Introduction to Computer Vision

Lecture 11 - Object Detection and Instance Segmentation

Prof. He Wang

Logistics

- Assignment 4 (Point Cloud Learning and Object Detection)
  - will release on 5/20
  - Due on 6/3 11:59PM

Video Analysis - Cont’d

Slides are borrowed from Stanford CS231N

Early Fusion vs. Late Fusion vs. 3DCNN

	Layer	Size (C x T x H x W)	Receptive Field (T x H x W)
Late Fusion	Input	3 x 20 x 64 x 64	
	Conv2D(3x3, 3->12)	12 x 20 x 64 x 64	1 x 3 x 3
	Pool2D(4x4)	12 x 20 x 16 x 16	1 x 6 x 6
	Conv2D(3x3, 12->24)	24 x 20 x 16 x 16	1 x 14 x 14
	GlobalAvgPool	24 x 1 x 1 x 1	20 x 64 x 64
Early Fusion	Input	3 x 20 x 64 x 64	
	Conv2D(3x3, 3*20->12)	12 x 64 x 64	20 x 3 x 3
	Pool2D(4x4)	12 x 16 x 16	20 x 6 x 6
	Conv2D(3x3, 12->24)	24 x 16 x 16	20 x 14 x 14
	GlobalAvgPool	24 x 1 x 1	20 x 64 x 64
3D CNN	Input	3 x 20 x 64 x 64	
	Conv3D(3x3x3, 3->12)	12 x 20 x 64 x 64	3 x 3 x 3
	Pool3D(4x4x4)	12 x 5 x 16 x 16	6 x 6 x 6
	Conv3D(3x3x3, 12->24)	24 x 5 x 16 x 16	14 x 14 x 14
	GlobalAvgPool	24 x 1 x 1	20 x 64 x 64

What is the difference?

Late Fusion: Build slowly in space, All-at-once in time at end

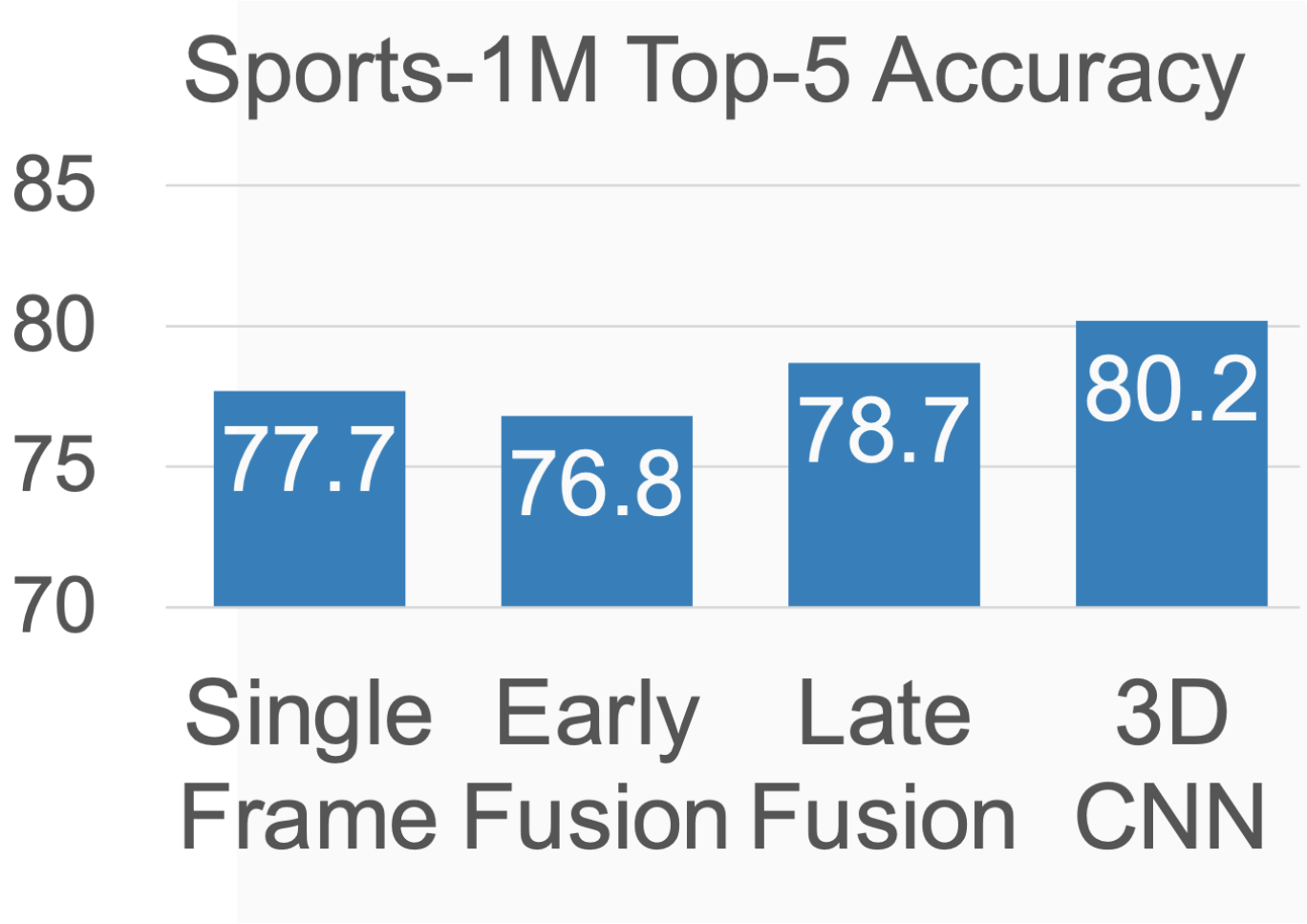
Early Fusion: Build slowly in space, All-at-once in time at start

3D CNN: Build slowly in space, Build slowly in time "Slow Fusion"

(Small example architectures, in practice much bigger)

Slide credit: Justin Johnson

Early Fusion vs. Late Fusion vs. 3DCNN



Single Frame model works well – always try this first!

3D CNNs have improved a lot since 2014!

C3D: The VGG of 3D CNNs

3D CNN that uses all 3x3x3 conv and 2x2x2 pooling (except Pool1 which is 1x2x2)

Released model pretrained on Sports-1M: Many people used this as a video feature extractor

Layer	Size
Input	3 x 16 x 112 x 112
Conv1 (3x3x3)	64 x 16 x 112 x 112
Pool1 (1x2x2)	64 x 16 x 56 x 56
Conv2 (3x3x3)	128 x 16 x 56 x 56
Pool2 (2x2x2)	128 x 8 x 28 x 28
Conv3a (3x3x3)	256 x 8 x 28 x 28
Conv3b (3x3x3)	256 x 8 x 28 x 28
Pool3 (2x2x2)	256 x 4 x 14 x 14
Conv4a (3x3x3)	512 x 4 x 14 x 14
Conv4b (3x3x3)	512 x 4 x 14 x 14
Pool4 (2x2x2)	512 x 2 x 7 x 7
Conv5a (3x3x3)	512 x 2 x 7 x 7
Conv5b (3x3x3)	512 x 2 x 7 x 7
Pool5	512 x 1 x 3 x 3
FC6	4096
FC7	4096
FC8	C

Tran et al, “Learning Spatiotemporal Features with 3D Convolutional Networks”, ICCV 2015

C3D: The VGG of 3D CNNs

3D CNN that uses all 3x3x3 conv and 2x2x2 pooling (except Pool1 which is 1x2x2)

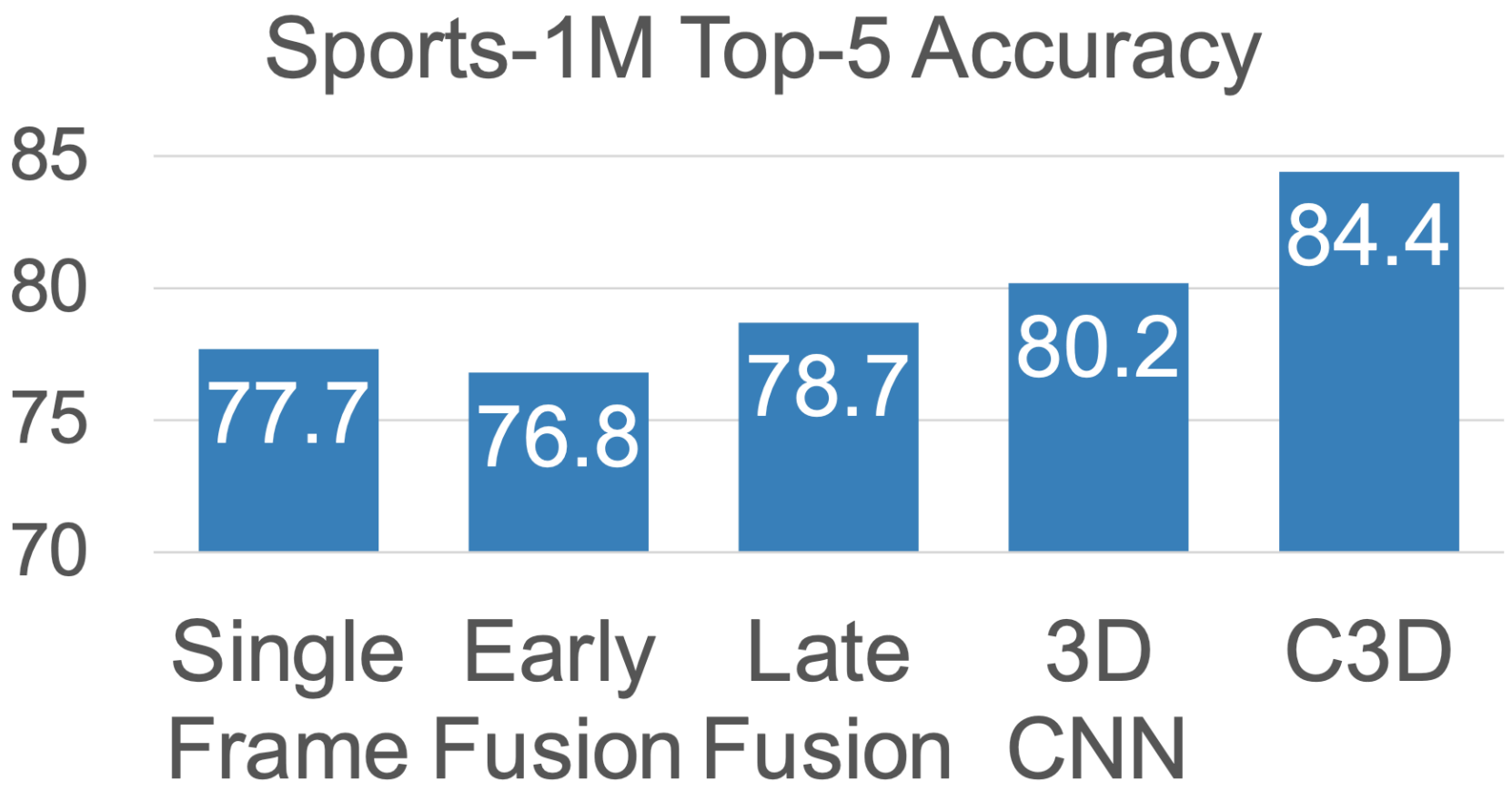
Released model pretrained on Sports-1M: Many people used this as a video feature extractor

Problem: 3x3x3 conv is very expensive!
AlexNet: 0.7 GFLOP
VGG-16: 13.6 GFLOP
C3D: **39.5 GFLOP (2.9x VGG!)**

Layer	Size	MFLOPs
Input	3 x 16 x 112 x 112	
Conv1 (3x3x3)	64 x 16 x 112 x 112	1.04
Pool1 (1x2x2)	64 x 16 x 56 x 56	
Conv2 (3x3x3)	128 x 16 x 56 x 56	11.10
Pool2 (2x2x2)	128 x 8 x 28 x 28	
Conv3a (3x3x3)	256 x 8 x 28 x 28	5.55
Conv3b (3x3x3)	256 x 8 x 28 x 28	11.10
Pool3 (2x2x2)	256 x 4 x 14 x 14	
Conv4a (3x3x3)	512 x 4 x 14 x 14	2.77
Conv4b (3x3x3)	512 x 4 x 14 x 14	5.55
Pool4 (2x2x2)	512 x 2 x 7 x 7	
Conv5a (3x3x3)	512 x 2 x 7 x 7	0.69
Conv5b (3x3x3)	512 x 2 x 7 x 7	0.69
Pool5	512 x 1 x 3 x 3	
FC6	4096	0.51
FC7	4096	0.45
FC8	C	0.05

Tran et al, “Learning Spatiotemporal Features with 3D Convolutional Networks”, ICCV 2015

Early Fusion vs. Late Fusion vs. 3DCNN

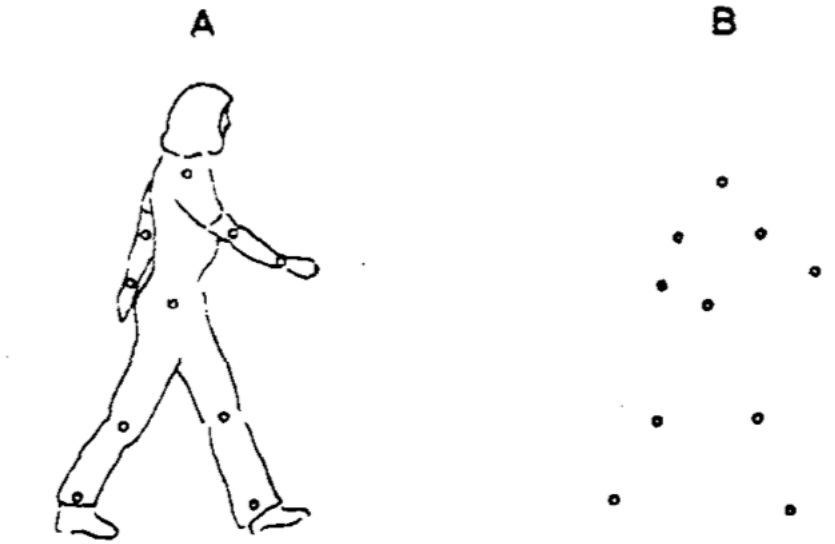


Recognizing Actions from Motion

Perception & Psychophysics
1973, Vol. 14, No. 2, 201-211

Visual perception of biological motion and a model for its analysis*

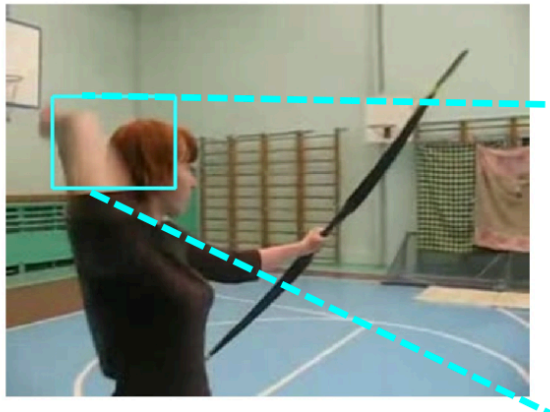
GUNNAR JOHANSSON
University of Uppsala, S:t Larsgatan 2, S-752 20 Uppsala, Sweden



Video link: https://www.youtube.com/watch?v=rEVB6kW9p6k

Measuring Motion

Image at frame t



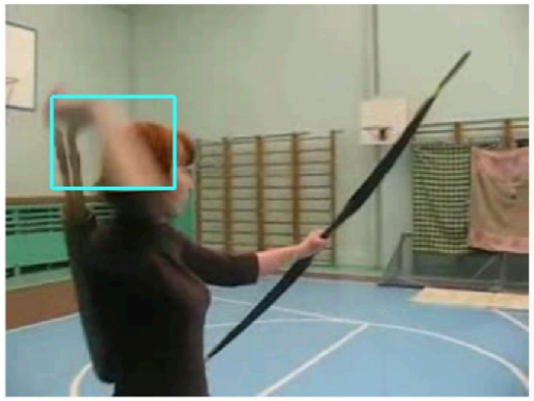
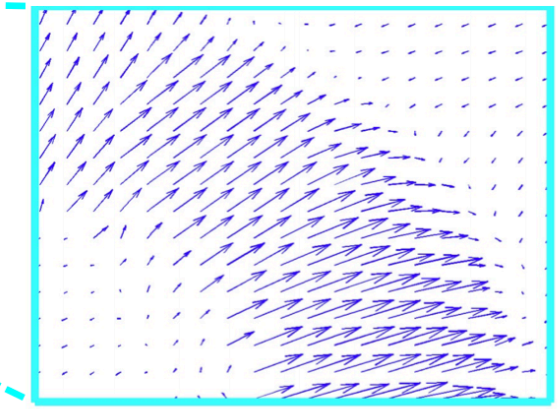


Image at frame t+1

Optical flow gives a displacement field F between images I_t and I_{t+1}



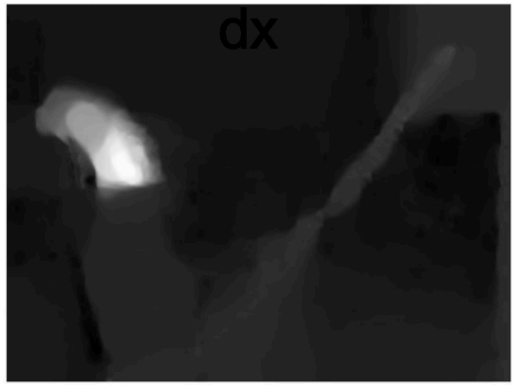
Tells where each pixel will move in the next frame:

