CNN applications

Samuel Cheng
(Slide credits: Fei-Fei Li, Andrej Karpathy, Justin Johnson, Serena Yeung)

School of ECE
University of Oklahoma

Spring, 2017

Overview

- We will look into several applications of CNNs besides image recognition
    - Semantic segmentation
    - Object localization
    - Object detection
    - Instance segmentation

So far: Image Classification

This image is CC0 public domain

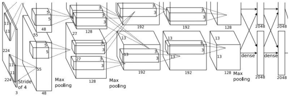


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

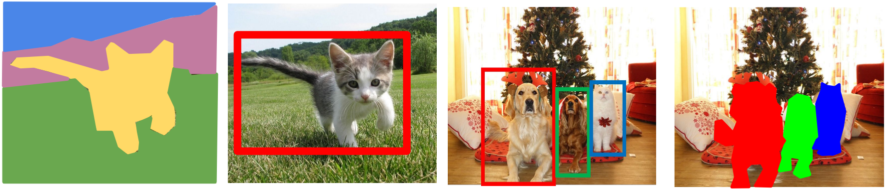
Vector: 4096

Fully-Connected: 4096 to 1000

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

Other Computer Vision Tasks

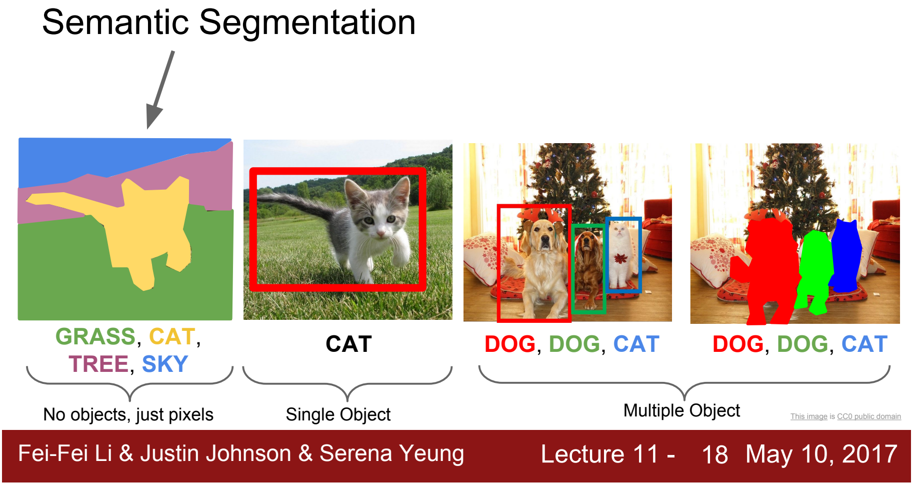
Semantic Segmentation | **Classification + Localization** | **Object Detection** | **Instance Segmentation**



GRASS, CAT, TREE, SKY — No objects, just pixels

CAT — Single Object

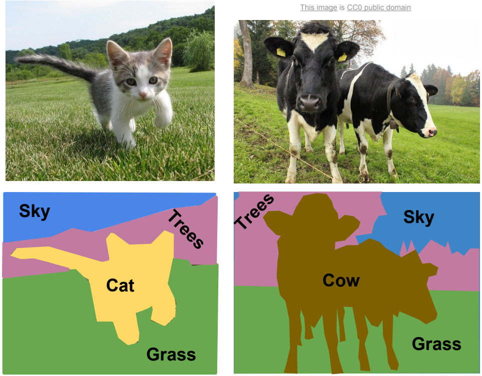
DOG, DOG, CAT | DOG, DOG, CAT — Multiple Object


Semantic Segmentation
GRASS, CAT,
TREE, SKY
CAT
DOG, DOG, CAT
DOG, DOG, CAT
No objects, just pixels
Single Object
Multiple Object
This image is CC0 public domain
Fei-Fei Li & Justin Johnson & Serena Yeung
Lecture 11 - 18 May 10, 2017

Semantic Segmentation

Label each pixel in the image with a category label

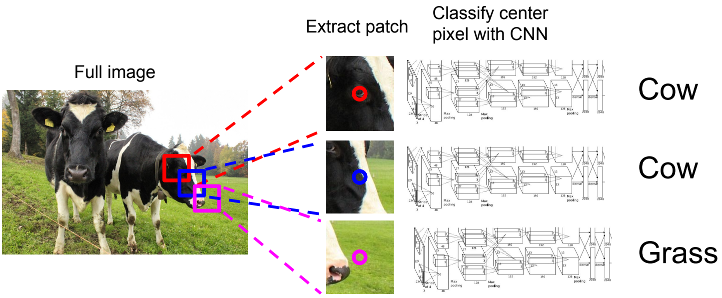
Don't differentiate instances, only care about pixels



This image is CC0 public domain

Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 19 May 10, 2017

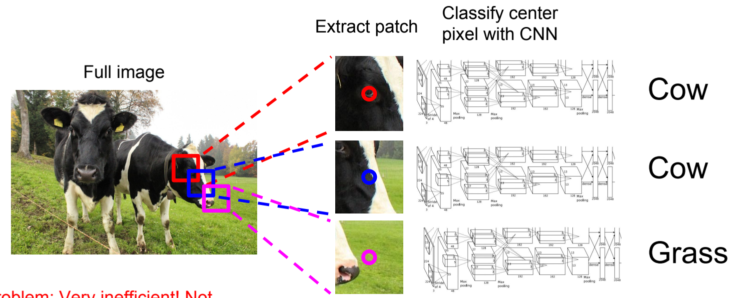
Semantic Segmentation Idea: Sliding Window



Farabet et al, “Learning Hierarchical Features for Scene Labeling,” TPAMI 2013
Pinheiro and Collobert, “Recurrent Convolutional Neural Networks for Scene Labeling”, ICML 2014

Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 20 May 10, 2017

Semantic Segmentation Idea: Sliding Window

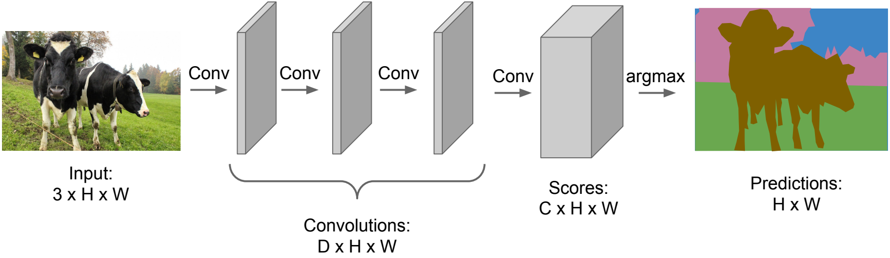


Problem: Very inefficient! Not reusing shared features between overlapping patches

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

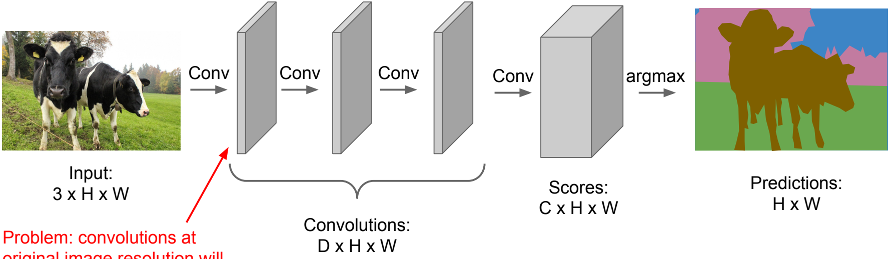
Design a network as a bunch of convolutional layers
to make predictions for pixels all at once!



Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 22 May 10, 2017

Semantic Segmentation Idea: Fully Convolutional

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

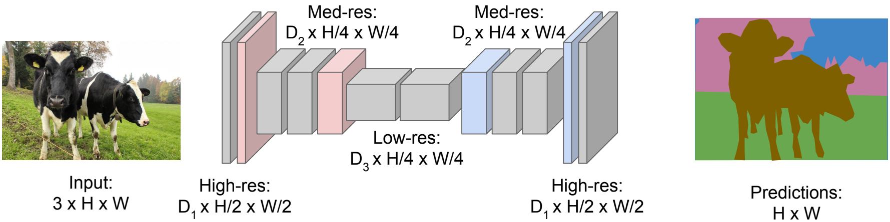


Problem: convolutions at original image resolution will be very expensive …

Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 23 May 10, 2017

Semantic Segmentation Idea: Fully Convolutional

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



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

Semantic Segmentation Idea: Fully Convolutional

Downsampling: Pooling, strided convolution

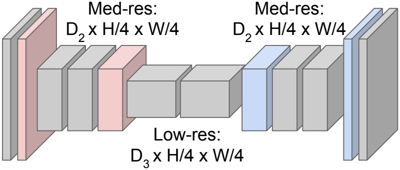
Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

Upsampling: ???

Input:
3 x H x W

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

Med-res:
D_2 x H/4 x W/4



Low-res:
D_3 x H/4 x W/4

Med-res:
D_2 x H/4 x W/4

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

Predictions:
H x W

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

In-Network upsampling: "Unpooling"

Nearest Neighbor

Input: 2 x 2

1	2
3	4

Output: 4 x 4

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

"Bed of Nails"

Input: 2 x 2

1	2
3	4

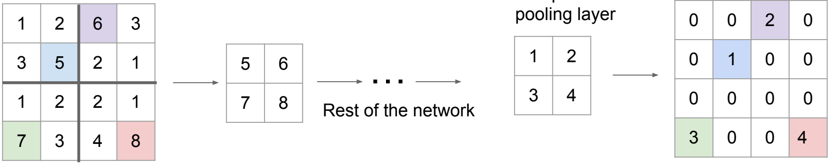
Output: 4 x 4

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

In-Network upsampling: “Max Unpooling”

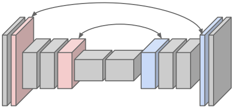
Max Pooling
Remember which element was max!

Max Unpooling
Use positions from pooling layer



Input: 4 x 4 Output: 2 x 2 Input: 2 x 2 Output: 4 x 4

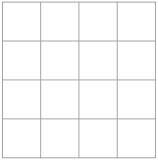
Corresponding pairs of downsampling and upsampling layers



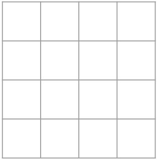
Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 27 May 10, 2017

Learnable Upsampling: Transpose Convolution

Recall: Typical 3 x 3 convolution, stride 1 pad 1



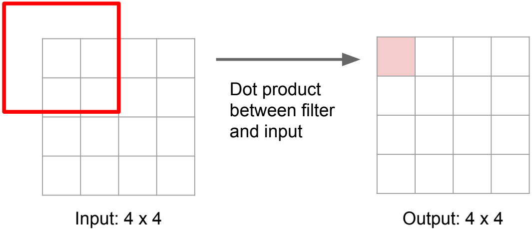
Input: 4 x 4



Output: 4 x 4

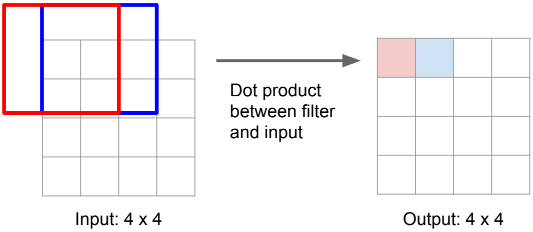
Learnable Upsampling: Transpose Convolution

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



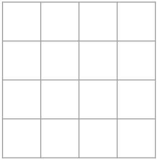
Learnable Upsampling: Transpose Convolution

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

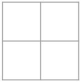


Learnable Upsampling: Transpose Convolution

Recall: Normal 3 x 3 convolution, <u>stride 2</u> pad 1



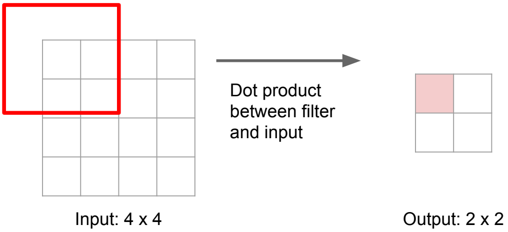
Input: 4 x 4



Output: 2 x 2

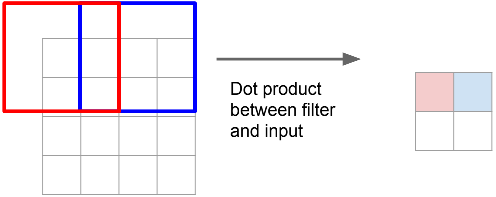
Learnable Upsampling: Transpose Convolution

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



Learnable Upsampling: Transpose Convolution

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



Input: 4 x 4

Output: 2 x 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

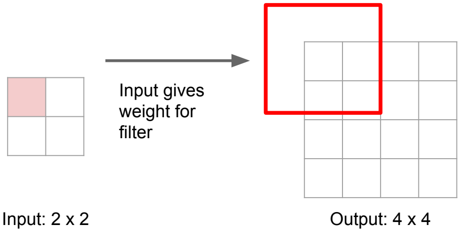
3 x 3 **transpose** convolution, stride 2 pad 1

