You Only Look Once

path to design a detector

Feng Wang

AIRD, Coretronic Co.

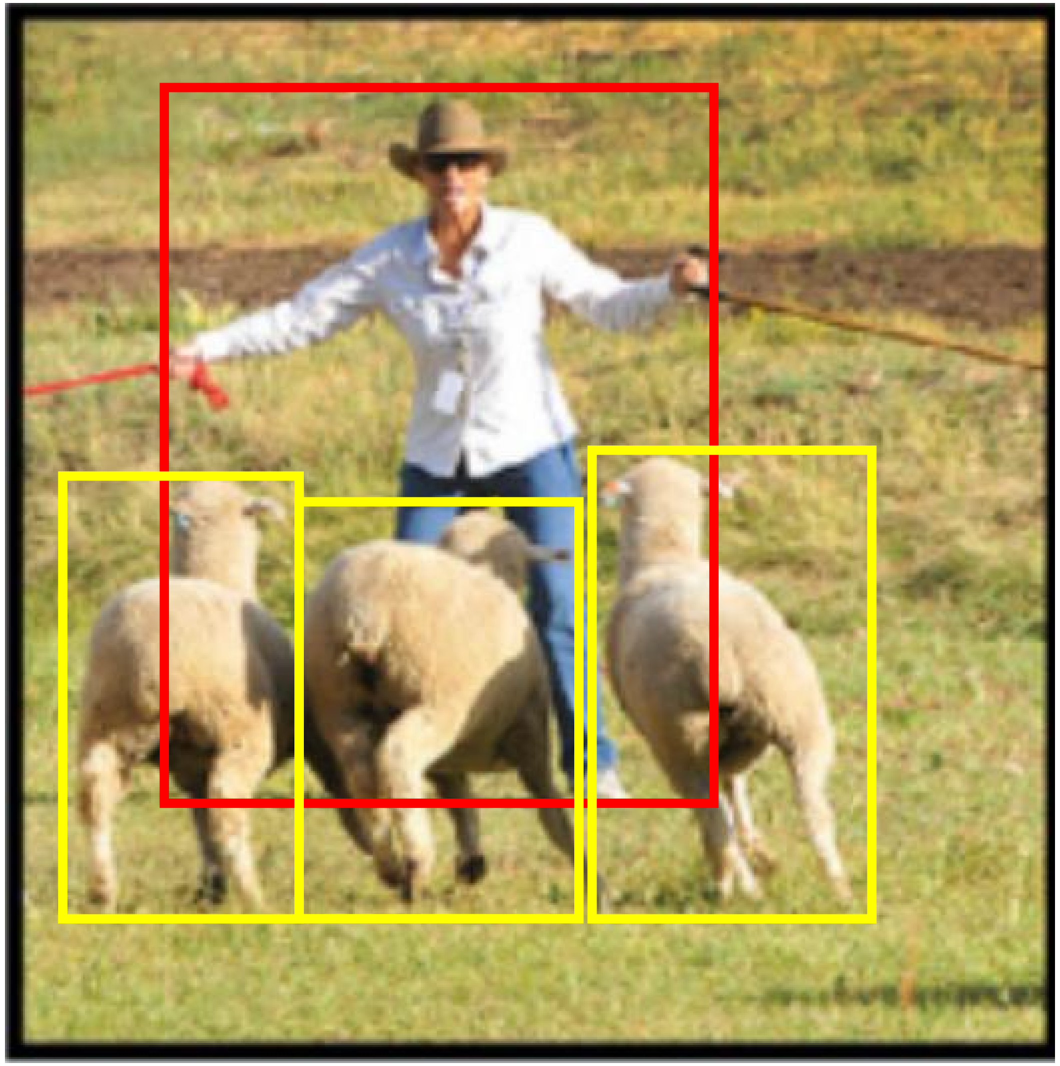
Apr 17, 2019

The slides and a list of references can be found from
https://github.com/fwcore/object-detection

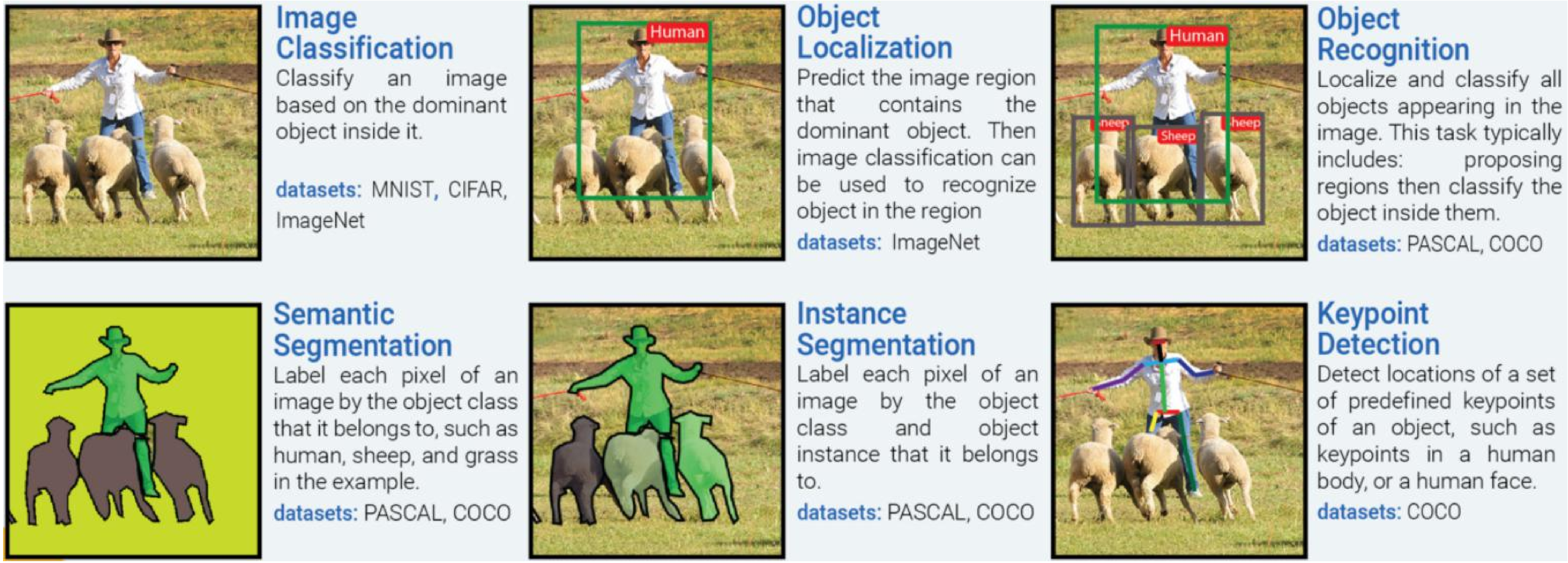
Outlines

- **Concepts in object detection**
- **A brief history of object detection**
- **YOLO**
  - design
  - loss function
  - training
  - weaknesses