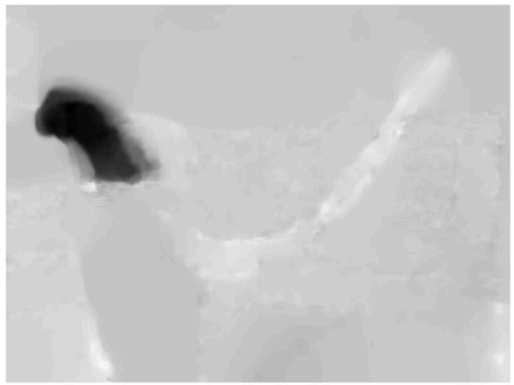
$F(x, y) = (dx, dy)$

$I_{t+1}(x+dx, y+dy) = I_t(x, y)$

Optical Flow highlights **local motion**

Horizontal flow



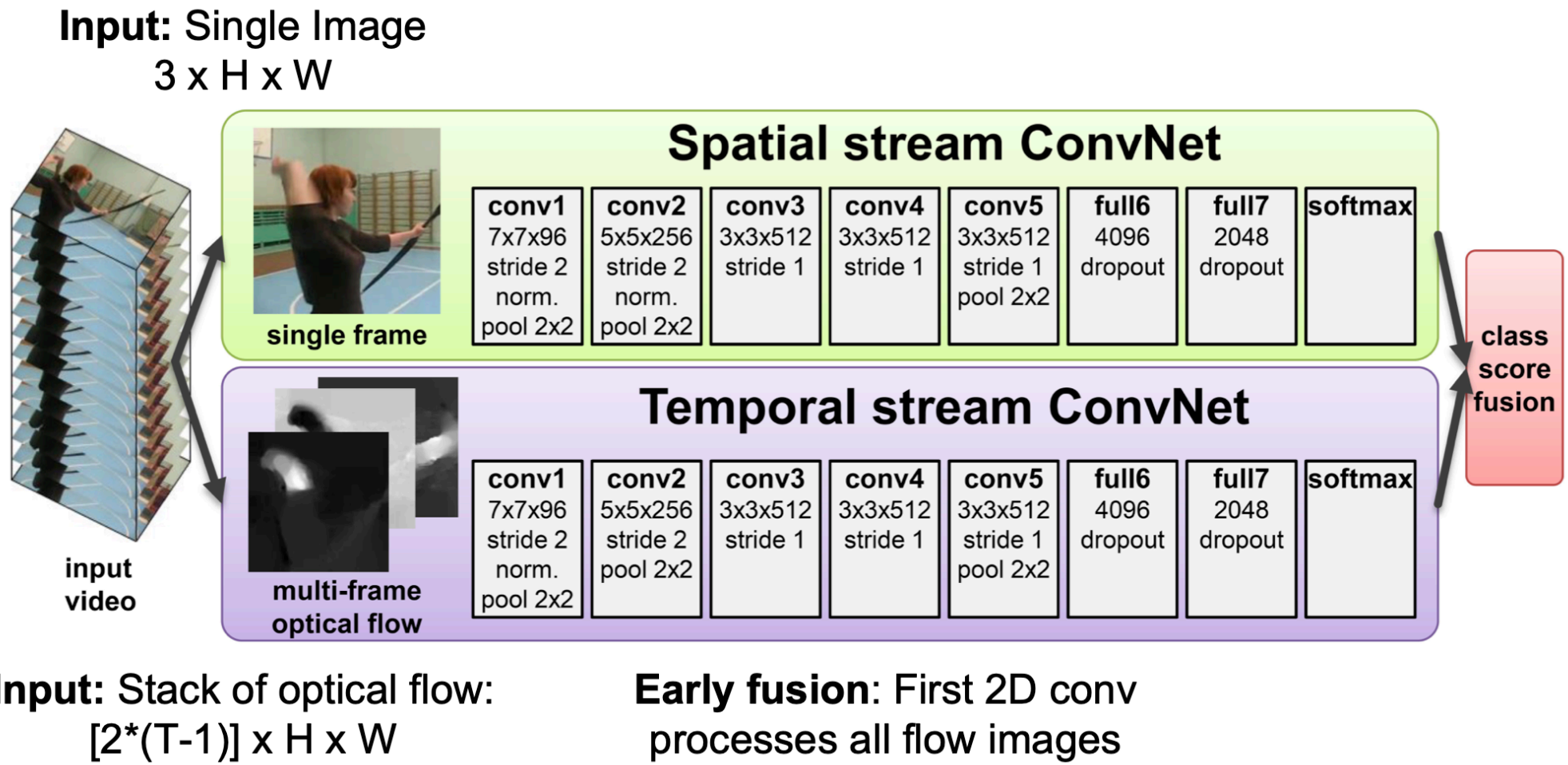


Vertical Flow dy

Slide credit: Justin Johnson

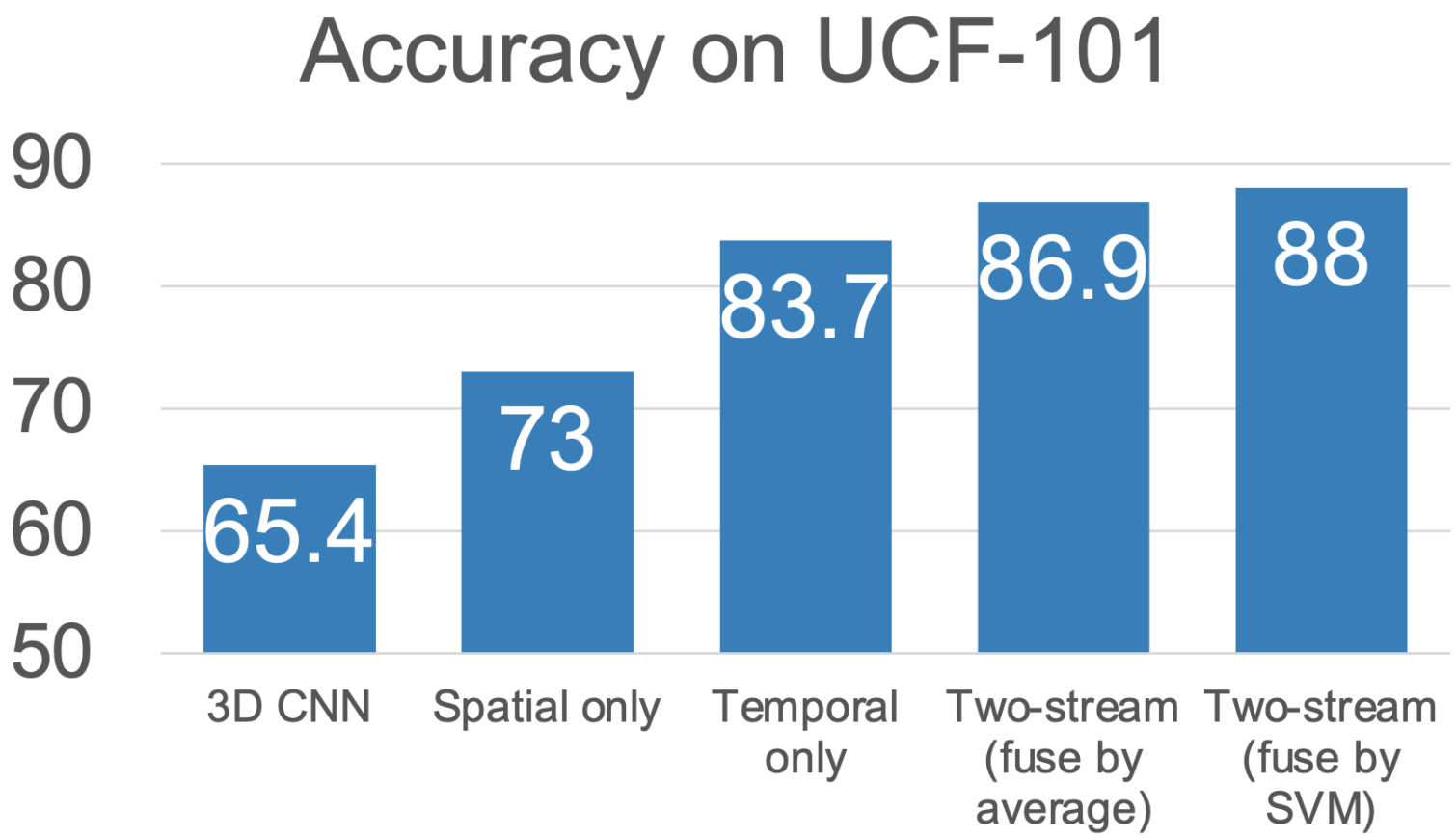
Simonyan and Zisserman, "Two-stream convolutional networks for action recognition in videos", NeurIPS 2014

Use Both Motion and Appearance: Two-Stream Fusion Networks



Simonyan and Zisserman, "Two-stream convolutional networks for action recognition in videos", NeurIPS 2014

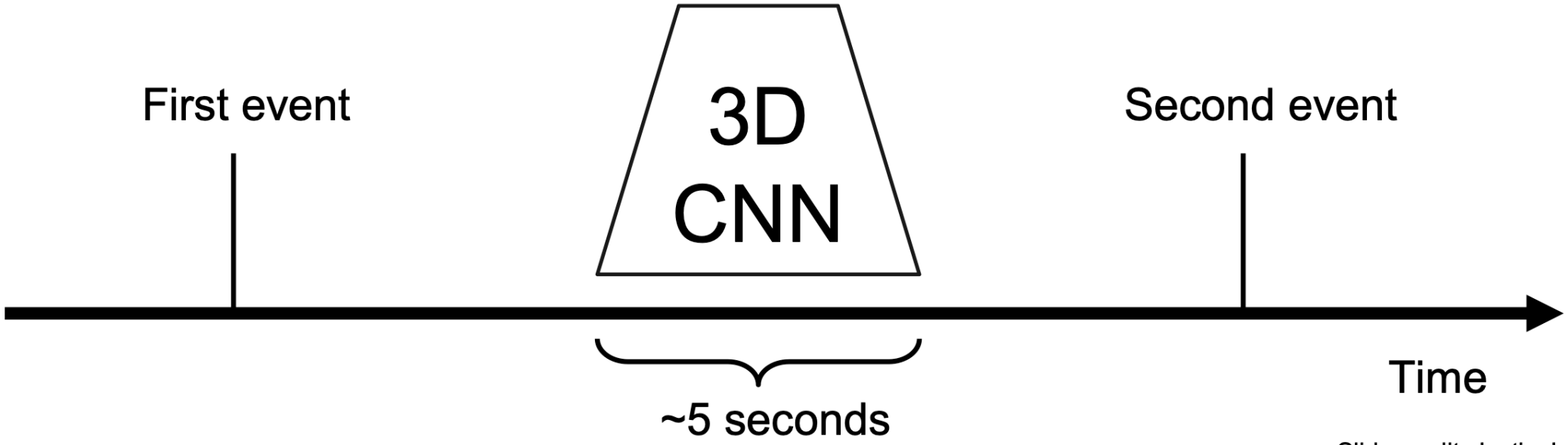
Use Both Motion and Appearance: Two-Stream Fusion Networks



Simonyan and Zisserman, “Two-stream convolutional networks for action recognition in videos”, NeurIPS 2014

Modeling Long-Term Temporal Structure

So far all our temporal CNNs only model local motion between frames in very short clips of ~2-5 seconds. What about long-term structure?

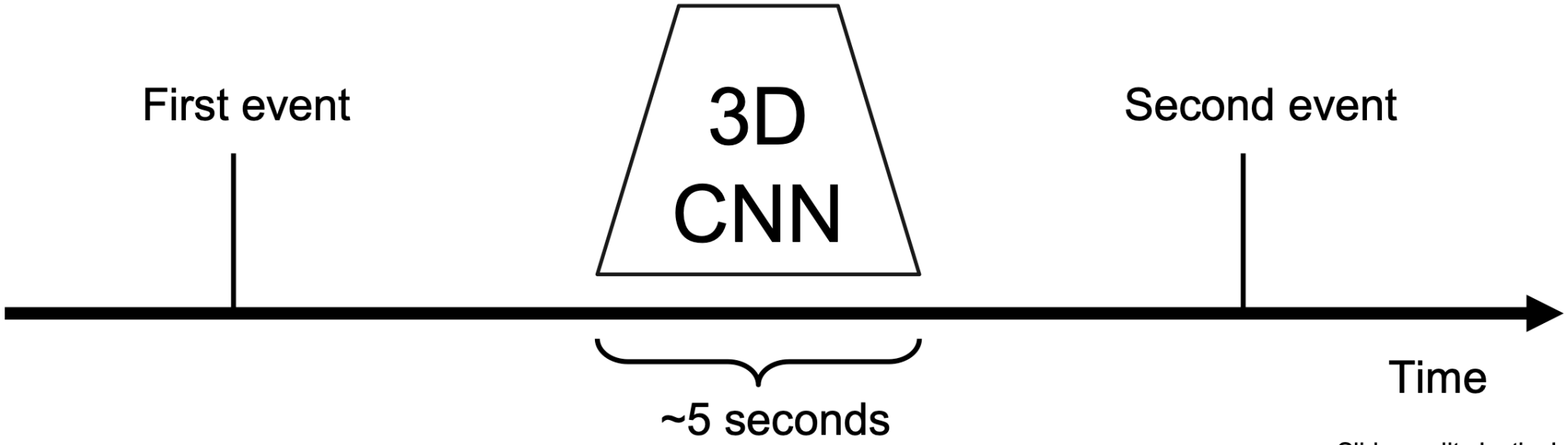


Slide credit: Justin Johnson

Modeling Long-Term Temporal Structure

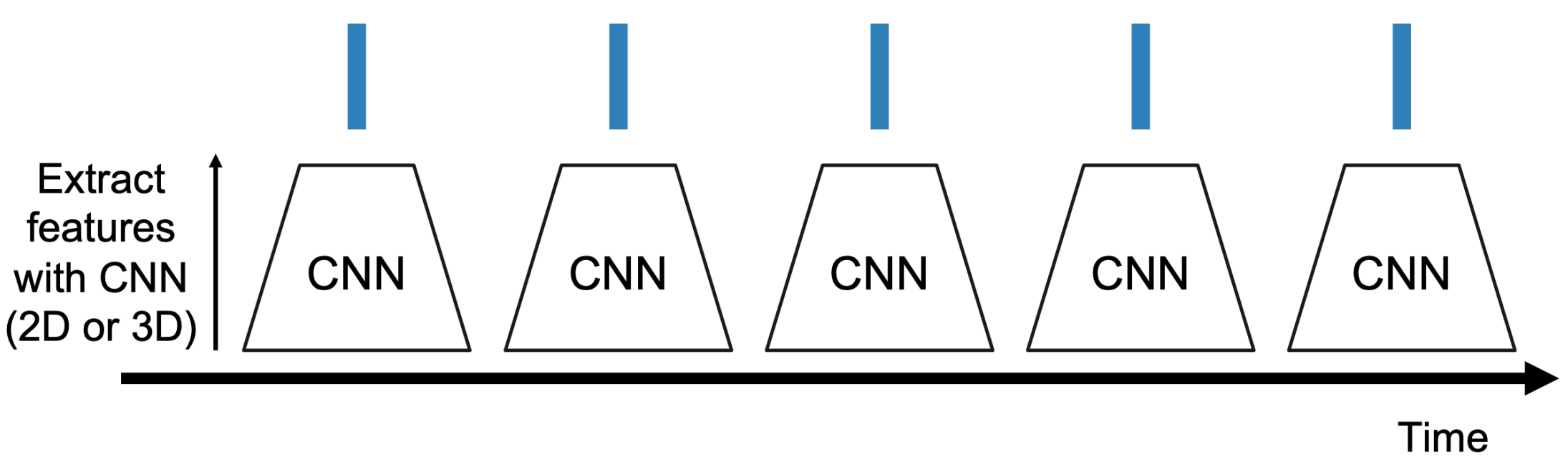
So far all our temporal CNNs only model local motion between frames in very short clips of ~2-5 seconds. What about long-term structure?

We know how to handle sequences! How about recurrent networks?



Slide credit: Justin Johnson

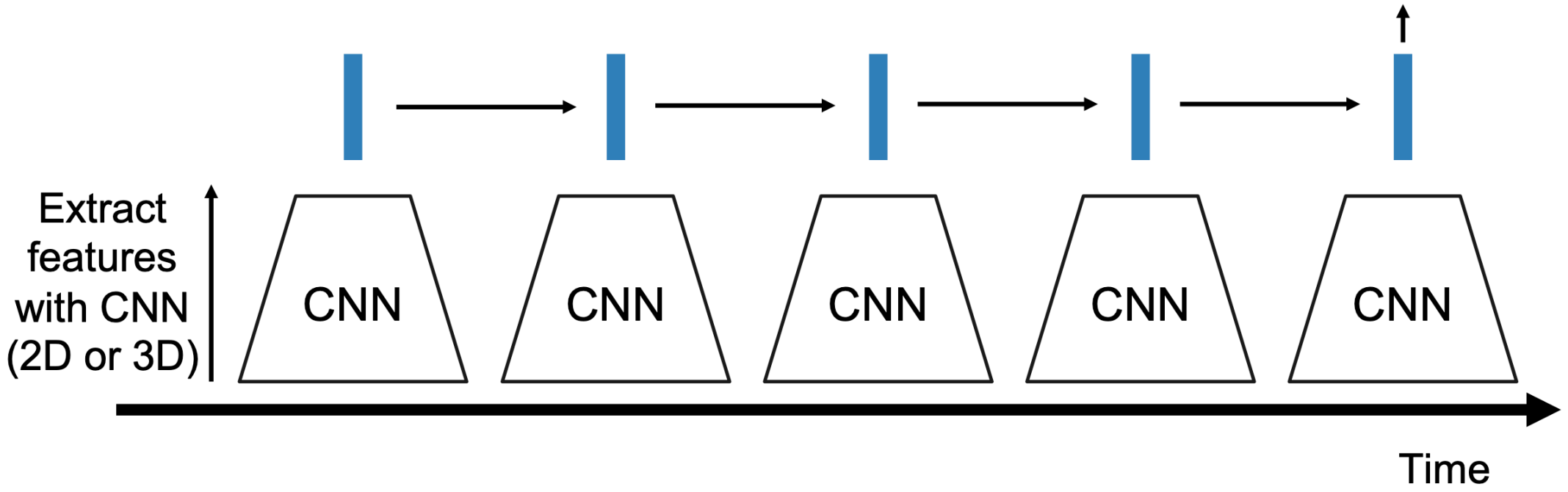
Modeling Long-Term Temporal Structure



Slide credit: Justin Johnson

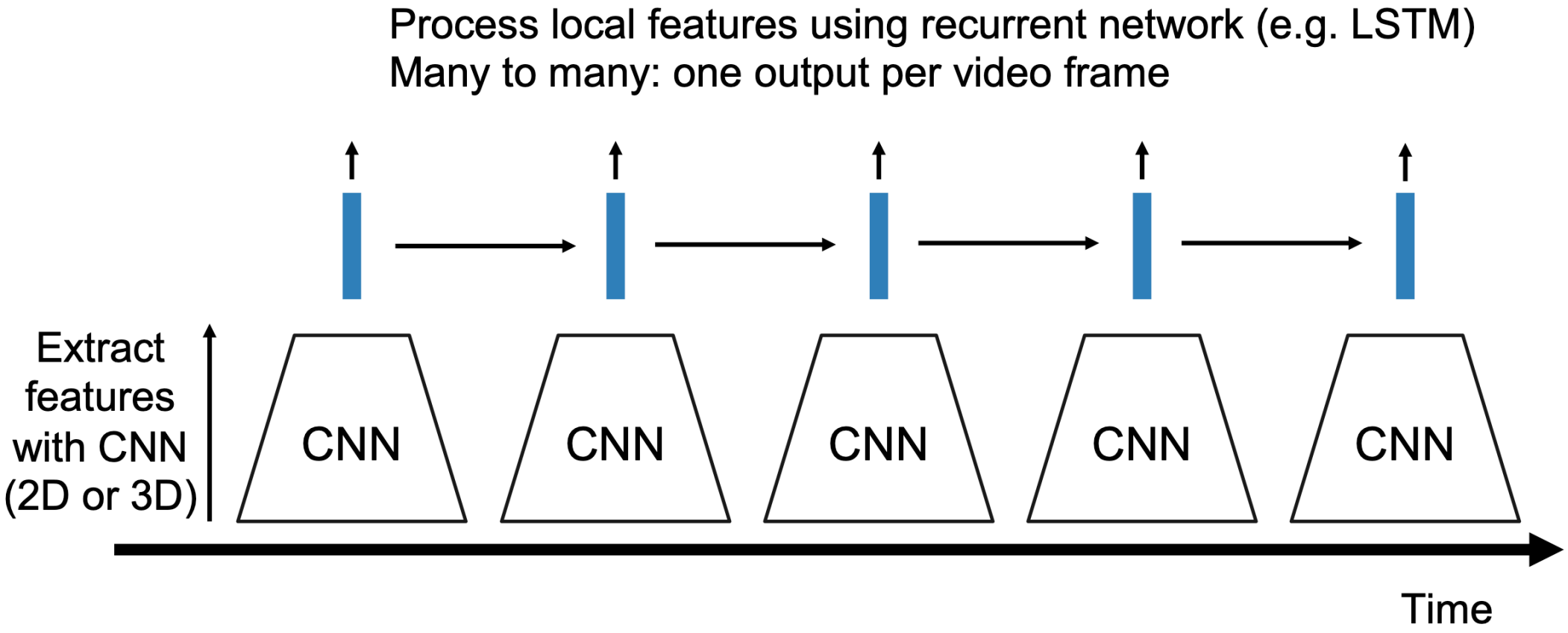
Modeling Long-Term Temporal Structure

Process local features using recurrent network (e.g. LSTM)
Many to one: One output at end of video