Input: 2 x 2

Output: 4 x 4

Learnable Upsampling: Transpose Convolution

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

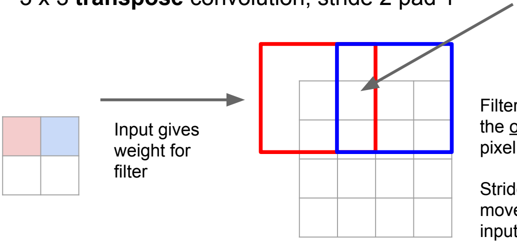


Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 35 May 10, 2017

Learnable Upsampling: Transpose Convolution

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

Sum where output overlaps



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

Stride gives ratio between movement in output and input

Input: 2 x 2

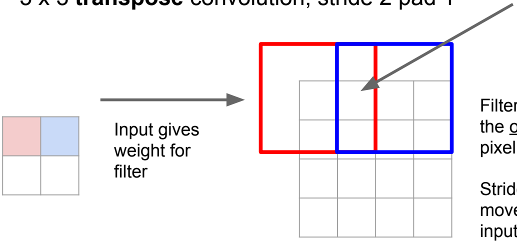
Output: 4 x 4

Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 36 May 10, 2017

Learnable Upsampling: Transpose Convolution

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

Sum where output overlaps



Input: 2 x 2

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

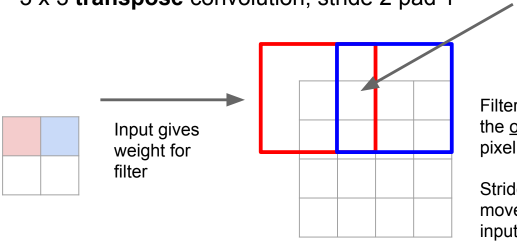
Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 37 May 10, 2017

Learnable Upsampling: Transpose Convolution

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

Other names:
-Deconvolution (bad)
-Upconvolution
-Fractionally strided convolution
-Backward strided convolution

Sum where output overlaps



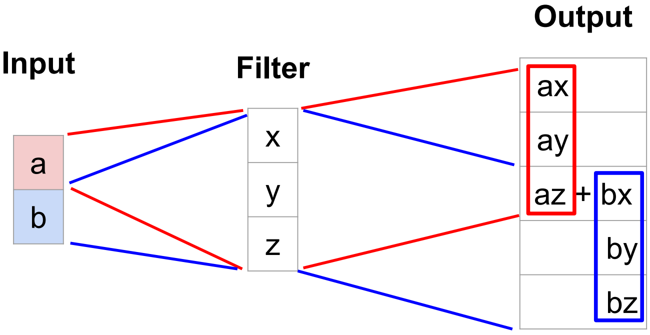
Input: 2 x 2

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

Transpose Convolution: 1D Example



Output contains copies of the filter weighted by the input, summing at where at overlaps in the output

Need to crop one pixel from output to make output exactly 2x input

Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 39 May 10, 2017

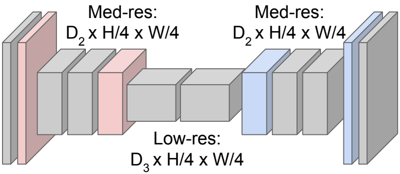
Semantic Segmentation Idea: Fully Convolutional

Downsampling: Pooling, strided convolution

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

Upsampling: Unpooling or strided transpose convolution

Input:
3 x H x W



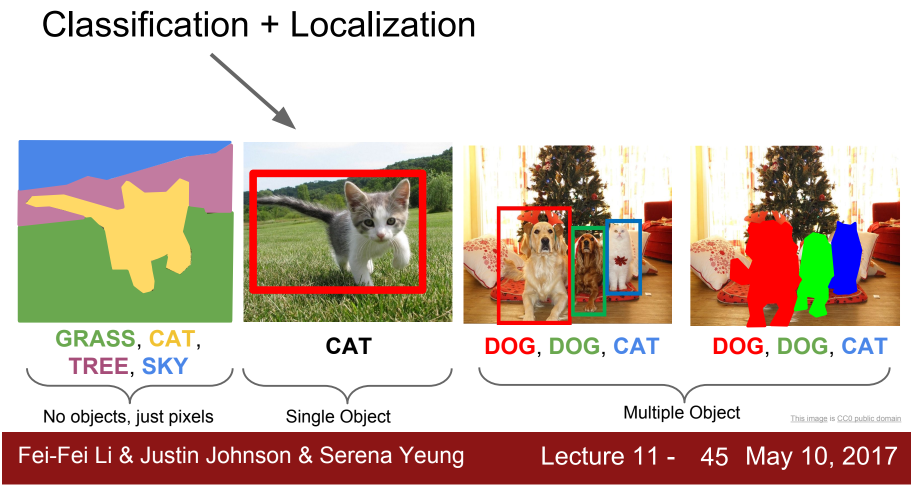
High-res:
D_1 x H/2 x W/2

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

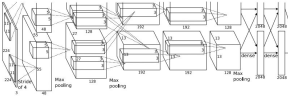
Predictions:
H x W

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

Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 44 May 10, 2017


Classification + Localization
GRASS, CAT,
TREE, SKY
No objects, just pixels
CAT
Single Object
DOG, DOG, CAT
DOG, DOG, CAT
Multiple Object
This image is CC0 public domain
Fei-Fei Li & Justin Johnson & Serena Yeung
Lecture 11 - 45
May 10, 2017

Classification + Localization



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)

Treat localization as a regression problem!

Classification + Localization

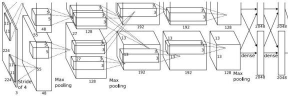
Correct label:
Cat

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

Softmax Loss

Fully Connected:
4096 to 1000

This image is CC0 public domain



Vector:
4096

Fully Connected:
4096 to 4

Box Coordinates
(x, y, w, h)

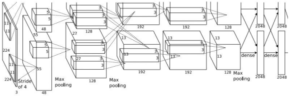
L2 Loss

Correct box:
(x’, y’, w’, h’)

Treat localization as a regression problem!

Classification + Localization

This image is CC0 public domain



Vector: 4096

Fully Connected: 4096 to 1000

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

Correct label: Cat

Softmax Loss

Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

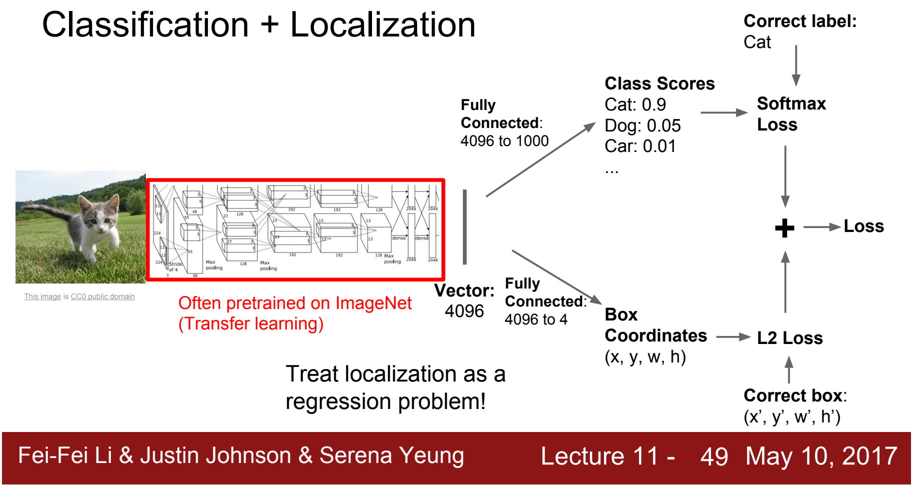
L2 Loss

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

Multitask Loss

\+ → **Loss**

Treat localization as a regression problem!


