

Object Detection and Segmentation

Meysam Madadi

So far: Image Classification



This image is [CC0 public domain](#)

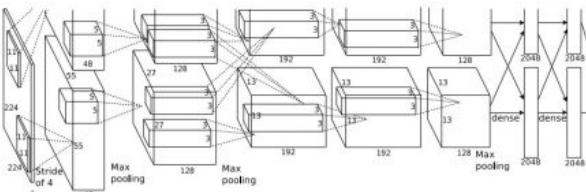


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Vector:
4096

Fully-Connected:
4096 to 1000

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

Other Computer Vision Tasks

Semantic Segmentation



GRASS, CAT,
TREE, SKY

No objects, just pixels

Classification + Localization



CAT

Single Object

Object Detection



DOG, DOG, CAT

Multiple Object

Instance Segmentation



DOG, DOG, CAT

[This image is CC0 public domain](#)

Semantic Segmentation



GRASS, CAT,
TREE, SKY

No objects, just pixels



CAT

Single Object



DOG, DOG, CAT

Multiple Object



DOG, DOG, CAT

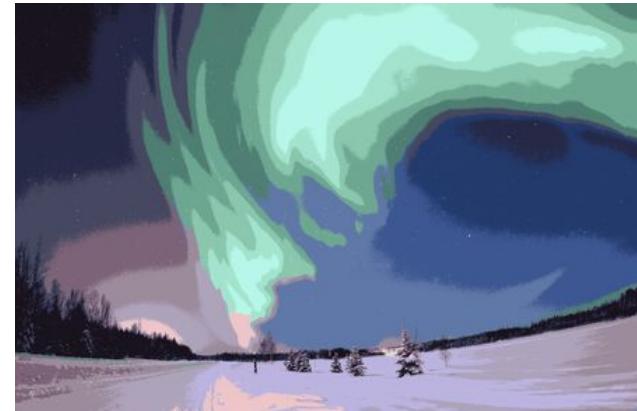
This image is CC0 public domain

Image segmentation

- Image segmentation is the process of **partitioning** an image into multiple **segments/super-pixels**, such that:
 - Every pixel is assigned a label such that pixels with the same label share certain characteristics like:
 - Color, intensity or texture,
 - Partitioning procedure does **not** attempt to understand what the segments represent.
- Possible solutions are:
 - Edge detection, color clustering/naming, graph partitioning, Markov Random Field, etc.

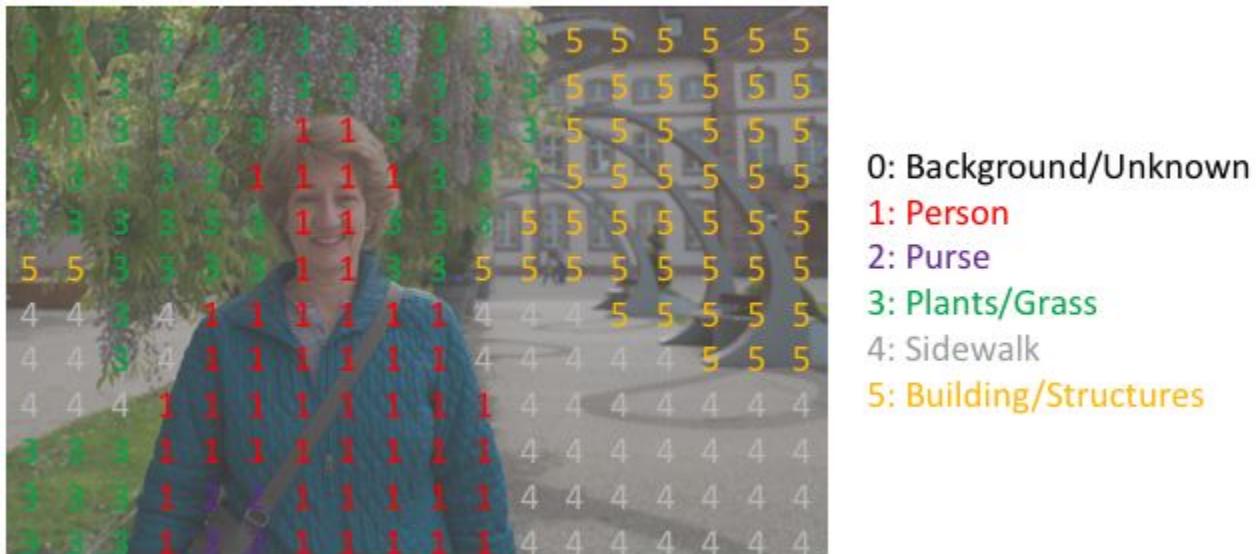


K-means



Semantic segmentation

- Semantic segmentation is the process of partitioning an image into a set of semantically meaningful segments,
 - Can be seen as a classification task where each pixel is classified into several pre-defined classes.



Representing the task



One-hot encoding

argmax

0: Background/Unknown

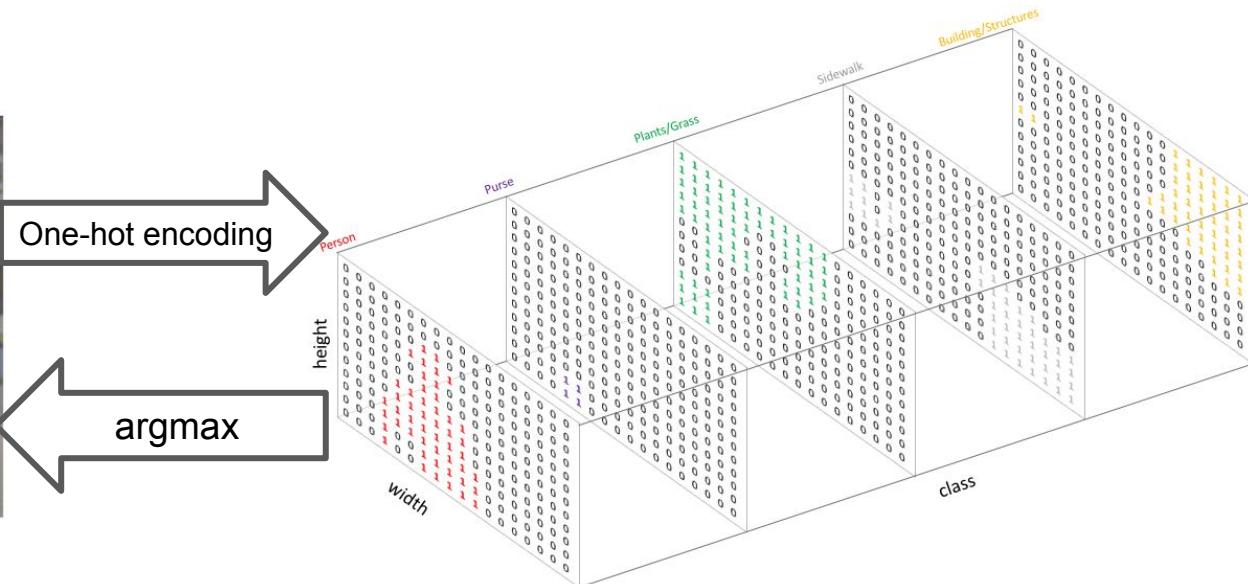
1: Person

2: Purse

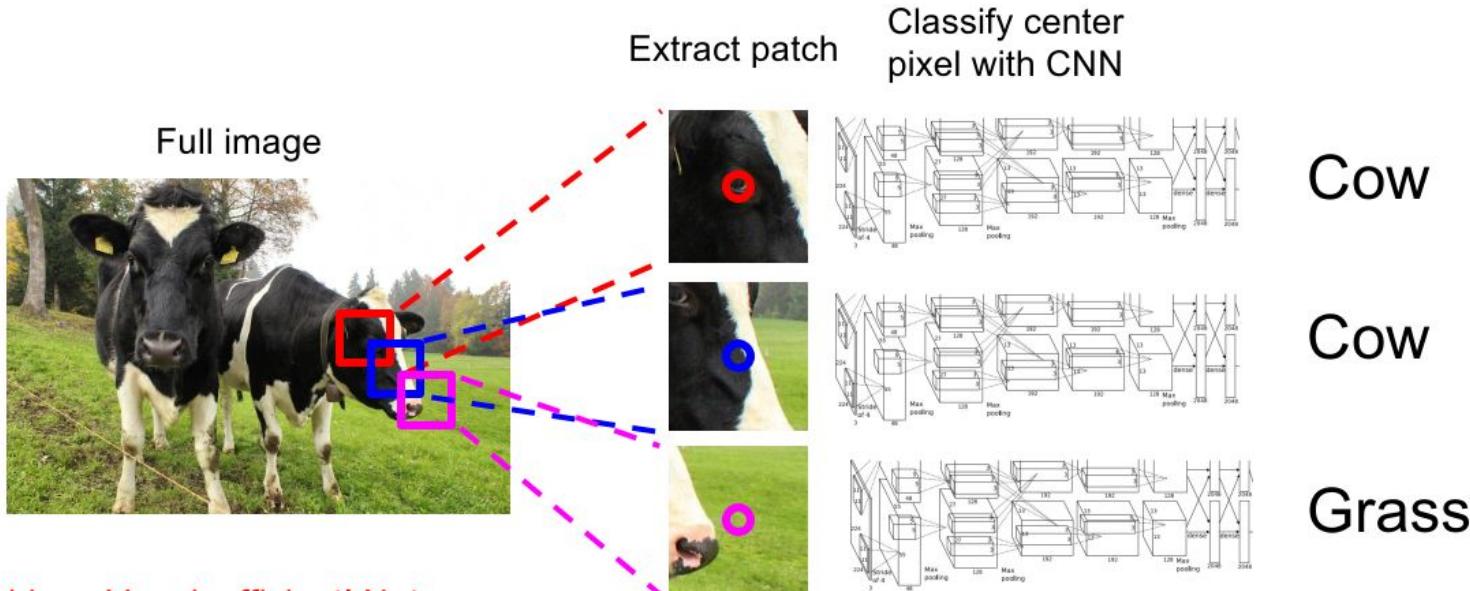
3: Plants/Grass

4: Sidewalk

5: Building/Structures



Semantic Segmentation Idea: Sliding Window



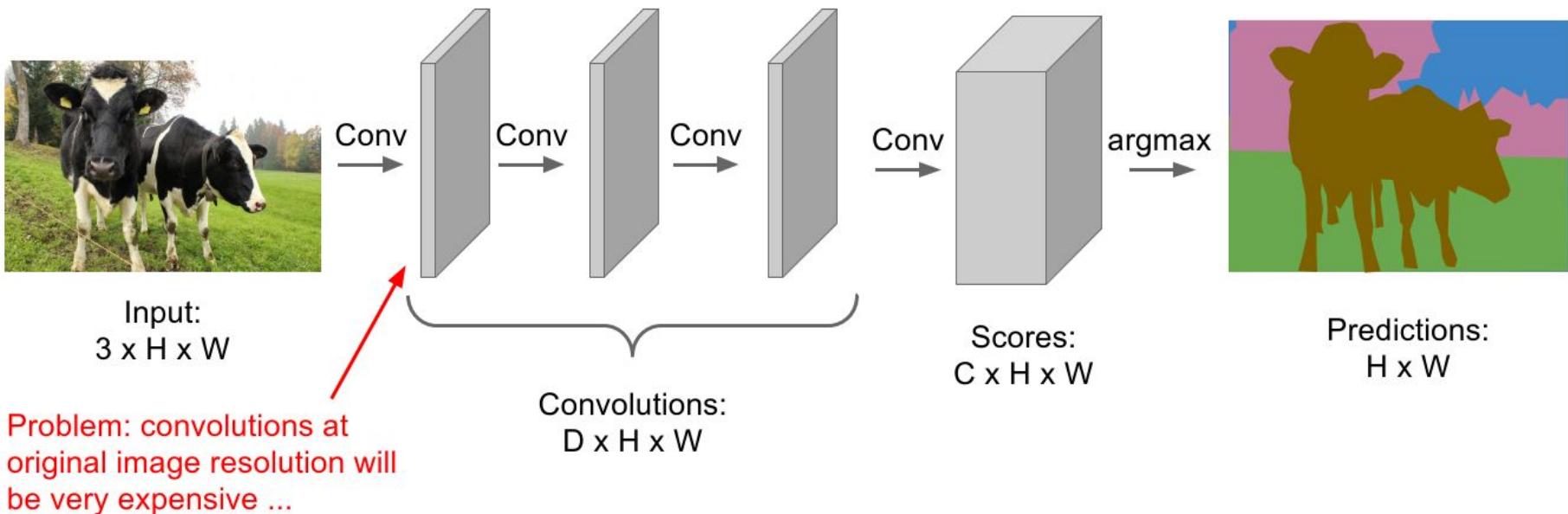
Problem: Very inefficient! Not reusing shared features between overlapping patches

Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013

Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014

Semantic Segmentation Idea: Fully Convolutional

Design a network as a bunch of convolutional layers
to make predictions for pixels all at once!



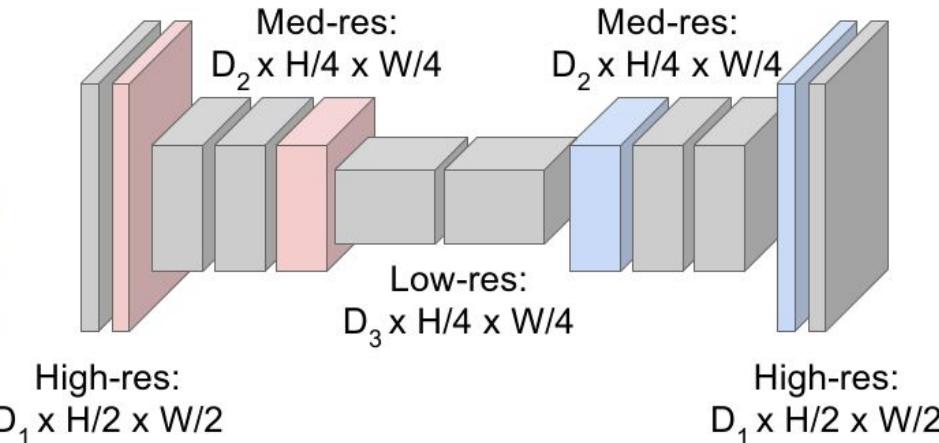
Semantic Segmentation Idea: Fully Convolutional

Downsampling:
Pooling, strided
convolution



Input:
 $3 \times H \times W$

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



Upsampling:
???



Predictions:
 $H \times W$

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

In-Network upsampling: “Unpooling”

Nearest Neighbor

1	2
3	4



1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Input: 2 x 2

Output: 4 x 4

“Bed of Nails”

1	2
3	4



1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Input: 2 x 2

Output: 4 x 4

In-Network upsampling: “Max Unpooling”

Max Pooling

Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4

5	6
7	8

Output: 2 x 2

Max Unpooling

Use positions from pooling layer

1	2
3	4

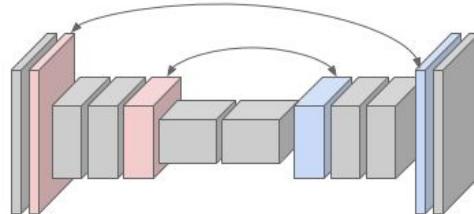
Rest of the network

Input: 2 x 2

0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

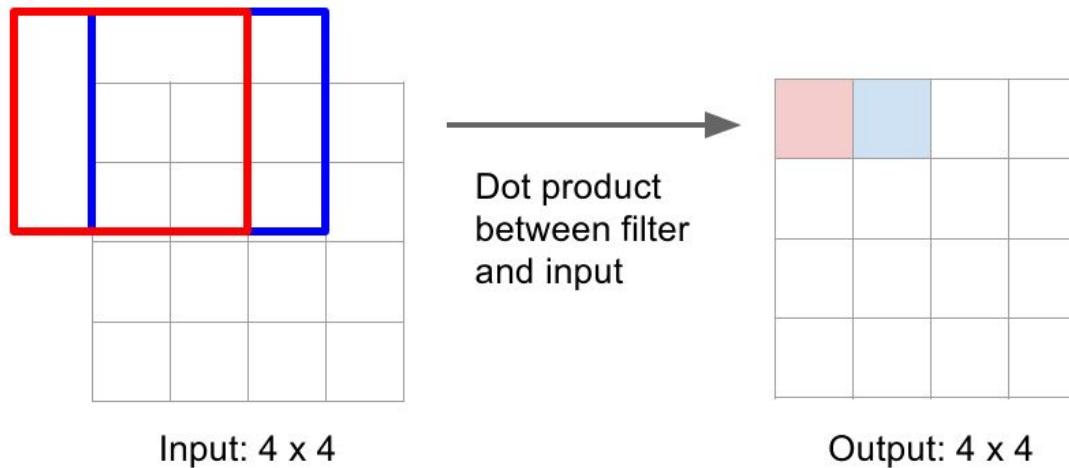
Output: 4 x 4

Corresponding pairs of
downsampling and
upsampling layers



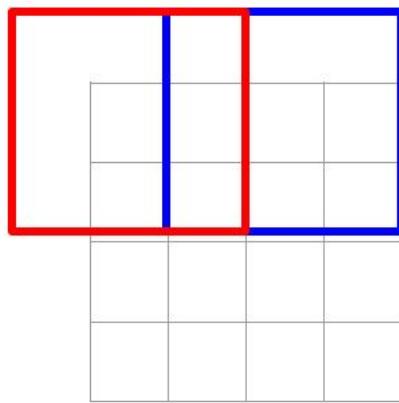
Learnable Upsampling: Transpose Convolution

Recall: Normal 3×3 convolution, stride 1 pad 1

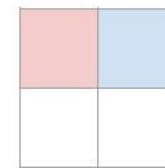


Learnable Upsampling: Transpose Convolution

Recall: Normal 3×3 convolution, stride 2 pad 1



Dot product
between filter
and input



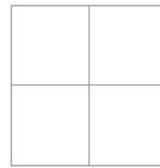
Output: 2×2

Filter moves 2 pixels in
the input for every one
pixel in the output

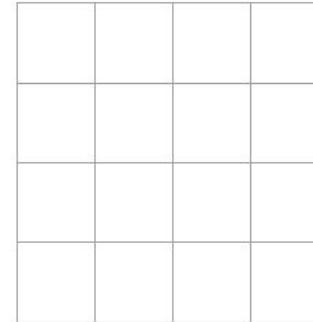
Stride gives ratio between
movement in input and
output