Slide credit: Justin Johnson

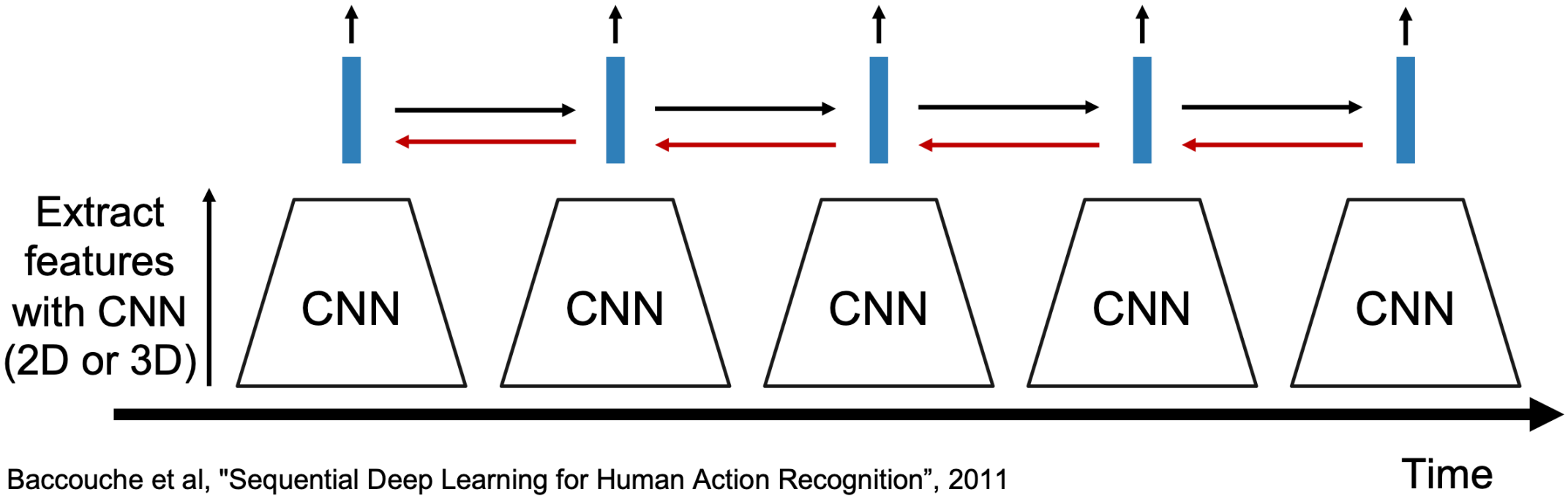
Modeling Long-Term Temporal Structure



Slide credit: Justin Johnson

Modeling Long-Term Temporal Structure

Sometimes don't backprop to CNN to save memory; pretrain and use it as a feature extractor

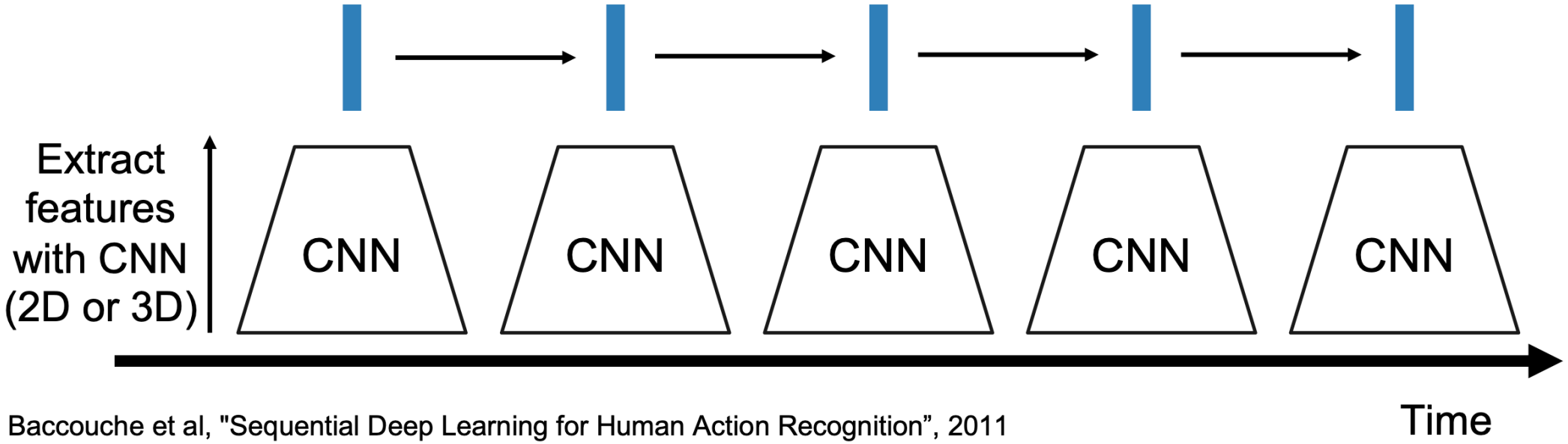


Baccouche et al, "Sequential Deep Learning for Human Action Recognition", 2011
Donahue et al, "Long-term recurrent convolutional networks for visual recognition and description", CVPR 2015

Slide credit: Justin Johnson

Modeling Long-Term Temporal Structure

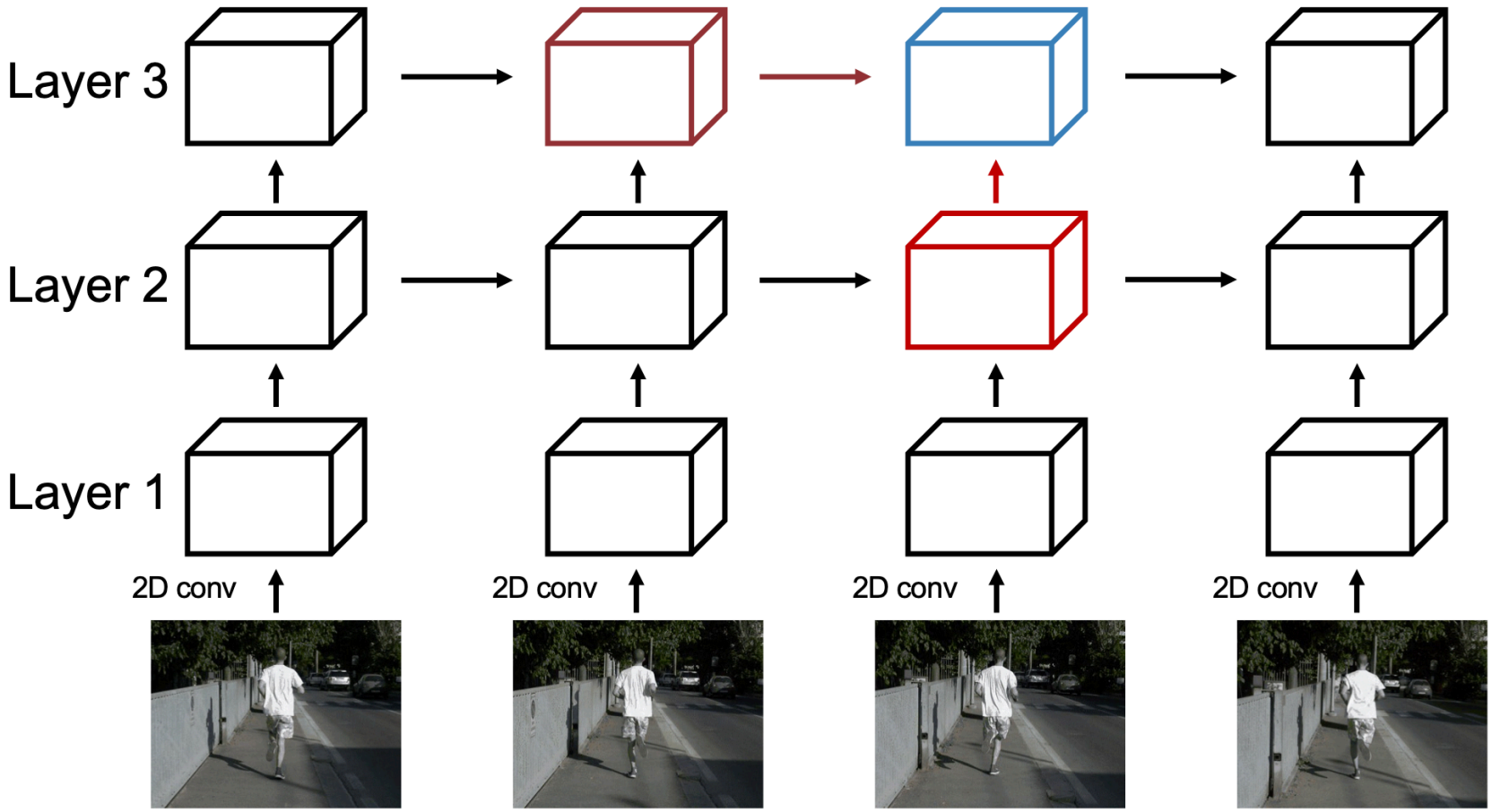
Inside CNN: Each value is a function of a fixed temporal window (local temporal structure)
Inside RNN: Each vector is a function of all previous vectors (global temporal structure)
Can we merge both approaches?



Baccouche et al, "Sequential Deep Learning for Human Action Recognition", 2011
Donahue et al, "Long-term recurrent convolutional networks for visual recognition and description", CVPR 2015

Slide credit: Justin Johnson

Recurrent Convolutional Network



Entire network uses 2D feature maps: C x H x W

Each depends on two inputs:
1. Same layer, previous timestep
2. Prev layer, same timestep

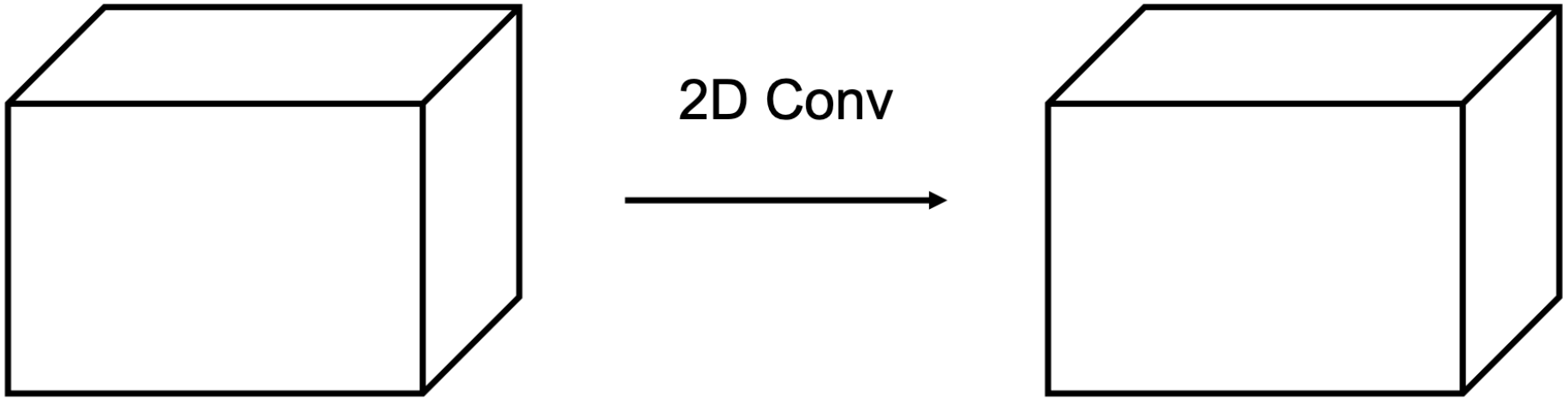
Use different weight at each layer, share weights across time

Ballas et al, "Delving Deeper into Convolutional Networks for Learning Video Representations", ICLR 2016

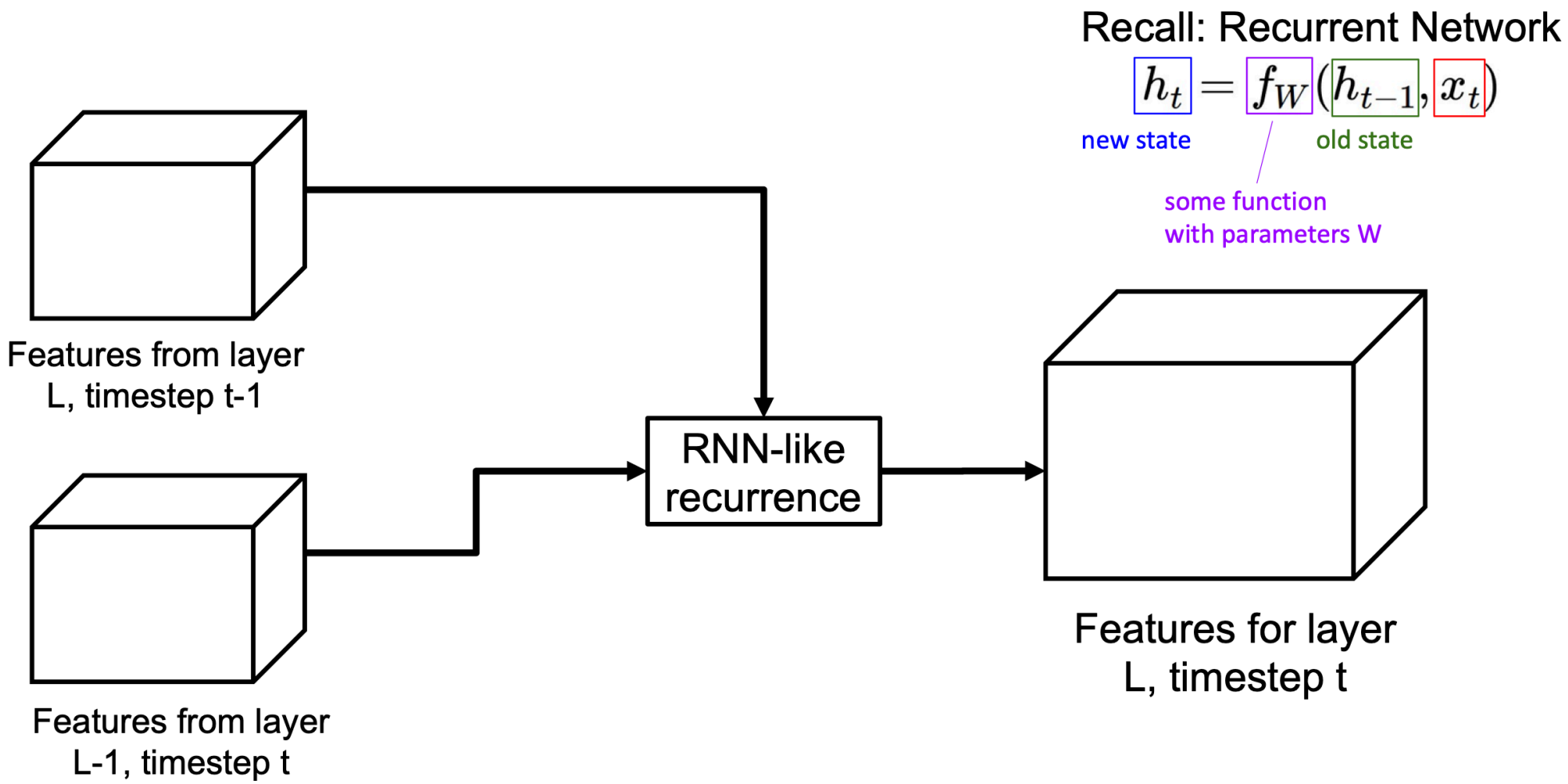
Slide credit: Justin Johnson

Recurrent Convolutional Network

Normal 2D CNN:



Recurrent Convolutional Network



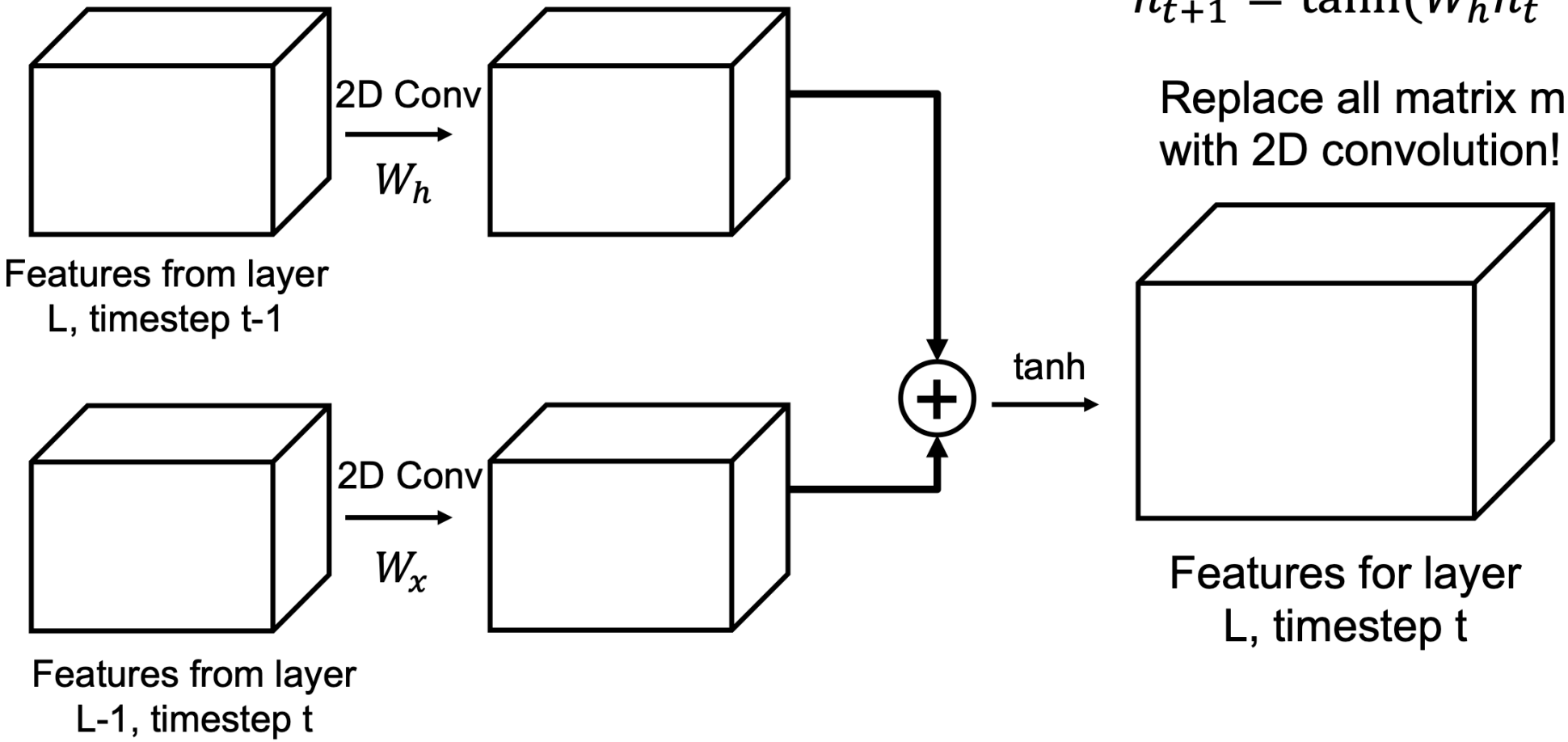
Ballas et al, "Delving Deeper into Convolutional Networks for Learning Video Representations", ICLR 2016

Recurrent Convolutional Network

Recall: Vanilla RNN

$$h_{t+1} = \tanh(W_h h_t + W_x x)$$

Replace all matrix multiply with 2D convolution!