Classification + Localization
This image is CC0 public domain
Often pretrained on ImageNet
(Transfer learning)
Vector:
4096
Fully
Connected:
4096 to 1000
Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...
Correct label:
Cat
Softmax
Loss
Loss
Fully
Connected:
4096 to 4
Box
Coordinates
(x, y, w, h)
L2 Loss
Correct box:
(x', y', w', h')
Treat localization as a
regression problem!
Fei-Fei Li & Justin Johnson & Serena Yeung
Lecture 11 - 49 May 10, 2017

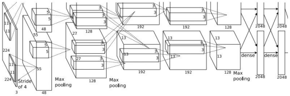
Aside: Human Pose Estimation

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

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)

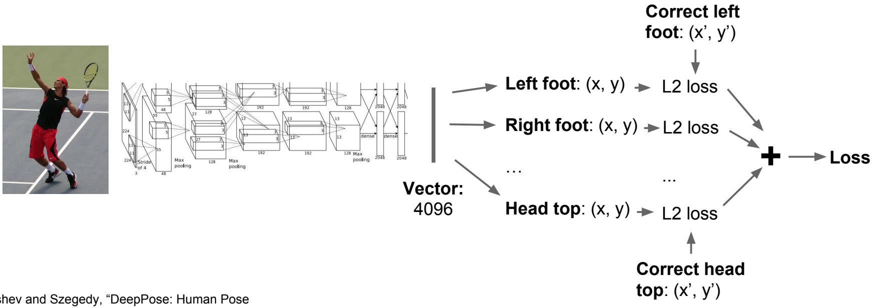
Right foot: (x, y)

…

Head top: (x, y)

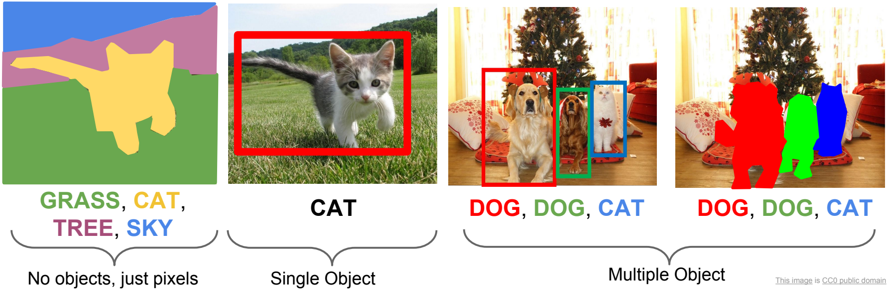
Toshev and Szegedy, “DeepPose: Human Pose Estimation via Deep Neural Networks”, CVPR 2014

Aside: Human Pose Estimation



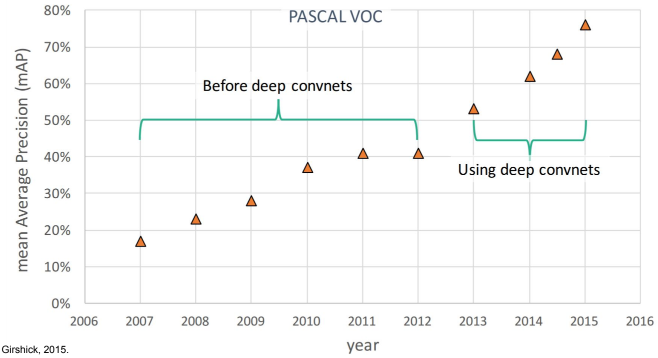
Toshev and Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks", CVPR 2014

Object Detection



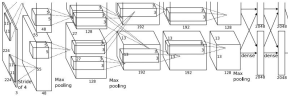
Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 53 May 10, 2017

Object Detection: Impact of Deep Learning

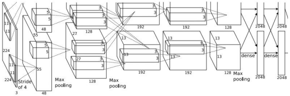


Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 54 May 10, 2017

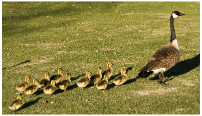
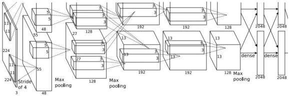
Object Detection as Regression?

CAT: (x, y, w, h)

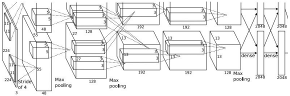
DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)

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

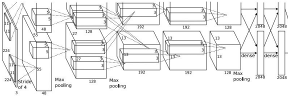
Object Detection as Regression?

Each image needs a different number of outputs!

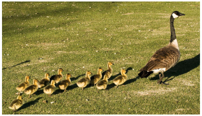
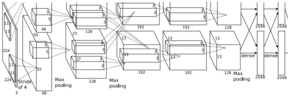
CAT: (x, y, w, h)

4 numbers

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

16 numbers

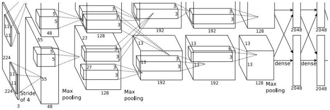



DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....

Many numbers!

Object Detection as Classification: Sliding Window

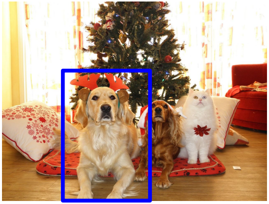
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

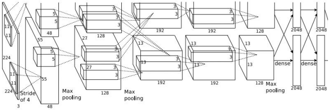


Dog? NO
Cat? NO
Background? YES

Object Detection as Classification: Sliding Window

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

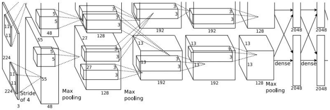




Dog? YES
Cat? NO
Background? NO

Object Detection as Classification: Sliding Window

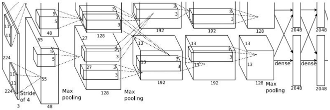
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

Object Detection as Classification: Sliding Window

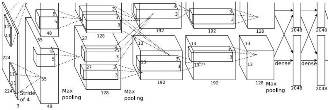
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? YES
Background? NO

Object Detection as Classification: Sliding Window

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

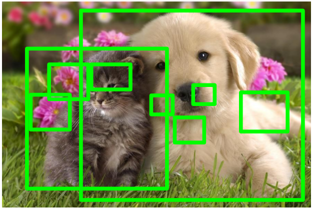


Dog? NO
Cat? YES
Background? NO

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



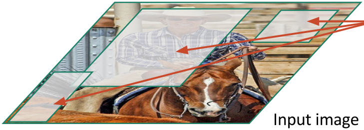
Alexe et al, "Measuring the objectness of image windows", TPAMI 2012
Uijlings et al, "Selective Search for Object Recognition", IJCV 2013
Cheng et al, "BING: Binarized normed gradients for objectness estimation at 300fps", CVPR 2014
Zitnick and Dollar, "Edge boxes: Locating object proposals from edges",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