Learnable Upsampling: Transpose Convolution

3×3 **transpose** convolution, stride 2 pad 1



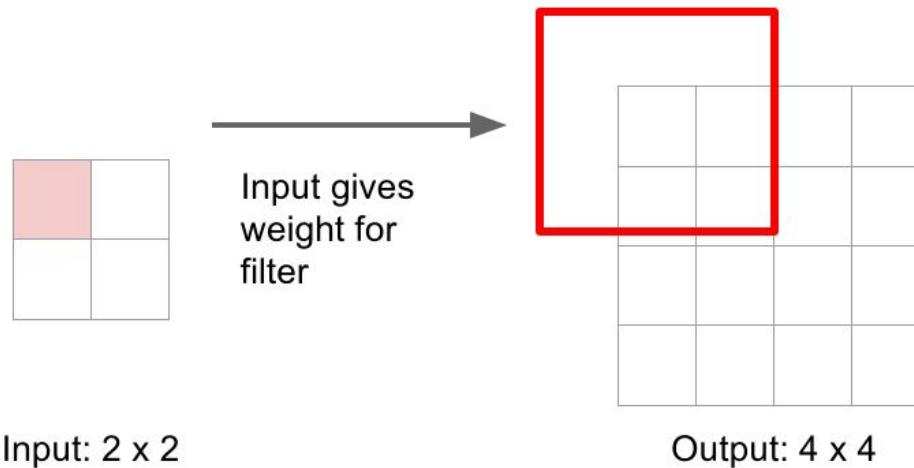
Input: 2×2



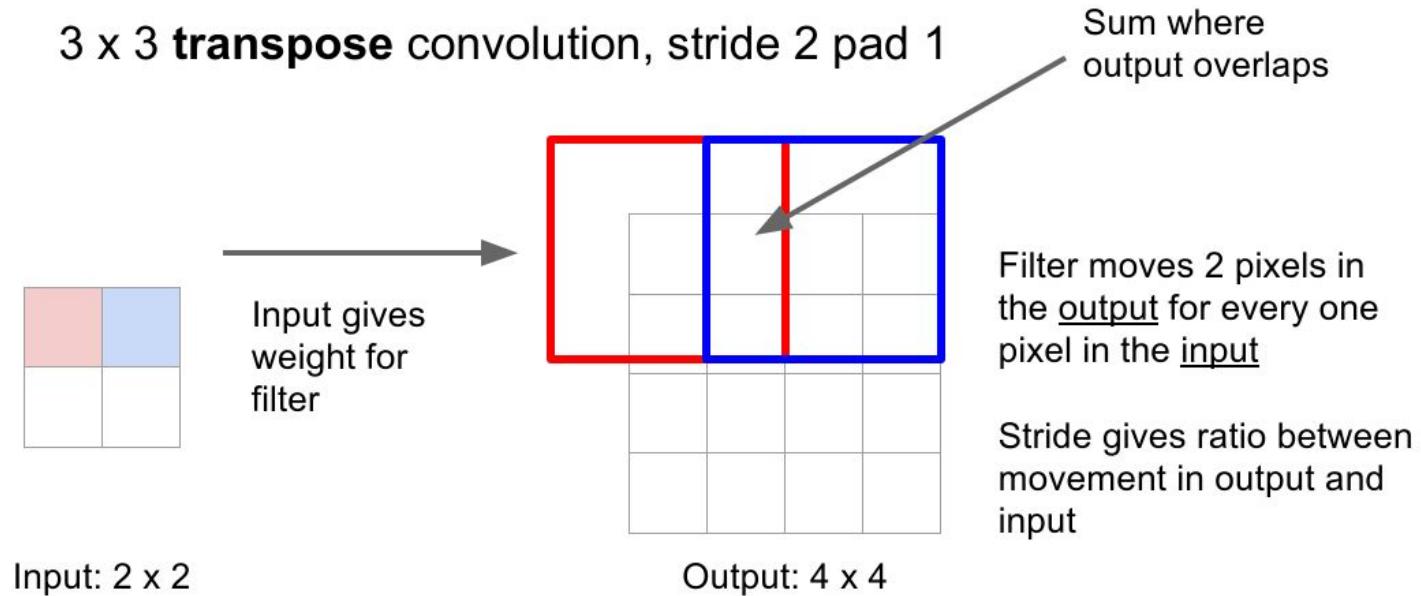
Output: 4×4

Learnable Upsampling: Transpose Convolution

3 x 3 **transpose** convolution, stride 2 pad 1



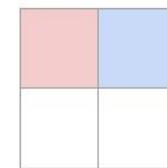
Learnable Upsampling: Transpose Convolution



Learnable Upsampling: Transpose Convolution

Other names:

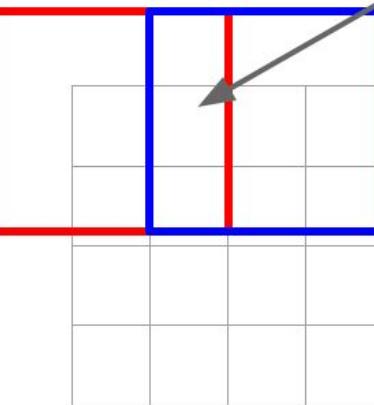
- Deconvolution (bad)
- Upconvolution
- Fractionally strided convolution
- Backward strided convolution



Input: 2 x 2

3 x 3 **transpose** convolution, stride 2 pad 1

Input gives weight for filter



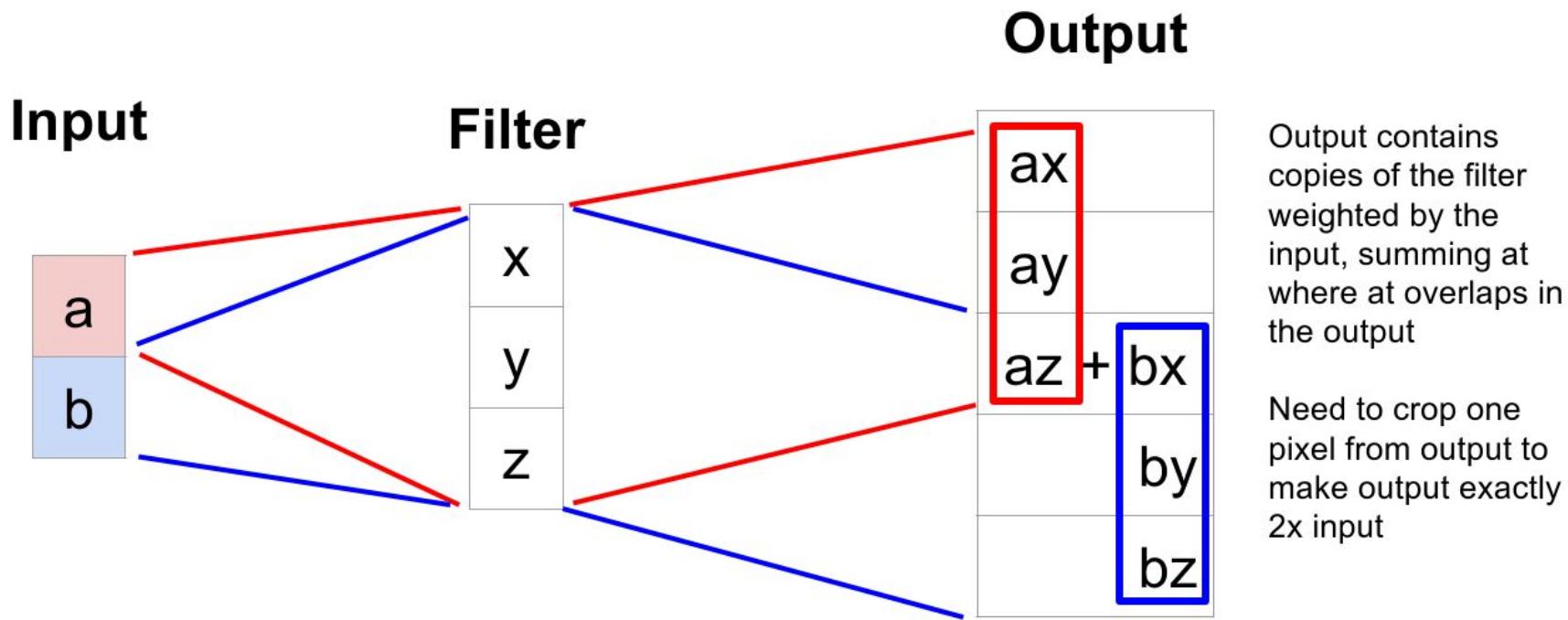
Output: 4 x 4

Filter moves 2 pixels in the output for every one pixel in the input

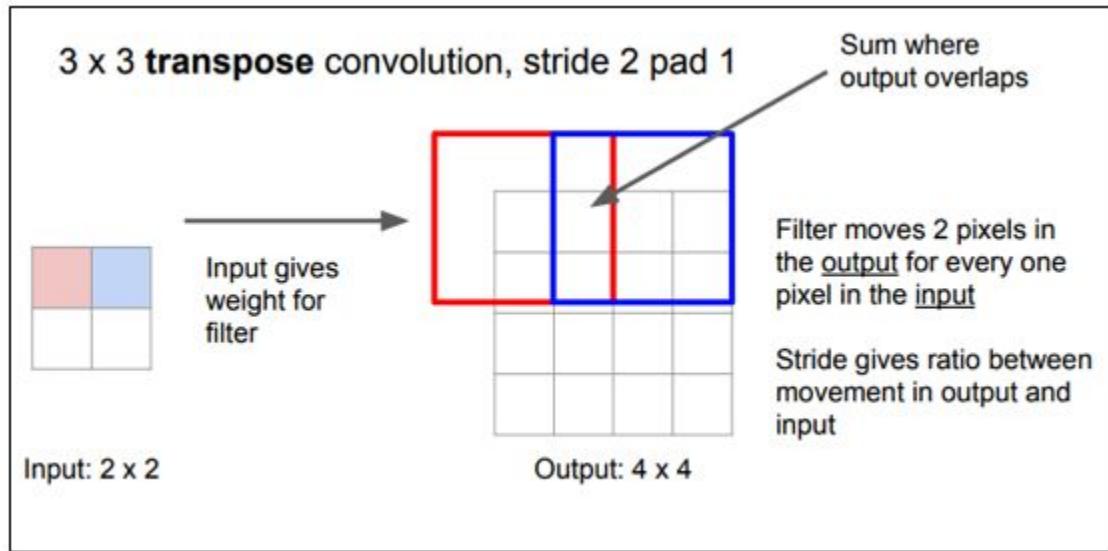
Stride gives ratio between movement in output and input

Sum where output overlaps

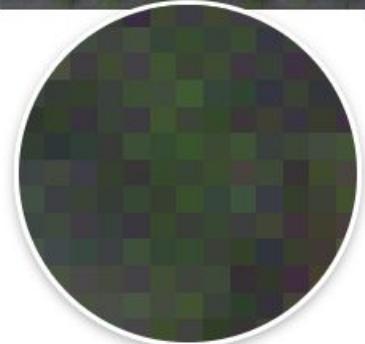
Transpose Convolution: 1D Example



Upsampling - transpose convolutions



Downside: Checkerboard artifact due to overlapped windows



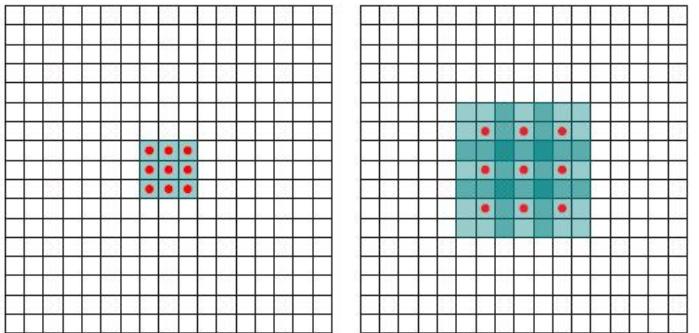
Dilated convolution

Red=Conv kernel, Green=receptive field

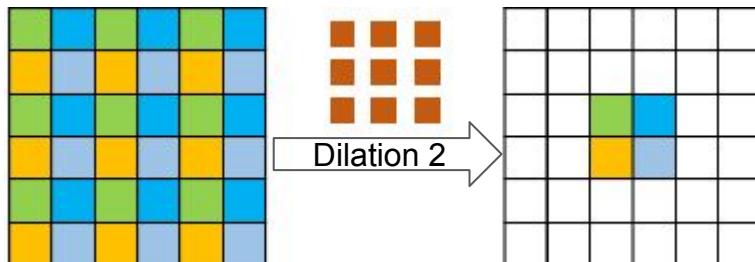
Normal Conv=Dilation 1

Dilation 2

Dilation 4



Upside: Supports exponential receptive field expansion without loss of resolution or coverage.



Downside: Has gridding artifacts.

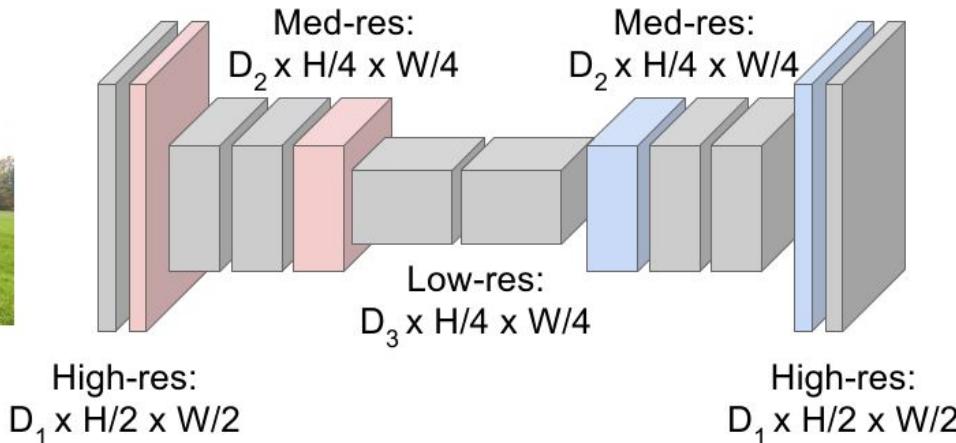
Semantic Segmentation Idea: Fully Convolutional

Downsampling:
Pooling, strided convolution



Input:
 $3 \times H \times W$

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



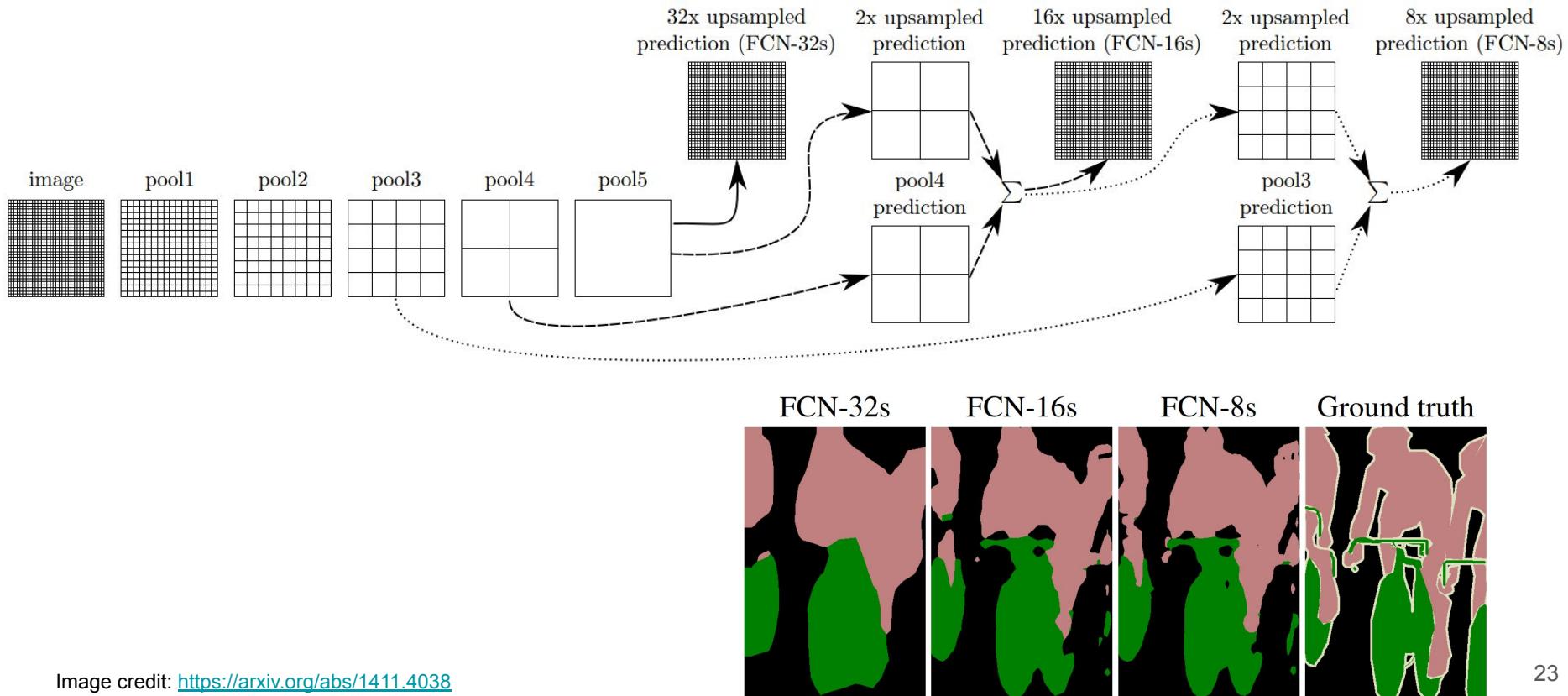
Upsampling:
Unpooling or strided transpose convolution



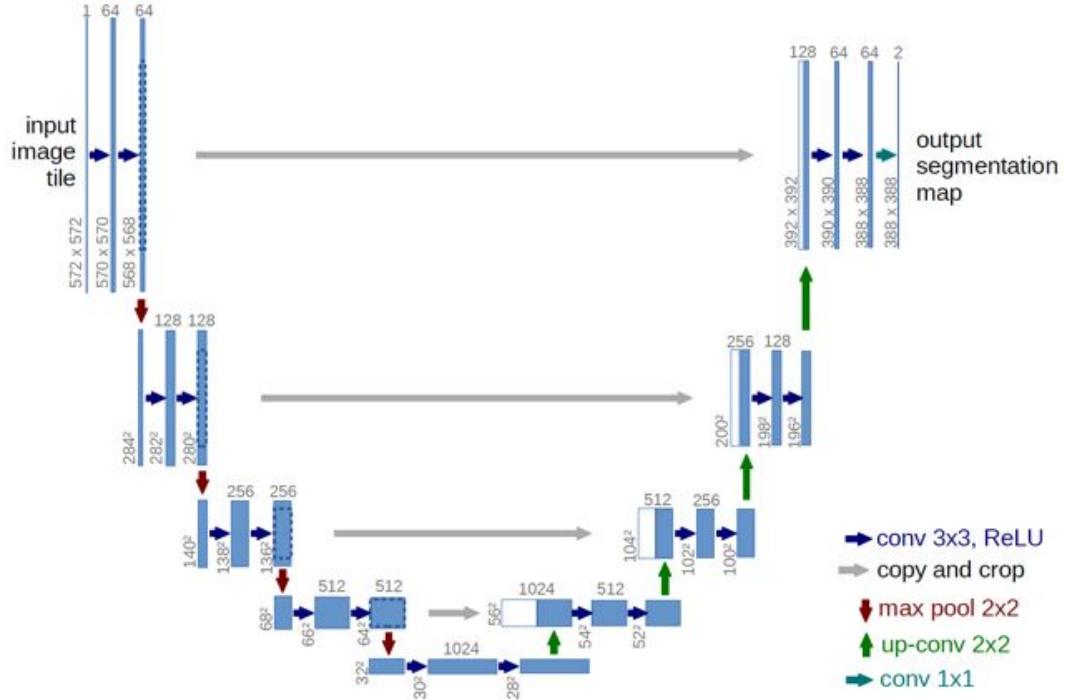
Predictions:
 $H \times W$

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

Skip connections ==> more fine grain details

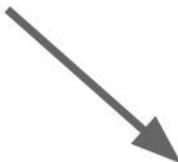


U-Net: deeper decoder with skip connection



- Simple to understand,
- Popular in many domains,
- Efficient on small dataset by the usage of data augmentation,
- Invariant to image scale due to fully convolutional layers,
- Decoder has more features than encoder.