Classification vs detection/recognition



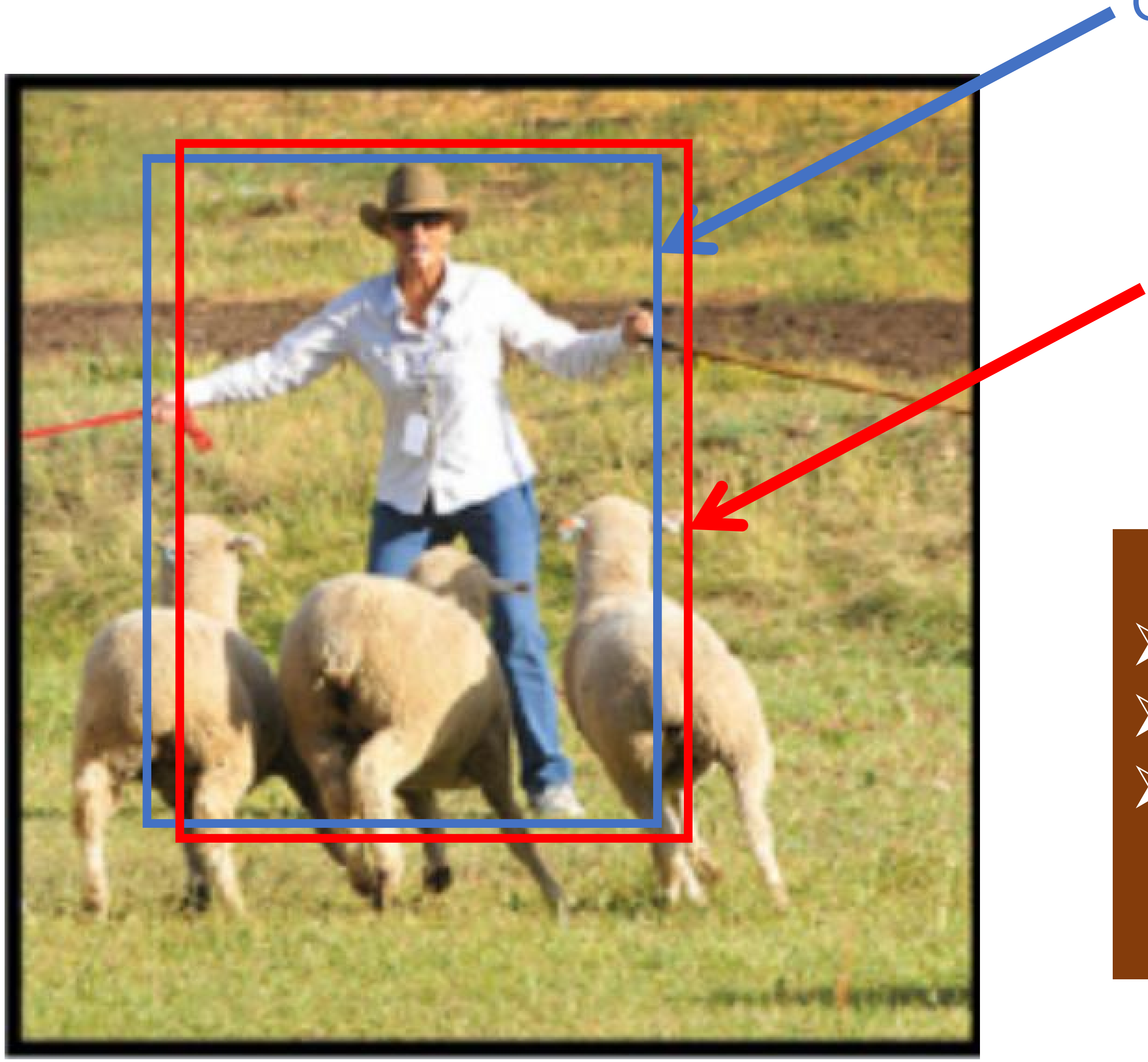
Common tasks on images



https://medium.com/@nikasa1889/the-modern-history-of-object-recognition-infographic-aea18517c318

Bounding box proposal

Region of interest, region proposal, box proposal

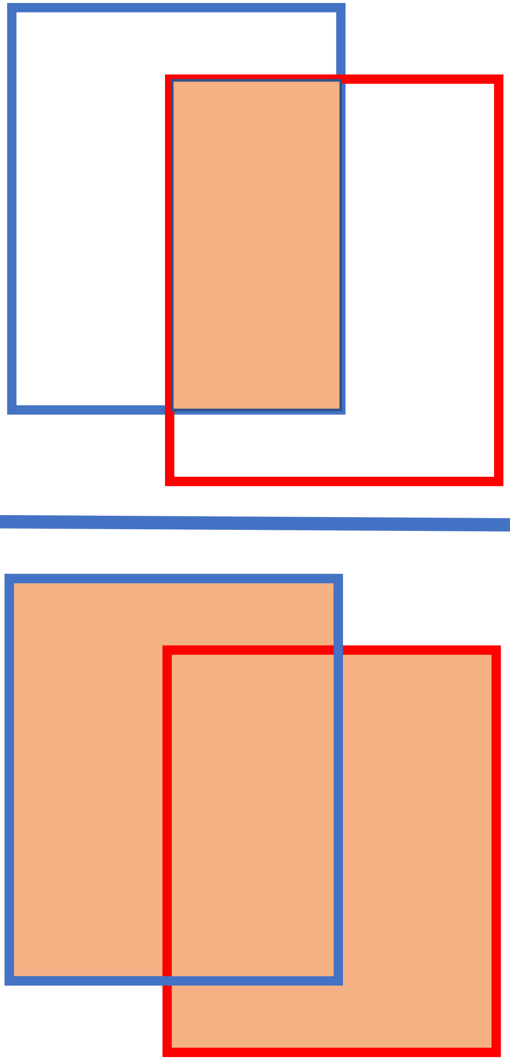


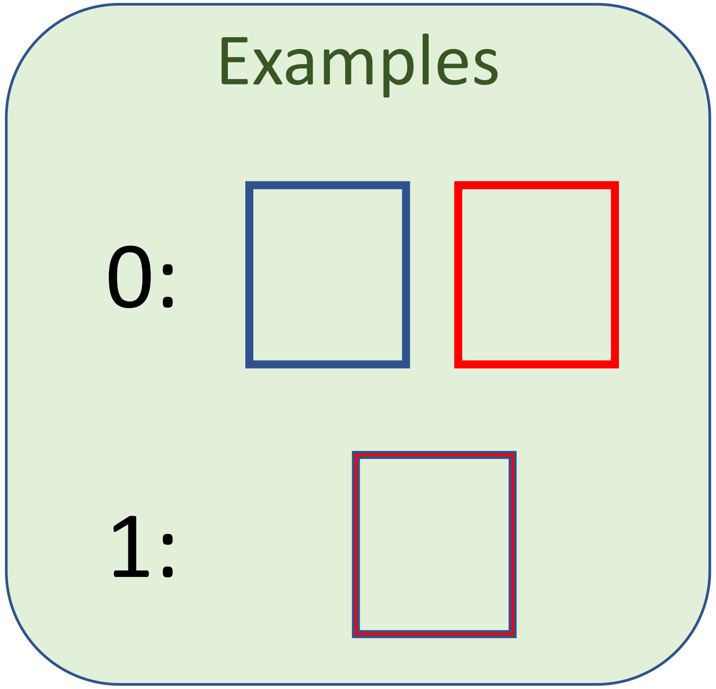
5 parameters

- w, h
- x, y
- confidence score: how likely it contains an object & accuracy of the box

How good: Intersection over Union (IOU)

$$\mathbf{IOU} = \frac{\textbf{Overlap Area}}{\textbf{Union Area}}$$

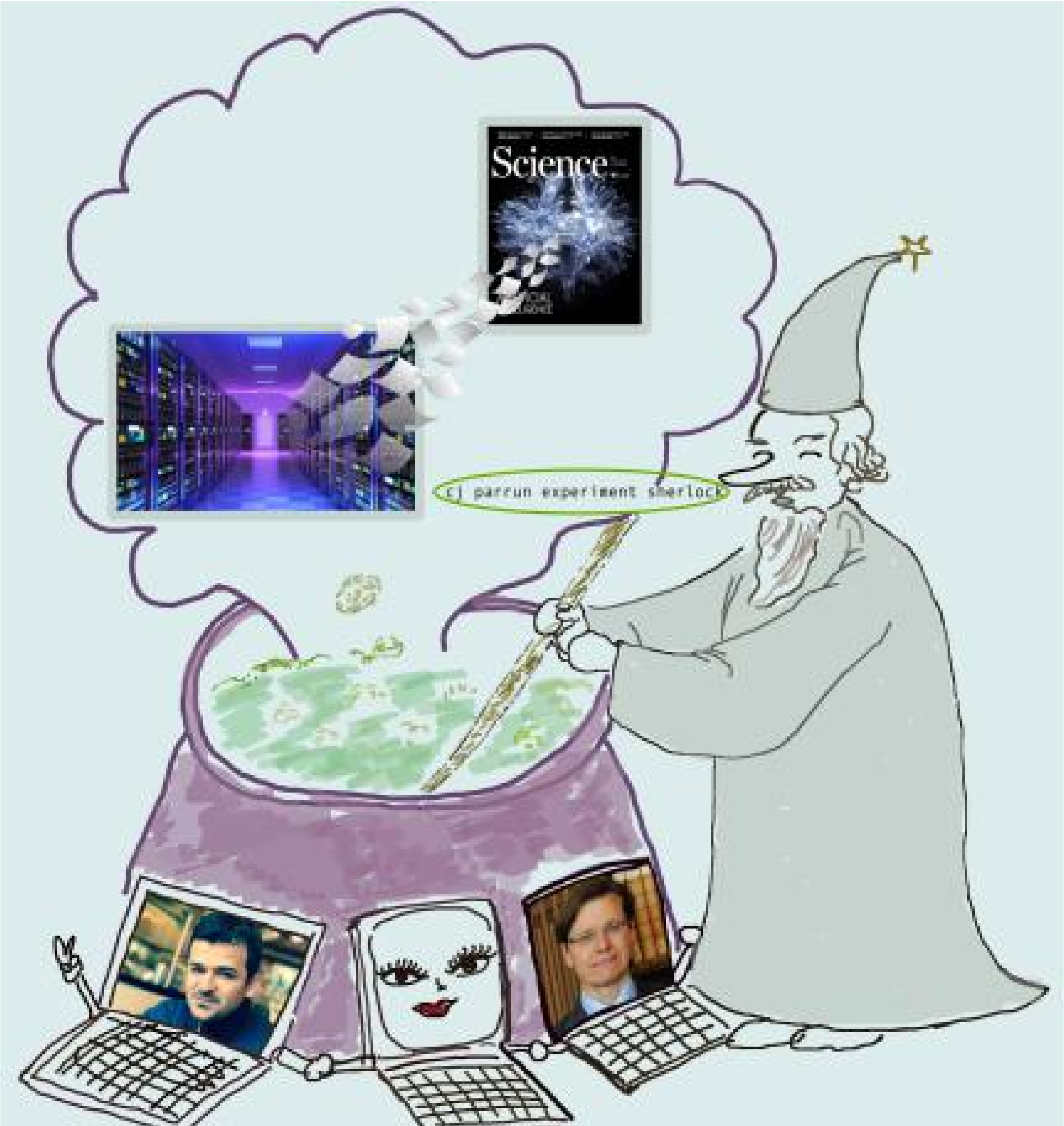




Outlines

- Concepts in object detection
- **A brief history of object detection**
- YOLO
  - design
  - loss function
  - training
  - weaknesses