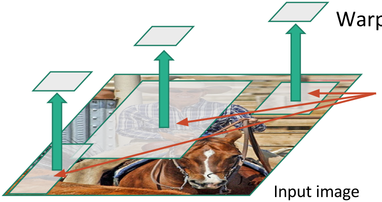
R-CNN



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

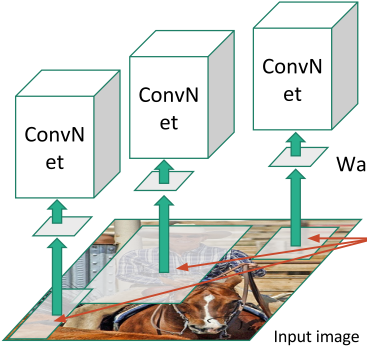
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

R-CNN



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

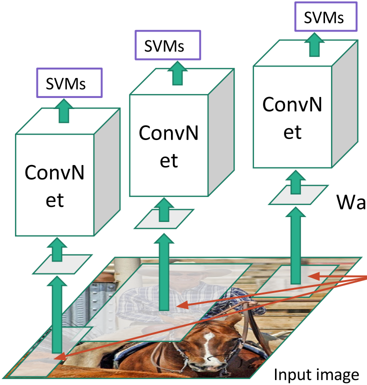
R-CNN



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

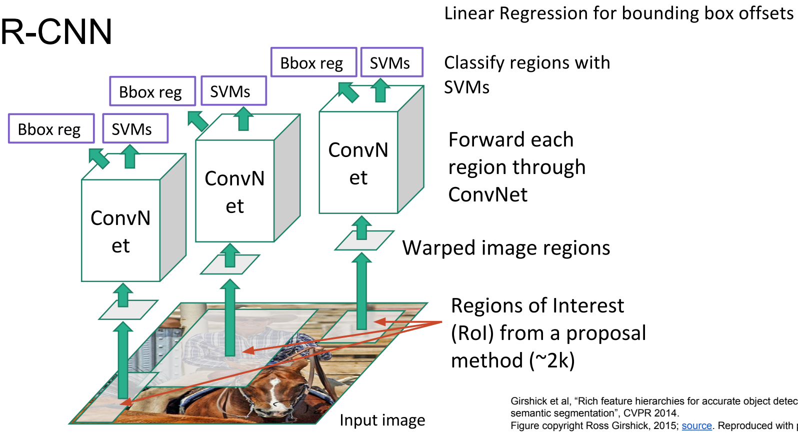
Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 66 May 10, 2017

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

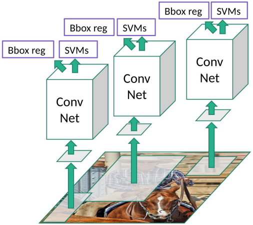


Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 68 May 10, 2017

R-CNN: Problems

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



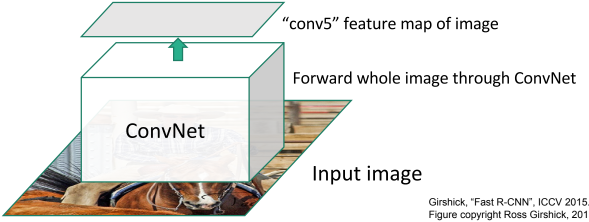
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

Fast R-CNN

Input image

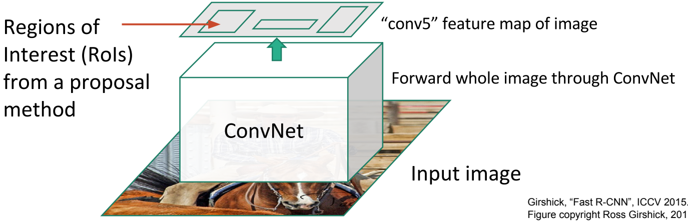
Girshick, "Fast R-CNN", ICCV 2015.

Fast R-CNN



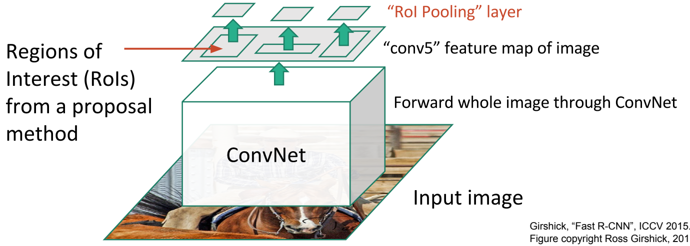
Girshick, "Fast R-CNN", ICCV 2015.

Fast R-CNN



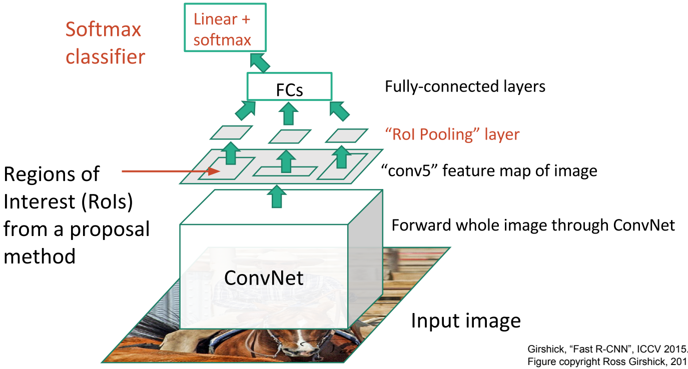
Girshick, "Fast R-CNN", ICCV 2015.

Fast R-CNN



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

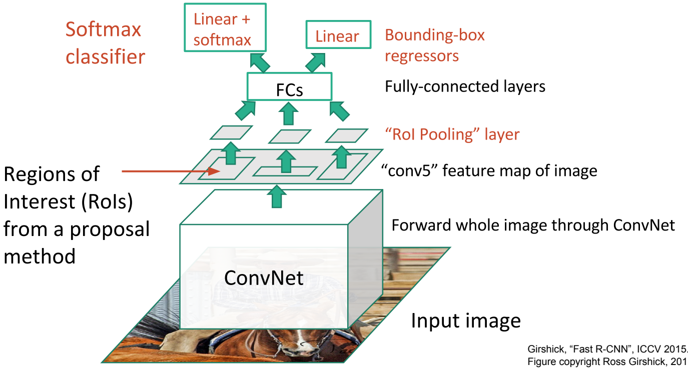
Fast R-CNN



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

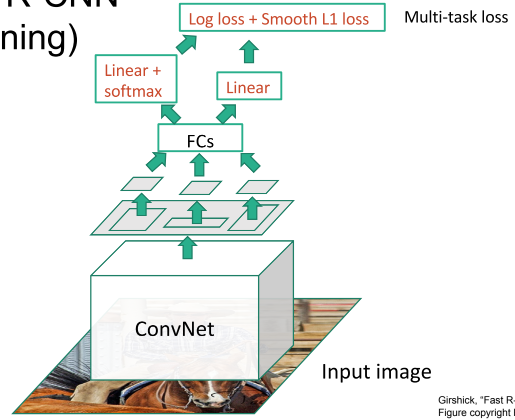
Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 74 May 10, 2017

Fast R-CNN



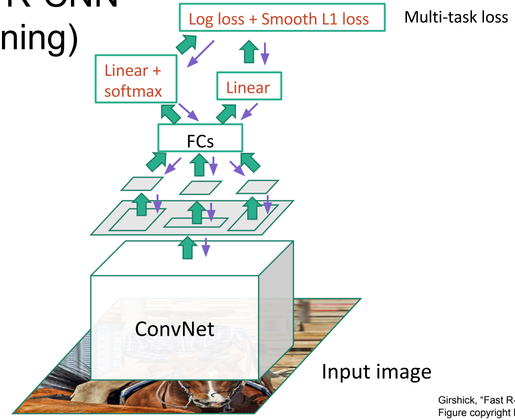
Girshick, "Fast R-CNN", ICCV 2015.