Classification + Localization



GRASS, CAT,
TREE, SKY

No objects, just pixels



CAT

Single Object



DOG, DOG, CAT

Multiple Object



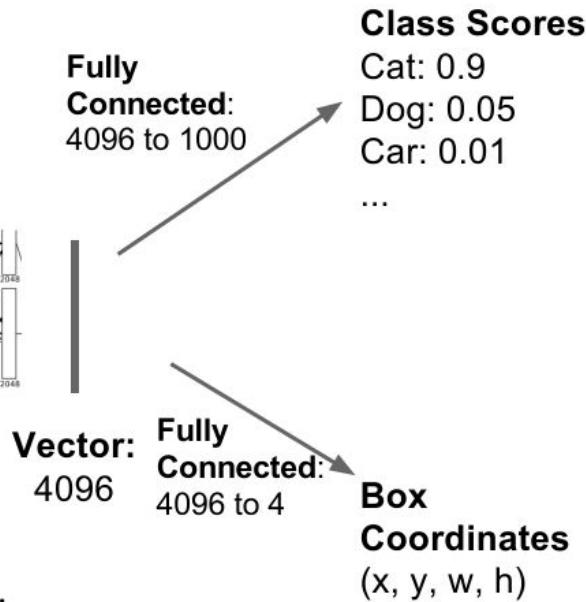
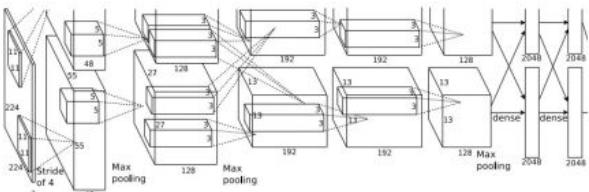
DOG, DOG, CAT

This image is CC0 public domain

Classification + Localization



This image is CC0 public domain

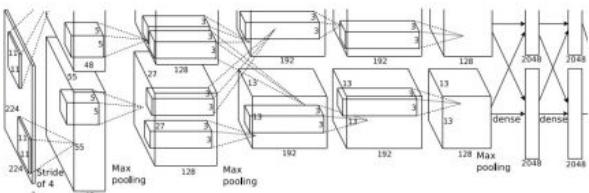


Treat localization as a regression problem!

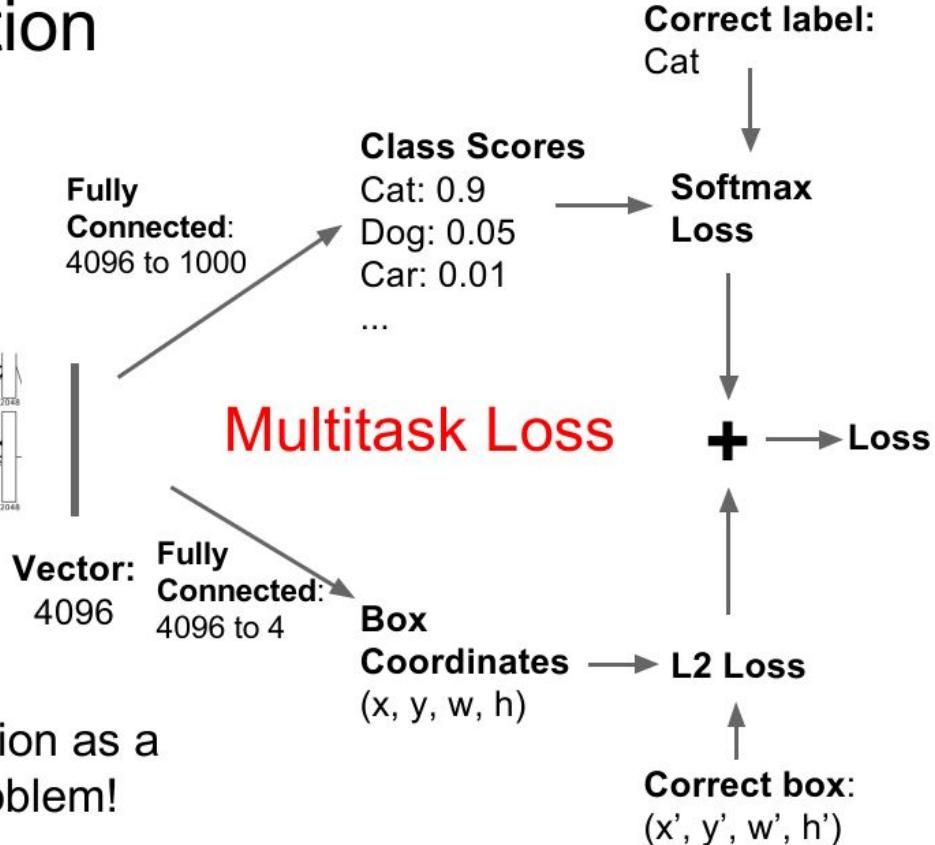
Classification + Localization



This image is CC0 public domain



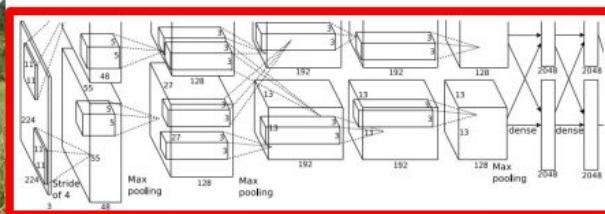
Treat localization as a regression problem!



Classification + Localization

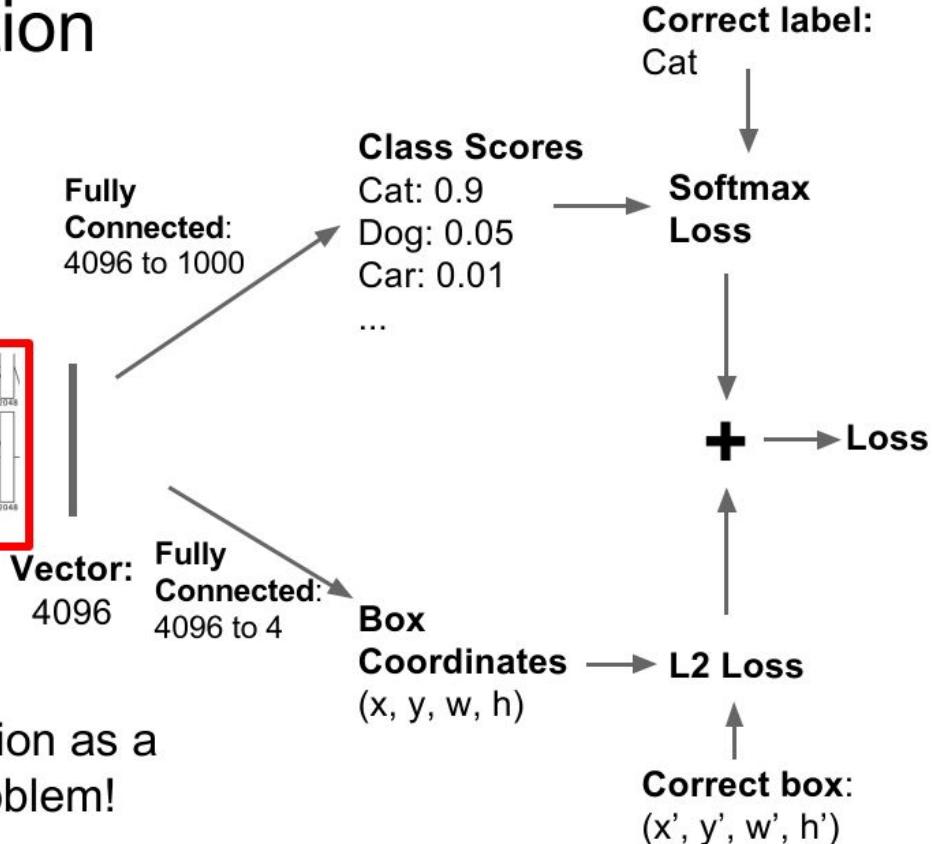


This image is CC0 public domain



Often pretrained on ImageNet
(Transfer learning)

Treat localization as a
regression problem!



Aside: Human Pose Estimation



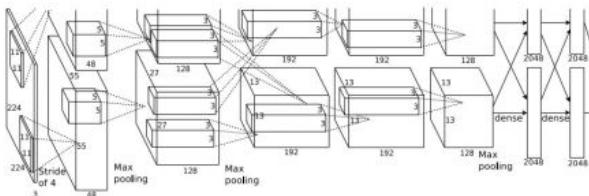
Represent pose as a set of 14 joint positions:

- Left / right foot
- Left / right knee
- Left / right hip
- Left / right shoulder
- Left / right elbow
- Left / right hand
- Neck
- Head top

This image is licensed under CC-BY 2.0.

Johnson and Everingham, "Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation", BMVC 2010

Aside: Human Pose Estimation



Vector:

4096

Left foot: (x, y)

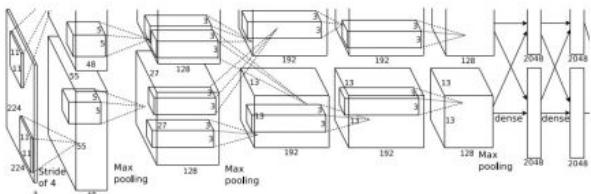
Right foot: (x, y)

...

Head top: (x, y)

Toshev and Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks", CVPR 2014

Aside: Human Pose Estimation



Vector:
4096

Correct left
foot: (x', y')

Left foot: (x, y) \rightarrow L2 loss

Right foot: (x, y) \rightarrow L2 loss

...

Head top: (x, y) \rightarrow L2 loss

Correct head
top: (x', y')

+ \rightarrow **Loss**

Toshev and Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks", CVPR 2014

Object Detection



GRASS, CAT,
TREE, SKY

No objects, just pixels



CAT

Single Object



DOG, DOG, CAT

Multiple Object



DOG, DOG, CAT

This image is CC0 public domain

Object Detection: Impact of Deep Learning

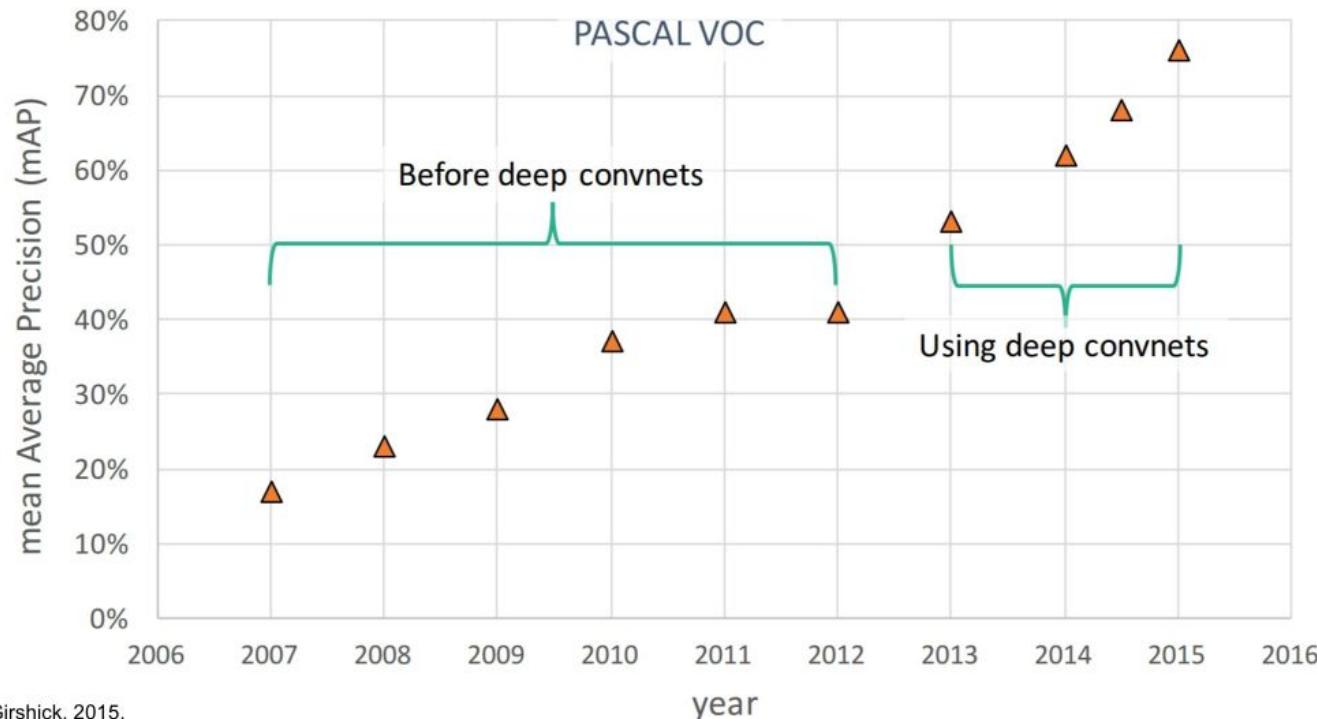
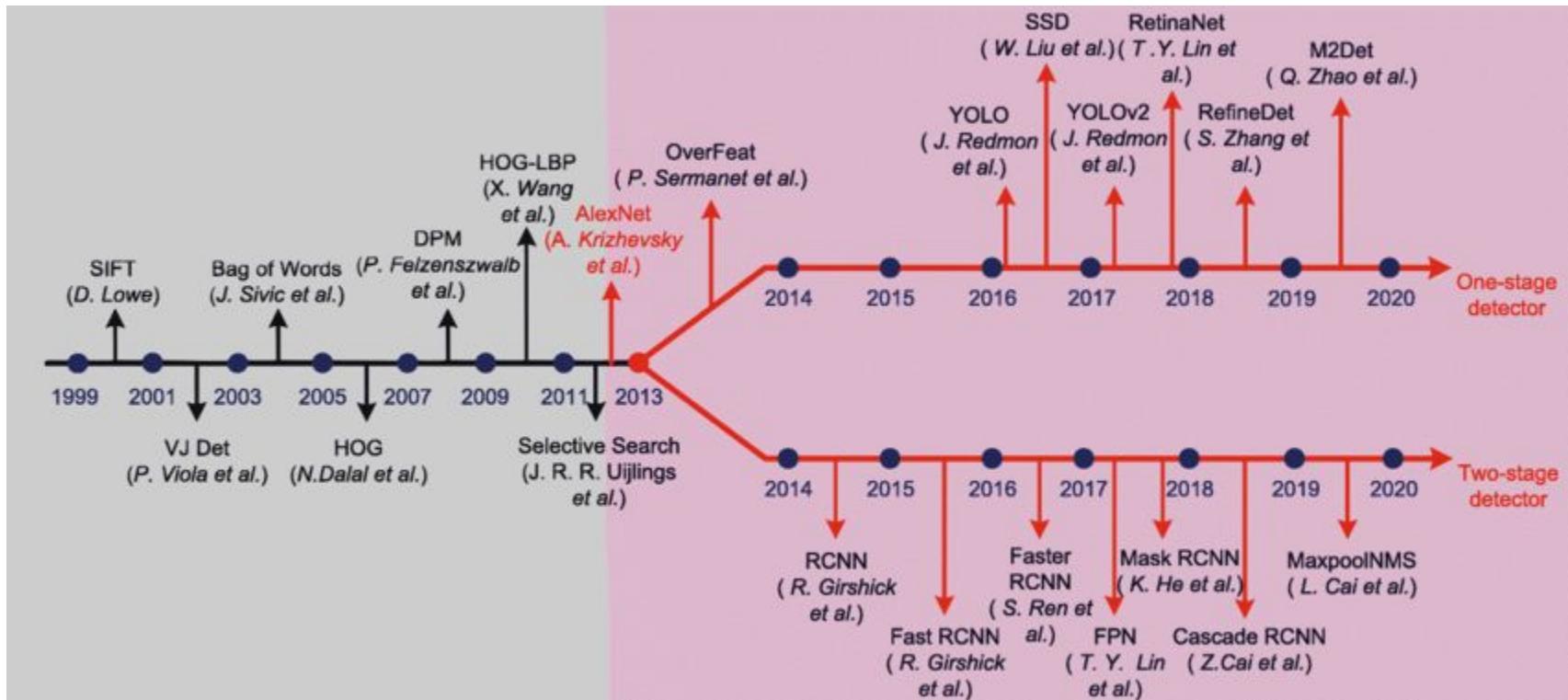
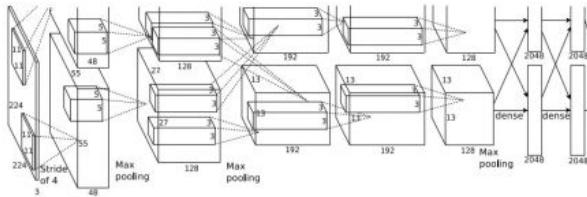


Figure copyright Ross Girshick, 2015.
Reproduced with permission.