A brief history of object detection

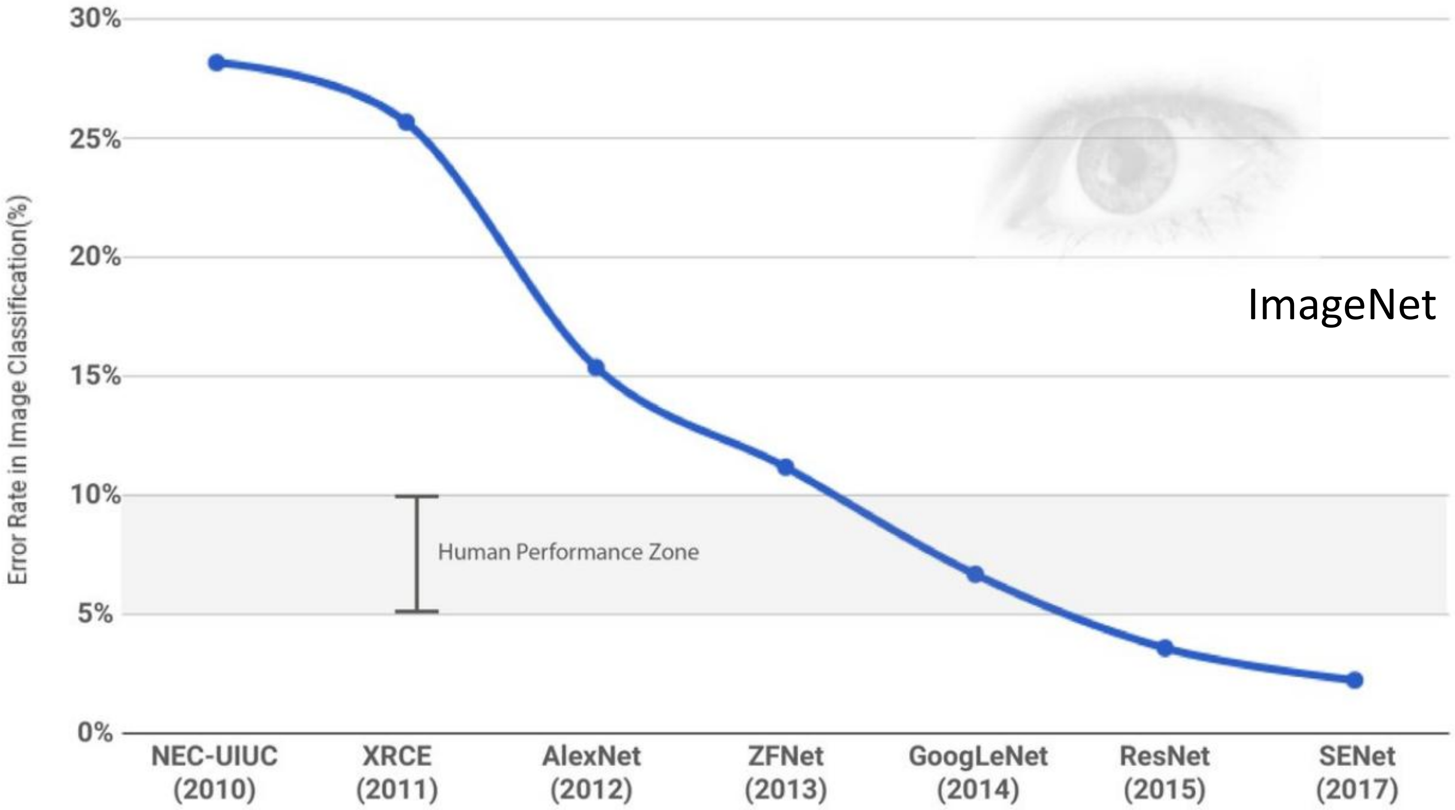


https://stats385.github.io

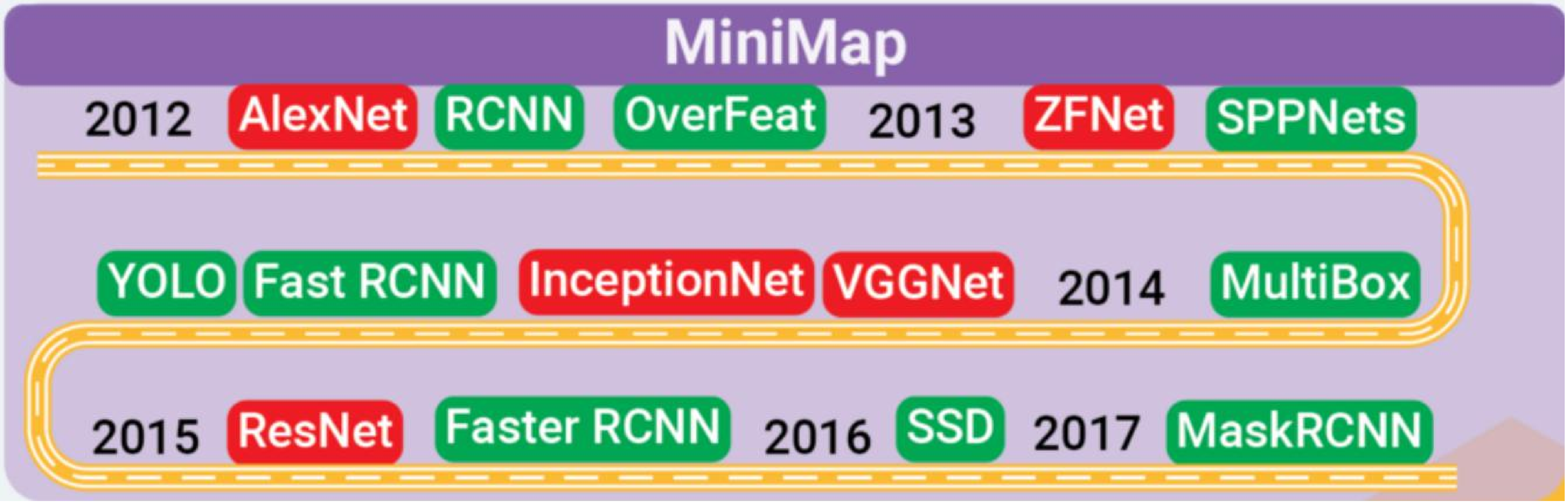
A brief history of object detection

- Before CNN, people use handcrafted features to locate and classify objects. (not too bad)

- CNN boosts the accuracy of classification



A brief history of object detection



Region proposal -> classification

- e.g. RCNN
- accurate
- slow

Single shot: Region proposal + classification

- e.g. YOLO, SSD
- fast
- less accurate

Outlines

- Concepts in object detection
- A brief history of object detection
- **YOLO**
  - design
  - loss function
  - training
  - weaknesses

YOLO: you look only once

Look once

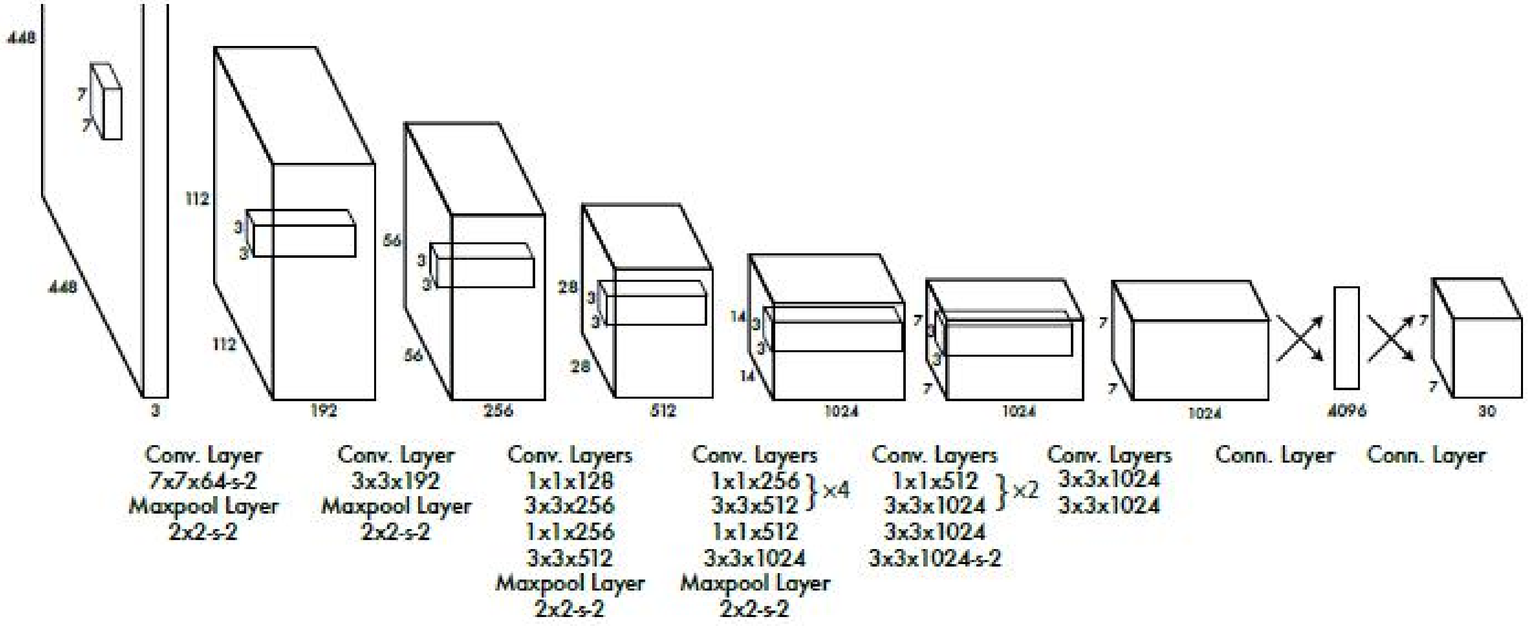
Results

- x, y, w, h
- confidence score: contain an object & box accuracy
- class score: belong to a class

Let's use CNN, since it's good.