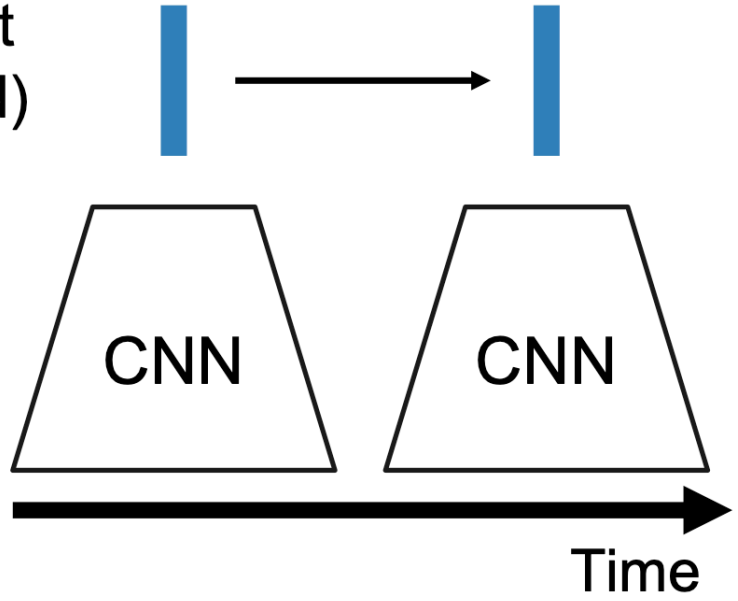
Ballas et al, "Delving Deeper into Convolutional Networks for Learning Video Representations", ICLR 2016

Modeling Long-Term Temporal Structure

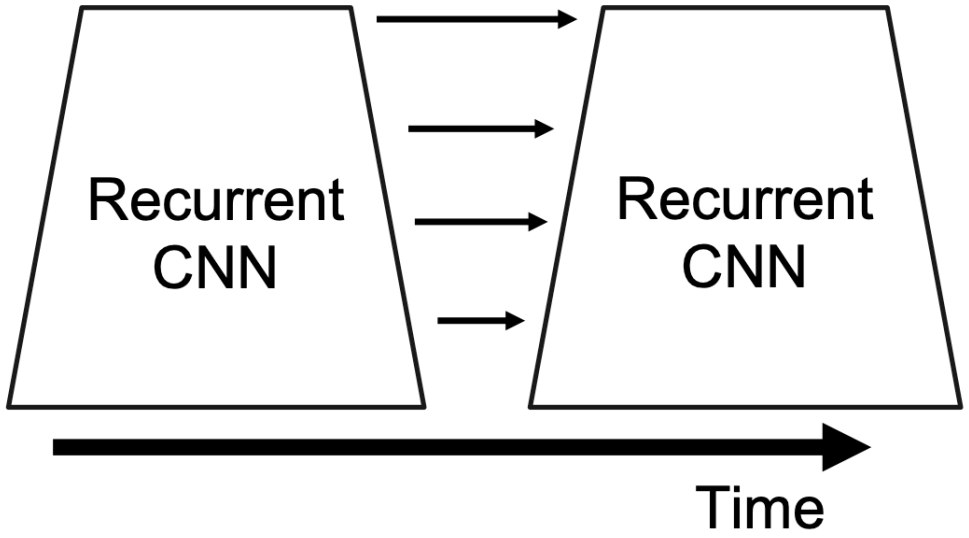
Problem: RNNs are slow for long sequences (can't be parallelized)

RNN: Infinite temporal extent (fully-connected)

CNN: finite temporal extent (convolutional)



Recurrent CNN: Infinite temporal extent (convolutional)



Baccouche et al, "Sequential Deep Learning for Human Action Recognition", 2011
Donahue et al, "Long-term recurrent convolutional networks for visual recognition and description", CVPR 2015

Ballas et al, "Delving Deeper into Convolutional Networks for Learning Video Representations", ICLR 2016

Slide credit: Justin Johnson

A lot of Variations

- ConvLSTM
- ConvGRU

- Direction:
  - Attention
  - Transformer
  - …

Video Tasks

So far: Classify short clips

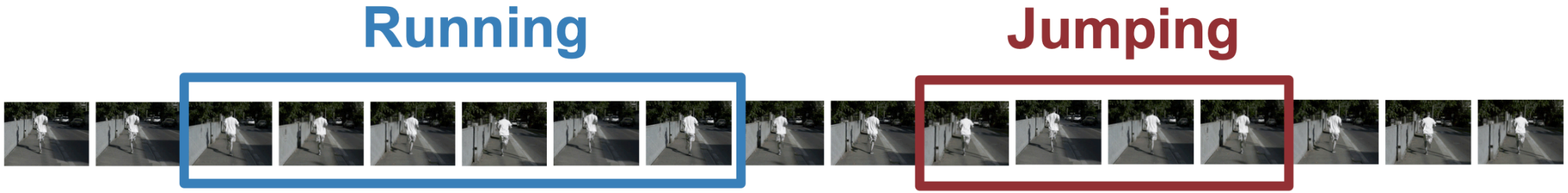
Videos: Recognize **actions**

Swimming
Running
Jumping
Eating
Standing

Video Tasks

Temporal Action Localization

Given a long untrimmed video sequence, identify frames corresponding to different actions



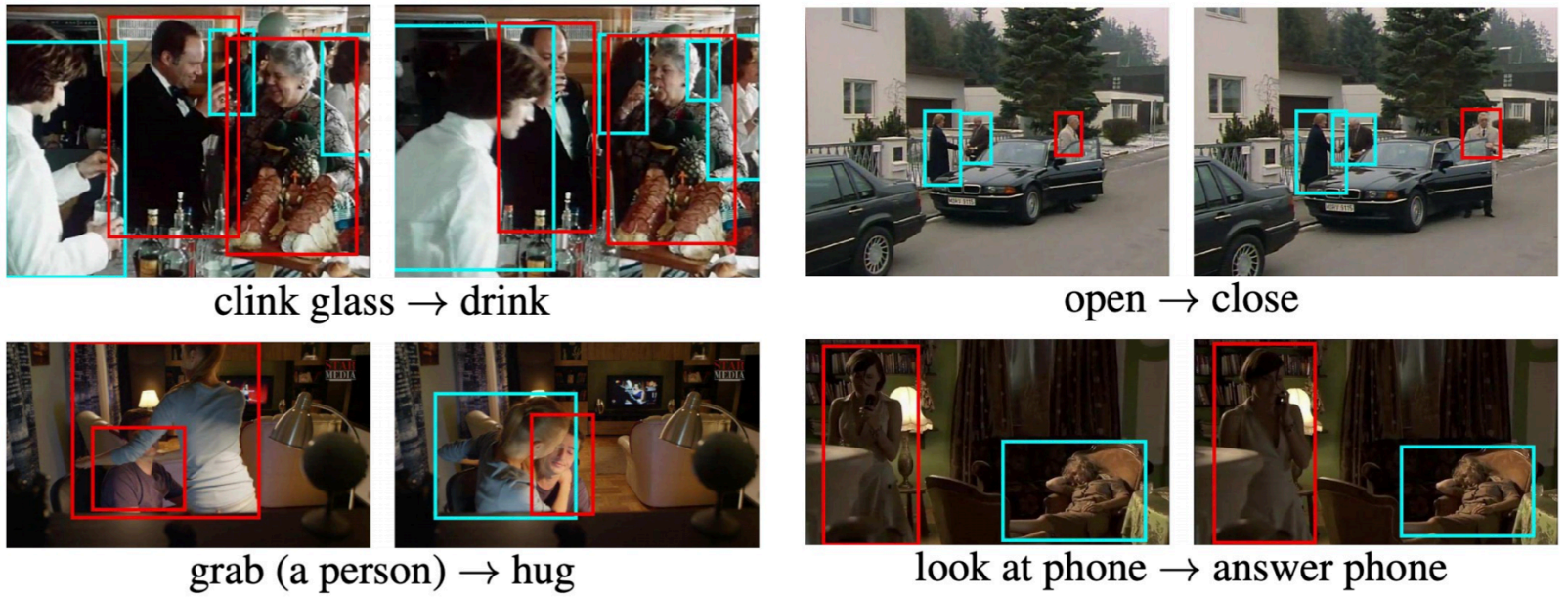
Can use architecture similar to Faster R-CNN:
first generate **temporal proposals** then **classify**

Chao et al, ” Rethinking the Faster R-CNN Architecture for Temporal Action Localization”, CVPR 2018

Video Tasks

Spatio-Temporal Detection

Given a long untrimmed video, detect all the people in both space and time and classify the activities they are performing. Some examples from AVA Dataset:



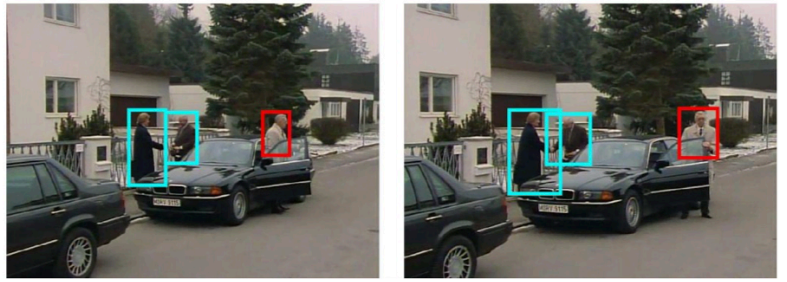
Gu et al, "AVA: A Video Dataset of Spatio-temporally Localized Atomic Visual Actions", CVPR 2018

Video Tasks

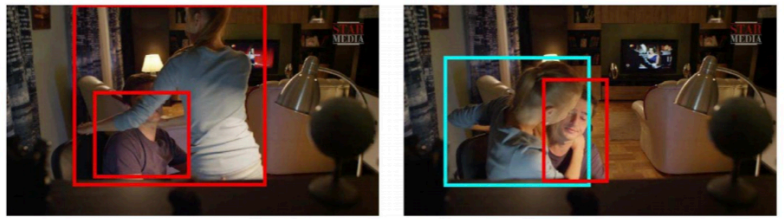
Spatio-Temporal Detection

Given a long untrimmed video, detect all the people in both space and time and classify the activities they are performing. Some examples from AVA Dataset:

clink glass → drink



open → close



grab (a person) → hug

look at phone → answer phone

Gu et al, "AVA: A Video Dataset of Spatio-temporally Localized Atomic Visual Actions", CVPR 2018

Summary

- 3D-CNN (2D + time)
- Two-Stream Network (appearance + motion)
- Recurrent CNN
- …

More on Temporal Data Analysis

- Audio is another kind of temporal data.

Musical instruments source separation

Train on 100,000 unlabeled multi-source video clips,
then separate audio for novel video.

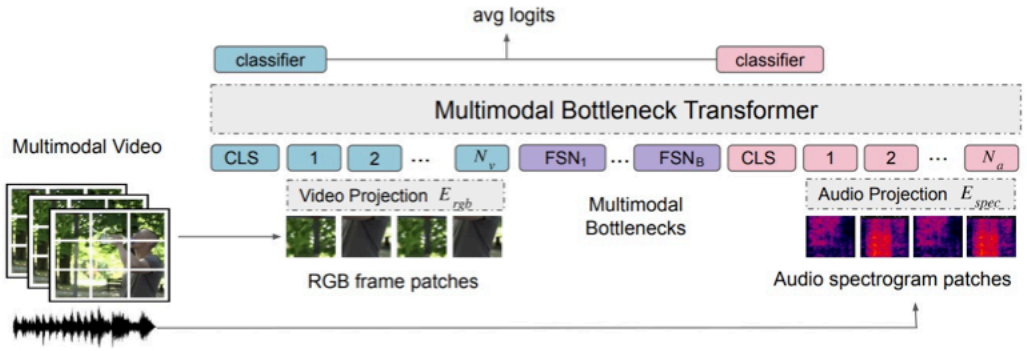
original video
(before separation)

object detections:
violin & flute

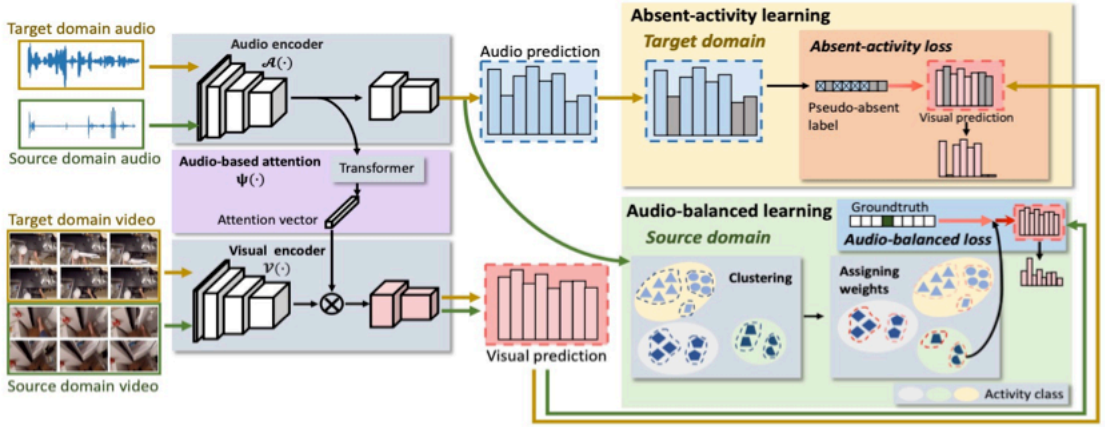
Gao & Grauman, Co-Separating Sounds of Visual Objects, ICCV 2019

More on Temporal Data Analysis

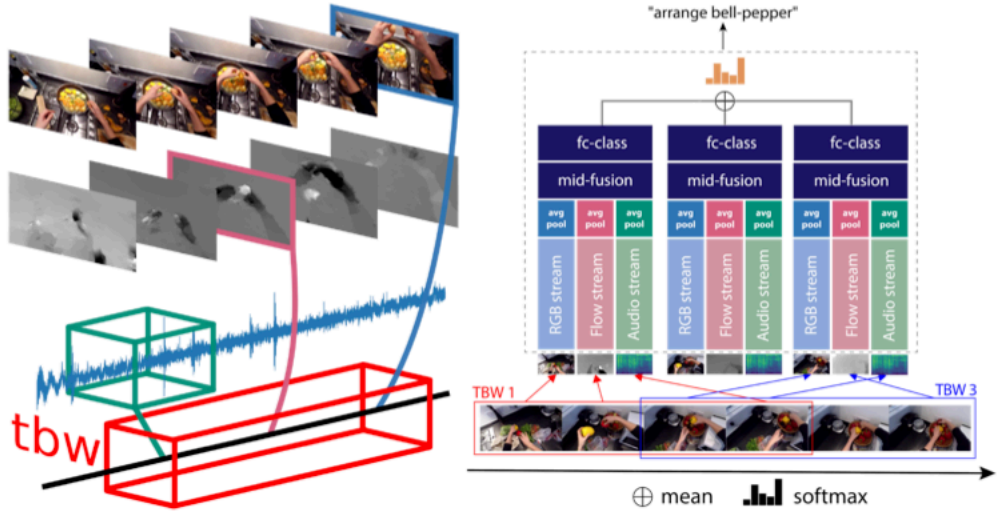
Multimodal Video Understanding



Attention Bottlenecks for Multimodal Fusion, Nagrani et al. NeurIPS 2021



Audio-Adaptive Activity Recognition Across Video Domains, Yunhua et al. CVPR 2022

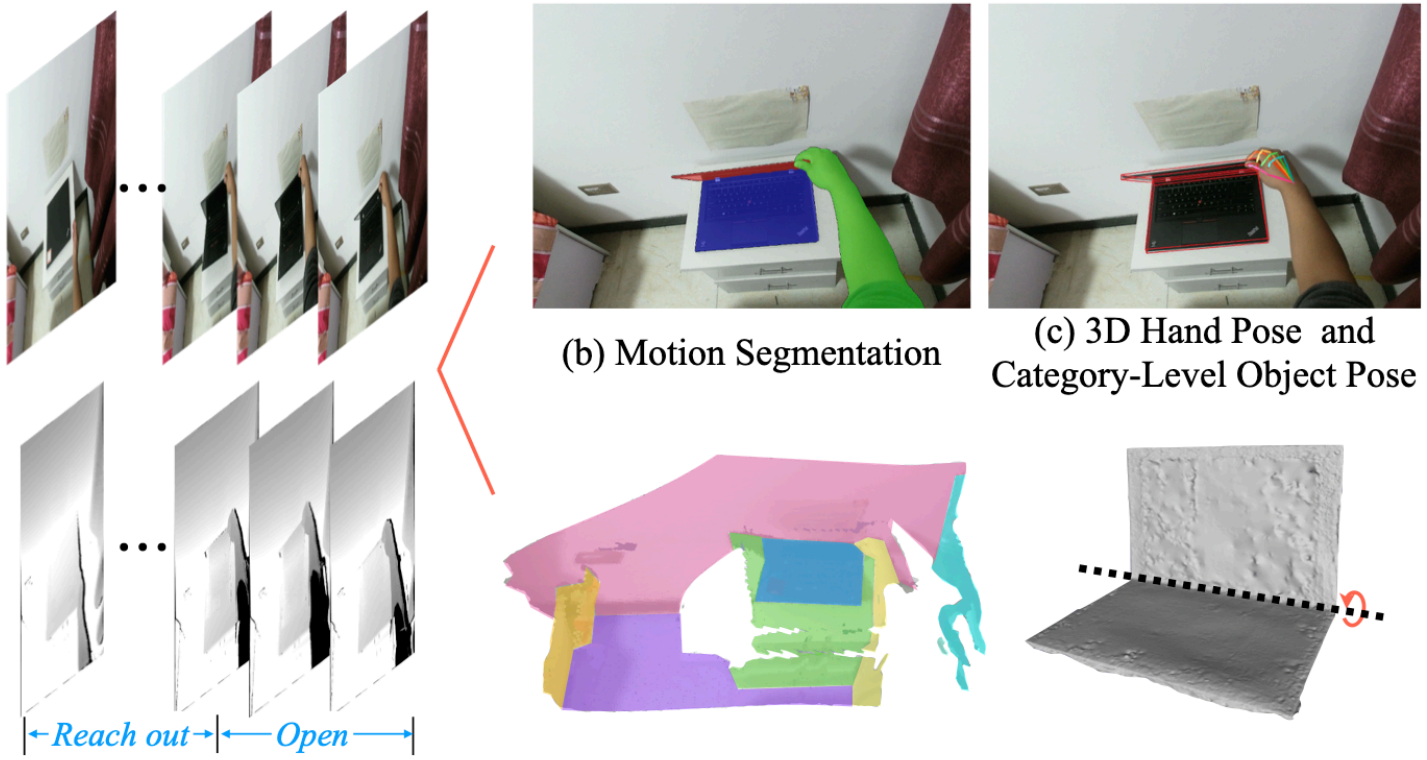


EPIC-Fusion: Audio-Visual Temporal Binding for Egocentric Action Recognition, Kazakos et al., ICCV 2019

More on Temporal Data Analysis

HOI4D: A 4D Egocentric Dataset for Category-Level Human-Object Interaction

Yunze Liu*[1,3], Yun Liu*[1], Che Jiang[1], Kangbo Lyu[1], Weikang Wan[2],
Hao Shen[2], Boqiang Liang[2], Zhoujie Fu[1], He Wang[2], Li Yi†[1,3]
[1] Tsinghua University, [2] Peking University, [3] Shanghai Qi Zhi Institute
https://hoi4d.github.io



(a) Hand Actions (b) Motion Segmentation (c) 3D Hand Pose and Category-Level Object Pose (d) Panoptic Segmentation (e) Reconstructed Object Mesh

More on Temporal Data Analysis

4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks

Christopher Choy
chrischoy@stanford.edu

JunYoung Gwak
jgwak@stanford.edu

Silvio Savarese
ssilvio@stanford.edu

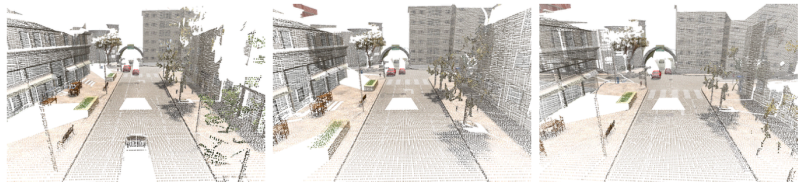


Figure 1: An example of 3D video: 3D scenes at different time steps. Best viewed on display.

