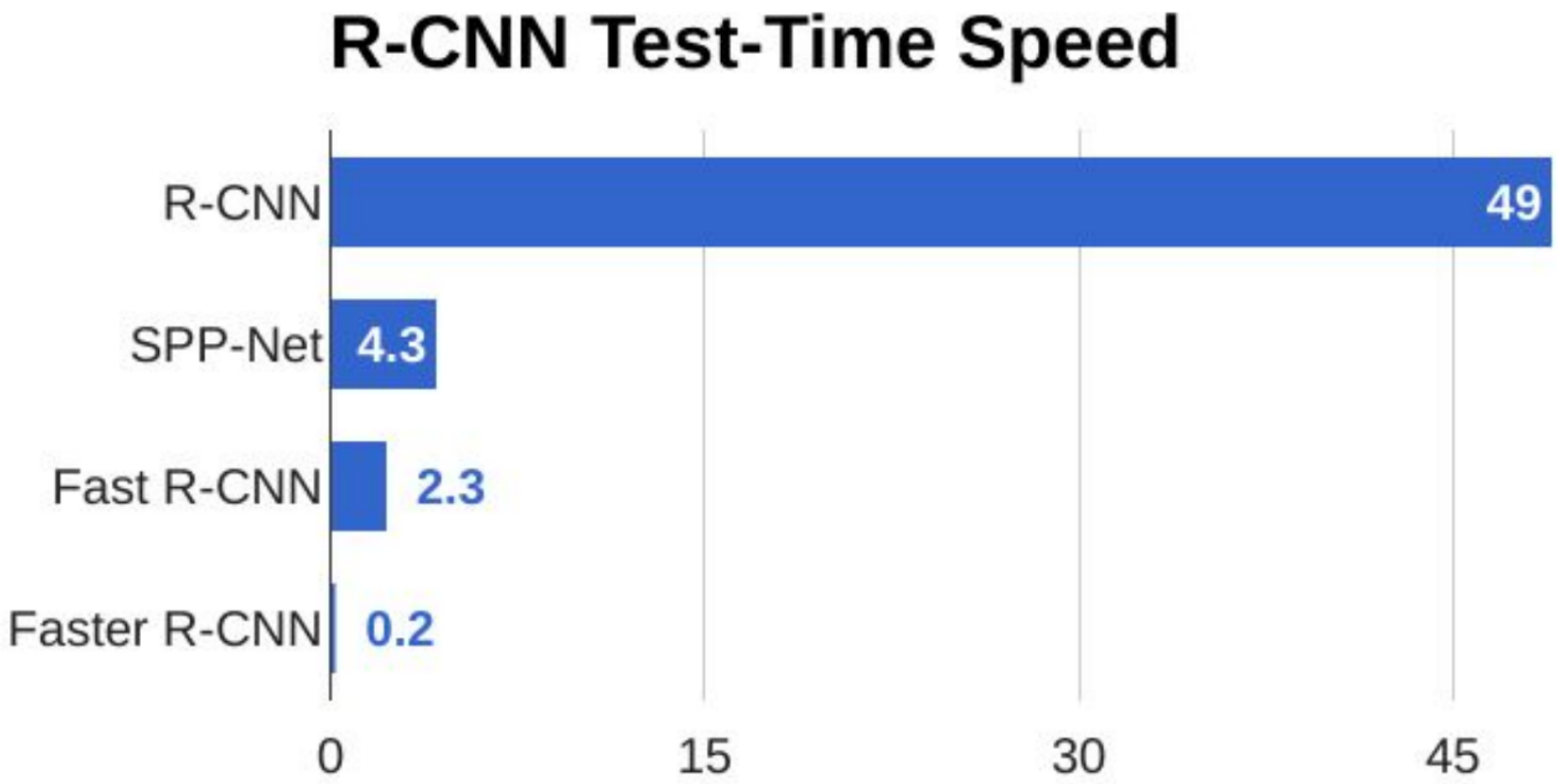
Faster R-CNN is a
Two-stage object detector

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

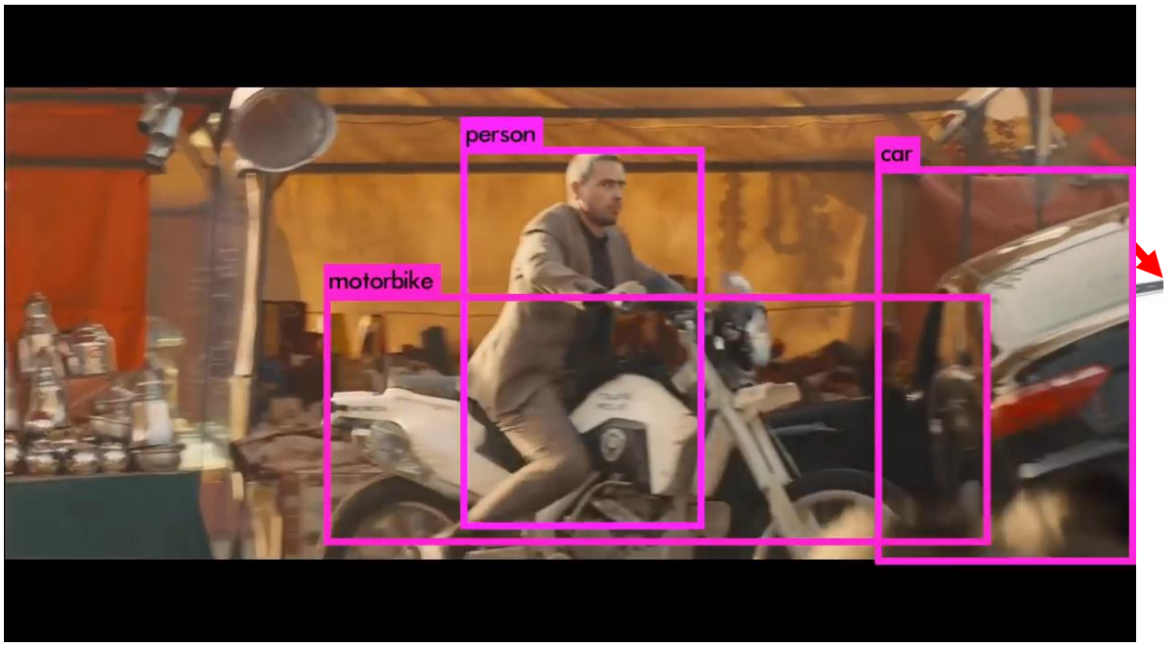
Second stage: Run once per region
- Crop features: RoI pool / align
- Predict object class
- Prediction bbox offset



Faster R-CNN (cont.)