Why not regress? They are just numbers.

Let's go to CNN



YOLO v1's CNN: GoogLeNet variant, 24 layers

	Type	Filters	Size	Output
	Convolutional	32	3 x 3	256 x 256
	Convolutional	64	3 x 3 / 2	128 x 128
1x	Convolutional	32	1 x 1	
	Convolutional	64	3 x 3	
	Residual			128 x 128
	Convolutional	128	3 x 3 / 2	64 x 64
2x	Convolutional	64	1 x 1	
	Convolutional	128	3 x 3	
	Residual			64 x 64
	Convolutional	256	3 x 3 / 2	32 x 32
8x	Convolutional	128	1 x 1	
	Convolutional	256	3 x 3	
	Residual			32 x 32
	Convolutional	512	3 x 3 / 2	16 x 16
8x	Convolutional	256	1 x 1	
	Convolutional	512	3 x 3	
	Residual			16 x 16
	Convolutional	1024	3 x 3 / 2	8 x 8
4x	Convolutional	512	1 x 1	
	Convolutional	1024	3 x 3	
	Residual			8 x 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

YOLO v3's CNN: darknet-53

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

YOLO v2's CNN: darknet-19, 19 layers

Let's do regression
-- wait, wait, how many bounding boxes? Where are they initially?

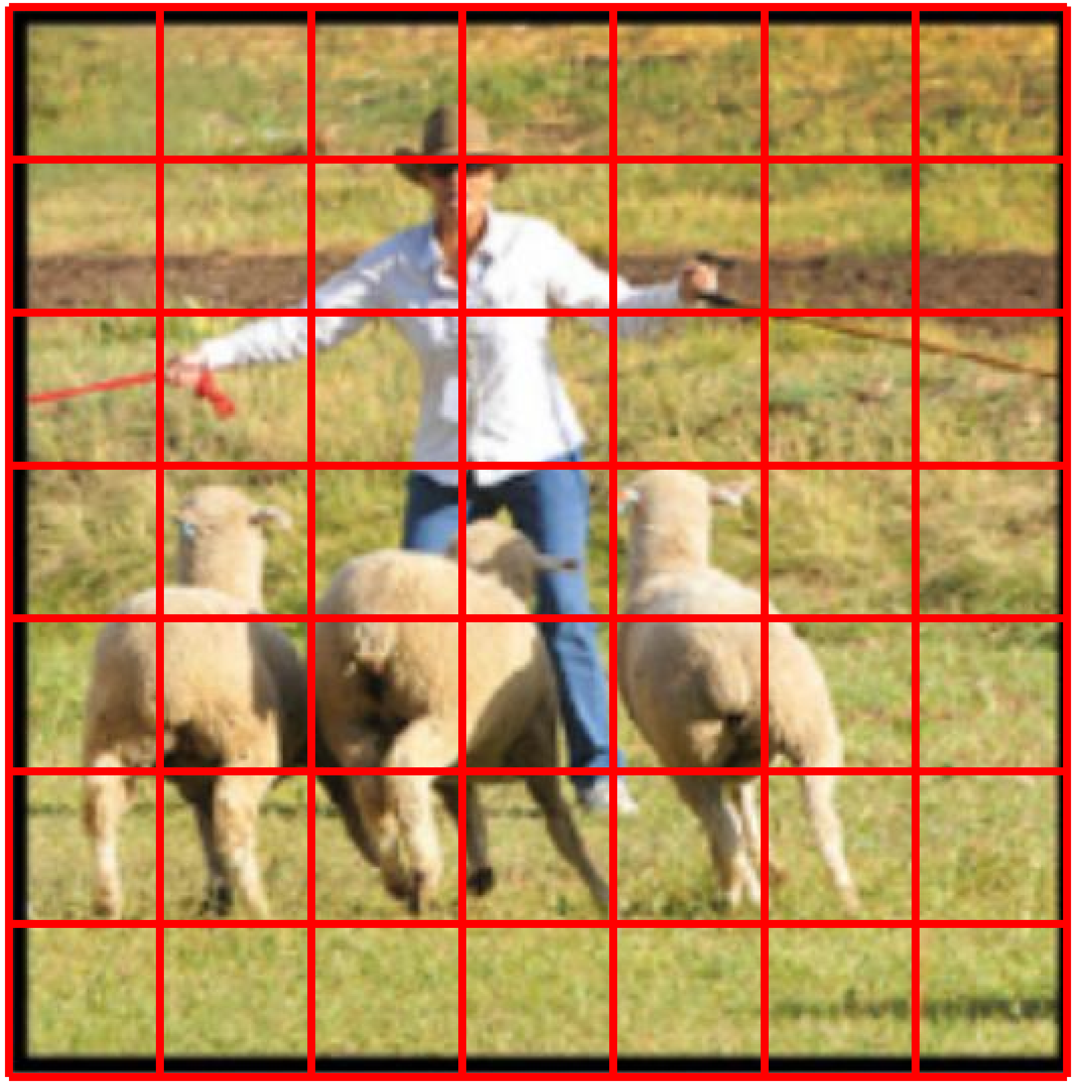
Results for one box

- x, y, w, h
- confidence score:
  contain an object &
  box accuracy
- class score:
  belong to a class

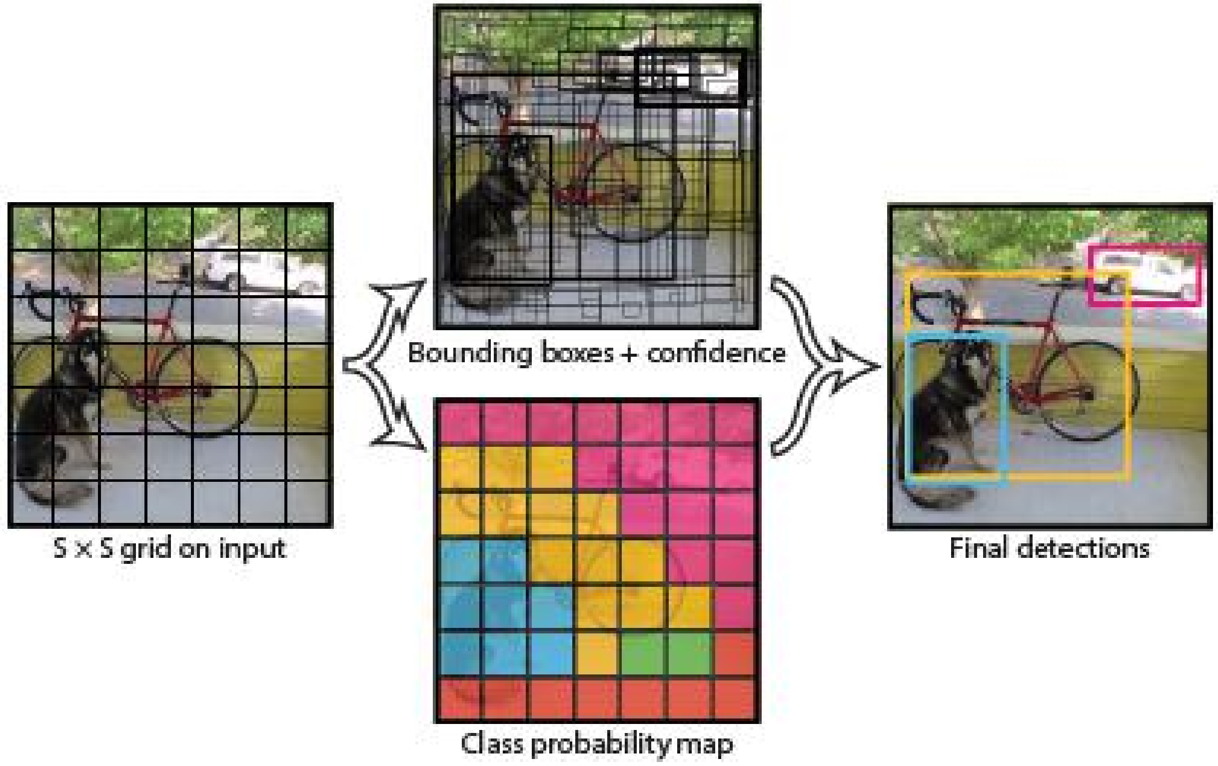
- Maybe set N as a large number?
- Maybe initially put them randomly?