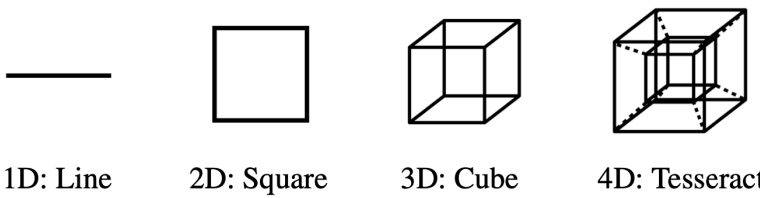


Figure 2: 2D projections of hypercubes in various dimensions

Object Detection

Some slides are borrowed from Stanford CS231N.

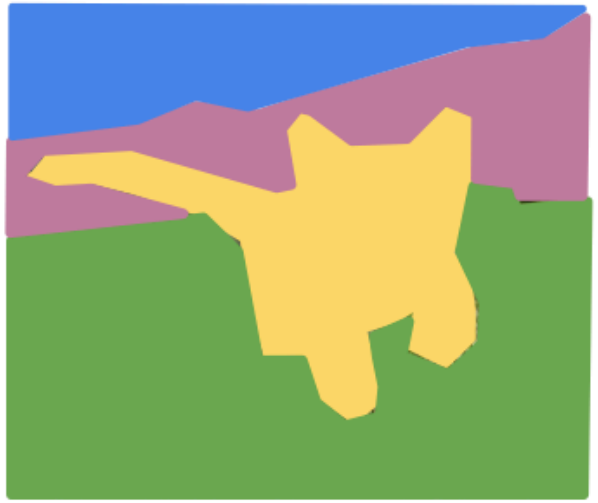
Computer Vision Tasks

Classification

CAT

No spatial extent

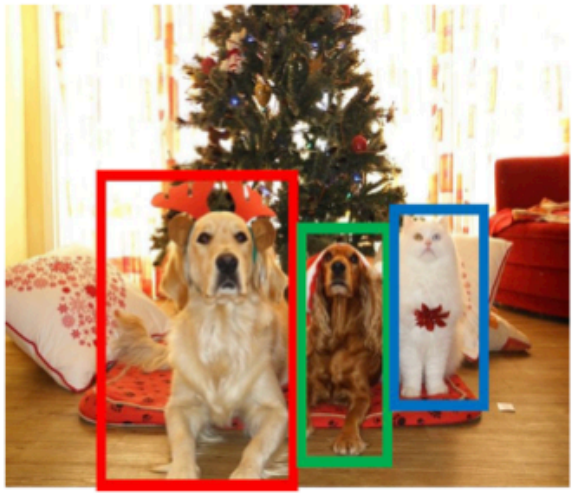
Semantic Segmentation



GRASS, CAT, TREE, SKY

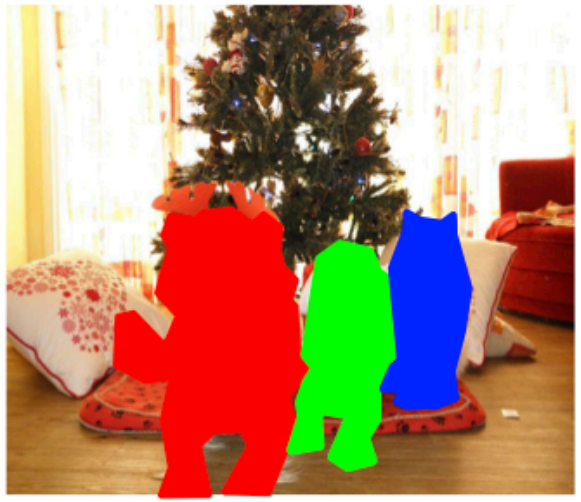
No objects, just pixels

Object Detection



DOG, DOG, CAT

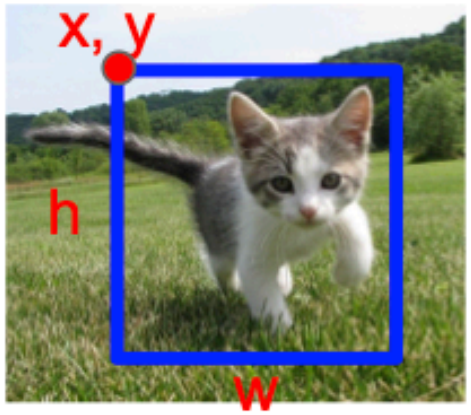
Instance Segmentation



DOG, DOG, CAT

Multiple Object

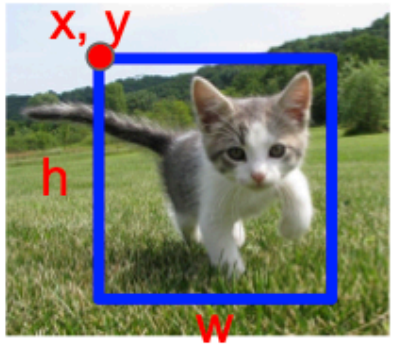
Object Detection: Single Object



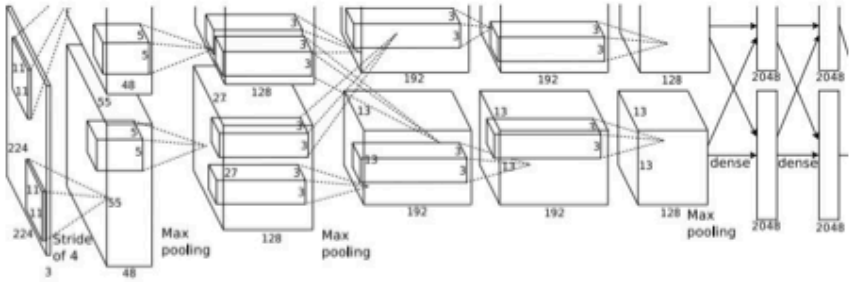
- Task: localization + classification
- Output: 2D (axis aligned) bounding box
  - How many degree-of-freedom?
    - 4 DoF
  - How to parameterize such a bounding box?
    - x,y,h,w

Object Detection: Single Object

- Localization + Classification



This image is CC0 public domain



Fully Connected: 4096 to 1000

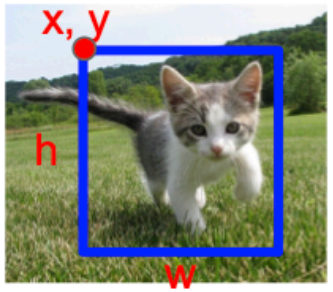
Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Vector: 4096

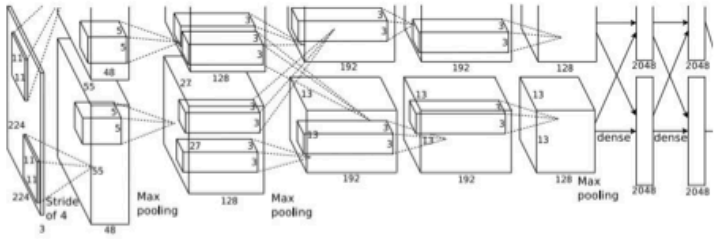
Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

Object Detection: Single Object



This image is CC0 public domain



Fully Connected: 4096 to 1000

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

Softmax Loss

Correct label: Cat

Vector: 4096

Fully Connected: 4096 to 4

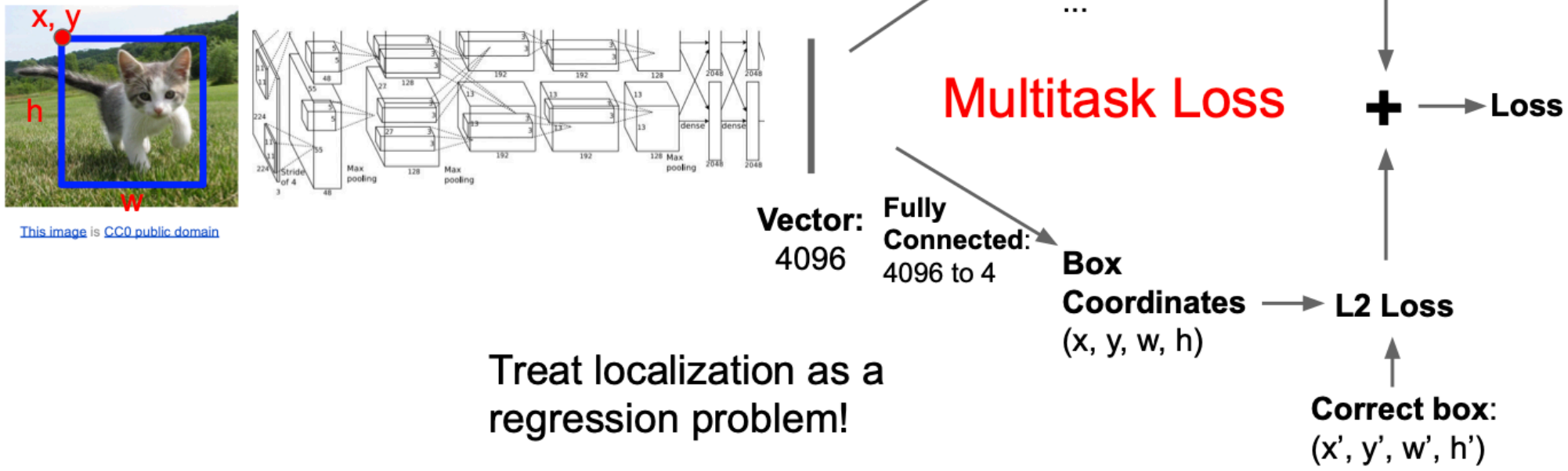
Box Coordinates (x, y, w, h)

L2 Loss

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

Treat localization as a regression problem!

Object Detection: Single Object



Regression Loss

- Error: $(\Delta x, \Delta y, \Delta w, \Delta h)$

- L1 loss: $\Sigma|\Delta_i|$ — robust, however not good at convergence
- L2 loss: $\Sigma\Delta_i^2$ (not the same to L2 norm) — not robust to a larger error, however good at convergence
- Rooted mean squared loss (RMSE): $\sqrt{\frac{1}{N}\Sigma\Delta_i^2}$ — the gradient of sqrt function is bad at 0

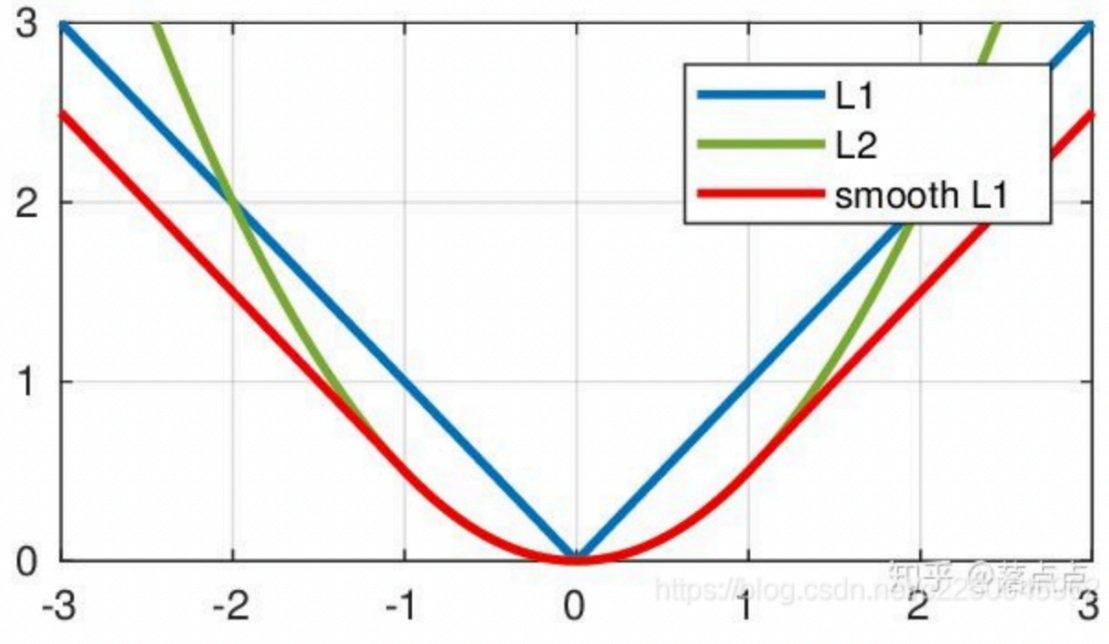
Regression Loss

- Smooth L1 loss (proposed by Fast RCNN, very similar to Huber loss widely used in robust optimization)

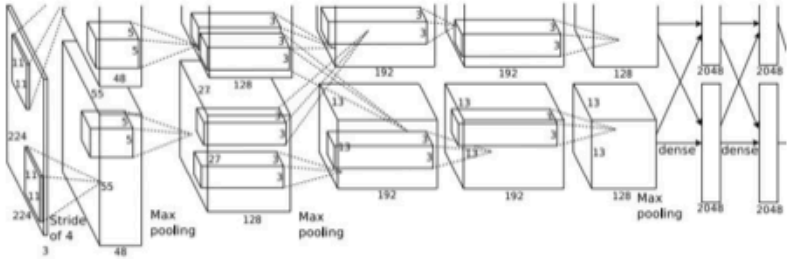
$$L_2(x) = x^2 \tag{1}$$

$$L_1(x) = |x| \tag{2}$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \tag{3}$$

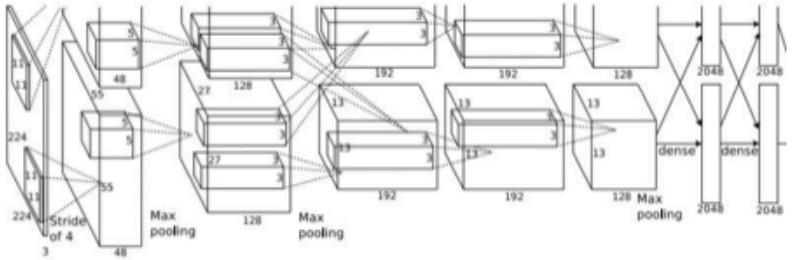


Object Detection: Multiple Objects



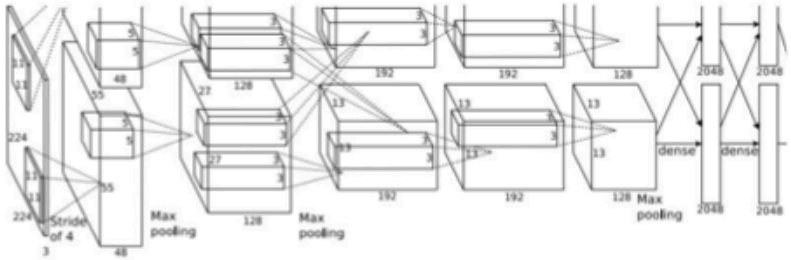
CAT: (x, y, w, h)

1 bounding box



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

3 bounding boxes



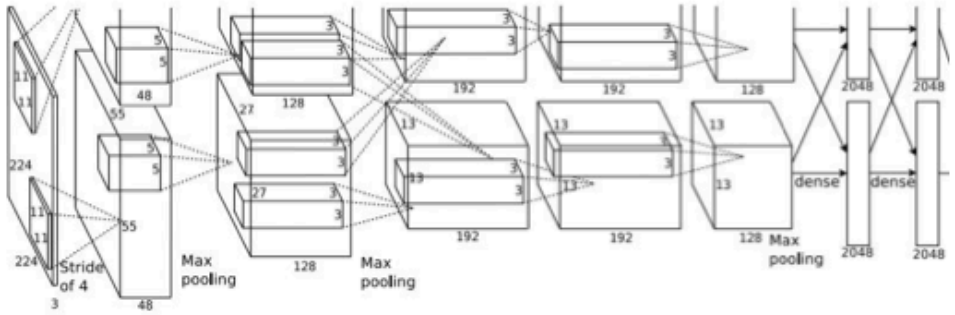
DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....

Many bounding boxes!

Different images need different numbers of outputs!