Fast R-CNN (Training)



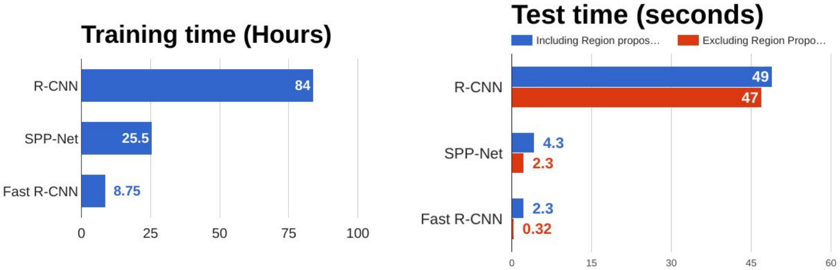
Girshick, "Fast R-CNN", ICCV 2015.

Fast R-CNN (Training)



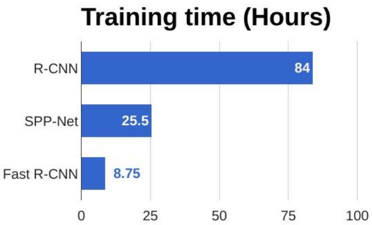
Girshick, "Fast R-CNN", ICCV 2015.

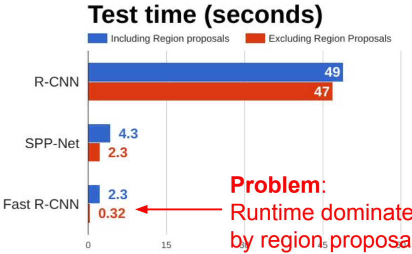
R-CNN vs SPP vs Fast R-CNN



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

R-CNN vs SPP vs Fast R-CNN





Problem: Runtime dominated by region proposals!

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

Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 80 May 10, 2017

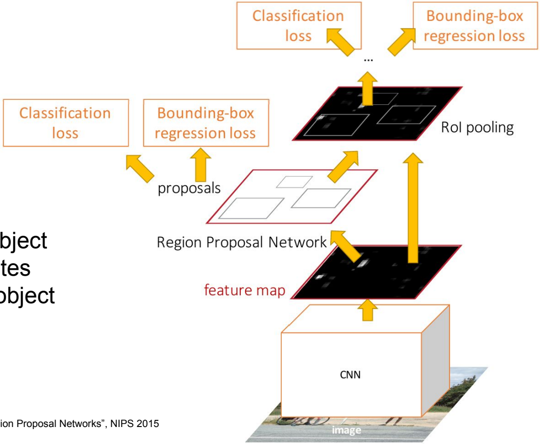
Faster R-CNN:

Make CNN do proposals!

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

Jointly train with 4 losses:

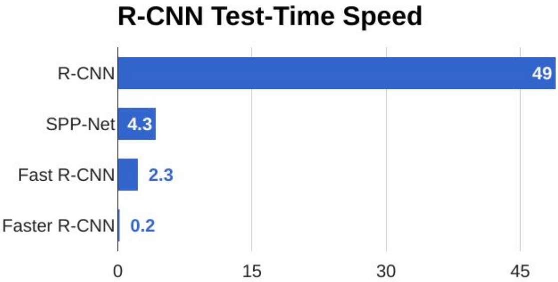
1. RPN classify object / not object
2. RPN regress box coordinates
3. Final classification score (object classes)
4. Final box coordinates



Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

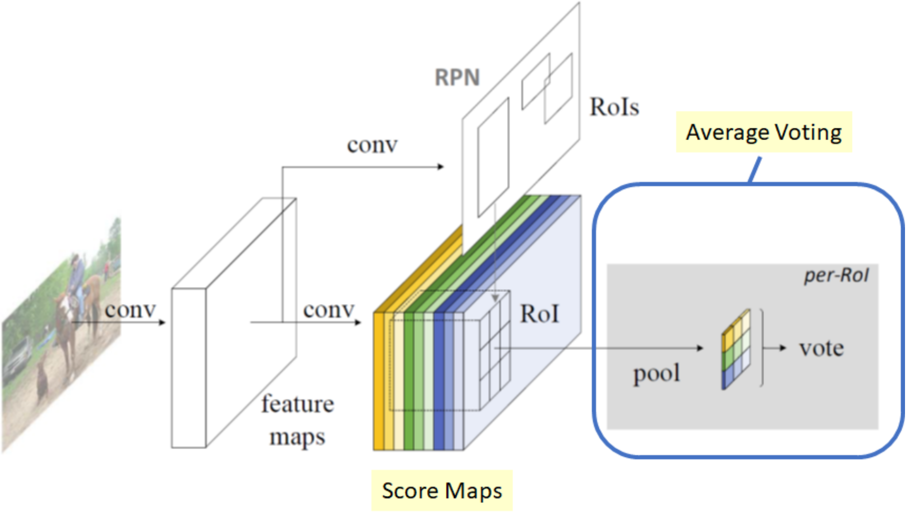
Fast**er** R-CNN:

Make CNN do proposals!



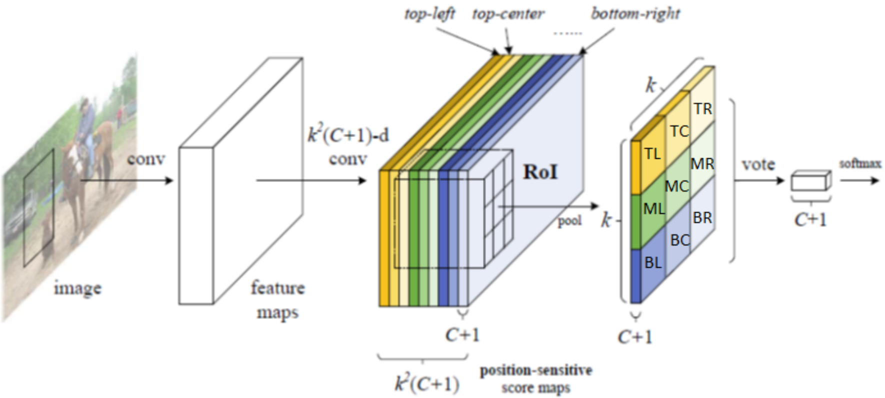
Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 82 May 10, 2017

Region-based fully convolutional network (R-FCN)