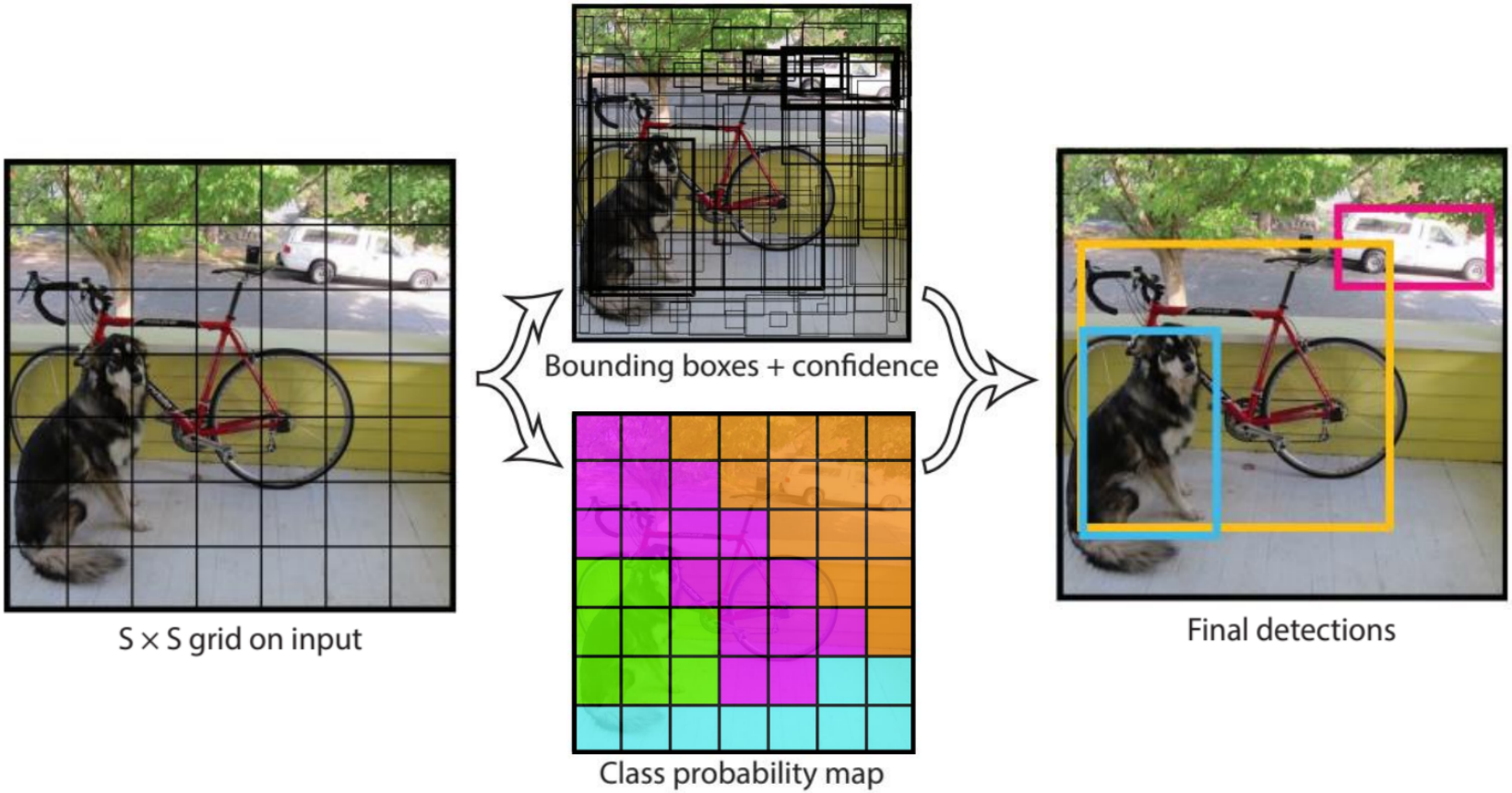
Single Shot Object Detection



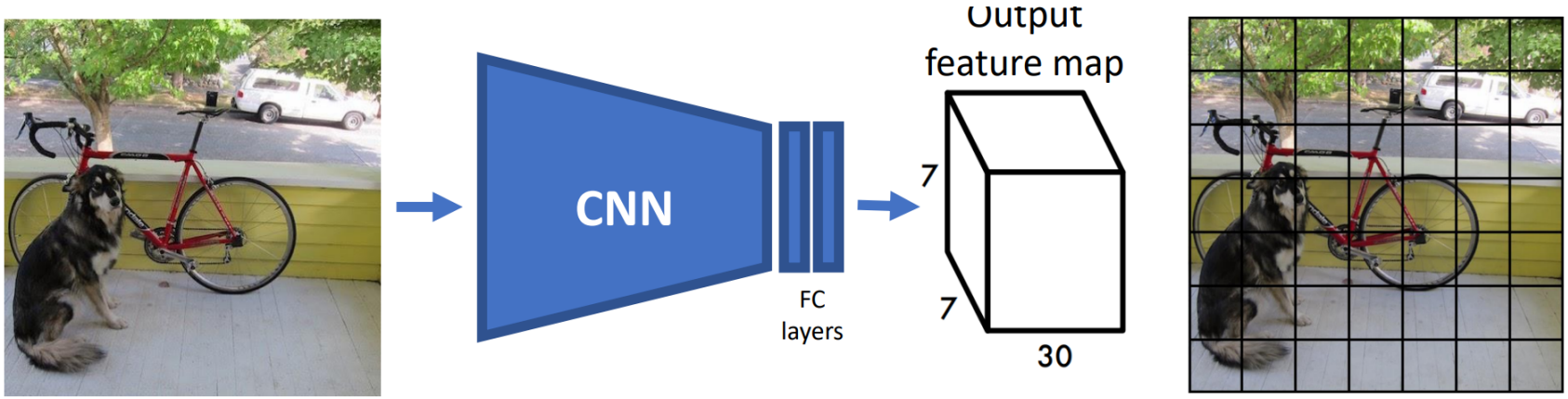
Single Shot:
SSD, YOLO ...