Sliding-Window based Multi-Object Detection

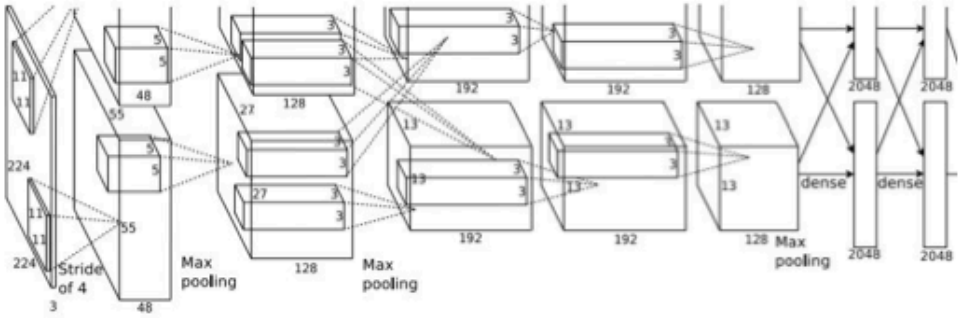
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? NO
Background? YES

Sliding-Window based Multi-Object Detection

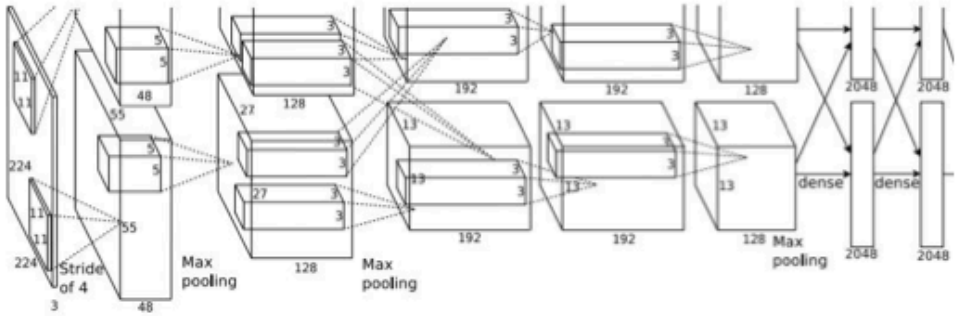
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

Sliding-Window based Multi-Object Detection

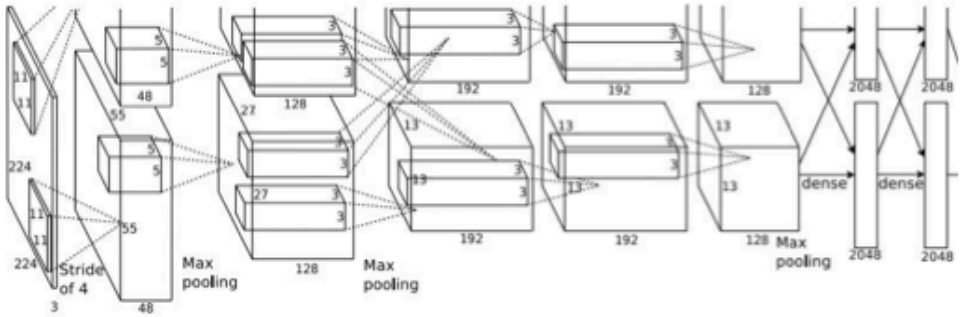
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

Sliding-Window based Multi-Object Detection

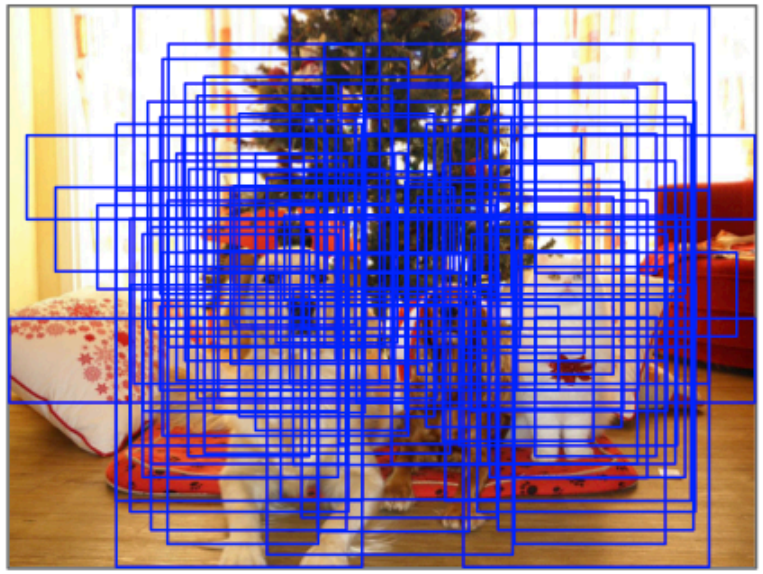
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



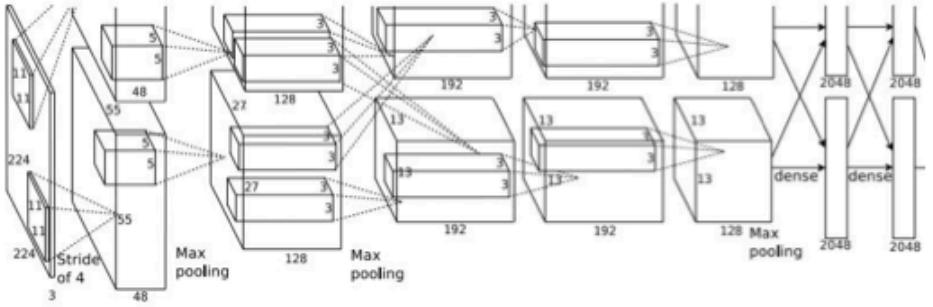
Dog? NO
Cat? YES
Background? NO

Q: What's the problem with this approach?

Sliding-Window based Multi-Object Detection



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

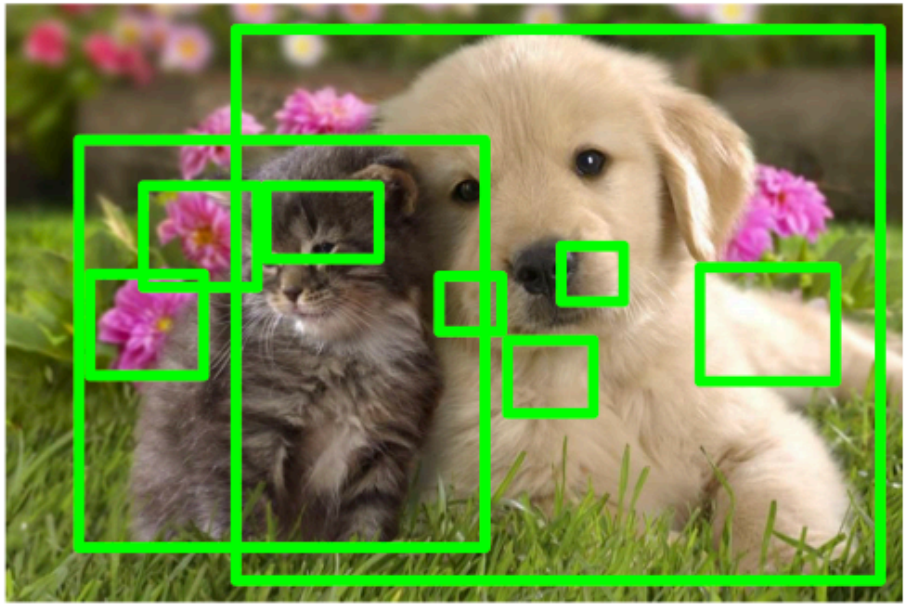


Dog? NO
Cat? YES
Background? NO

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

Region Proposals: Selective Search

- Find “blobby” image regions that are likely to contain objects
- Relatively fast to run; e.g. Selective Search gives 2000 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.

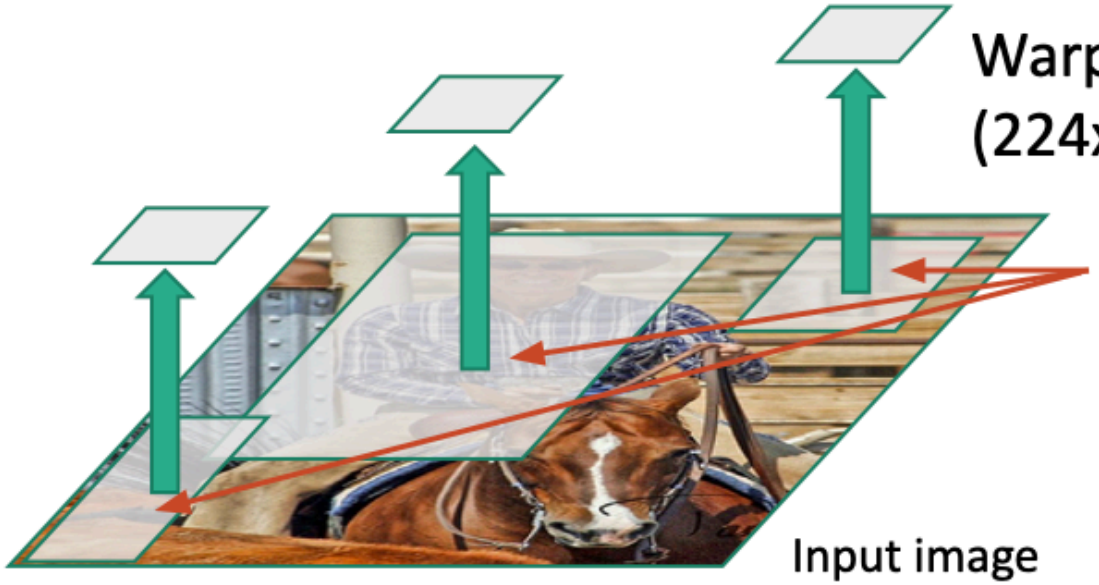
R-CNN

Input image

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

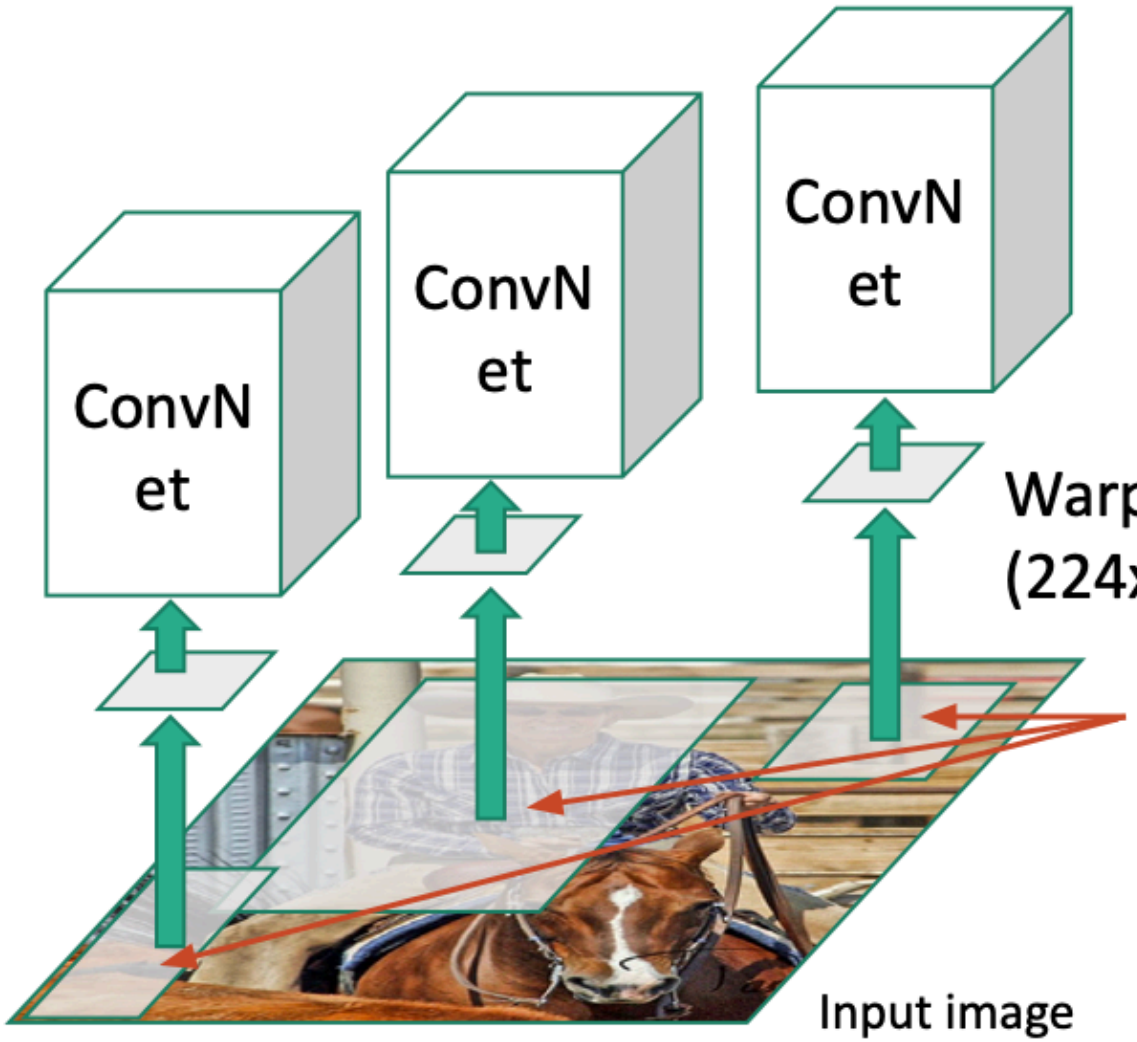
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

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



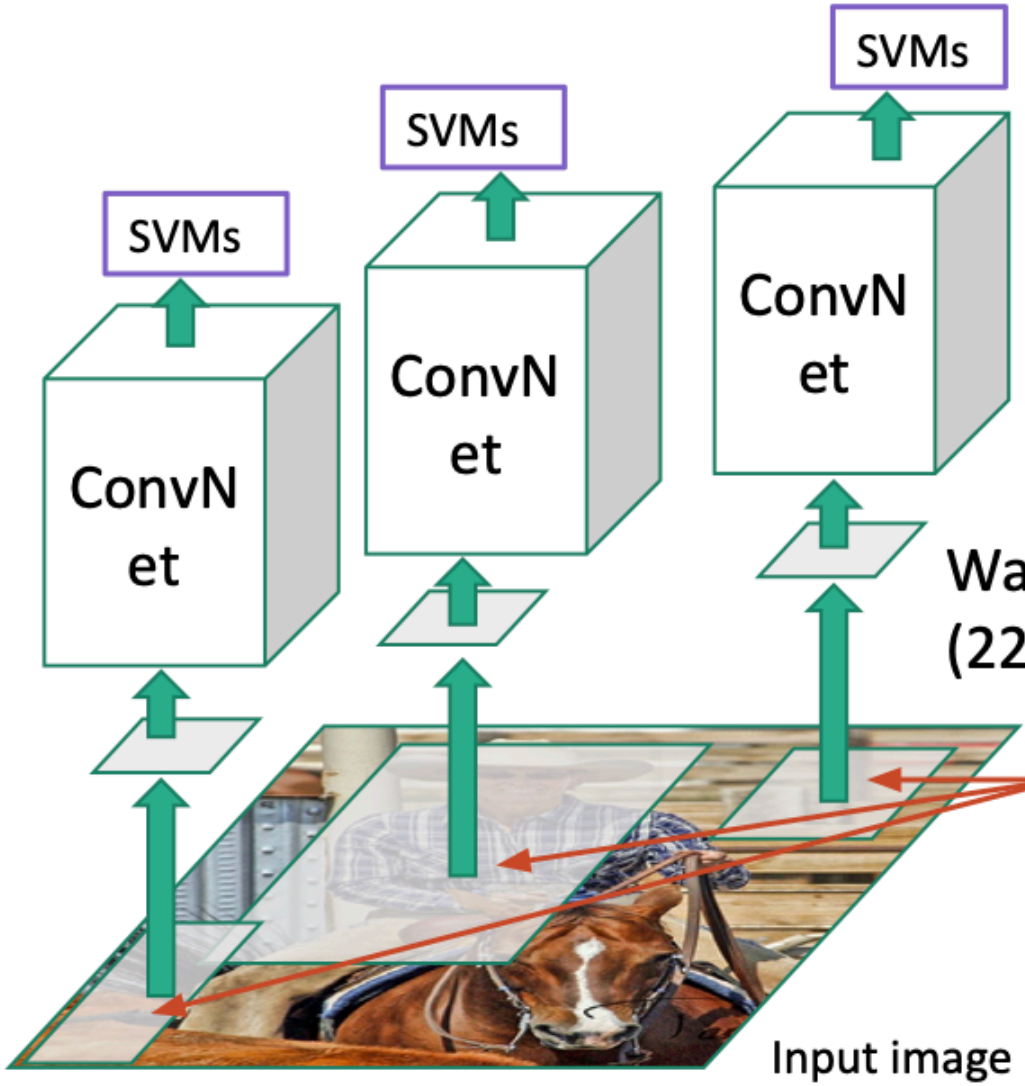
Forward each region through ConvNet (ImageNet-pretranied)

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



Classify regions with SVMs

Forward each region through ConvNet (ImageNet-pretranied)

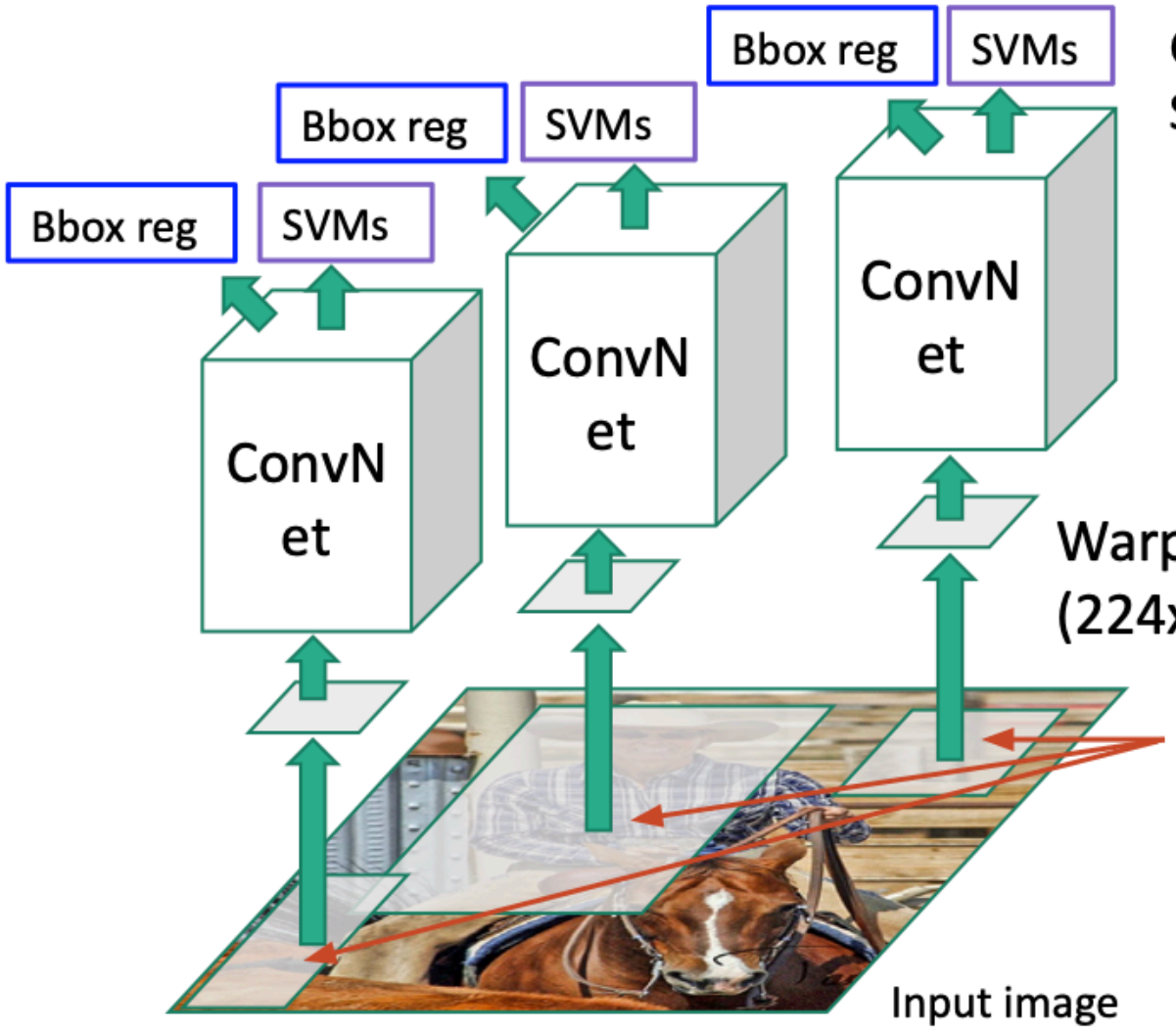
Warped image regions (224x224 pixels)