Note: N is large, but much smaller than R-CNN's region proposal.

Better solution: using grids



Let's do regression with non-maximal suppression



We can use CNN to extract features, and finally perform a regression to detect objects.

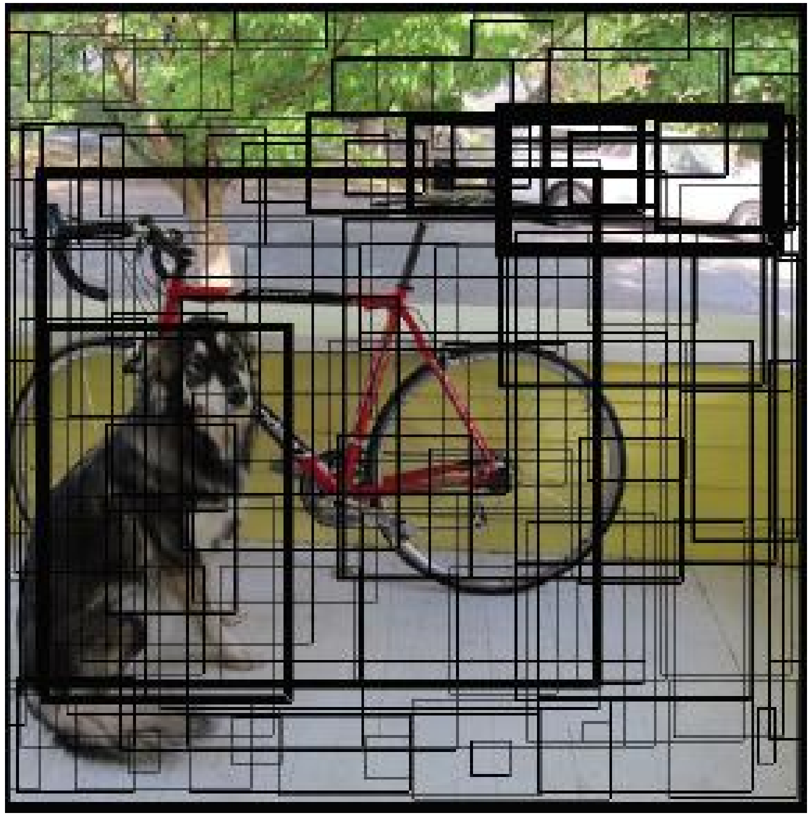
- YOLO v1: fully connected layers
- v2 & v3: convolutional layers

arXiv: 1506.02640, 1612.08242, 1804.02767

Grid 1	Proposed box 1 x, y, w, h confidence score	Proposed box 2 x, y, w, h confidence score	Class scores class 1 class 2, ... class 20
⋮	⋮	⋮	⋮
Grid SxS	Proposed box 1 x, y, w, h confidence score	Proposed box 2 x, y, w, h confidence score	Class scores class 1 class 2, ... class 20

vector size: SxSx(5x2+20)

Loss function



Problems

- ➢ One object is partially/fully covered by several boxes.
- ➢ Most boxes has no objects.
- ➢ Multi-task training problem: location & class
- ➢ Small objects need more accurate location & box size.

Solution

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

Oh, no math please. Let's speak human language

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \boxed{\mathbb{1}_{ij}^{\text{obj}}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \boxed{\mathbb{1}_{ij}^{\text{obj}}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \boxed{\mathbb{1}_{ij}^{\text{obj}}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \boxed{\mathbb{1}_{ij}^{\text{noobj}}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \boxed{\mathbb{1}_{i}^{\text{obj}}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

Problem 1:
One object is partially/fully covered by several boxes.

- Each true object has one proposed box “responsible” to it.
  Rule: the one with highest overlap with the ground truth boxes.
- When inference, we use non-maximal suppression to select the best among the proposals.

Human language

$$\boxed{\lambda_{\textbf{coord}}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

5

$$+ \boxed{\lambda_{\textbf{coord}}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \boxed{\lambda_{\textbf{noobj}}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

0.5

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

<u>Problem 2</u>: Most boxes has no objects.

Human language

$$\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

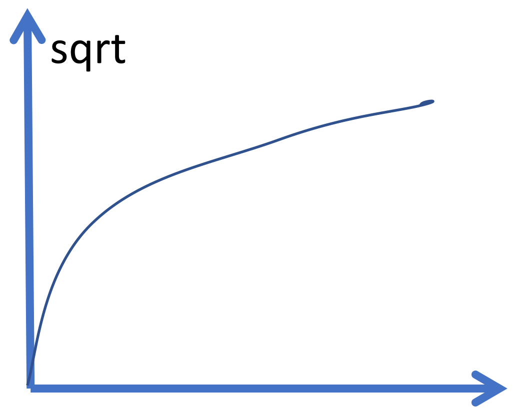
$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

Problem 3:
Multi-task training problem: location & class.

Weighted sum: here the problem is left untouched.

Human language

$$
\begin{aligned}
&\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\boxed{\left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2} \right] \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2
\end{aligned}
$$



<u>Problem 4</u>:
Small objects need more accurate location & box size.

Other problems

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \boxed{\left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2\right]}$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2\right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \boxed{\left(C_i - \hat{C}_i\right)^2}$$

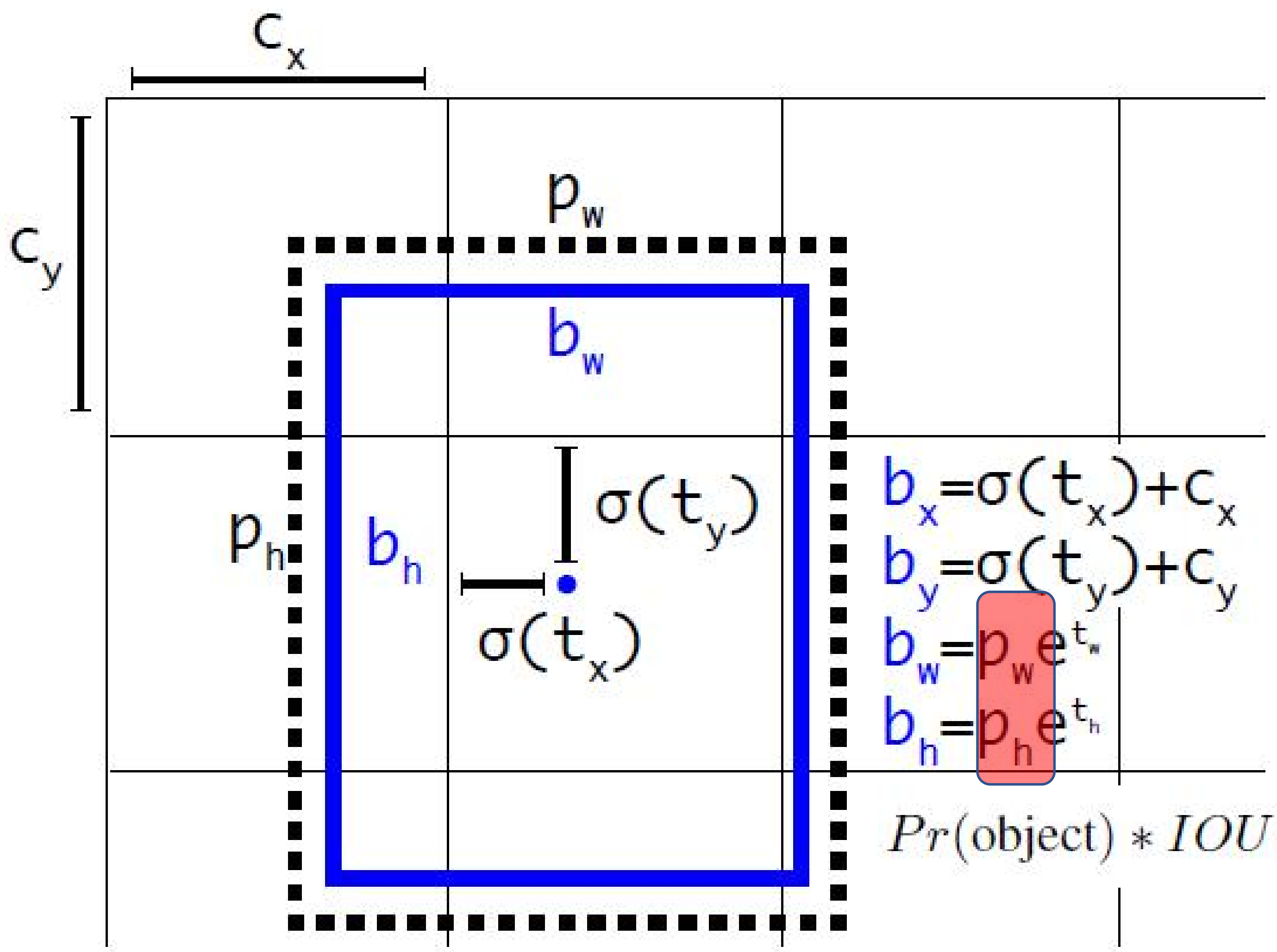
$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \boxed{\left(C_i - \hat{C}_i\right)^2}$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \boxed{(p_i(c) - \hat{p}_i(c))^2}$$