Fast
High false rate

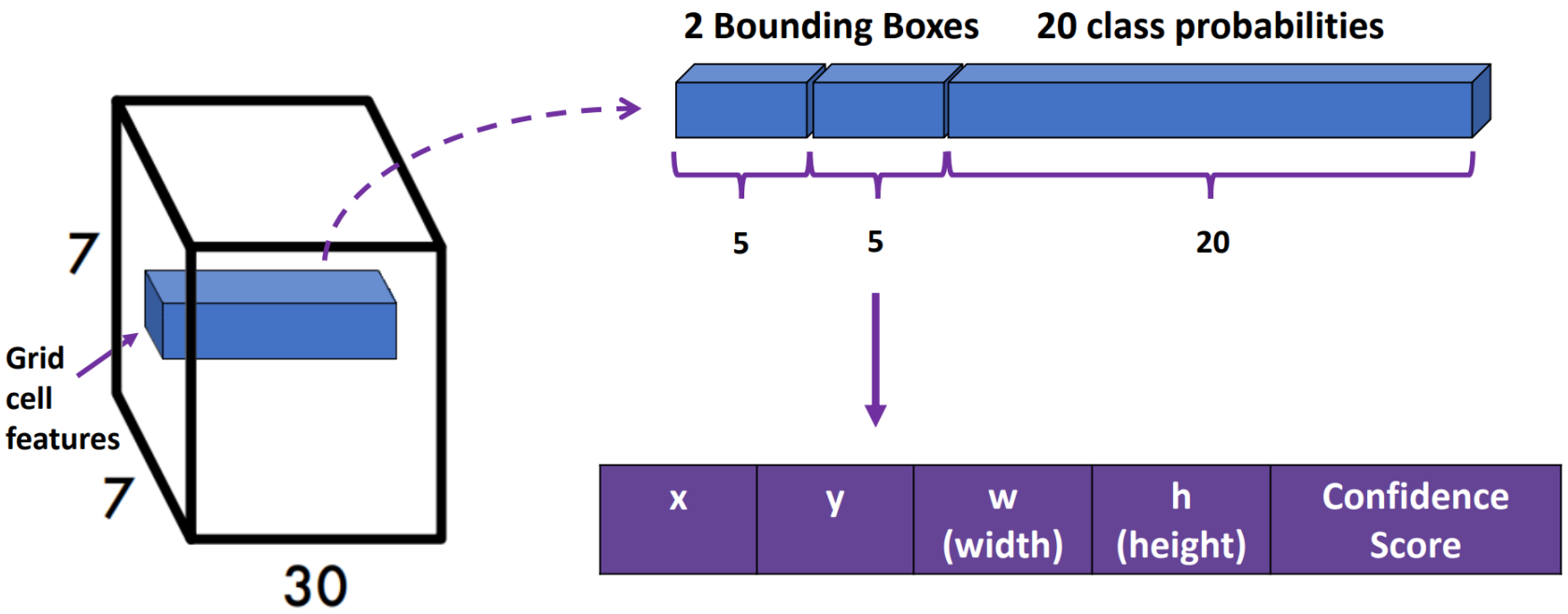
YOLO - Overview

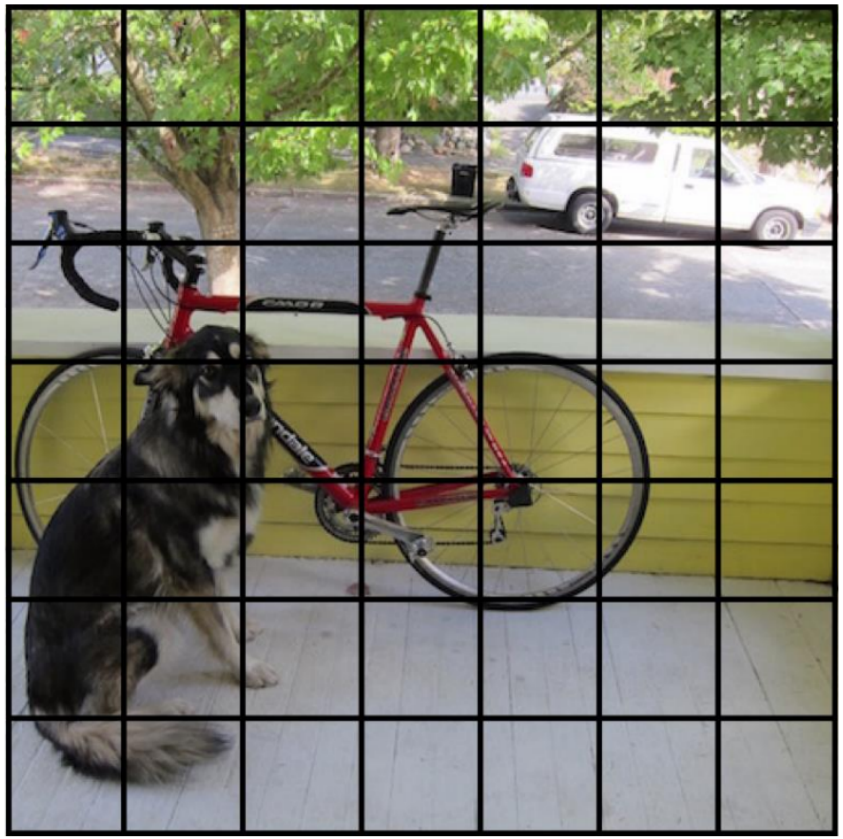


YOLO - Overview

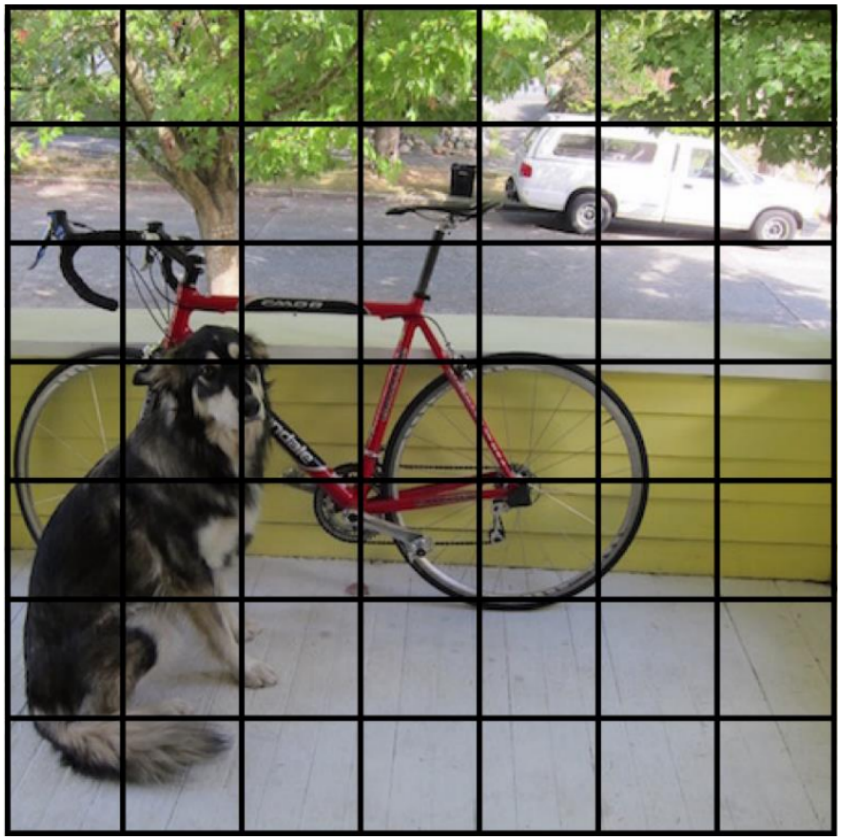


YOLO - Overview



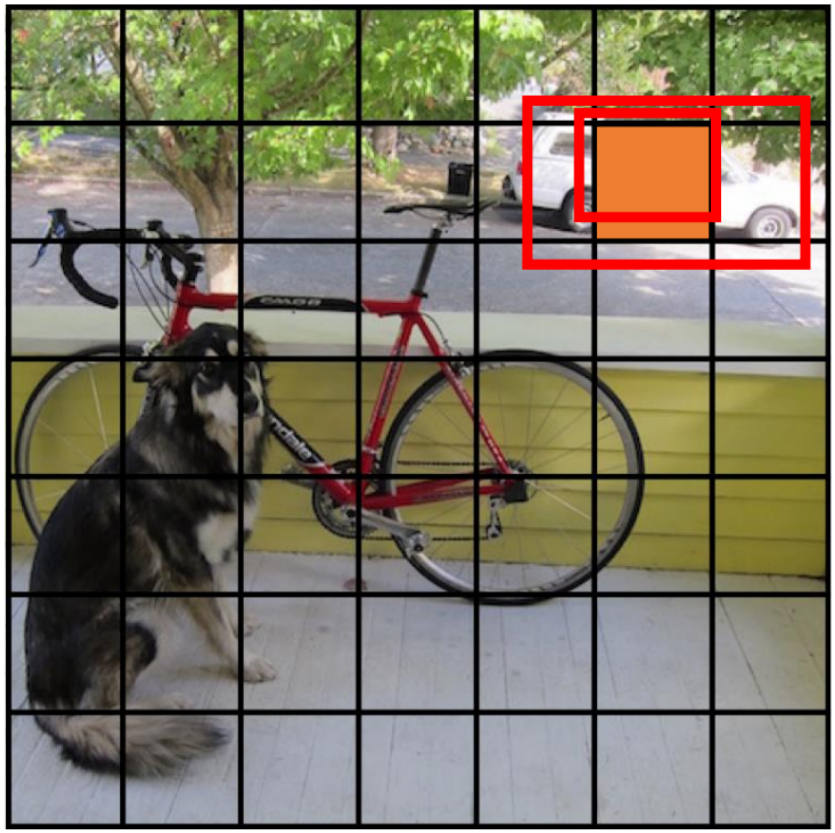


Each cell predicts



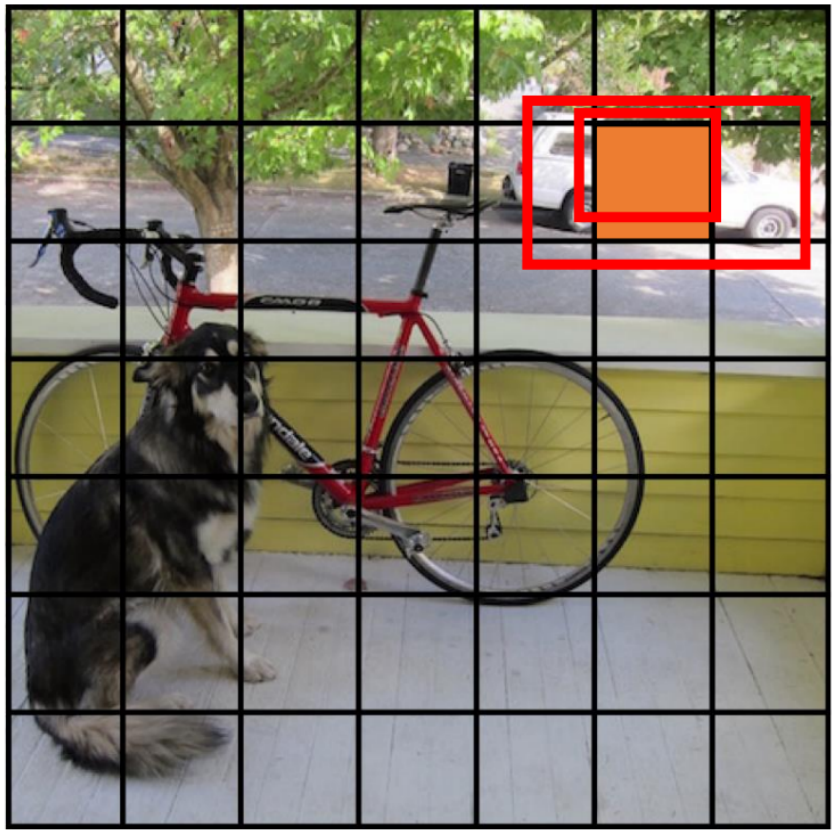
Each cell predicts

- $B = 2$ bounding boxes $(x, y, w, h)+$ confidence score