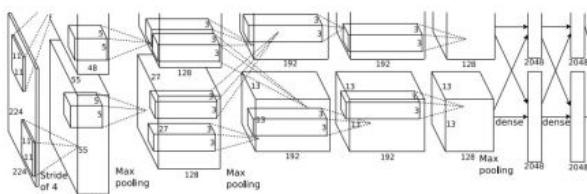
Evolution of object detection



Object Detection as Regression?



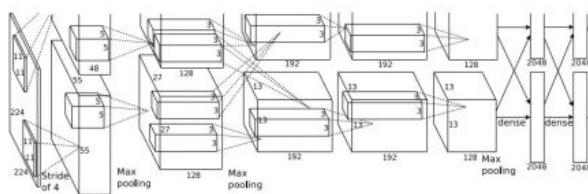
CAT: (x, y, w, h)



DOG: (x, y, w, h)

DOG: (x, y, w, h)

CAT: (x, y, w, h)



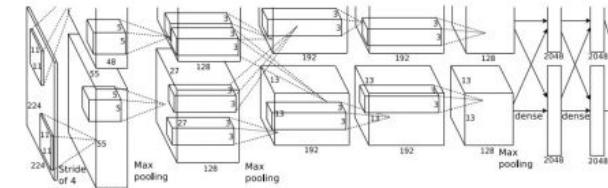
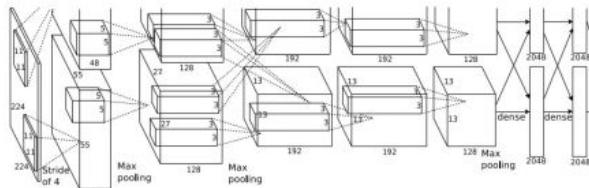
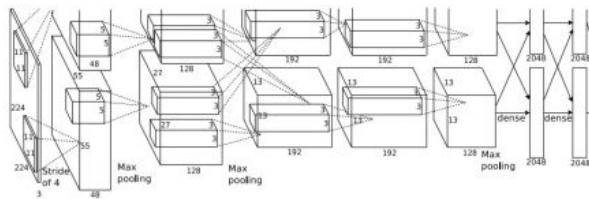
DUCK: (x, y, w, h)

DUCK: (x, y, w, h)

...

Object Detection as Regression?

Each image needs a different number of outputs!



CAT: (x, y, w, h) 4 numbers

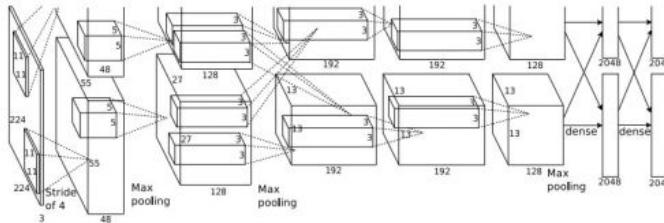
DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)

DUCK: (x, y, w, h) Many numbers!
DUCK: (x, y, w, h)

...

Object Detection as Classification: Sliding Window

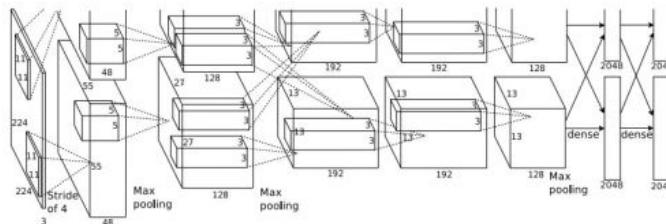
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? NO
Background? YES

Object Detection as Classification: Sliding Window

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

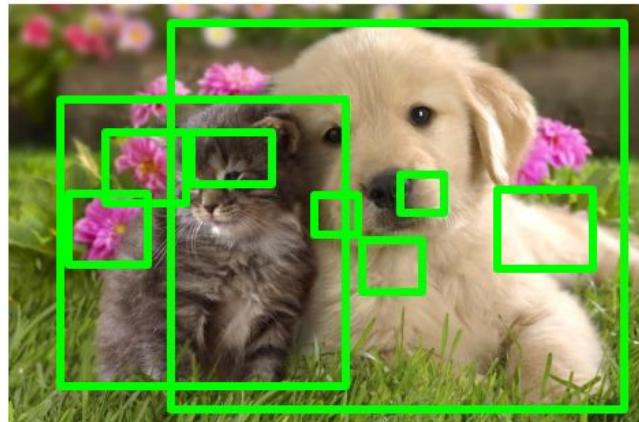


Dog? NO
Cat? YES
Background? NO

Problem: Need to apply CNN to huge number of locations and scales, very computationally expensive!

Region Proposals

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



Alexe et al, "Measuring the objectness of image windows", TPAMI 2012

Uijlings et al, "Selective Search for Object Recognition", IJCV 2013

Cheng et al, "BING: Binarized normed gradients for objectness estimation at 300fps", CVPR 2014

Zitnick and Dollar, "Edge boxes: Locating object proposals from edges", ECCV 2014

R-CNN



Input image

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

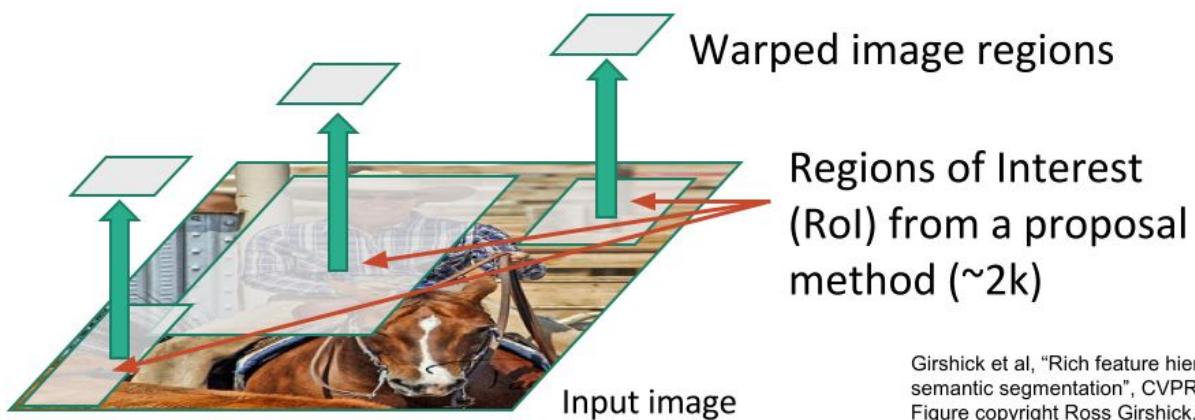
R-CNN



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

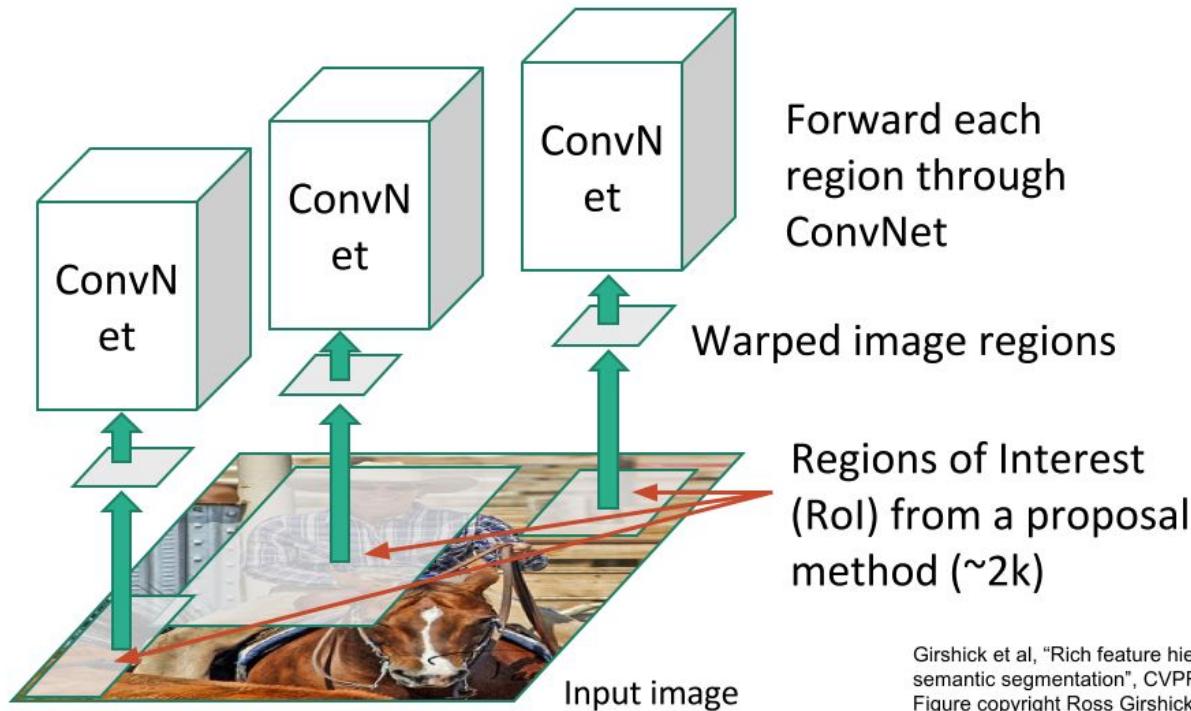
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

R-CNN



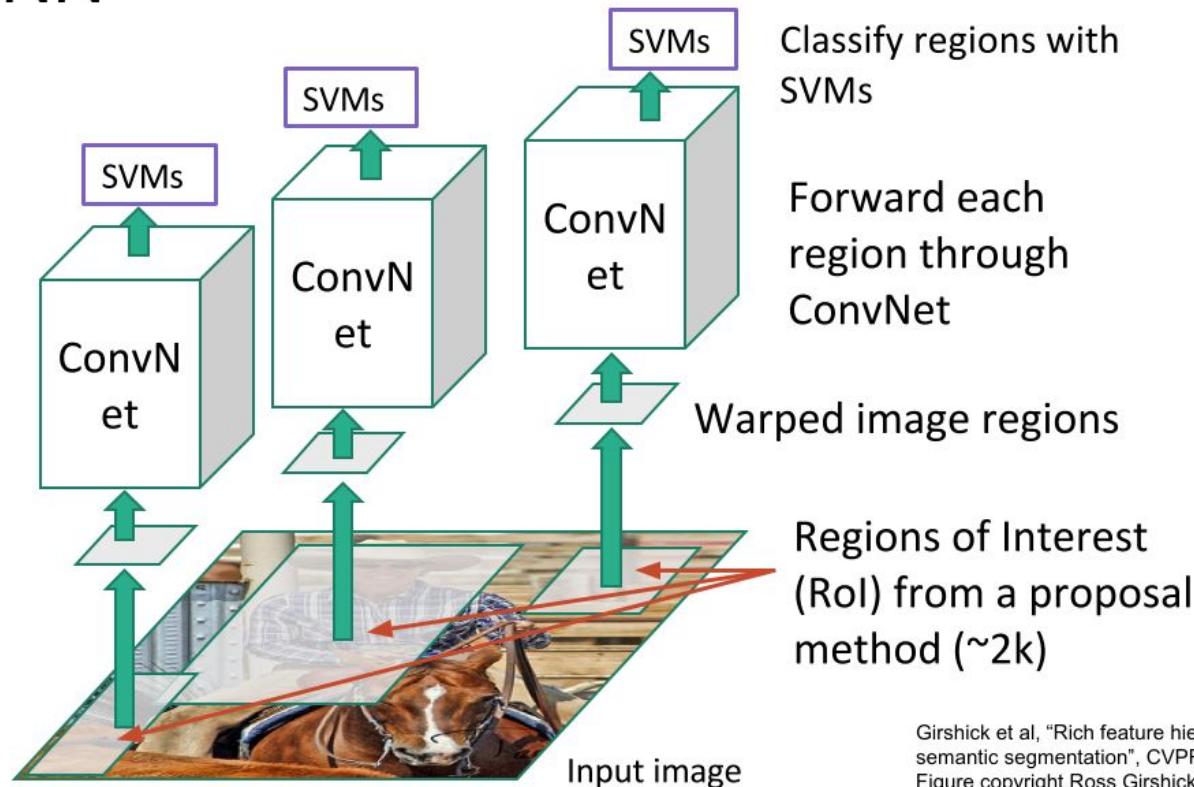
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

R-CNN



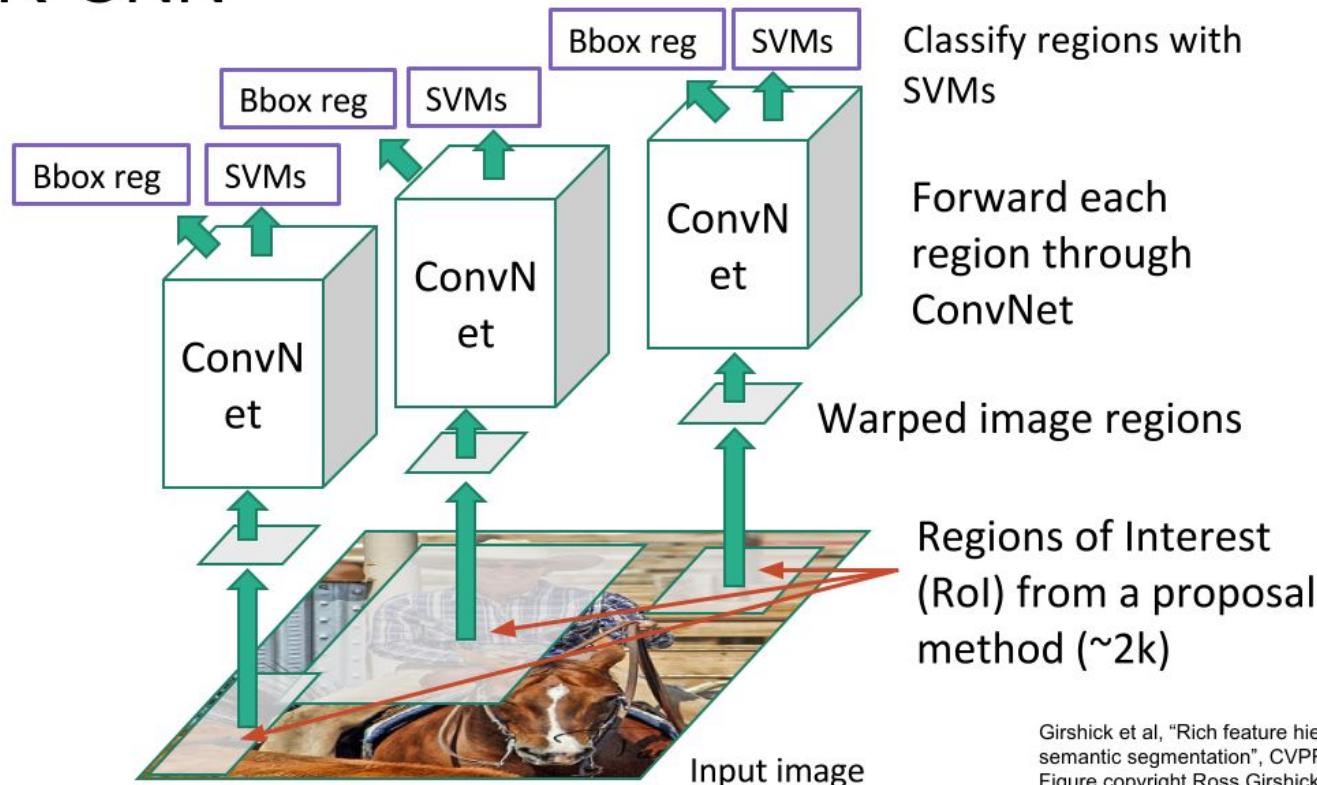
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

R-CNN



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

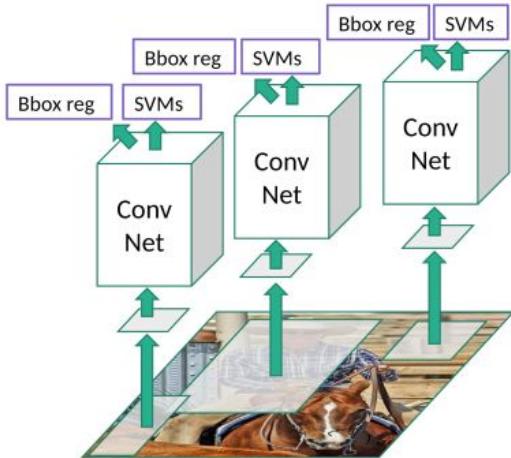
R-CNN



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: Problems

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



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

Slide copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

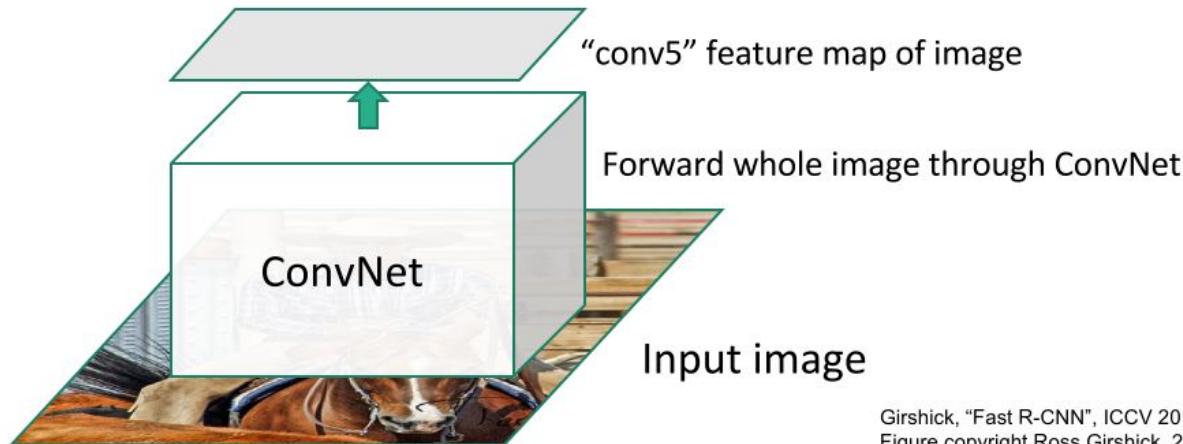
Fast R-CNN



Input image

Girshick, "Fast R-CNN", ICCV 2015.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

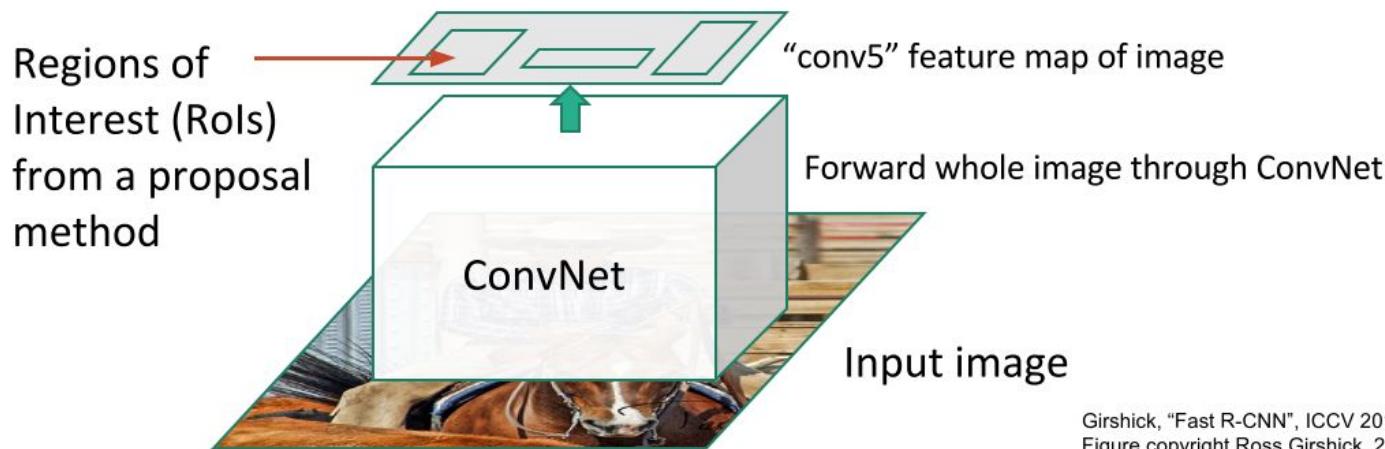
Fast R-CNN



Girshick, "Fast R-CNN", ICCV 2015.