- x, y can be out of the grid cell
- smaller objects can locate worse than the largers

- probability can be out of [0, 1]

Fix them in YOLO v2



Pre-defined box size

$$Pr(\text{object}) * IOU(b, \text{object}) = \sigma(t_o)$$

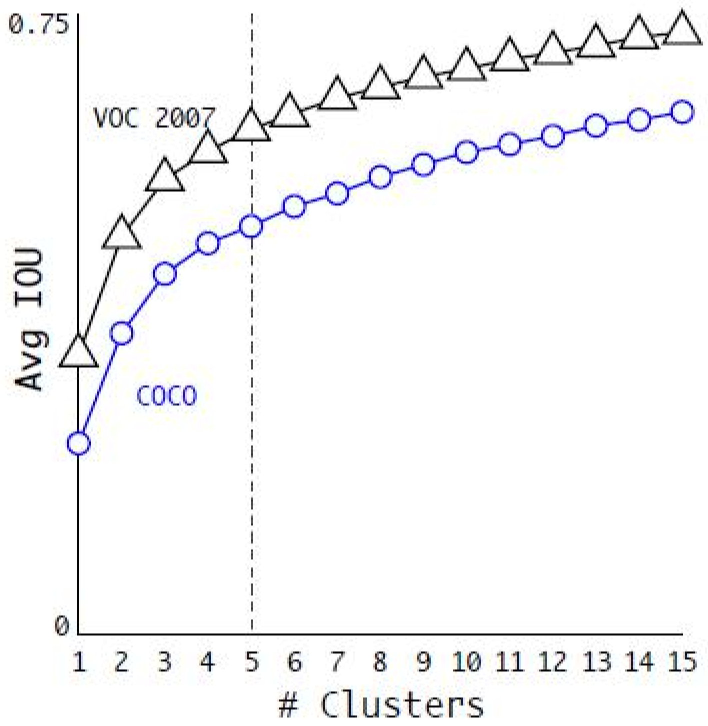
Pre-defined box: anchor

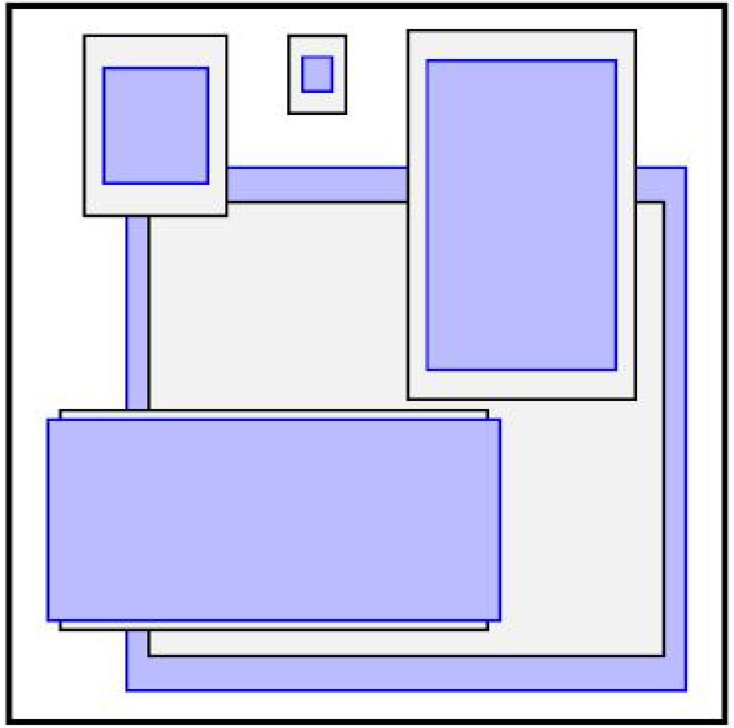
- Naturally, objects have special aspect ratios and sizes.
  - This can be a good starting point.
  - We don't need randomly initialized boxes' shapes.

- Handcrafted box size vs clustering algorithms

- Box can reshape during training.

- The number of pre-defined boxes is a hyperparameter
  - v2 uses 5
  - v3 uses 9





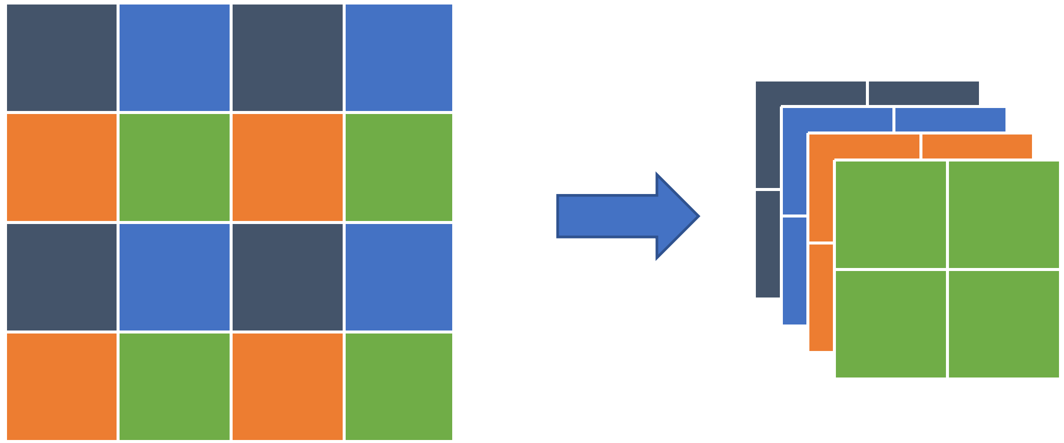
anchors used in YOLO v2

Anchor-free detection is a research topic, see https://arxiv.org/abs/1904.01355 for an instance.

Improvements (in v2)

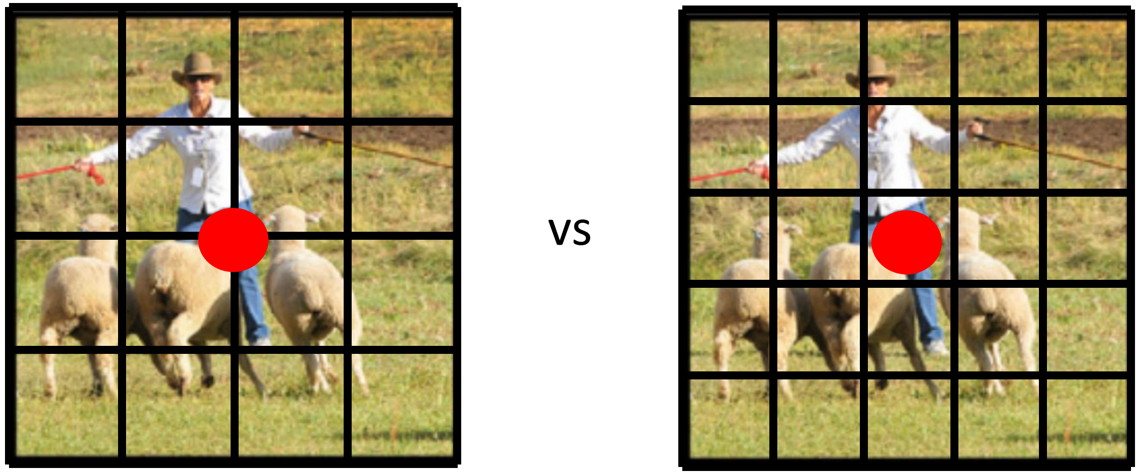
- Resizing image sizes randomly during training: {320, 352, ..., 608}
  - CNN only reduce an image by a constant factor (here 32), hence is robust to input image size
  - resize every 10 epochs.
  - multi-scale training