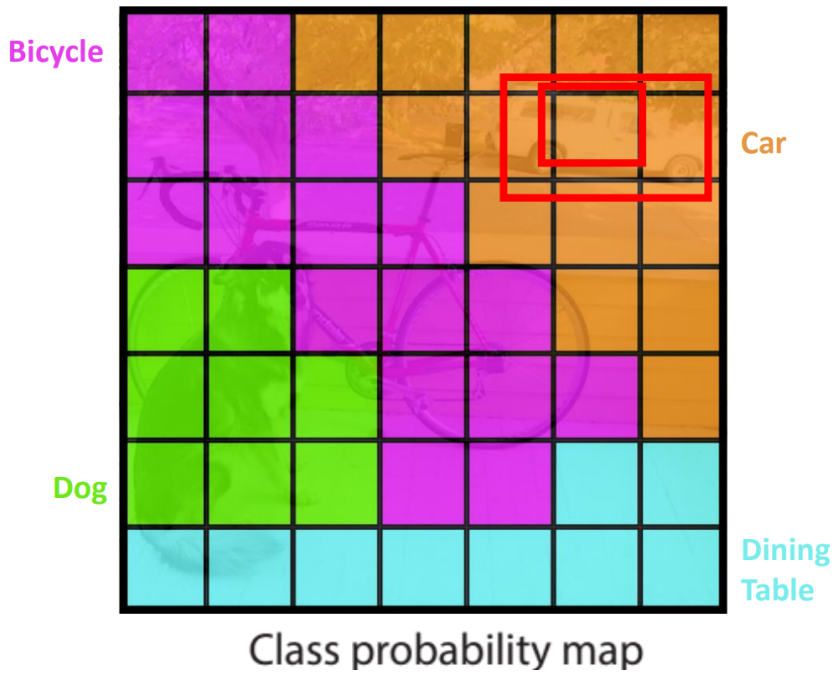
Each cell predicts

- $B = 2$ bounding boxes $(x, y, w, h)+$ confidence score
- $C = 20$ class probabilities



Each cell predicts

- $B = 2$ bounding boxes $(x, y, w, h)+$ confidence score
- $C = 20$ class probabilities



Each cell predicts

- $B = 2$ bounding boxes $(x, y, w, h)+$ confidence score
- $C = 20$ class probabilities