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

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



Classify regions with SVMs

Forward each region through ConvNet (ImageNet-pretranied)

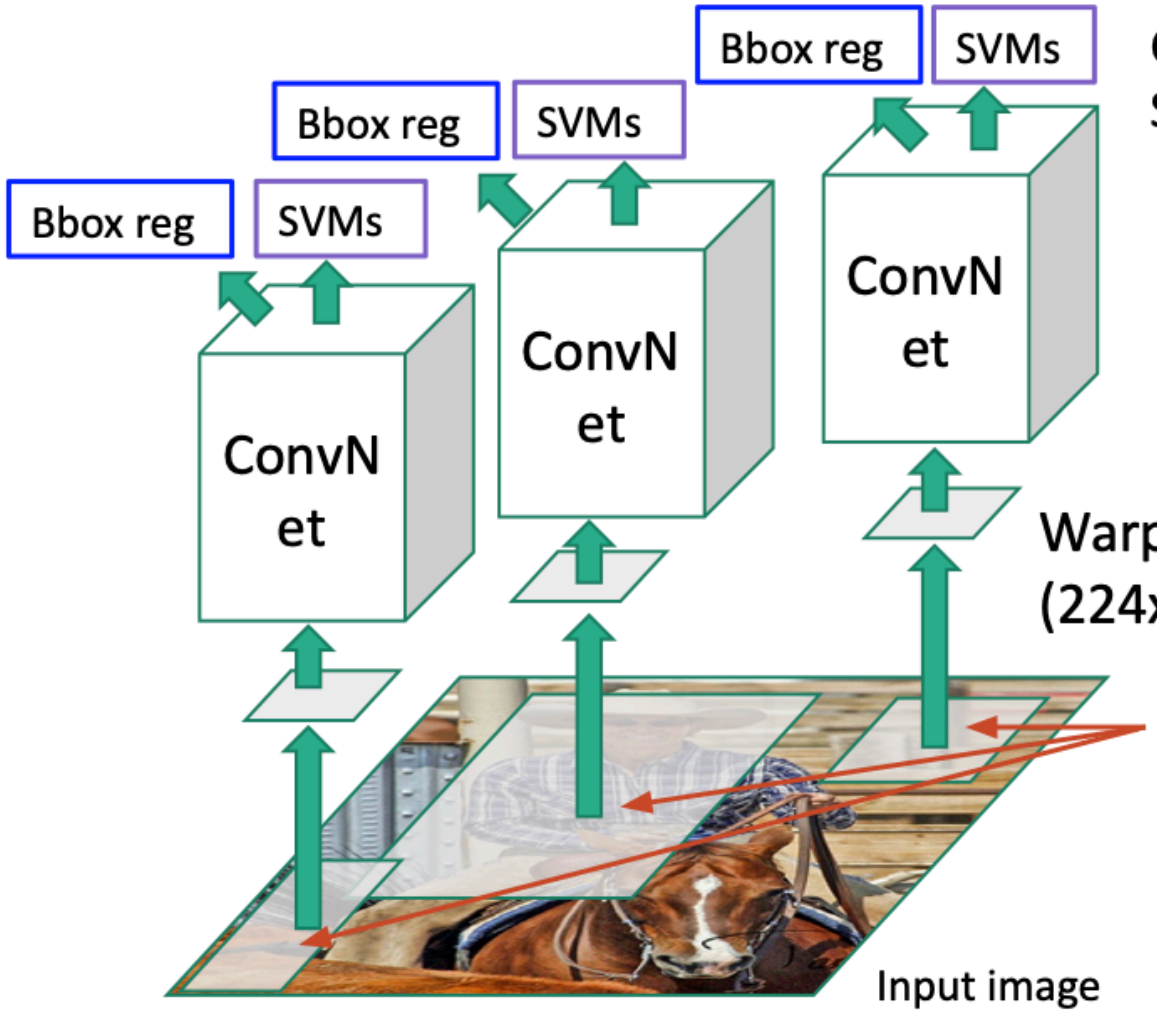
Warped image regions (224x224 pixels)

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

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)

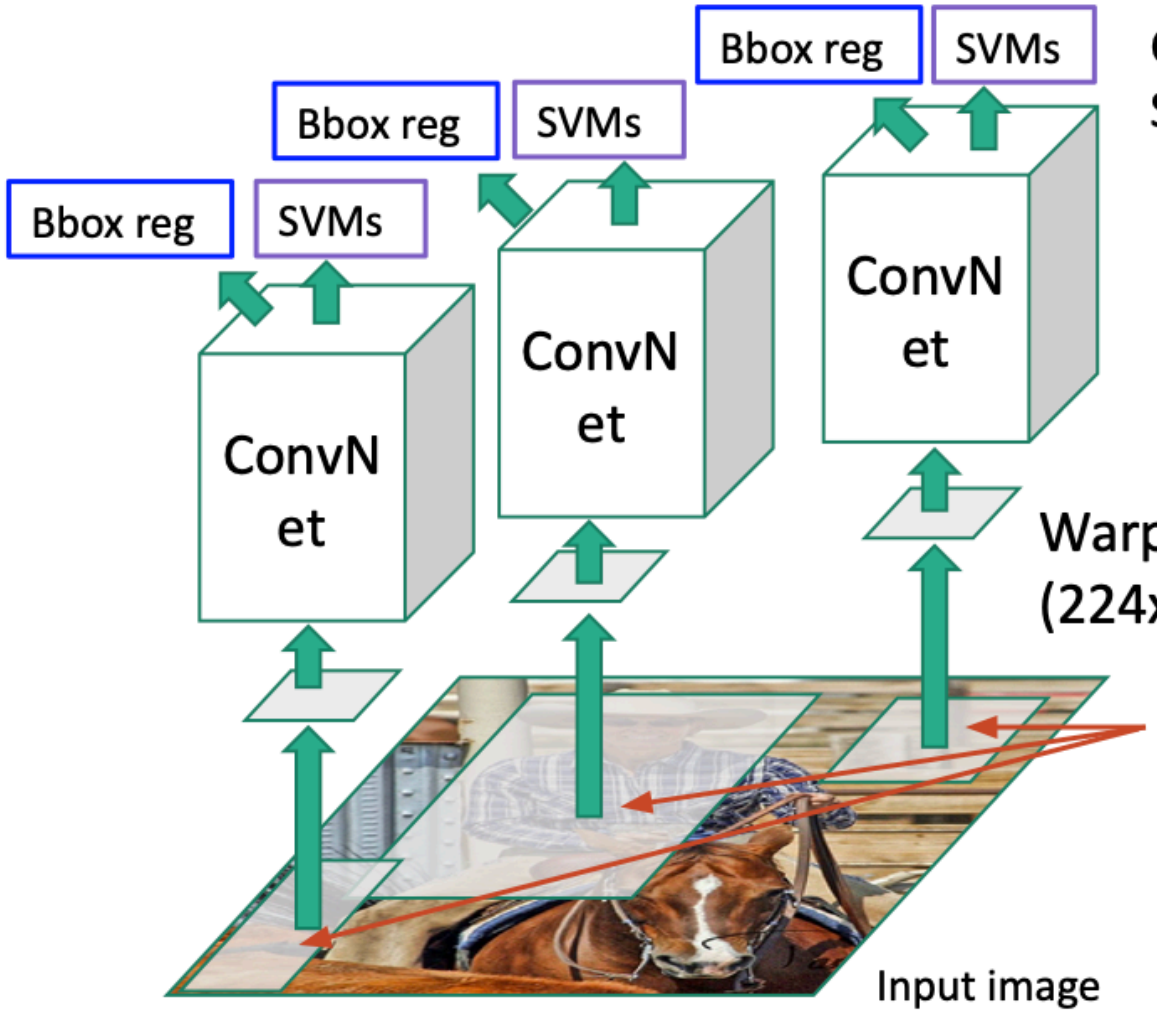
Regions of Interest (RoI) from a proposal method (~2k)

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

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)

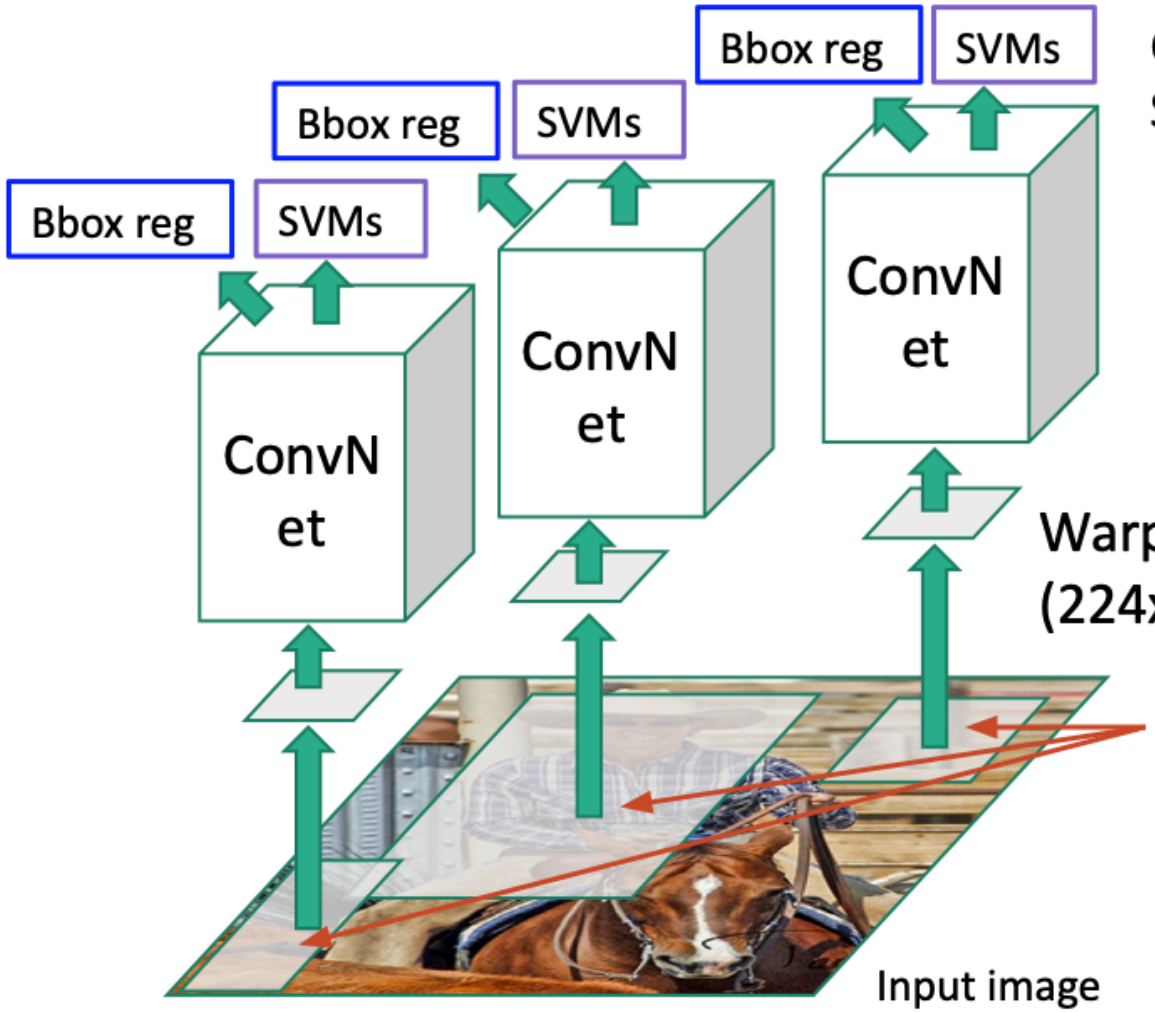
Problem: Very slow! Need to do ~2k independent forward passes for each image!

Problem 2: The cropped region doesn't contain sufficient information to regress bounding box refinements.

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

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)

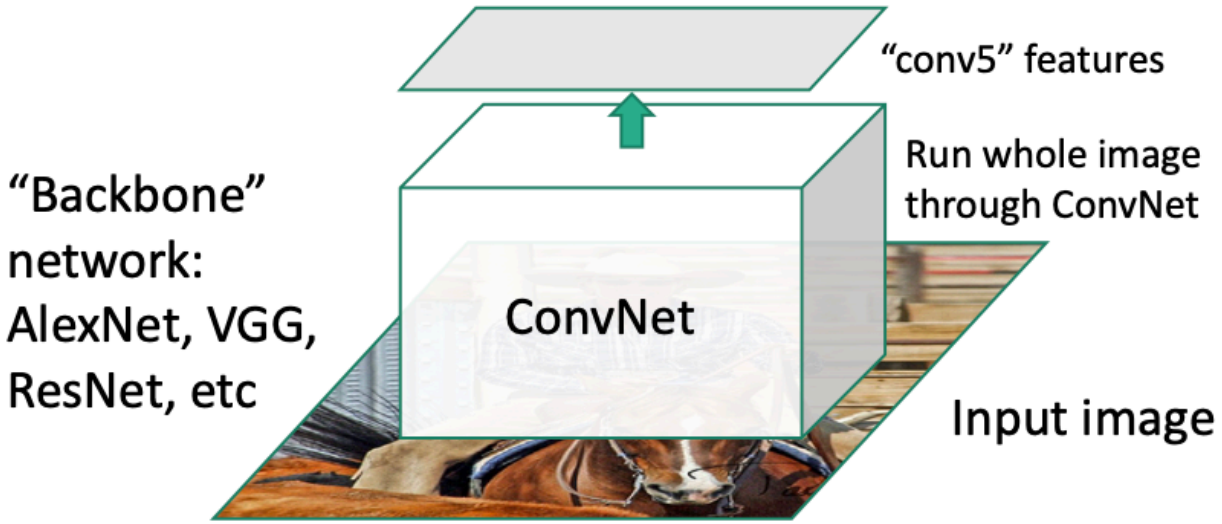
Regions of Interest (RoI) from a proposal method (~2k)

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!

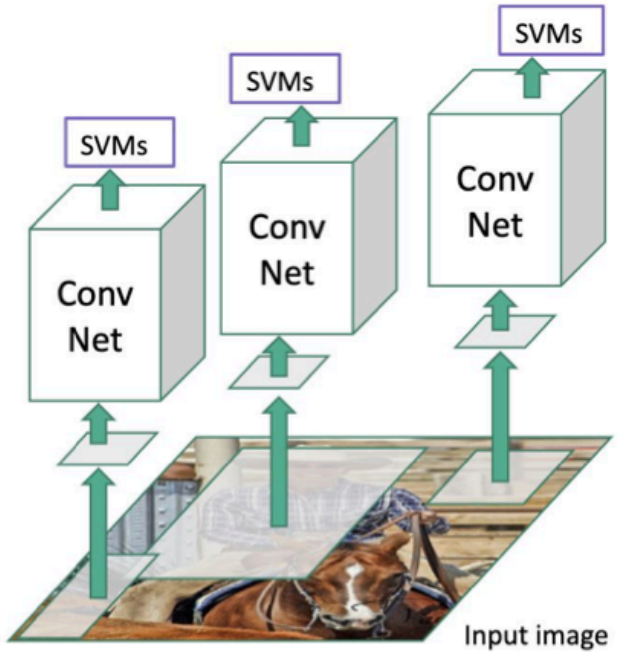
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN



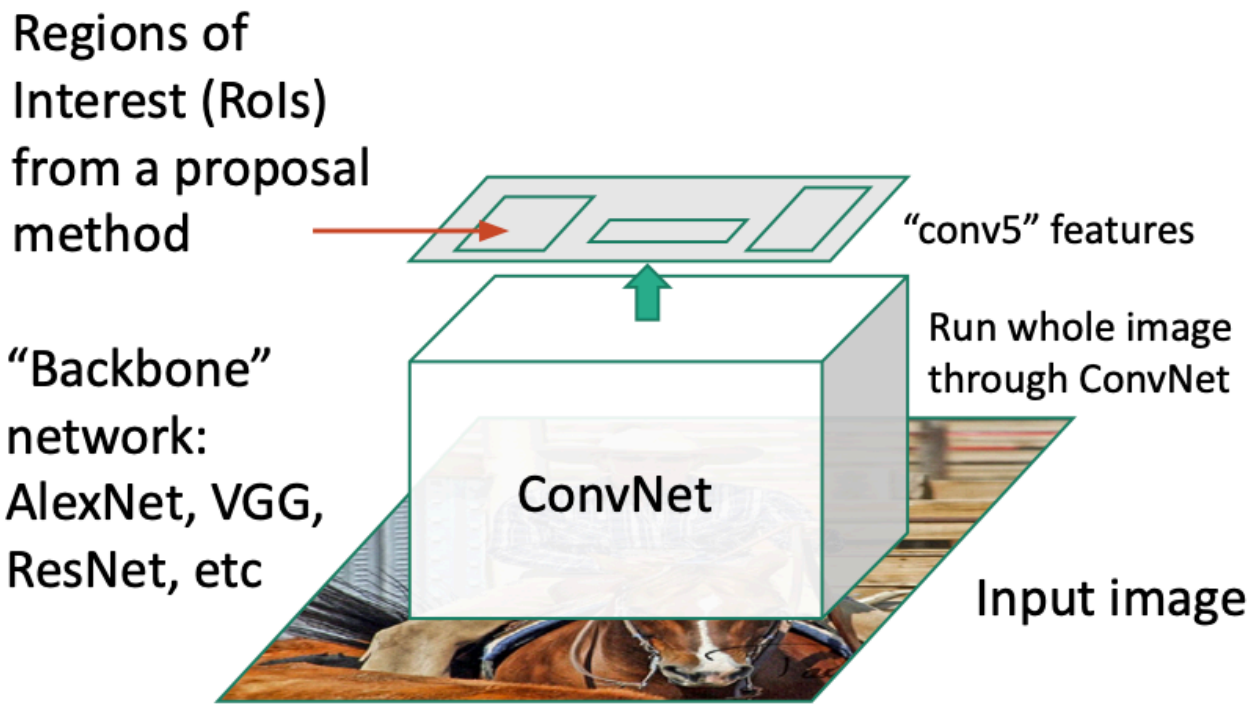
Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast RCNN