- Passthrough layer
  - No loss to perform reshaping



Feature map

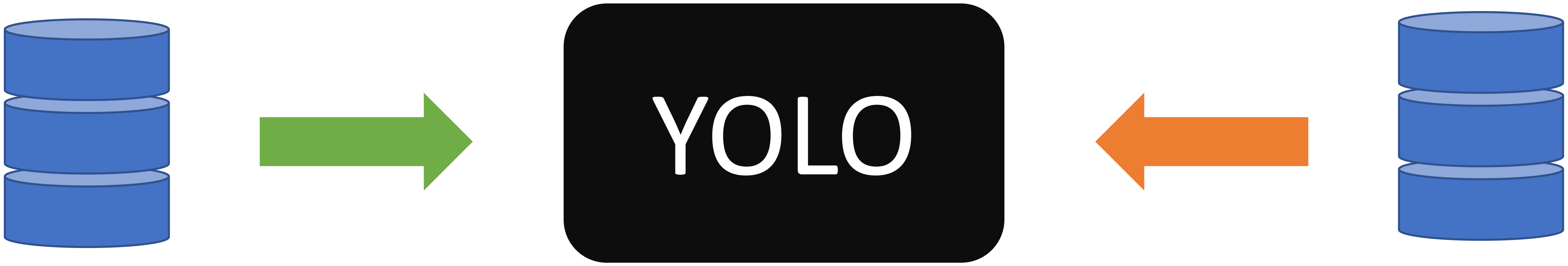
- Odd number of grid cells



Training

ImageNet:
classification dataset

COCO/PASCAL VOC:
detection dataset

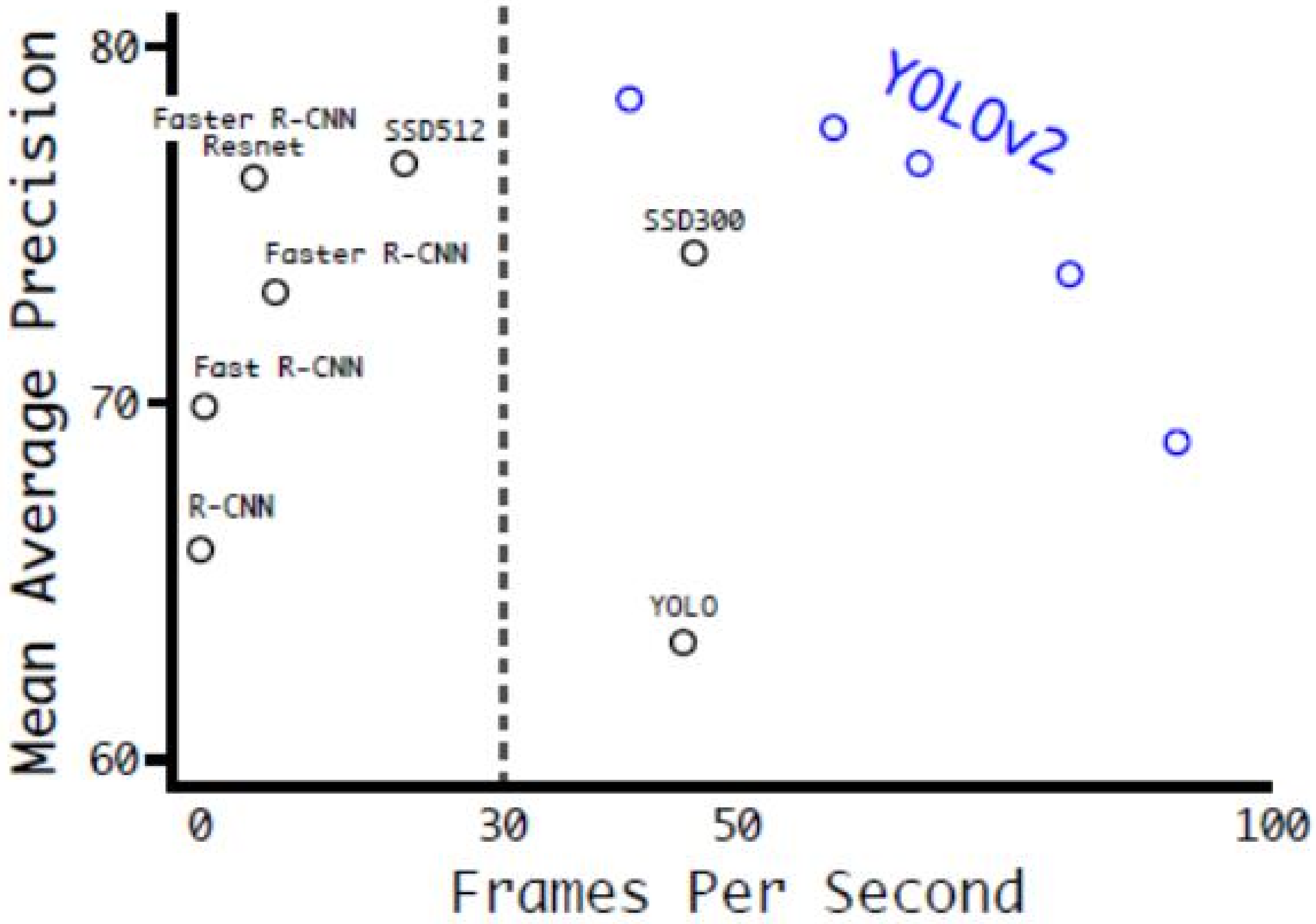


Step 1:
- ➢ train classification backbone

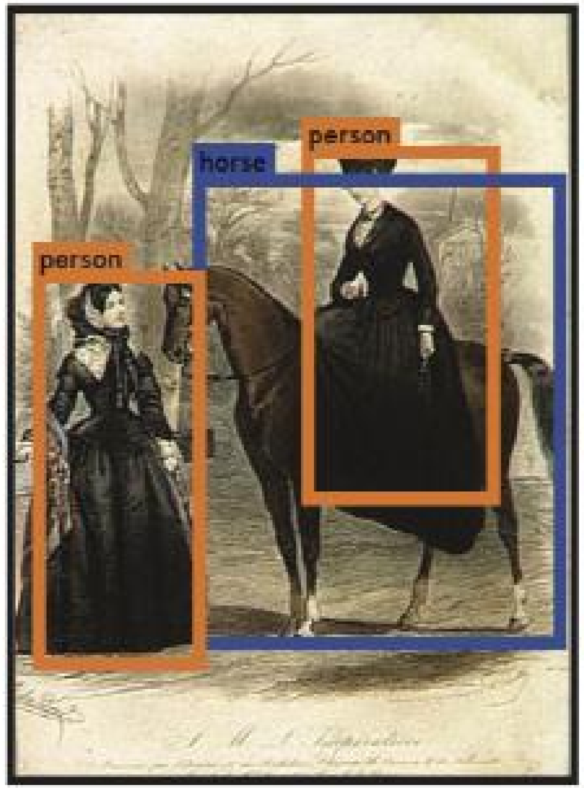
Step 2 (transfer learning):
- ➢ remove head layers
- ➢ add regression as new head
- ➢ fine-tune backbone & train head

Training tricks
- ➢ decaying learning rate
- ➢ batch normalization
- ➢ data augmentation

Performance



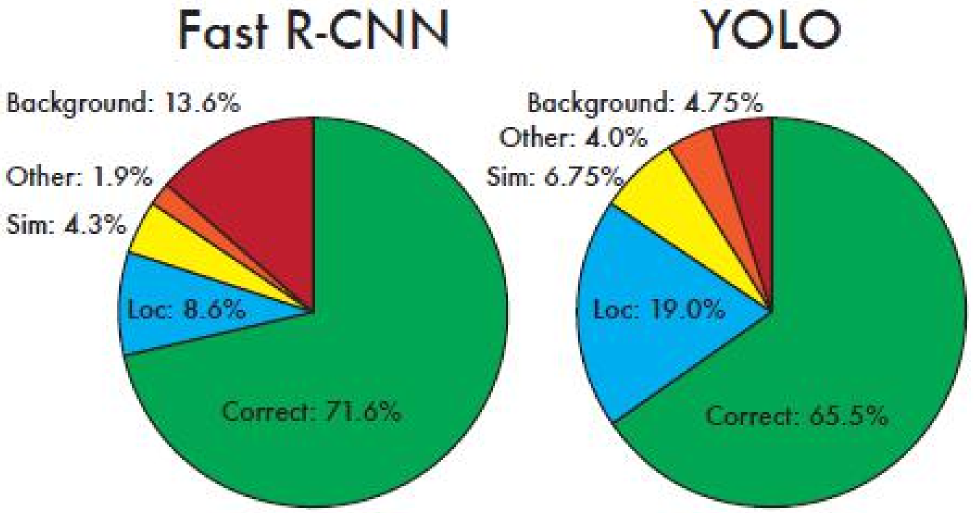
Generalizability



Picasso & People-Art dataset

But ... no free lunch

- ➢ YOLO is not as accurate as RCNN-series models
  - ➢ multi-task problem:
    YOLO wins in less background error,
    however, loses in localization error.