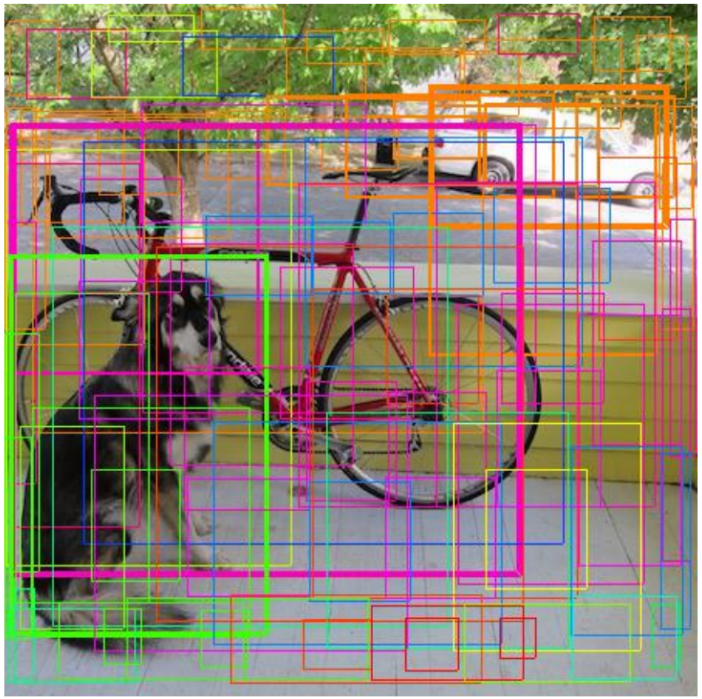
SxSxB Bounding-Boxes (S=7,B=2 → 96 Bboxs)

S × S grid on input

Each cell predicts

- $B = 2$ bounding boxes $(x, y, w, h)+$ confidence score
- $C = 20$ class probabilities

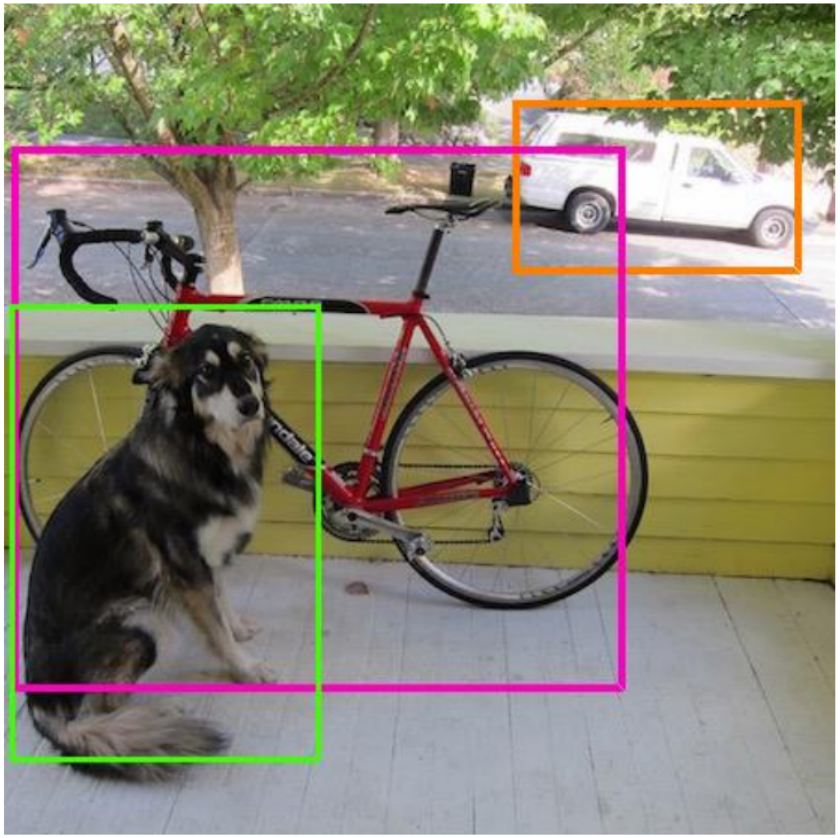
SxSxB Bounding-Boxes (S=7,B=2 → 96 Bboxs)



Each cell predicts

- $B = 2$ bounding boxes $(x, y, w, h)+$ confidence score
- $C = 20$ class probabilities