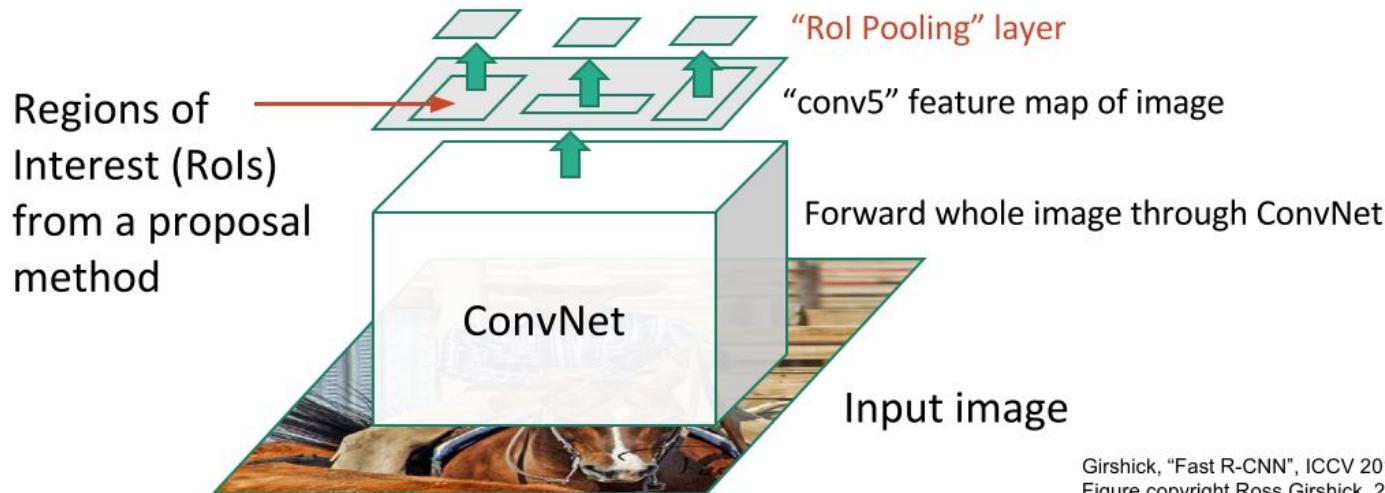
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

Fast R-CNN



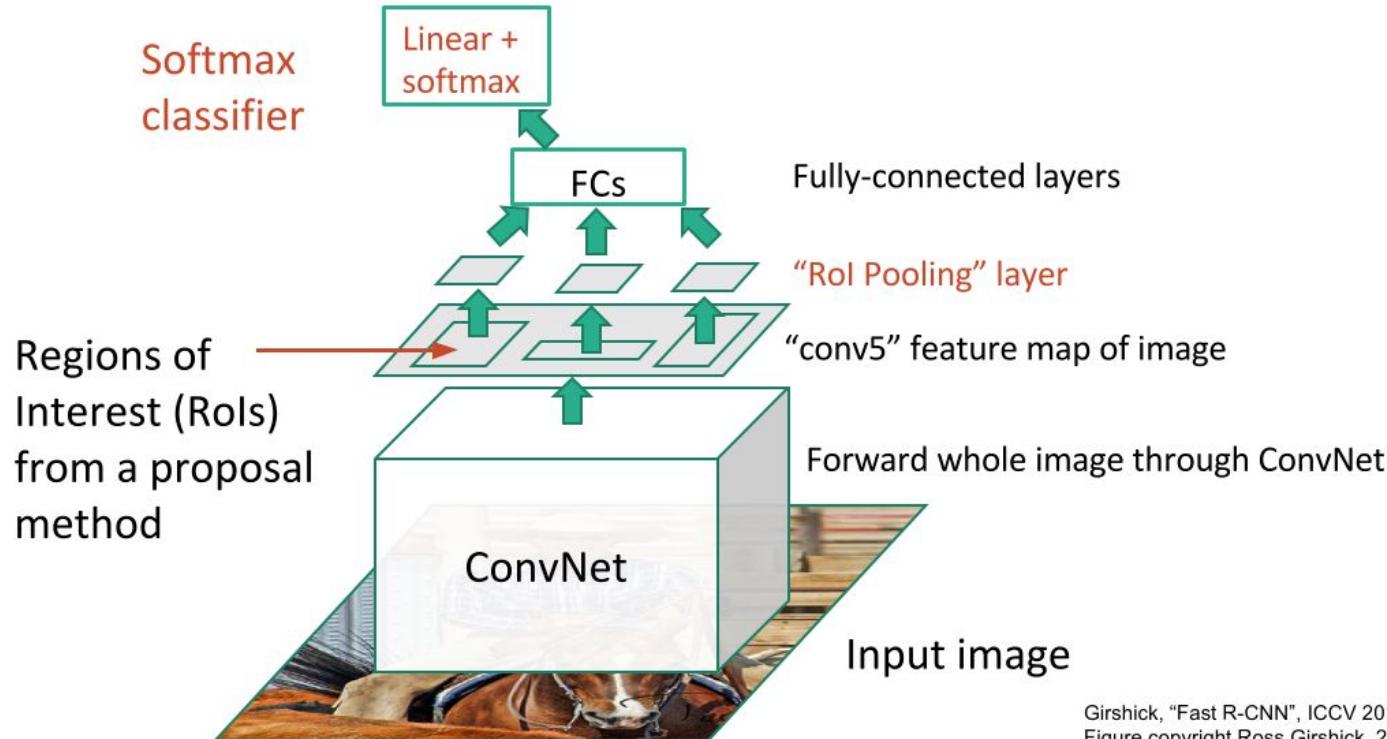
Girshick, “Fast R-CNN”, ICCV 2015.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

Fast R-CNN



Girshick, "Fast R-CNN", ICCV 2015.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

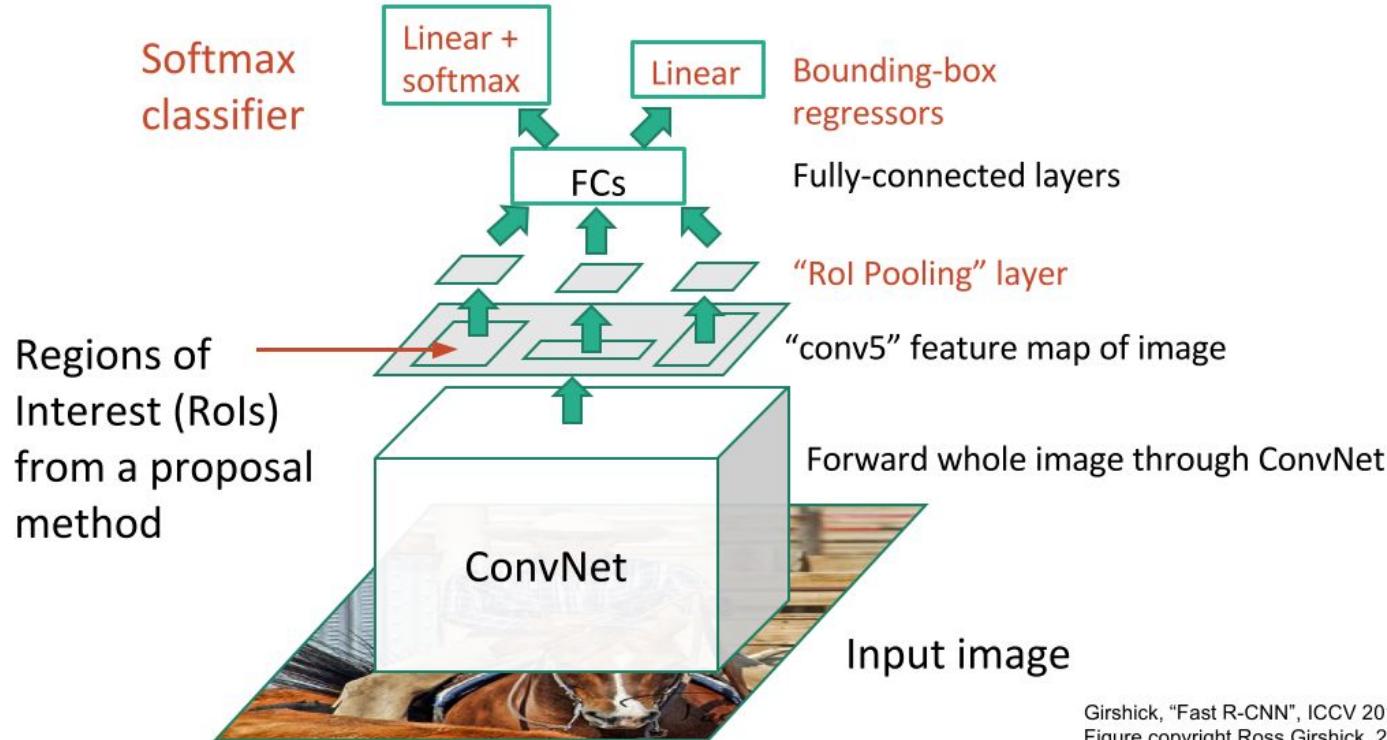
Fast R-CNN



Girshick, "Fast R-CNN", ICCV 2015.

Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

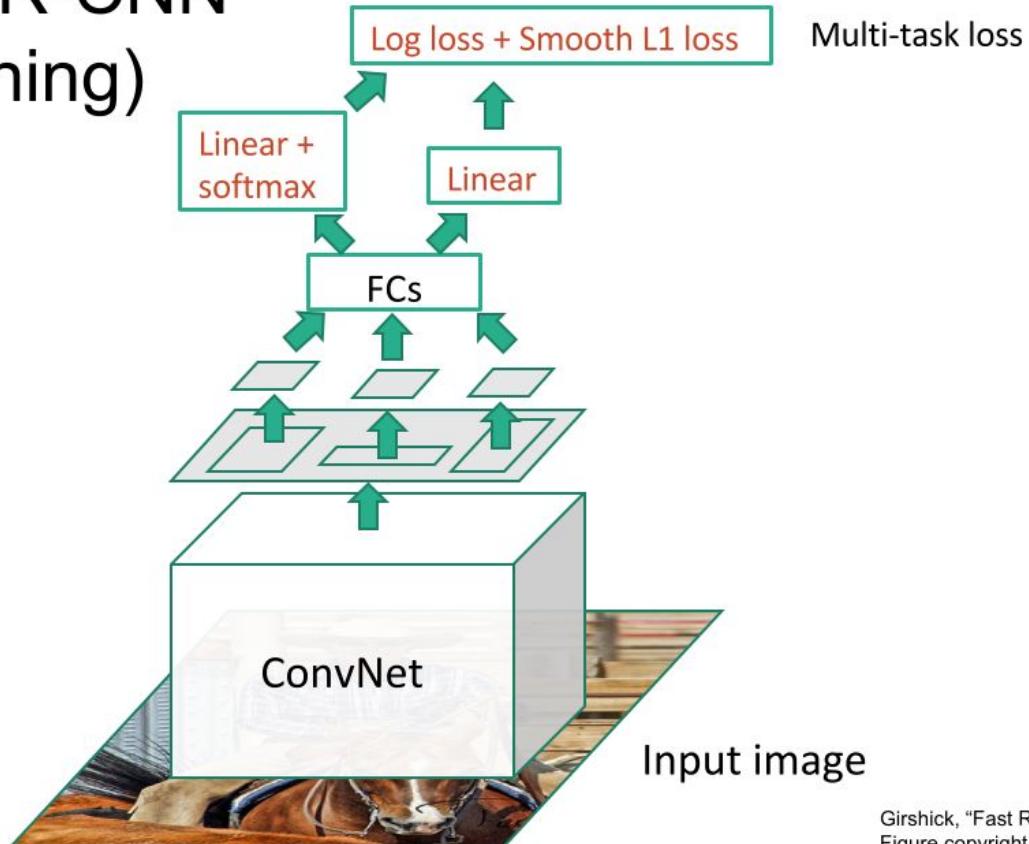
Fast R-CNN



Girshick, "Fast R-CNN", ICCV 2015.

Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

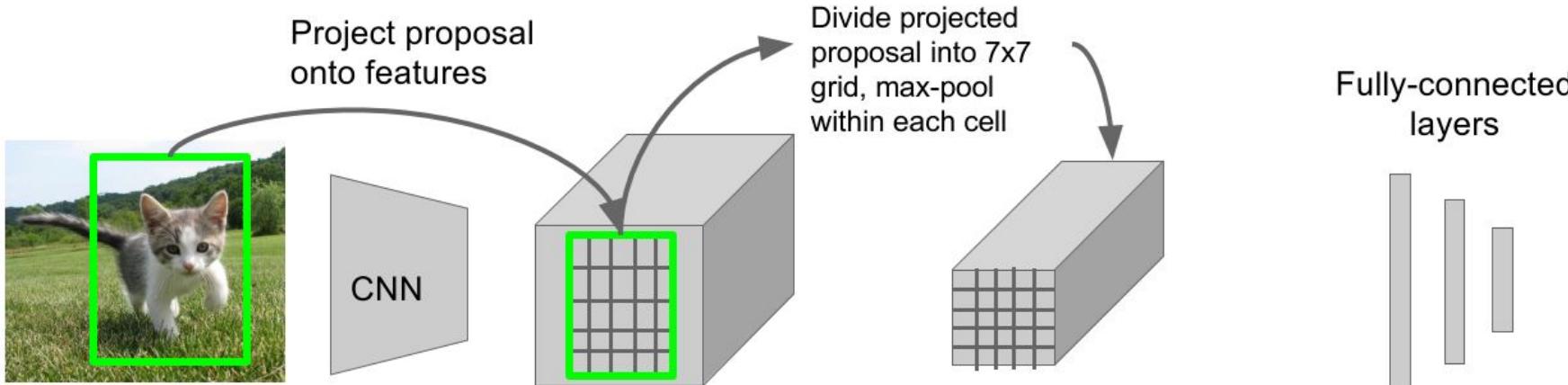
Fast R-CNN (Training)



Girshick, "Fast R-CNN", ICCV 2015.

Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

Fast R-CNN: RoI Pooling



Hi-res input image:
 $3 \times 640 \times 480$
with region
proposal

Hi-res conv features:
 $512 \times 20 \times 15$;

Projected region
proposal is e.g.
 $512 \times 18 \times 8$
(varies per proposal)

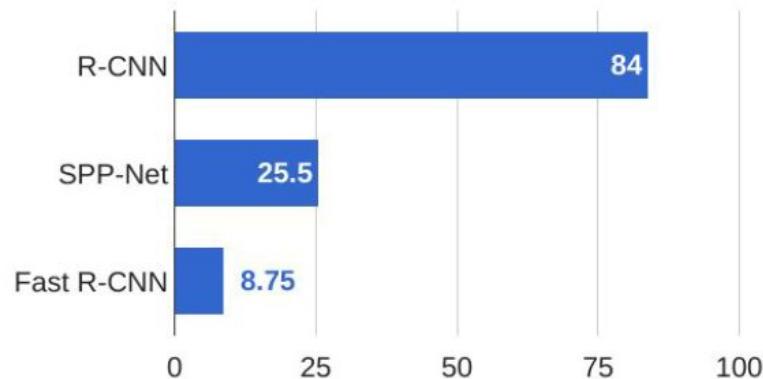
RoI conv features:
 $512 \times 7 \times 7$
for region proposal

Fully-connected layers expect
low-res conv features:
 $512 \times 7 \times 7$

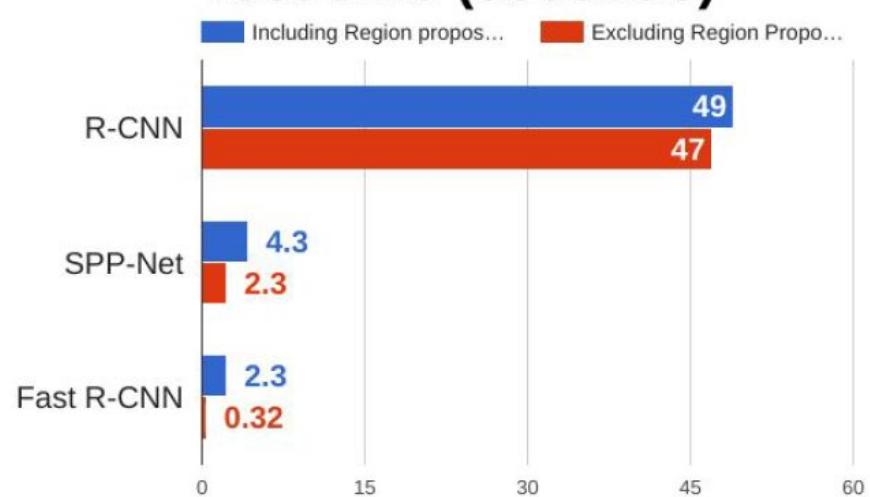
Girshick, "Fast R-CNN", ICCV 2015.