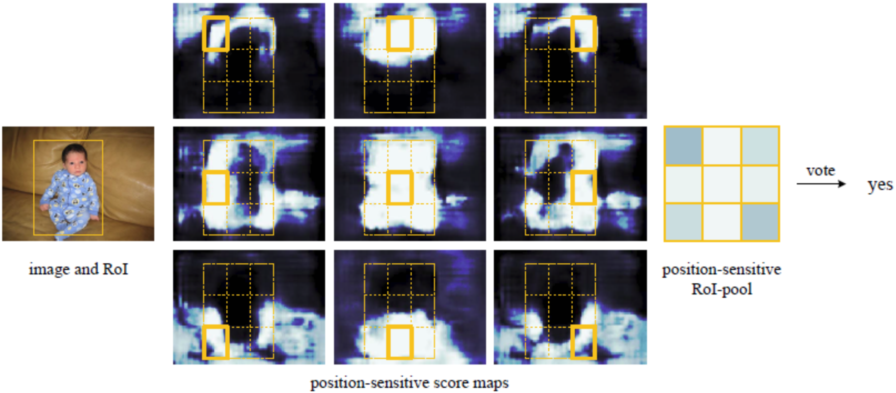


Fully connected layers are replaced by average pooling

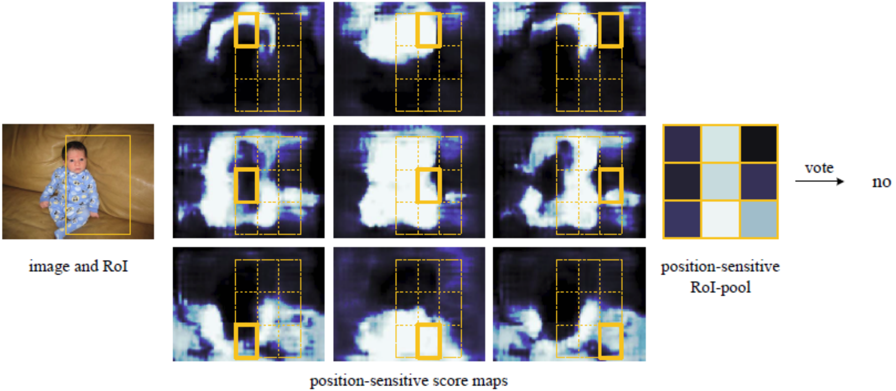
Region-based fully convolutional network (R-FCN)



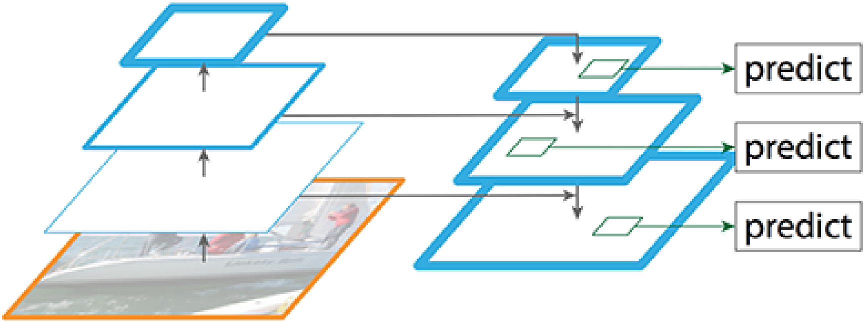
Region-based fully convolutional network (R-FCN)



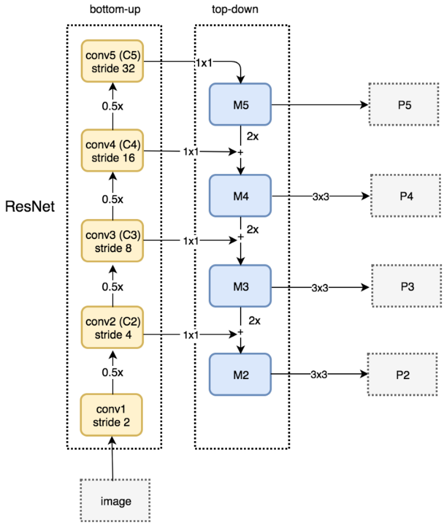
Region-based fully convolutional network (R-FCN)



Feature pyramid network (FPN)

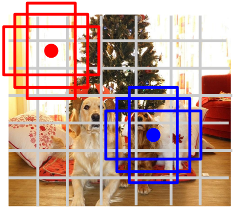


Feature pyramid network (FPN)



Detection without Proposals: YOLO / SSD

Input image
3 x H x W



Divide image into grid
7 x 7

Image a set of **base boxes** centered at each grid cell
Here B = 3

Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers:
  (dx, dy, dh, dw, confidence)
- Predict scores for each of C classes (including background as a class)

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

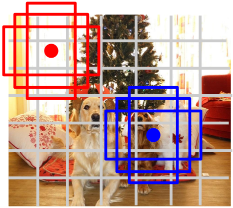
Redmon et al, "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016
Liu et al, "SSD: Single-Shot MultiBox Detector", ECCV 2016

Fei-Fei Li & Justin Johnson & Serena Yeung   Lecture 11 - 83   May 10, 2017

Detection without Proposals: YOLO / SSD

Go from input image to tensor of scores with one big convolutional network!

Input image
3 x H x W



Divide image into grid
7 x 7

Image a set of **base boxes** centered at each grid cell
Here B = 3

Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers: (dx, dy, dh, dw, confidence)
- Predict scores for each of C classes (including background as a class)

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

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

Fei-Fei Li & Justin Johnson & Serena Yeung Lecture 11 - 84 May 10, 2017

Focal loss

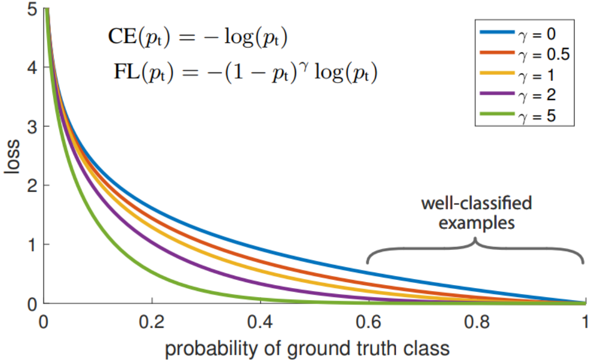


Figure 1. We propose a novel loss we term the *Focal Loss* that adds a factor $(1 - p_t)^{\gamma}$ to the standard cross entropy criterion. Setting $\gamma > 0$ reduces the relative loss for well-classified examples ($p_t > .5$), putting more focus on hard, misclassified examples. As our experiments will demonstrate, the proposed focal loss enables training highly accurate dense object detectors in the presence of vast numbers of easy background examples.