- Apply Non-Maximum Suppression



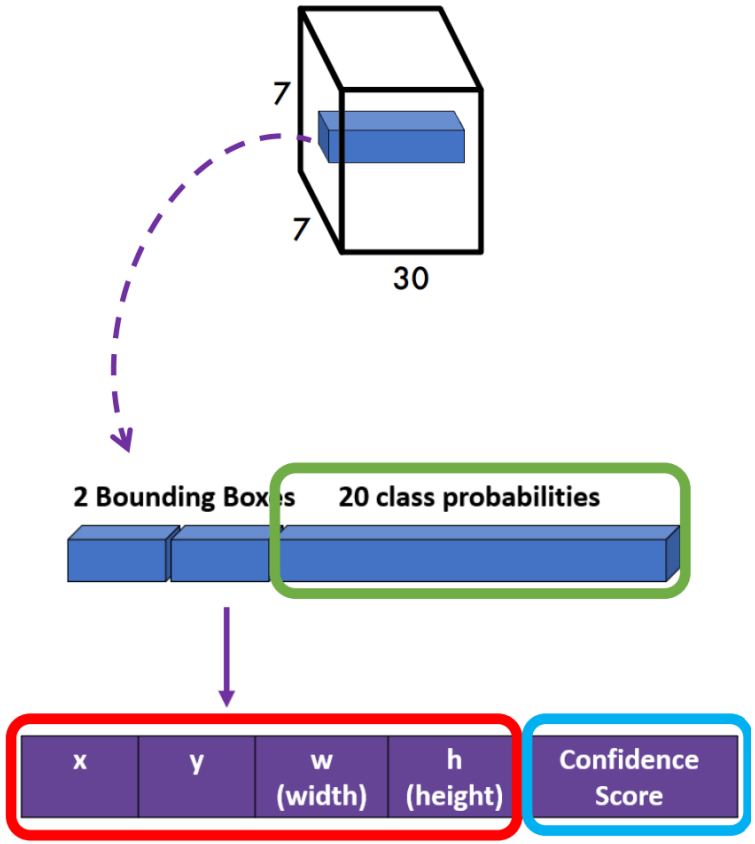
YOLO - Loss Function

YOLO – Loss function

$$\mathcal{L} = \mathcal{L}_{Localization\ Loss}$$

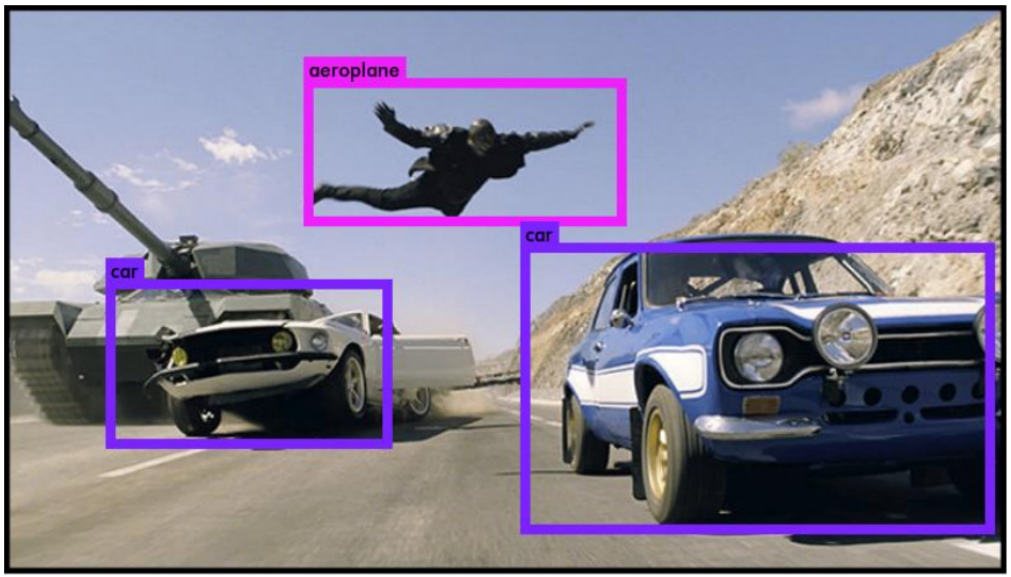
$$+ \mathcal{L}_{Confidence\ Loss}$$

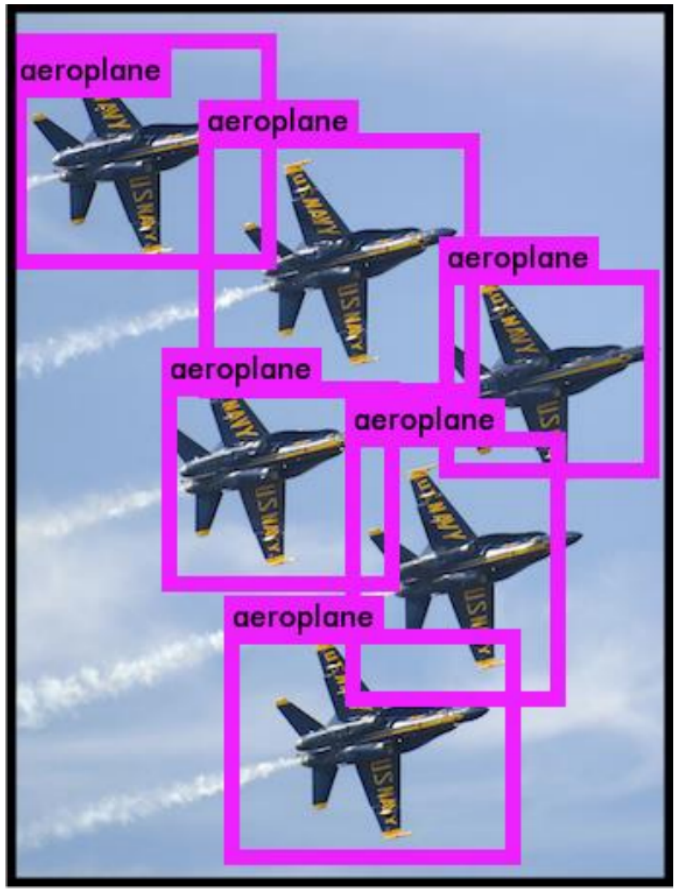
$$+ \mathcal{L}_{Classification\ Loss}$$