R-CNN vs SPP vs Fast R-CNN

Training time (Hours)



Test time (seconds)

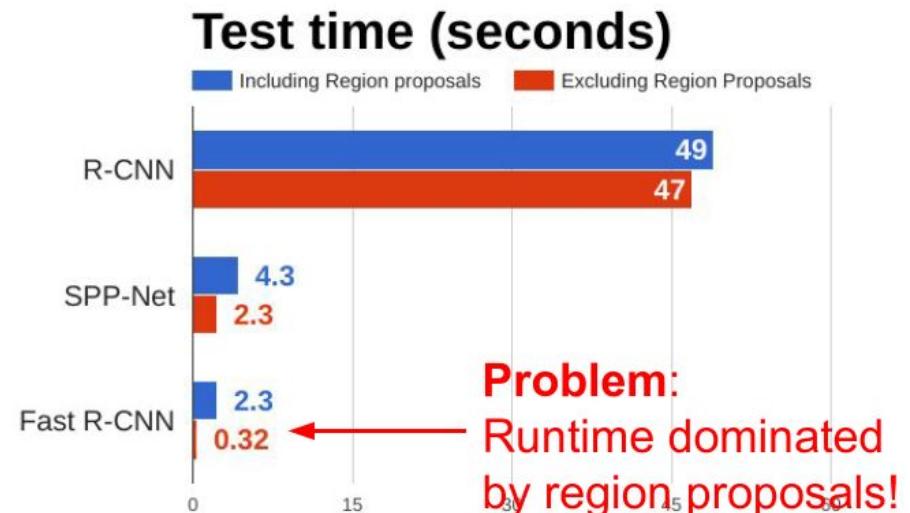
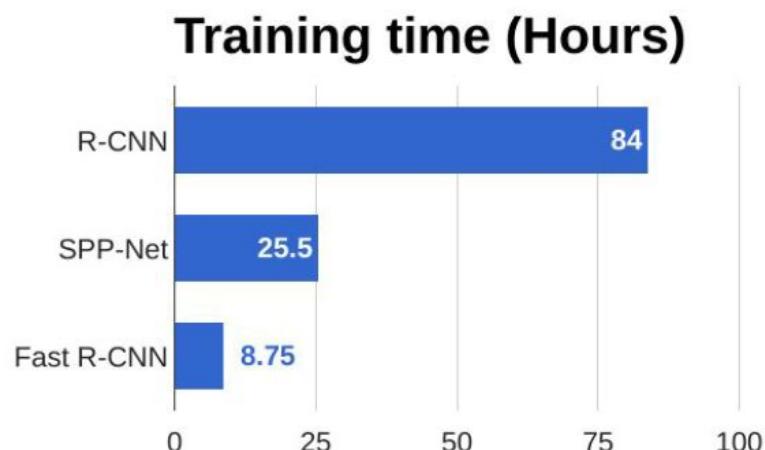


Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

He et al, "Spatial pyramid pooling in deep convolutional networks for visual recognition", ECCV 2014

Girshick, "Fast R-CNN", ICCV 2015

R-CNN vs SPP vs Fast R-CNN



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

He et al, "Spatial pyramid pooling in deep convolutional networks for visual recognition", ECCV 2014

Girshick, "Fast R-CNN", ICCV 2015

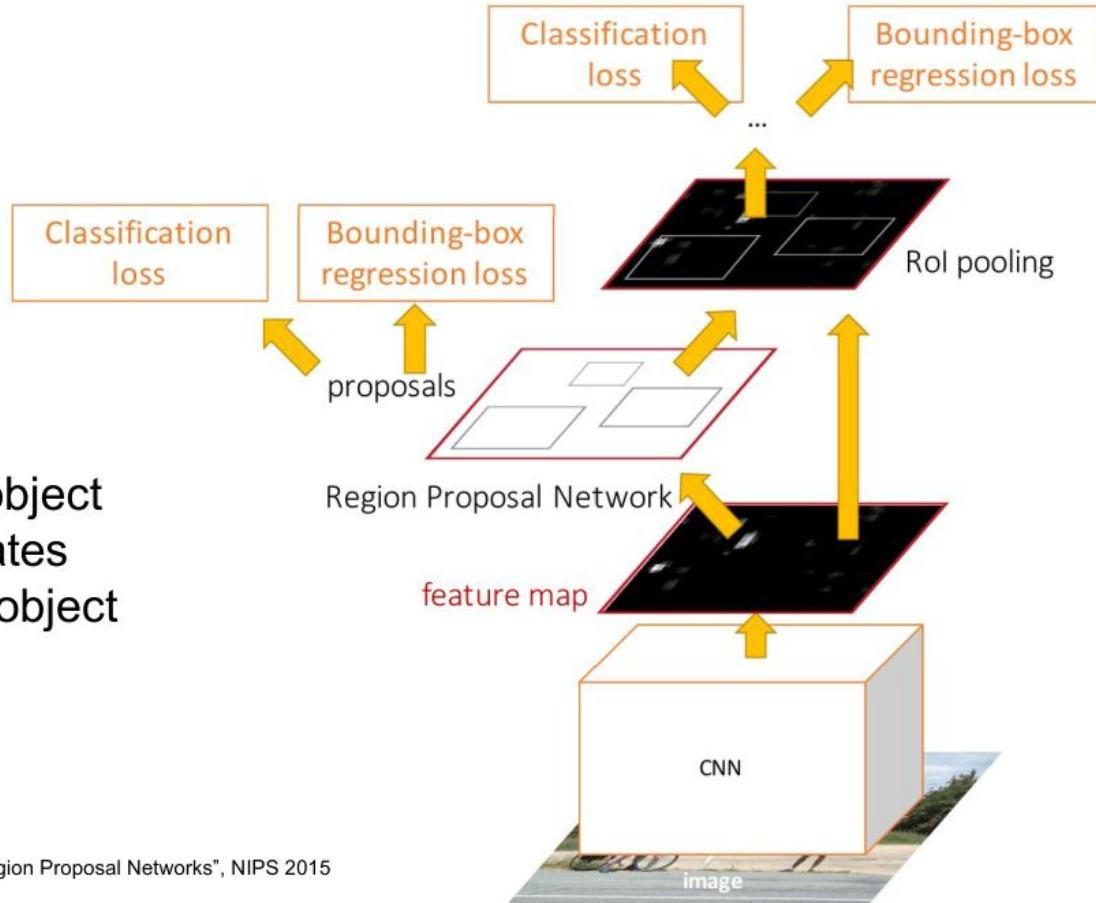
Faster R-CNN:

Make CNN do proposals!

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

Jointly train with 4 losses:

1. RPN classify object / not object
2. RPN regress box coordinates
3. Final classification score (object classes)
4. Final box coordinates

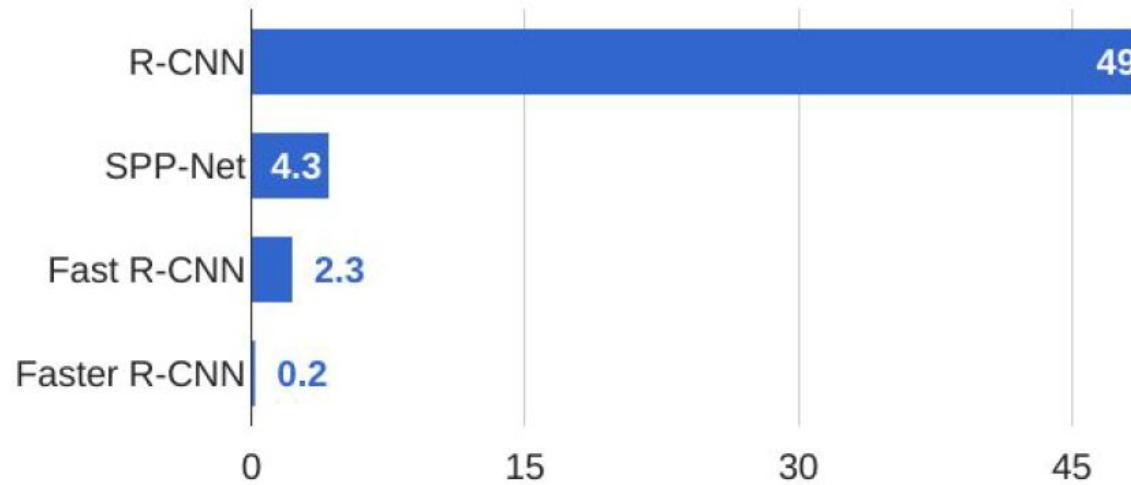


Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission

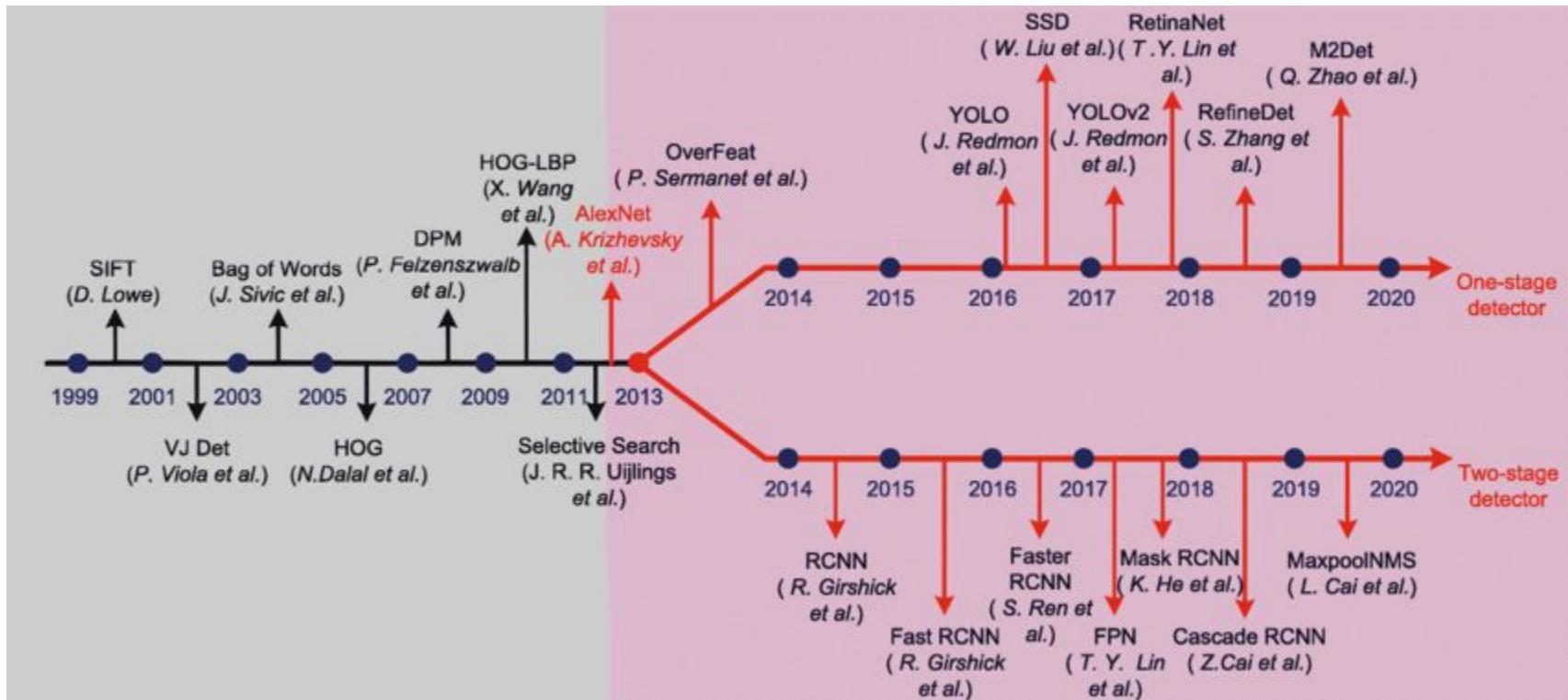
Faster R-CNN:

Make CNN do proposals!

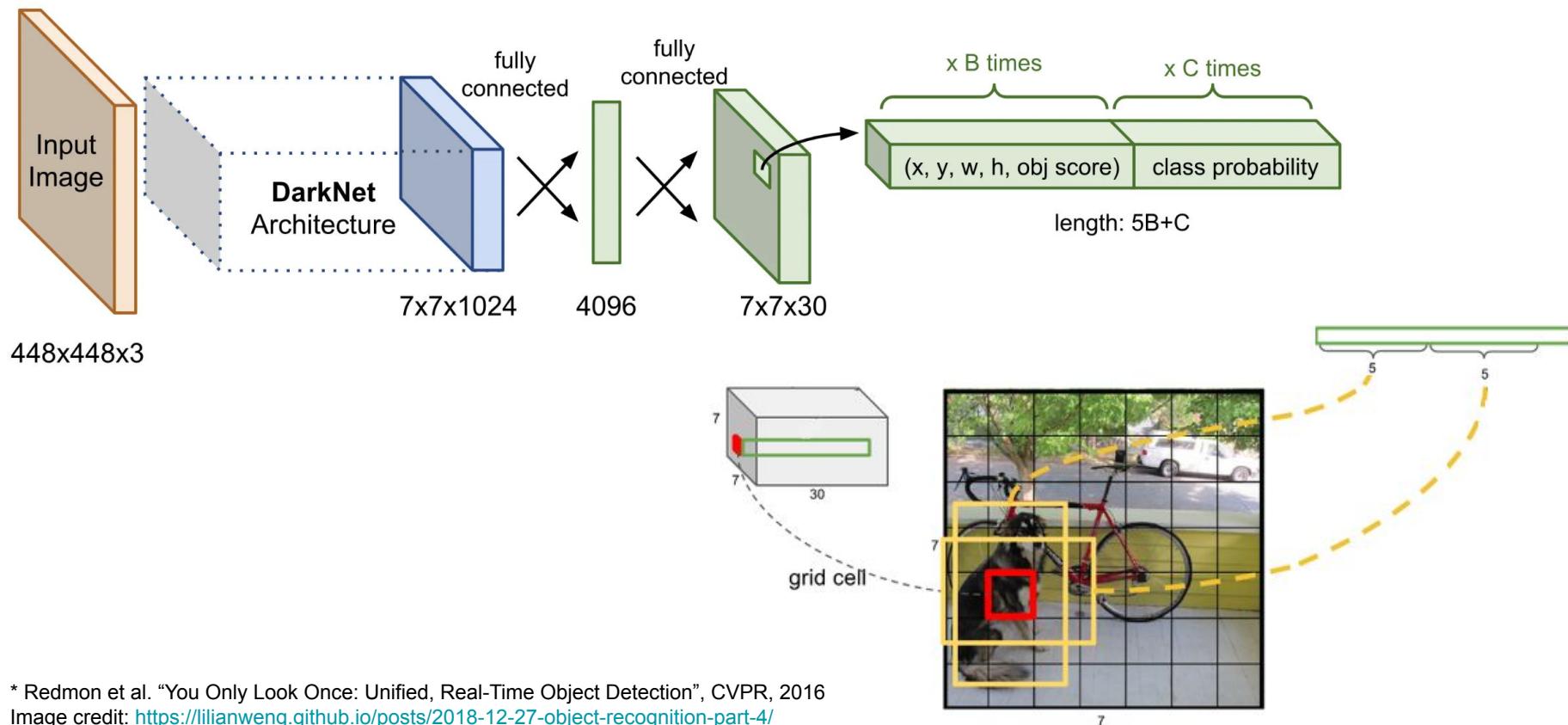
R-CNN Test-Time Speed



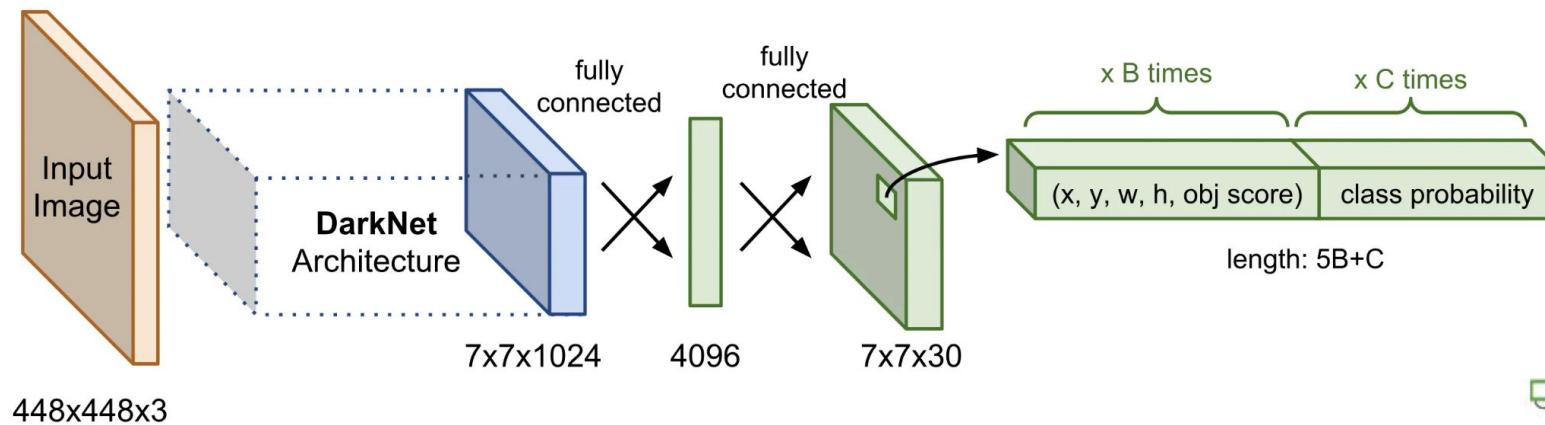
Evolution of object detection



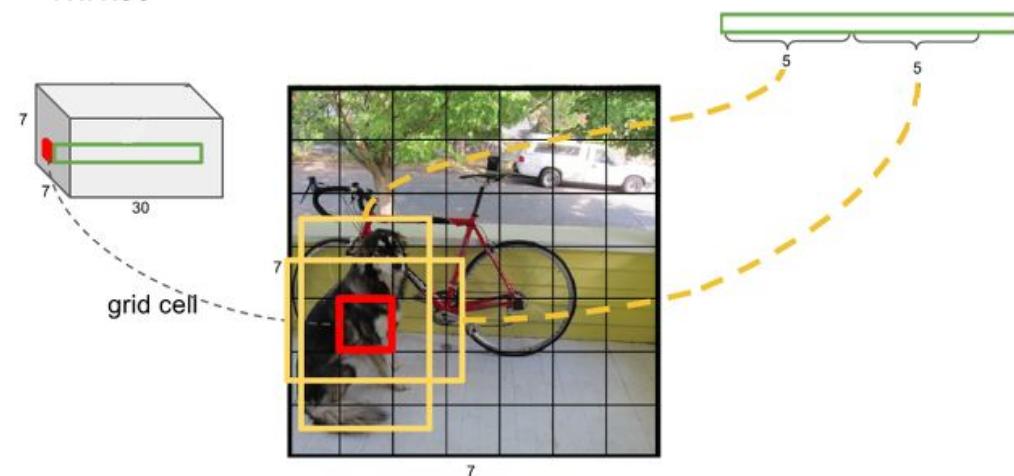
Single stage object detection - YOLO v1*