RetinaNet

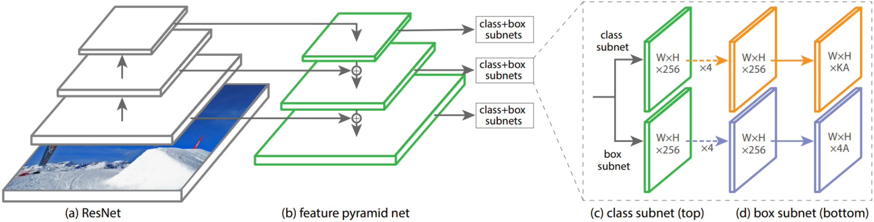


Figure 3. The one-stage **RetinaNet** network architecture uses a Feature Pyramid Network (FPN) [20] backbone on top of a feedforward ResNet architecture [16] (a) to generate a rich, multi-scale convolutional feature pyramid (b). To this backbone RetinaNet attaches two subnetworks, one for classifying anchor boxes (c) and one for regressing from anchor boxes to ground-truth object boxes (d). The network design is intentionally simple, which enables this work to focus on a novel focal loss function that eliminates the accuracy gap between our one-stage detector and state-of-the-art two-stage detectors like Faster R-CNN with FPN [20] while running at faster speeds.

RetinaNet

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [16]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [20]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [17]	Inception-ResNet-v2 [34]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [32]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [27]	DarkNet-19 [27]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [22, 9]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [9]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet (ours)	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet (ours)	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2

Object Detection: Lots of variables ...

Base Network
VGG16
ResNet-101
Inception V2
Inception V3
Inception ResNet
MobileNet

Object Detection architecture
Faster R-CNN
R-FCN
SSD

Image Size
 # Region Proposals
…

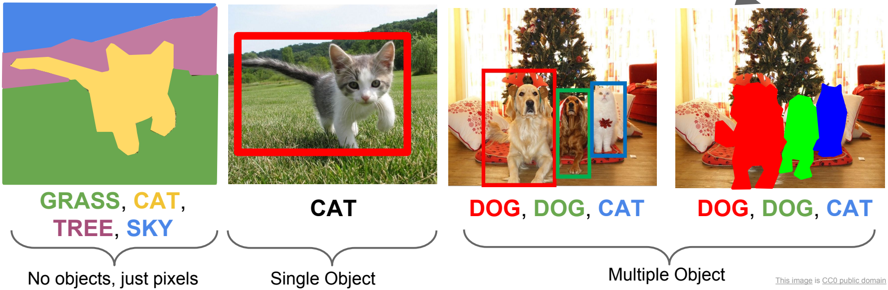
Takeaways
Faster R-CNN is slower but more accurate

SSD is much faster but not as accurate

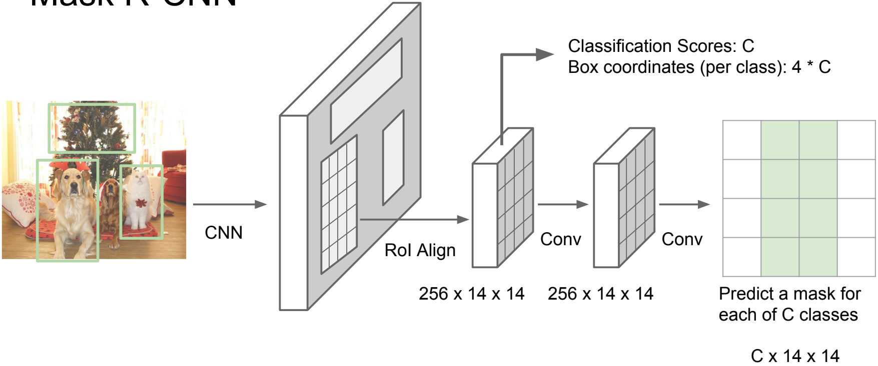
Huang et al, "Speed/accuracy trade-offs for modern convolutional object detectors", CVPR 2017

R-FCN: Dai et al, "R-FCN: Object Detection via Region-based Fully Convolutional Networks", NIPS 2016
Inception-V2: Ioffe and Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", ICML 2015
Inception V3: Szegedy et al, "Rethinking the Inception Architecture for Computer Vision", arXiv 2016
Inception ResNet: Szegedy et al, "Inception-V4, Inception-ResNet and the Impact of Residual Connections on Learning", arXiv 2016
MobileNet: Howard et al, "Efficient Convolutional Neural Networks for Mobile Vision Applications", arXiv 2017

Instance Segmentation

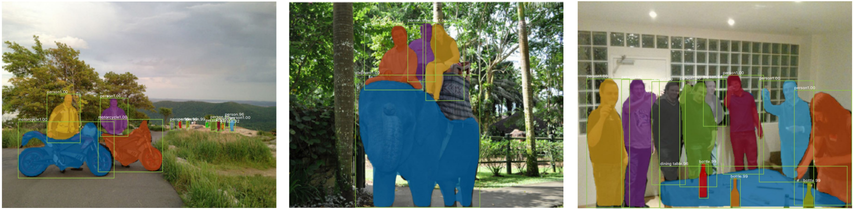


Mask R-CNN



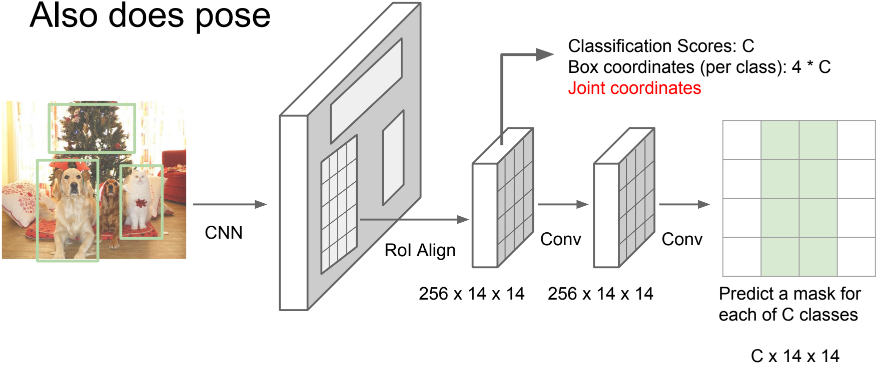
He et al, "Mask R-CNN", arXiv 2017

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

Fei-Fei Li & Justin Johnson & Serena Yeung | Lecture 11 - 92 | May 10, 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.

Fei-Fei Li & Justin Johnson & Serena Yeung   Lecture 11 -   93   May 10, 2017

Brain score

Brain-Score   Leaderboard   About   Compare   Participate

Brain Rank▼	Model	Brain-Score	V1: Your data here!	V2: Your data here!	V4: dicarlo.Majaj2015	IT: dicarlo.Majaj2015	Behavior: dicarlo.Raja...2018	Classification: Imagenet2012
	Ceiling	.729			.892	.817	.479	100.0
1	densenet-169 *Huang et al., 2016*	.549			.663	.606	.378	75.9
2	cornet_s *Kubilius et al., 2018*	.544			.650	.600	.382	74.7
3	resnet-101_v2 *He et al., 2015*	.542			.653	.585	.389	77.4
4	densenet-201 *Huang et al., 2016*	.541			.655	.601	.368	77.2
5	densenet-121 *Huang et al., 2016*	.541			.657	.597	.369	74.5