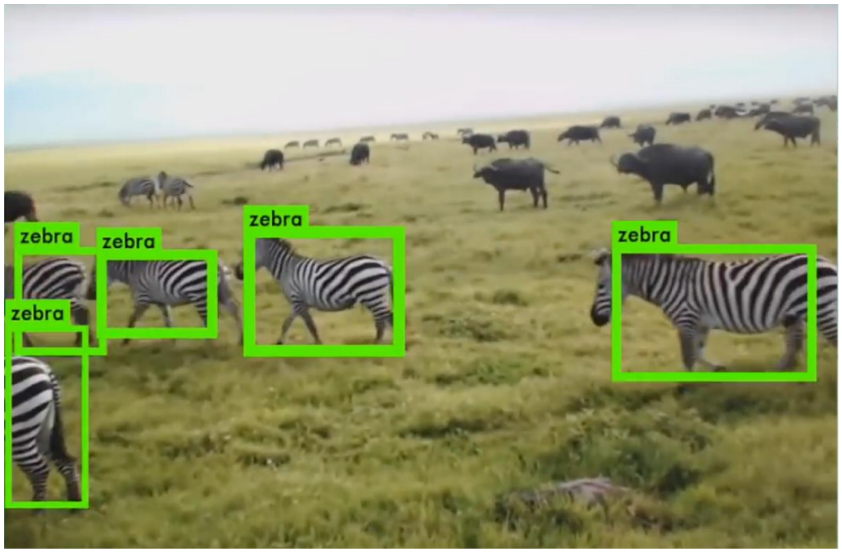
YOLO - Benefits

- Fast. Good for real-time processing
- End-to-end training



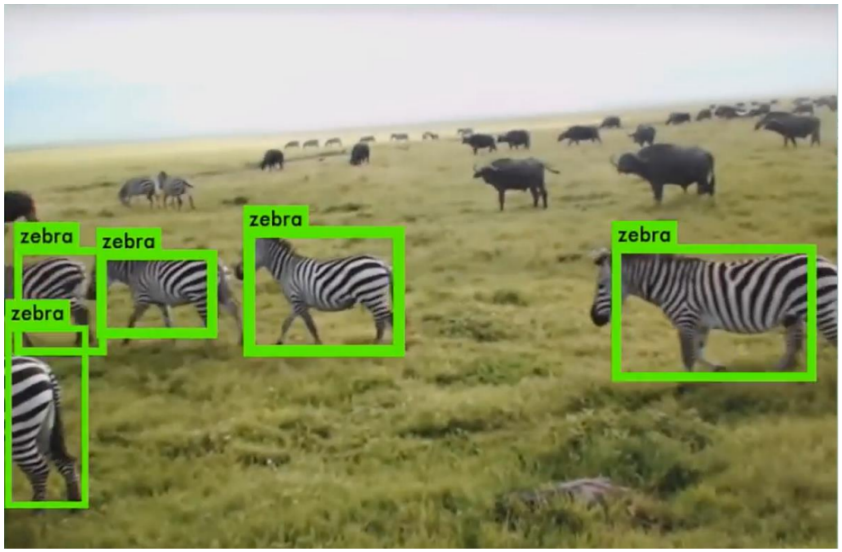


YOLO - Limitations



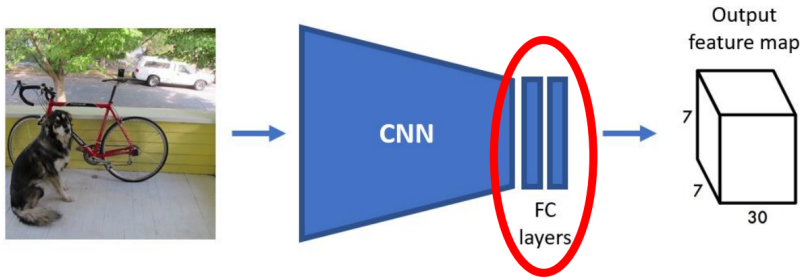
YOLO - Limitations

- Difficult to detect small objects
- Coarse predictions



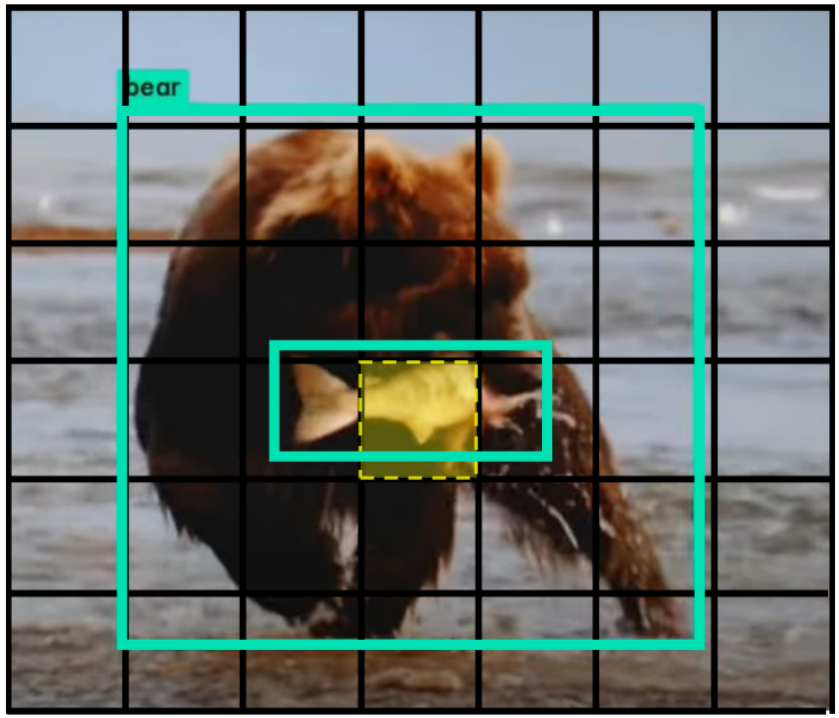
YOLO - Limitations

- Difficult to detect small objects
- Coarse predictions
- Fixed input size



YOLO - Limitations

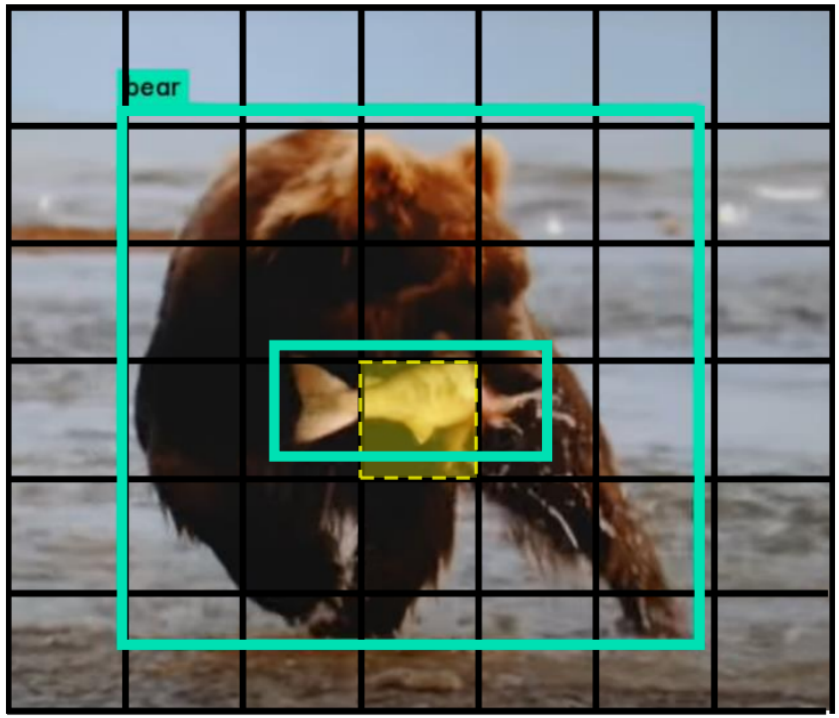
- Difficult to detect small objects
- Coarse predictions
- Fixed input size
- A grid cell can predict only one class



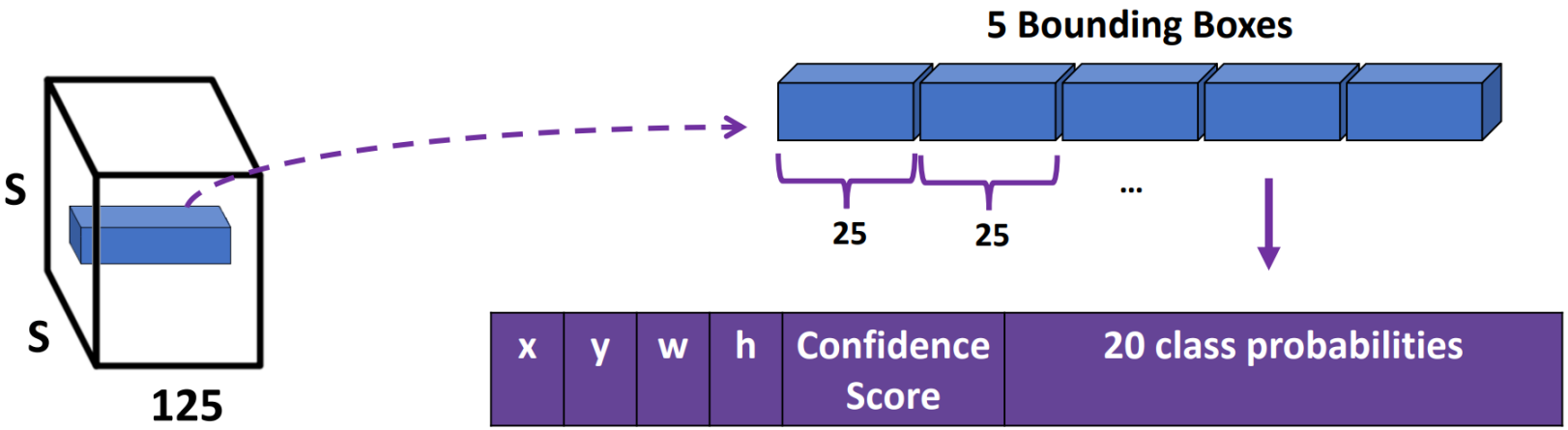
YOLO - Limitations

- Difficult to detect small objects
- Coarse predictions
- Fixed input size
- A grid cell can predict only one class

- Solutions:
  - Remove fc layers!
  - Predict class per bbox (not per cell)



- Removed fully connected layers
- A grid cell predicts class probabilities for each box



There's always room for improvement!

- YOLOv3
  - J. Redmon, A. Farhadi. Yolov3: An incremental improvement, 2018
- YOLOv4
  - A. Bochkovskiy, C. Wang, H. Liao. Yolov4: Optimal speed and accuracy of object detection (Feb. 2020)
- YOLOv5
  - YOLOv5 by ultralytics (June 2020)
- PP-YOLO
  - X. Long, K. Deng, G. Wang, Y. Zhang, Q. Dang, Y. Gao, H. Shen, J. Ren, S. Han, E. Ding, S. Wen. Pp-yolo: An effective and efficient implementation of object detector (June 2020)
- PP-YOLOv2 (2021)