Single stage object detection - YOLO v1



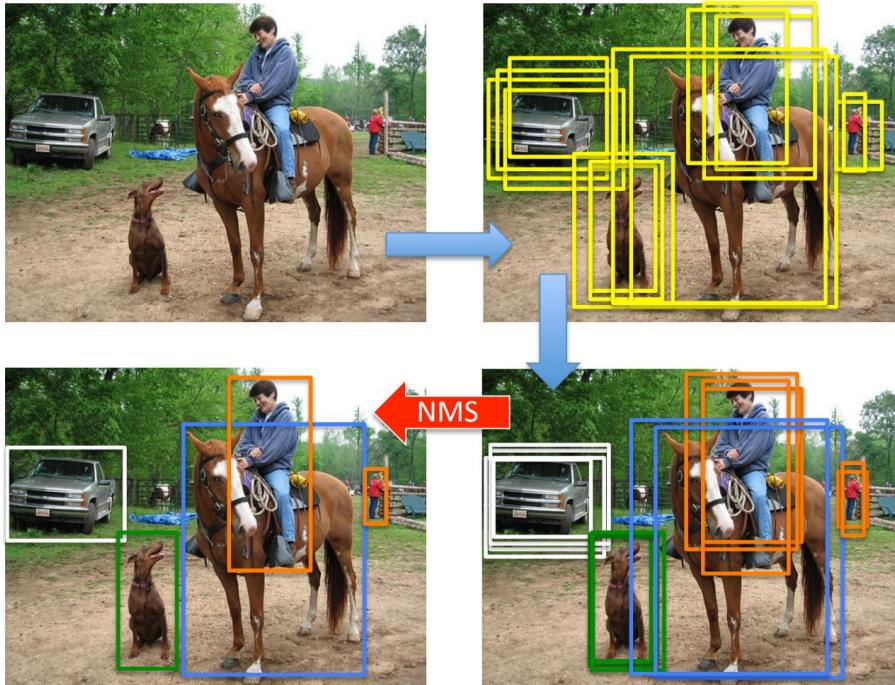
- **Darknet (26 layers)**: inspired from Googlenet. Inception module replaced by 1×1 and 3×3 conv layers.
- **x, y**: relative to the cell coordinate (top-left).
- **w, h**: normalized by image width and height.
- During the training, **obj score** is computed by **IOU** between predicted box and groundtruth.
- **Class probabilities** are multiplied by obj score.



Single stage object detection - YOLO v1

- **Pros:**
 - It is fast,
 - Generalizes well to unseen domains.
- **Cons:**
 - Its accuracy is not great,
 - Does not work well on different image resolutions,
 - Only predicts one class per cell.

Single stage object detection - Non Maximum Suppression



Input : $\mathcal{B} = \{b_1, \dots, b_N\}$, $\mathcal{S} = \{s_1, \dots, s_N\}$, N_t
 \mathcal{B} is the list of initial detection boxes
 \mathcal{S} contains corresponding detection scores
 N_t is the NMS threshold

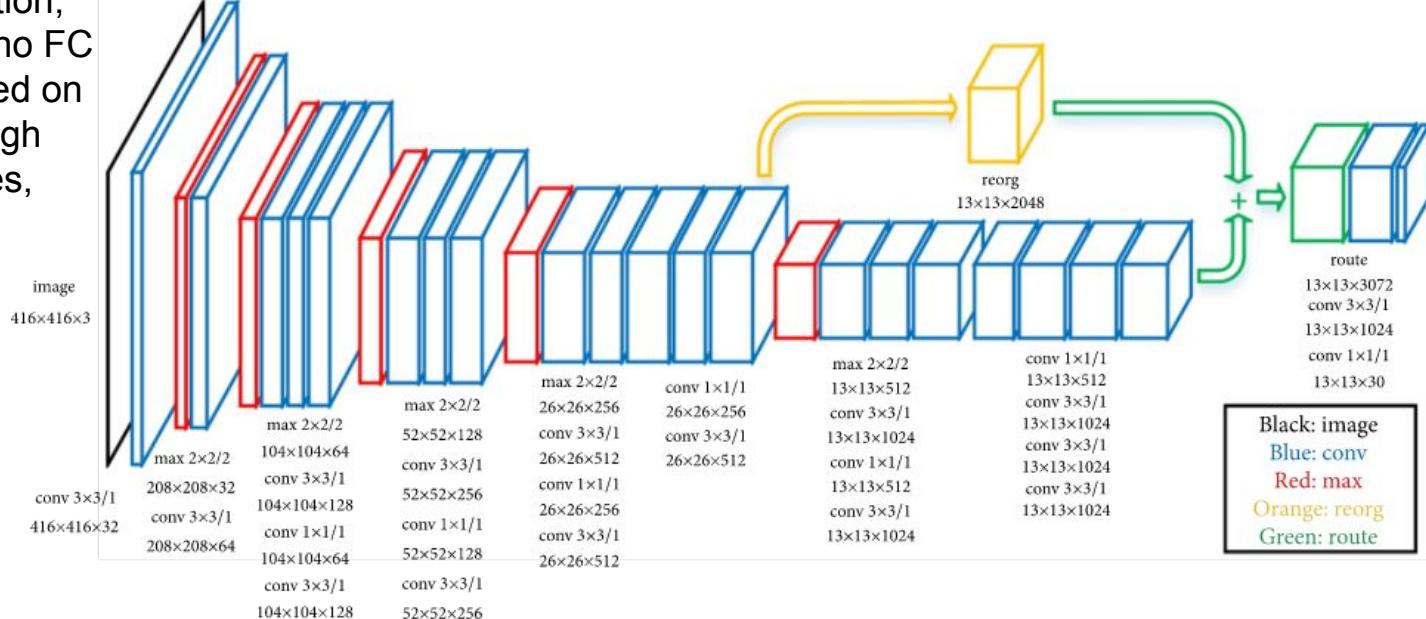
```
begin
   $\mathcal{D} \leftarrow \{\}$ 
  while  $\mathcal{B} \neq \text{empty}$  do
     $m \leftarrow \text{argmax } \mathcal{S}$ 
     $\mathcal{M} \leftarrow b_m$ 
     $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{M}; \mathcal{B} \leftarrow \mathcal{B} - \mathcal{M}$ 
    for  $b_i$  in  $\mathcal{B}$  do
      if  $\text{iou}(\mathcal{M}, b_i) \geq N_t$  then
        |  $\mathcal{B} \leftarrow \mathcal{B} - b_i; \mathcal{S} \leftarrow \mathcal{S} - s_i$ 
      end
    end
     $s_i \leftarrow s_i f(\text{iou}(\mathcal{M}, b_i))$ 
  end
  return  $\mathcal{D}, \mathcal{S}$ 
```

$s_i = \begin{cases} s_i, & \text{iou}(\mathcal{M}, b_i) < N_t \\ s_i(1 - \text{iou}(\mathcal{M}, b_i)), & \text{iou}(\mathcal{M}, b_i) \geq N_t \end{cases}$

Single stage object detection - YOLO v2 *

Improvements over YOLOv1

- Darknet with 19 layers, batch normalization, reorg layer and no FC layers, pre-trained on imagenet with high resolution images,



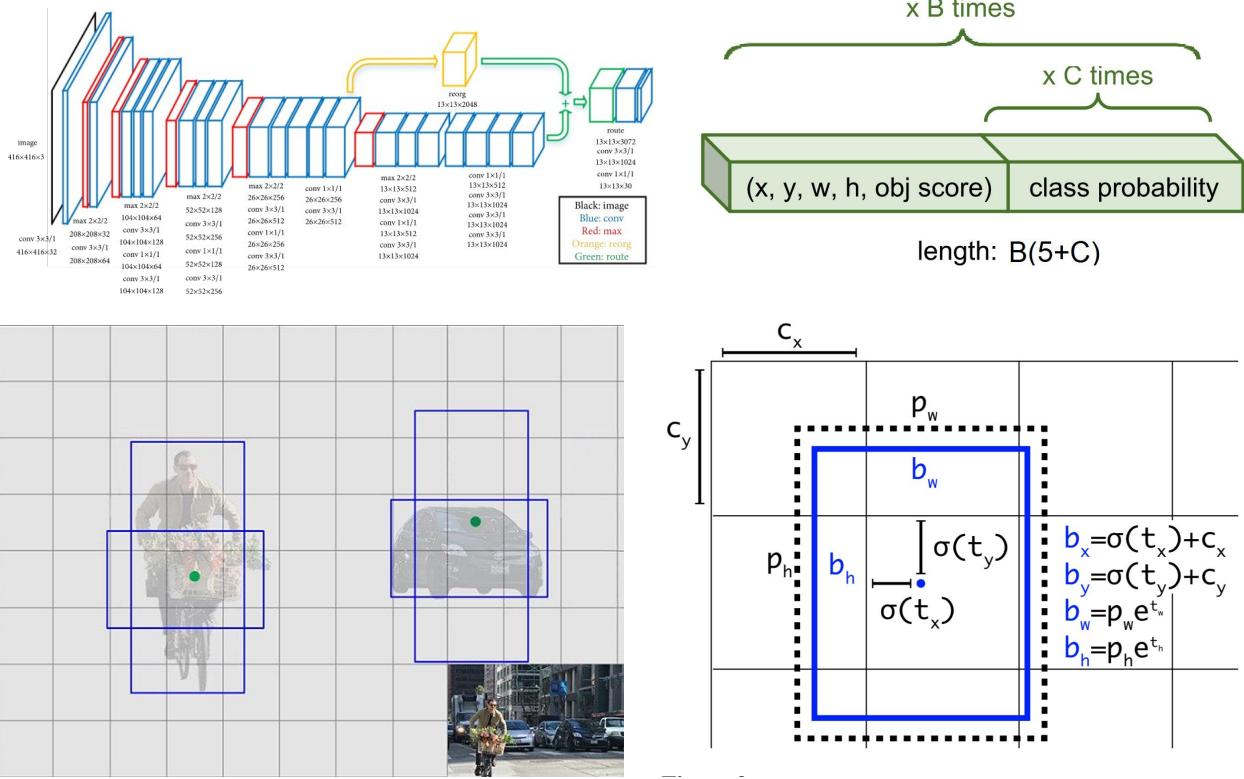
* <https://arxiv.org/pdf/1612.08242.pdf>

Image credit: <https://www.hindawi.com/journals/mpe/2018/3518959/>

Single stage object detection - YOLO v2

Improvements over YOLOv1

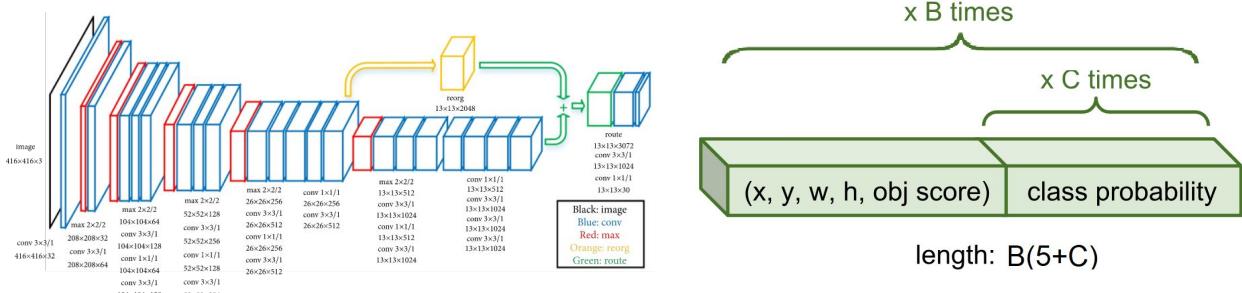
- Darknet with 19 layers, batch normalization, reorg layer and no FC layers, pre-trained on imagenet with high resolution images,
 - Introduction of anchor boxes optimized by kmeans rather than predefined boxes,
 - Object classes defined for each anchor in each cell,
 - Multi-scale training every 10 epochs.



Single stage object detection - YOLO v2

Improvements over YOLOv1

- Darknet with 19 layers, batch normalization, reorg layer and no FC layers, pre-trained on imagenet with high resolution images,
- Introduction of anchor boxes optimized by kmeans rather than predefined boxes,
- Object classes defined for each anchor in each cell,
- Multi-scale training every 10 epochs.

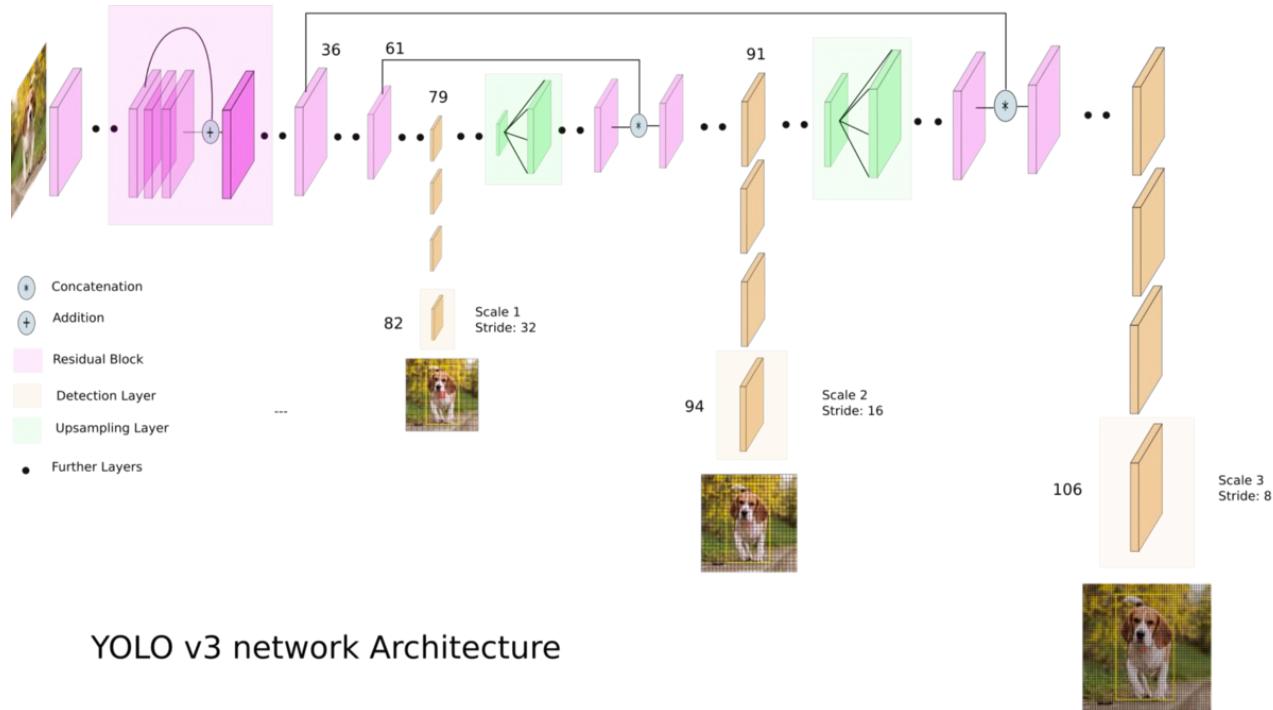


Localization improved but still low recall on small objects

Single stage object detection - YOLO v3

Improvements over YOLOv2

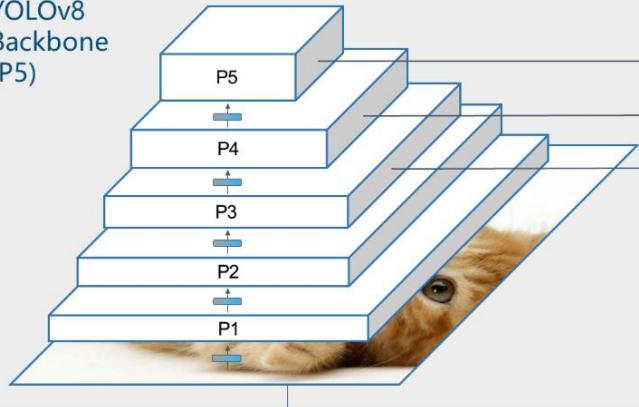
- Darknet with 53 layers, residual connections and feature pyramid layers at the end,
- Logistic regression for object score,
- Sigmoid activation and binary cross-entropy for class prediction.