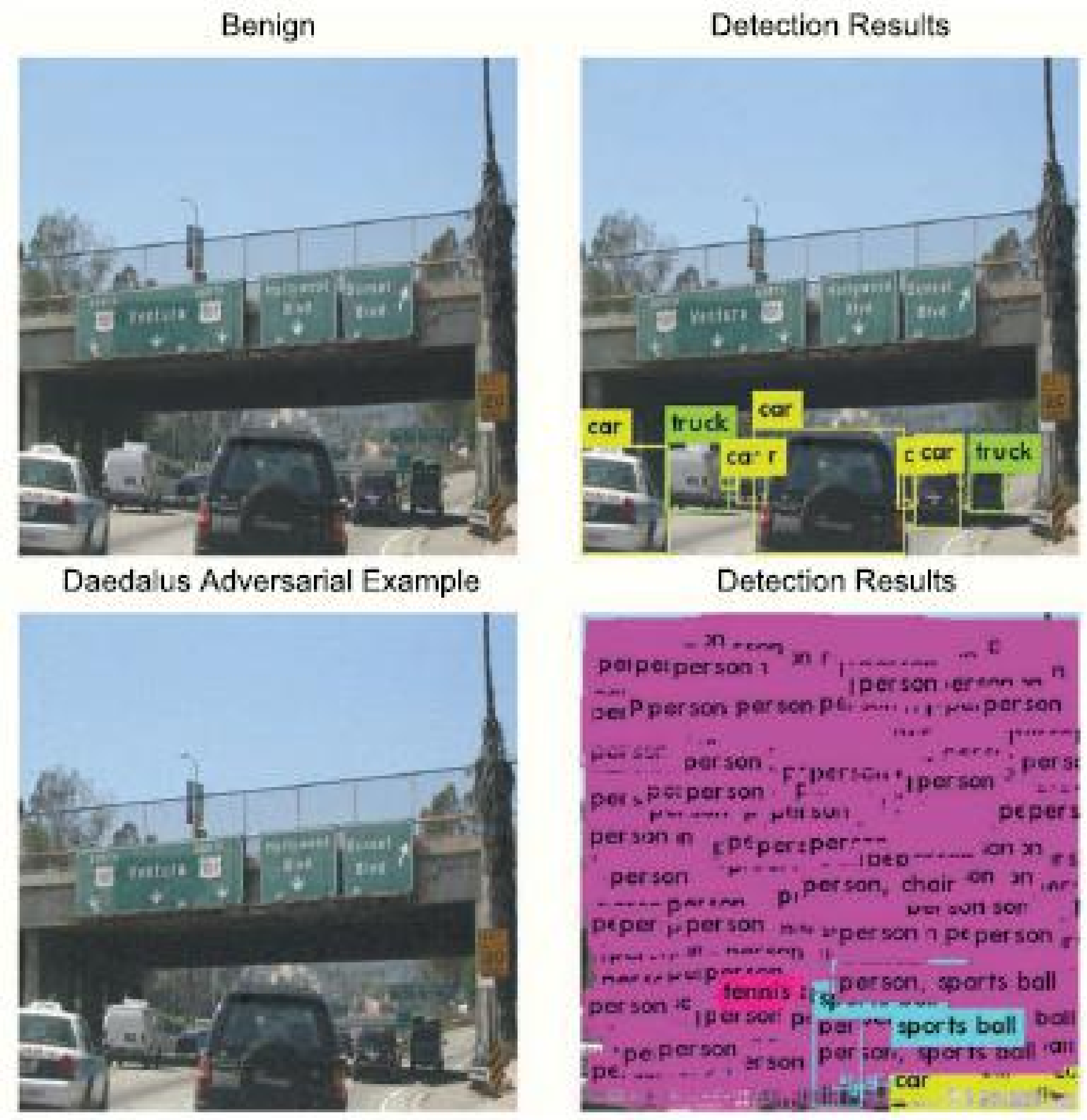
- ➢ YOLO is poor for detecting small objects
  - ➢ CNN: training on ImageNet may not generalize well for small objects (classification)
  - ➢ loss function equalizes location weights for small & large objects (localization)

50+ years

- ➢ YOLO is not good at crowd objects
  - ➢ non-maximal suppression. See an improvement: Adaptive NMS (arXiv:1904.03629)

- ➢ YOLO is bad when encountering strange aspect ratio
  - ➢ pre-defined anchors, or anchors learned from data. Go anchor-free (arXiv:1904.01355).

Security

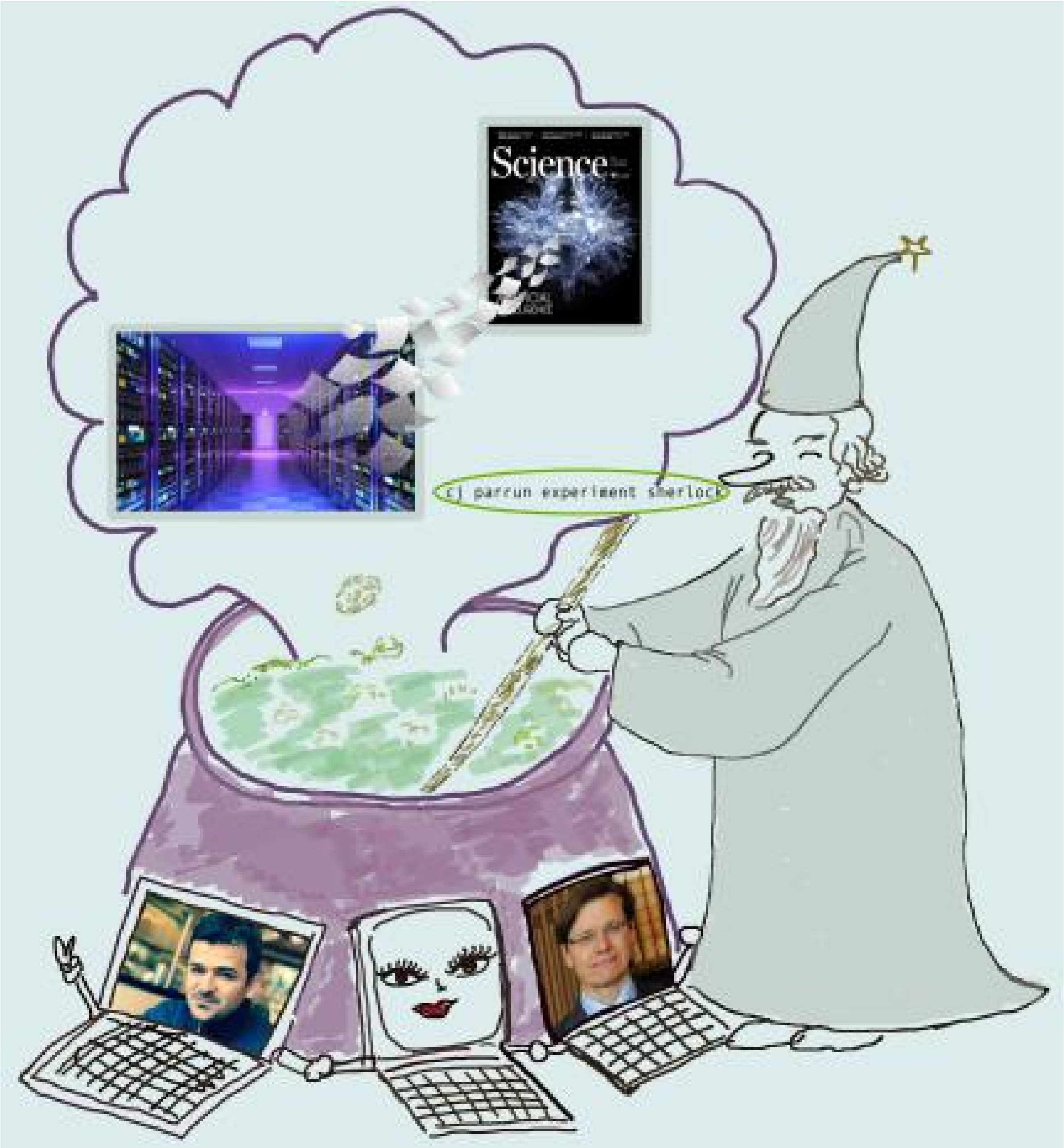


<u>CNN (classification) can be fooled, as well as YOLO, and the issues can be even worse.</u>

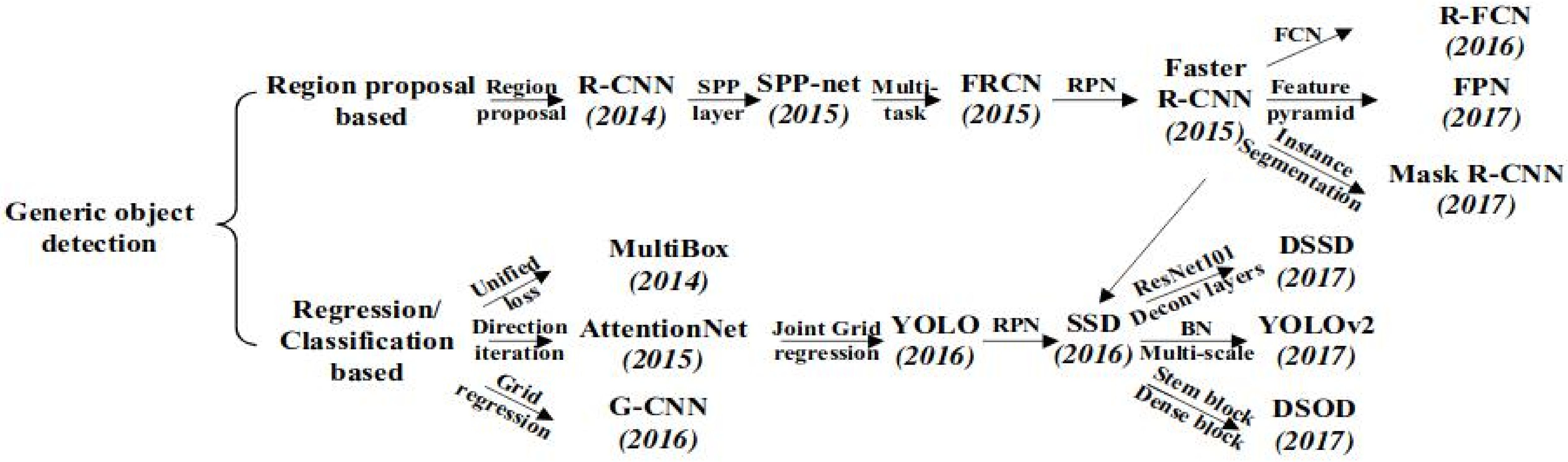
Non-maximal suppression is fooled.

Daedalus: Breaking Non-Maximum Suppression in Object Detection via Adversarial Examples. arXiv:1902.02067

Is there anything helpful to improve?



Darwin's evolution



arXiv: 1807.05511