  - J. X. Huang, X. Wang, W. Lv, X. Bai, X. Long, K. Deng, Q. Dang, S. Han, Q. Liu, X. Hu, D. Yu, Y. Ma, O. Yoshie. PP-YOLOv2: A Practical Object Detector (2021)

Object Detection: Summary

- Object detection is a complex task that involves identifying and localizing objects within images.
- Key challenges include:
  - Variability in object appearance, scale, and orientation.
  - Occlusion and clutter in images.
  - Real-time processing requirements for applications like autonomous driving.
- Techniques range from traditional methods like sliding windows to advanced deep learning approaches like YOLO and Faster R-CNN.
- Intersection over Union (IoU) is a critical metric for evaluating the accuracy of object detection models.
- YOLO (You Only Look Once) is a popular real-time object detection algorithm that processes images in a single pass, making it efficient for applications requiring speed.

References

These slides have been adapted from

- Fei-Fei Li, Yunzhu Li & Ruohan Gao, Stanford CS231n: Deep Learning for Computer Vision
- Assaf Shocher, Shai Bagon, Meirav Galun & Tali Dekel, WAIC DL4CV Deep Learning for Computer Vision: Fundamentals and Applications
- Justin Johnson, UMich EECS 498.008/598.008: Deep Learning for Computer Vision

Credits

Dr. Prashant Aparajeya

Computer Vision Scientist — Director(AISimply Ltd)

p.aparajeya@aisimply.uk

This project benefited from external collaboration, and we acknowledge their contribution with gratitude.