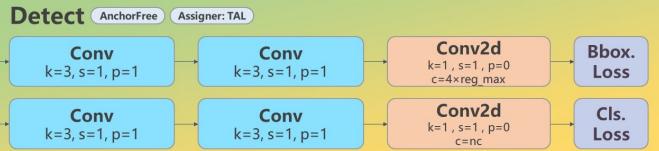
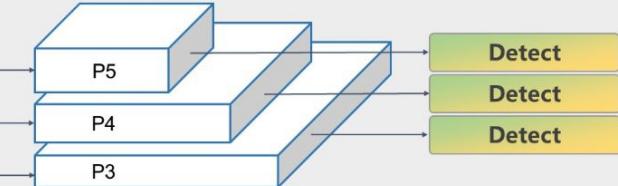
Single stage object detection - YOLO v8

Backbone

YOLOv8
Backbone
(P5)



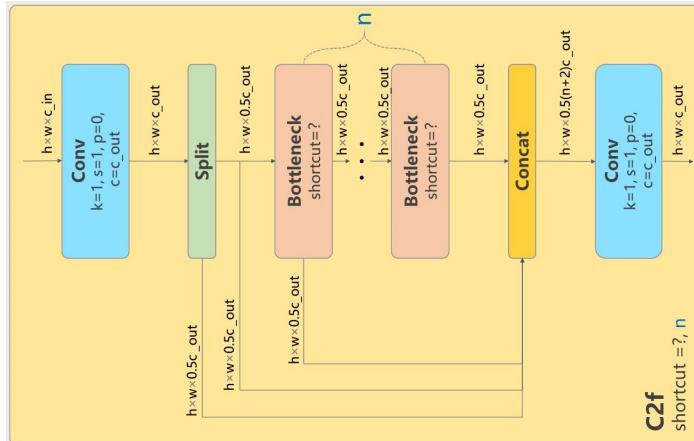
Head YOLOv8Head



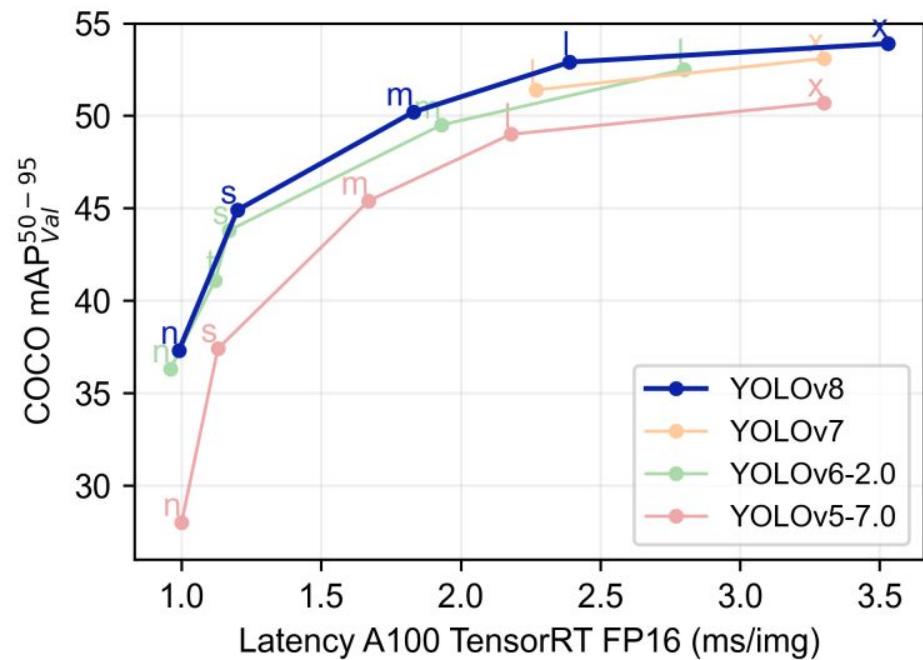
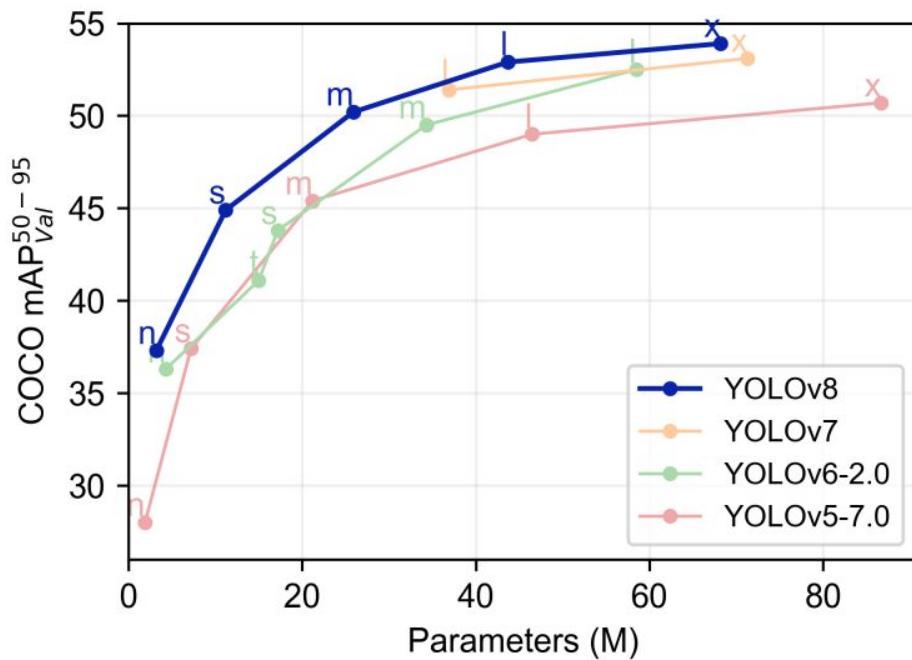
Features:

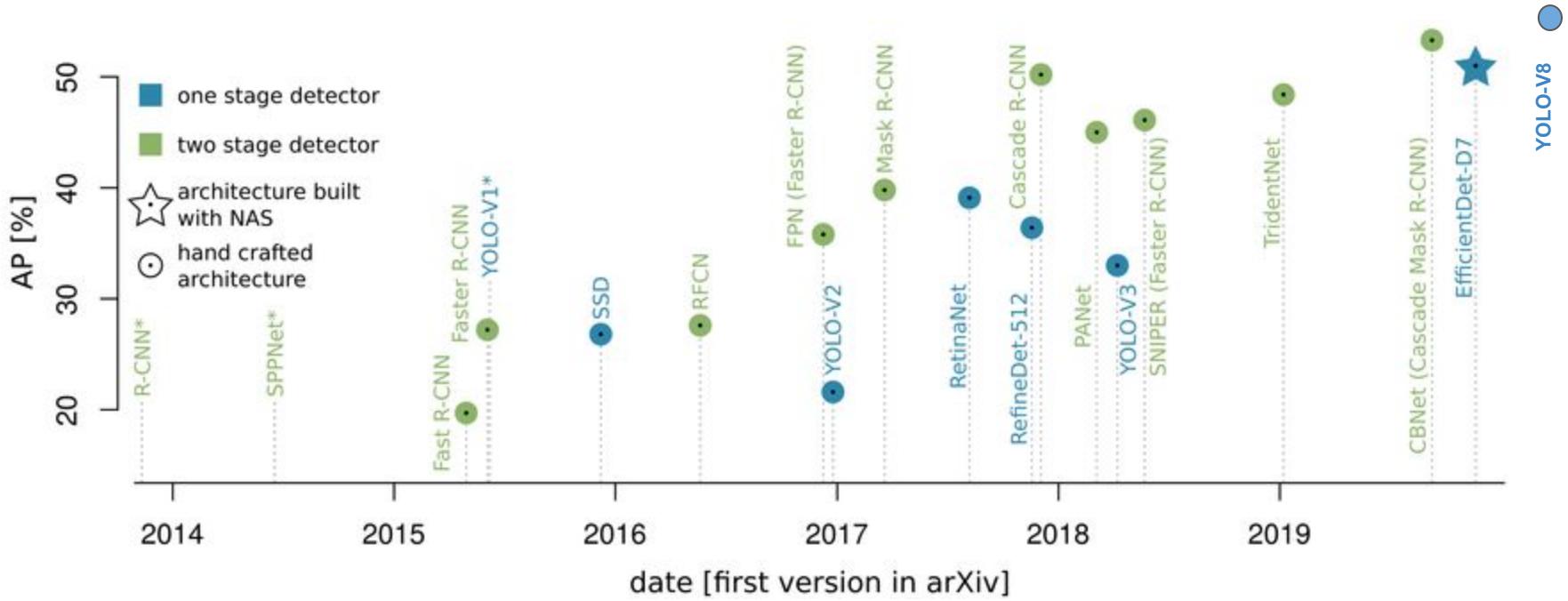
- New architecture,
- Fast detection with state-of-the-art accuracy,
- Anchor-free detections, which help to speed-up NMS,
- A well-structured python package with developer convenience features like semantic segmentation and object classification,

Image credit: RangeKing <https://github.com/ultralytics/ultralytics/issues/189>



Single stage object detection - YOLO v8





Instance Segmentation



GRASS, CAT,
TREE, SKY

No objects, just pixels



CAT

Single Object



DOG, DOG, CAT

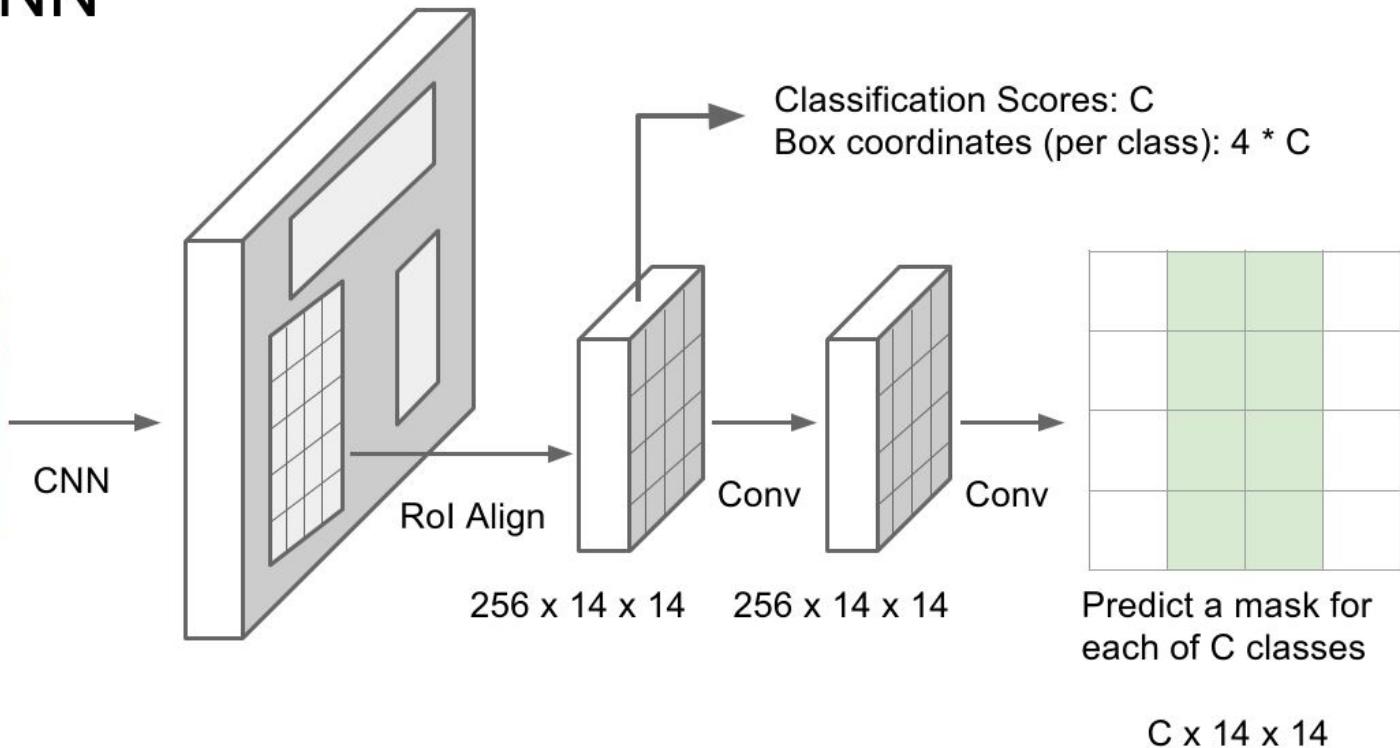
Multiple Object



DOG, DOG, CAT

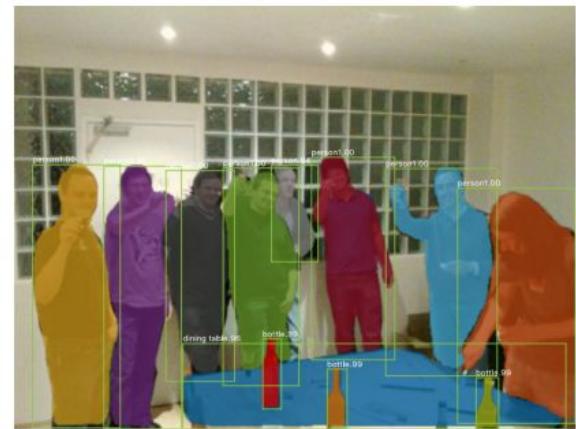
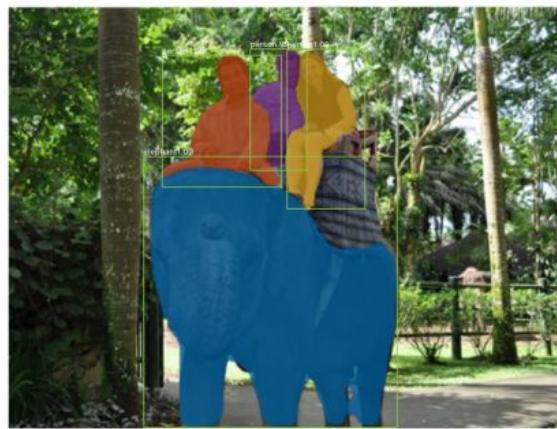
This image is CC0 public domain

Mask R-CNN



He et al, "Mask R-CNN", arXiv 2017

Mask R-CNN: Very Good Results!

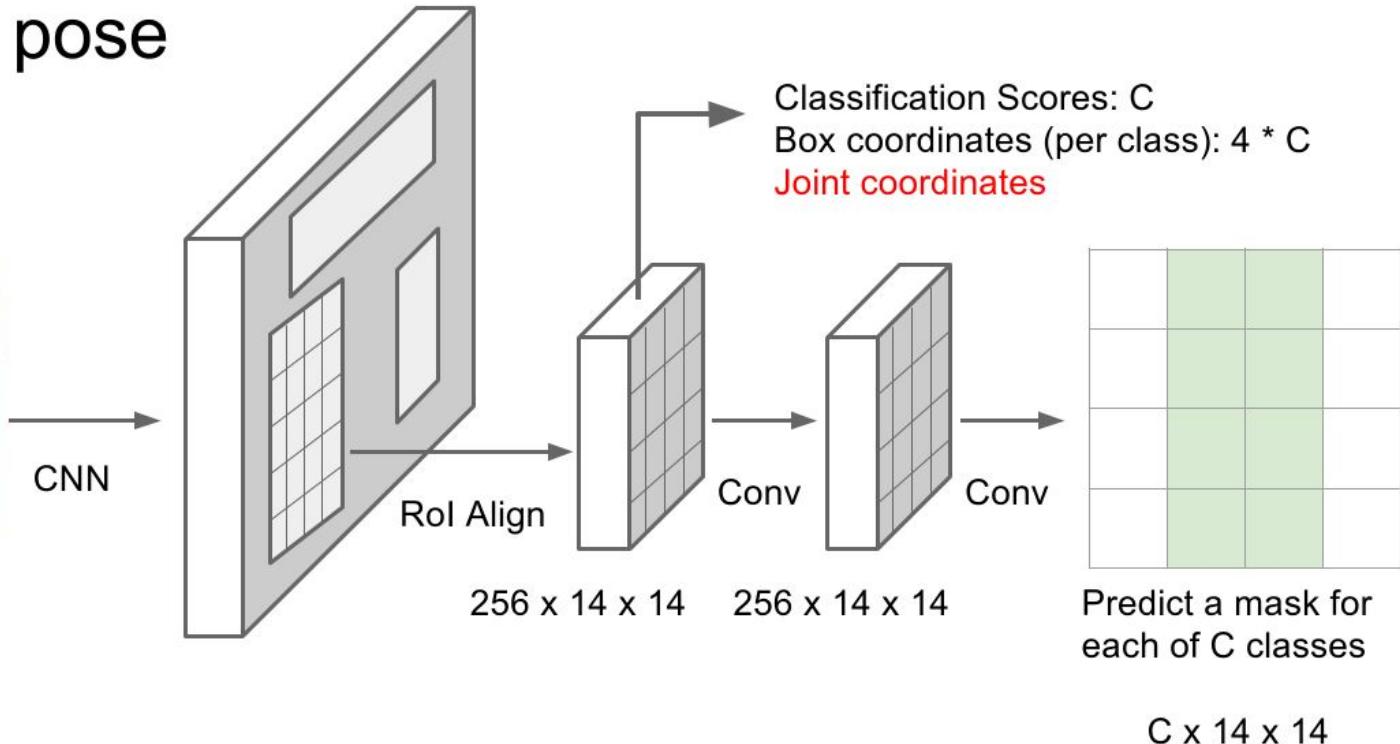


He et al, "Mask R-CNN", arXiv 2017

Figures copyright Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, 2017.
Reproduced with permission.

Mask R-CNN

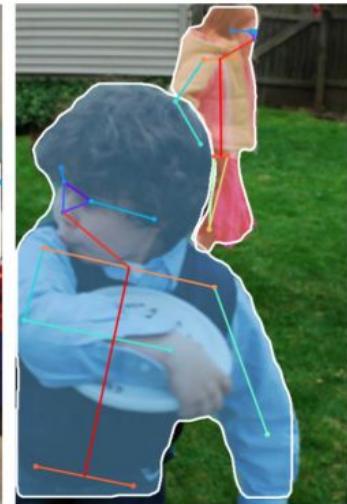
Also does pose



He et al, "Mask R-CNN", arXiv 2017

Mask R-CNN

Also does pose



He et al, "Mask R-CNN", arXiv 2017

Figures copyright Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, 2017.
Reproduced with permission.