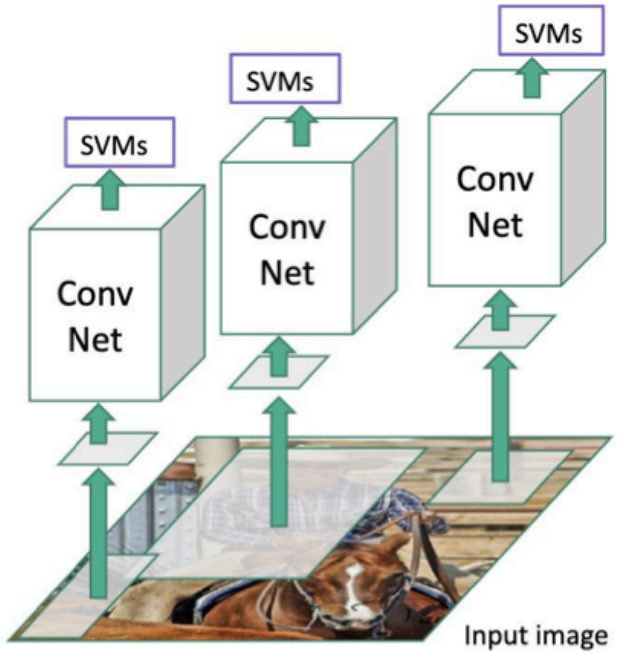
R-CNN

Fast R-CNN



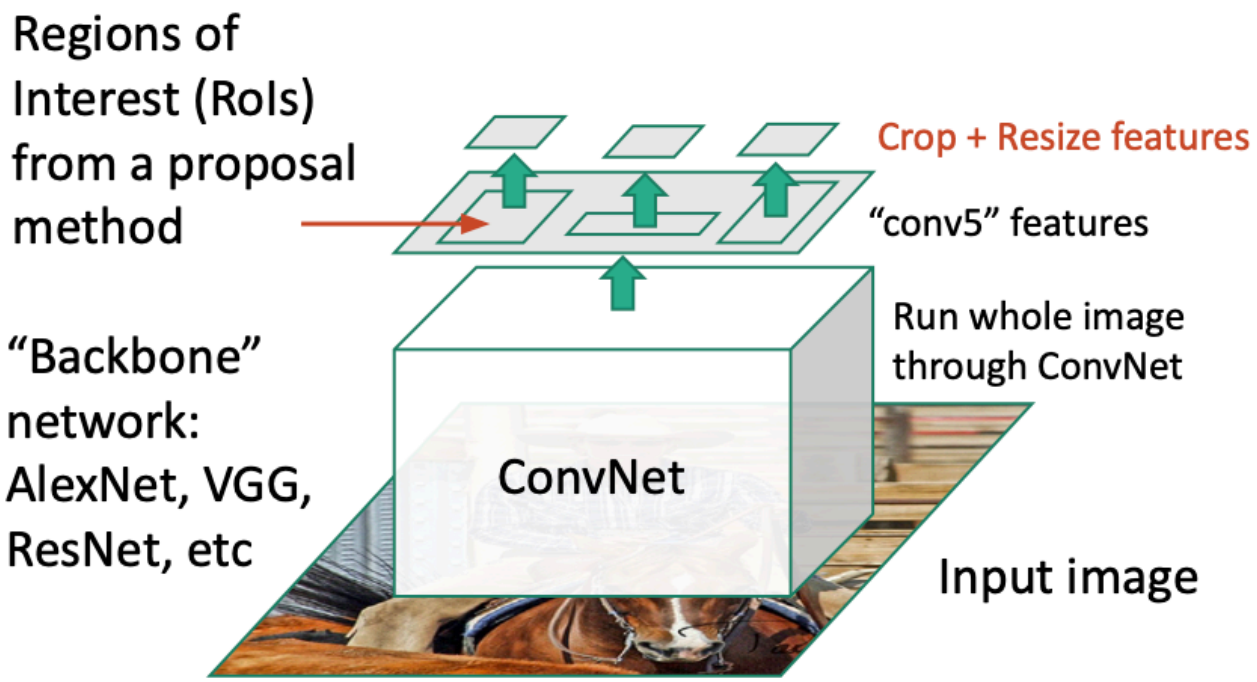
Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast RCNN

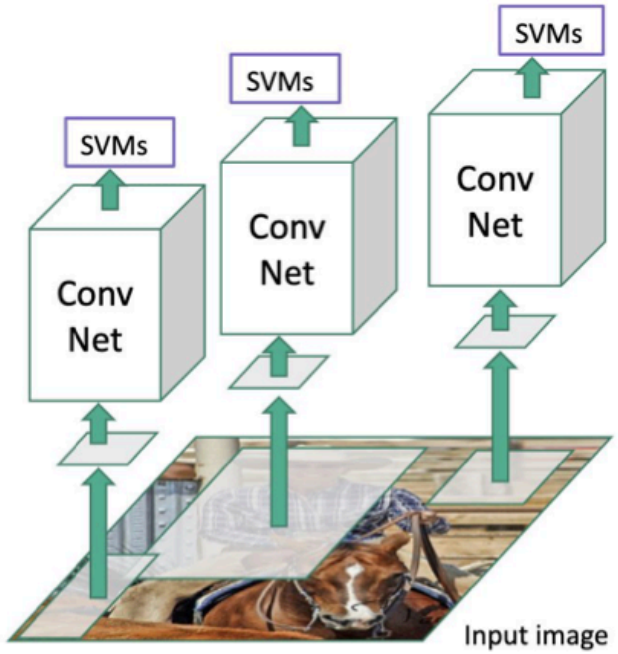


R-CNN

Fast R-CNN



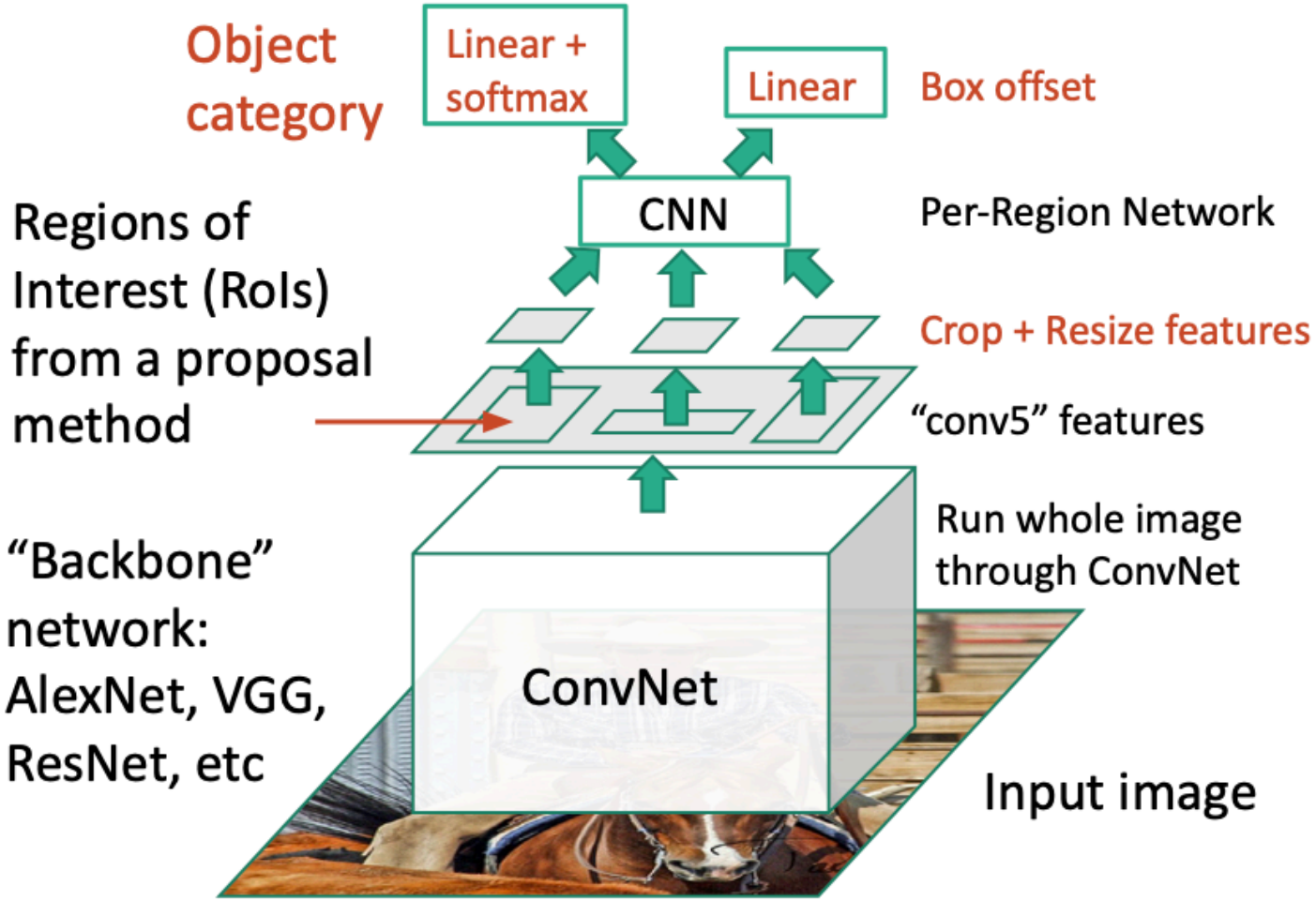
Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.



Fast RCNN

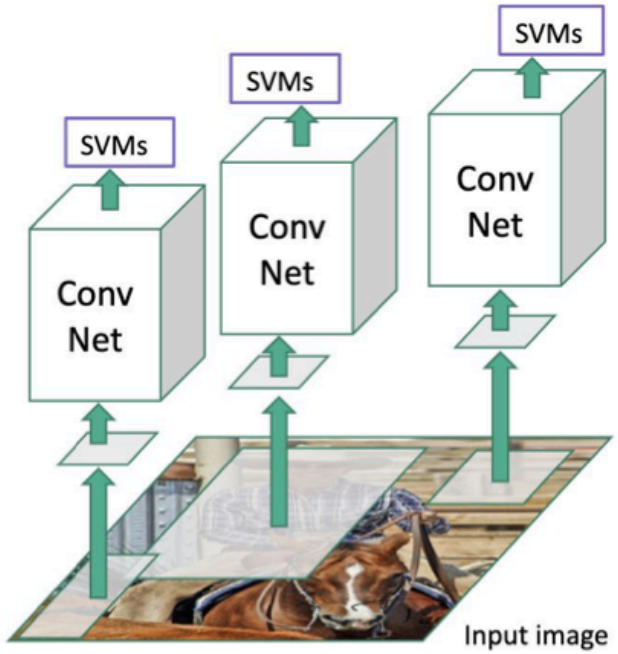
R-CNN

Fast R-CNN



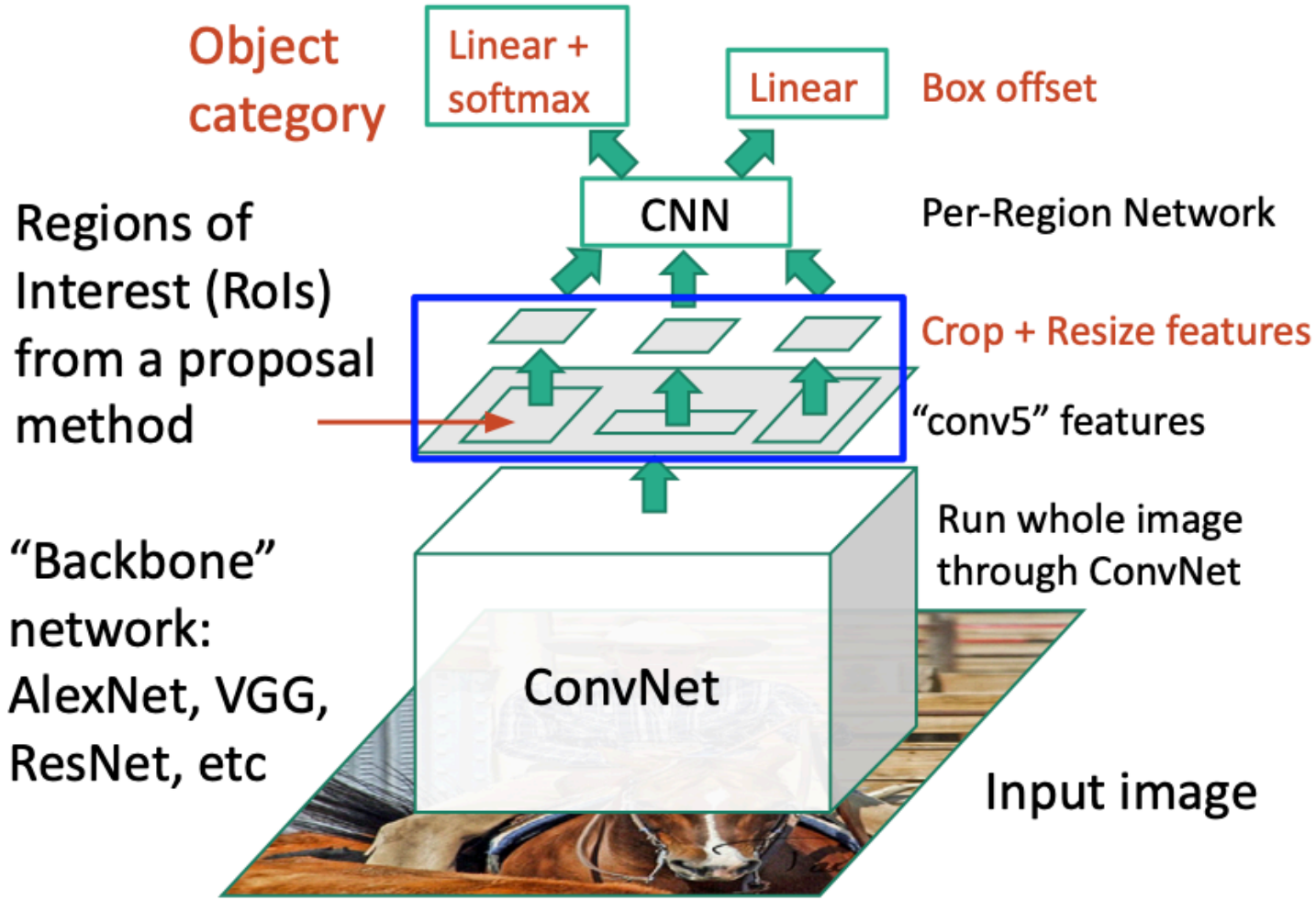
Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast RCNN

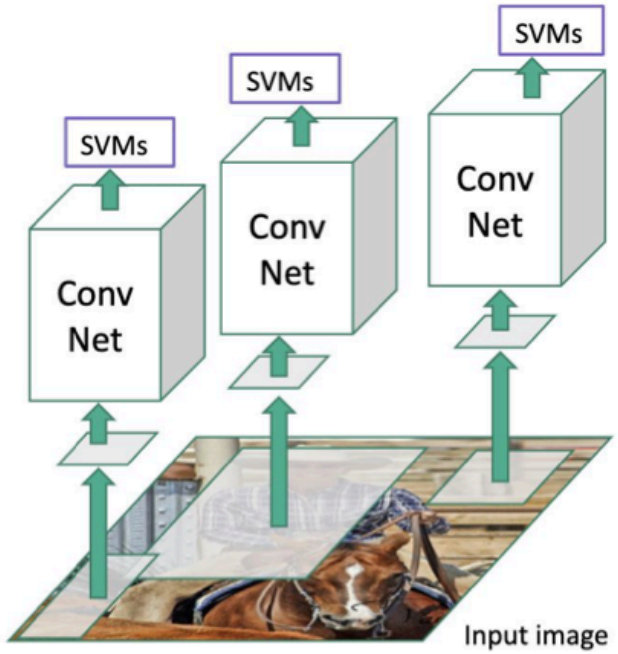


R-CNN

Fast R-CNN

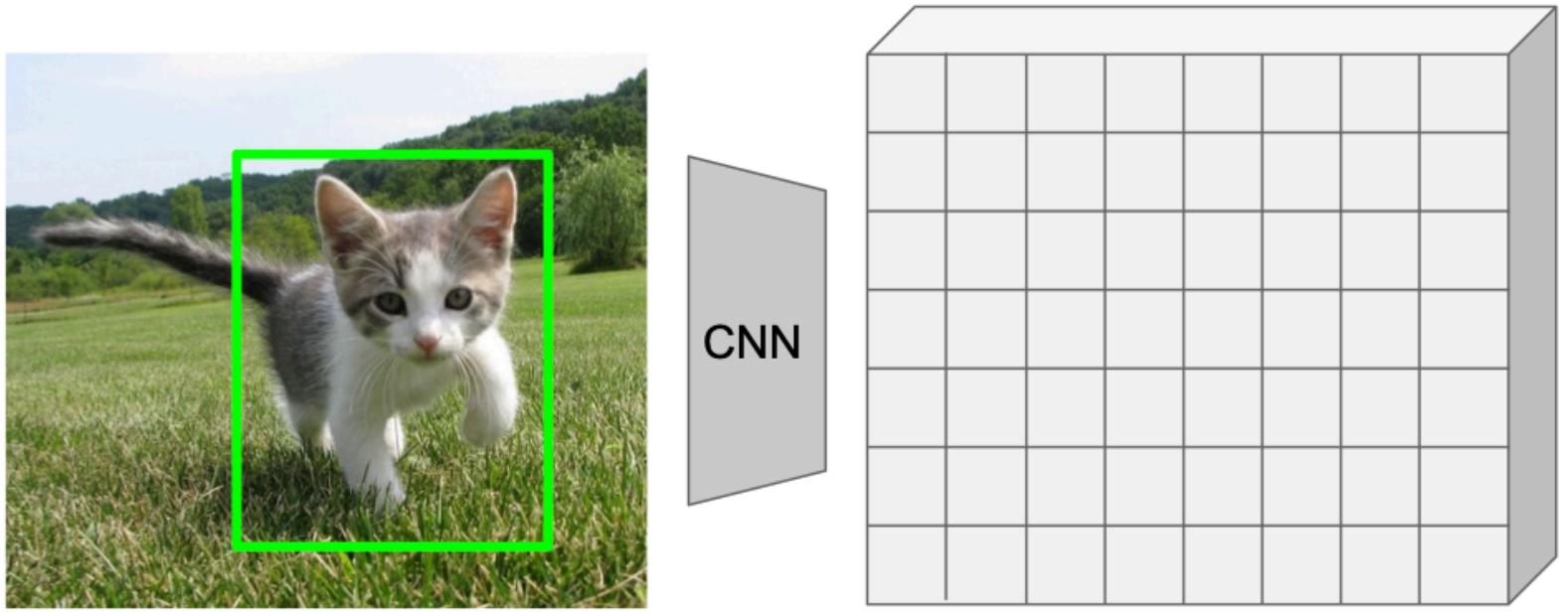


Fast RCNN



R-CNN

Cropping Features: RoI Pool

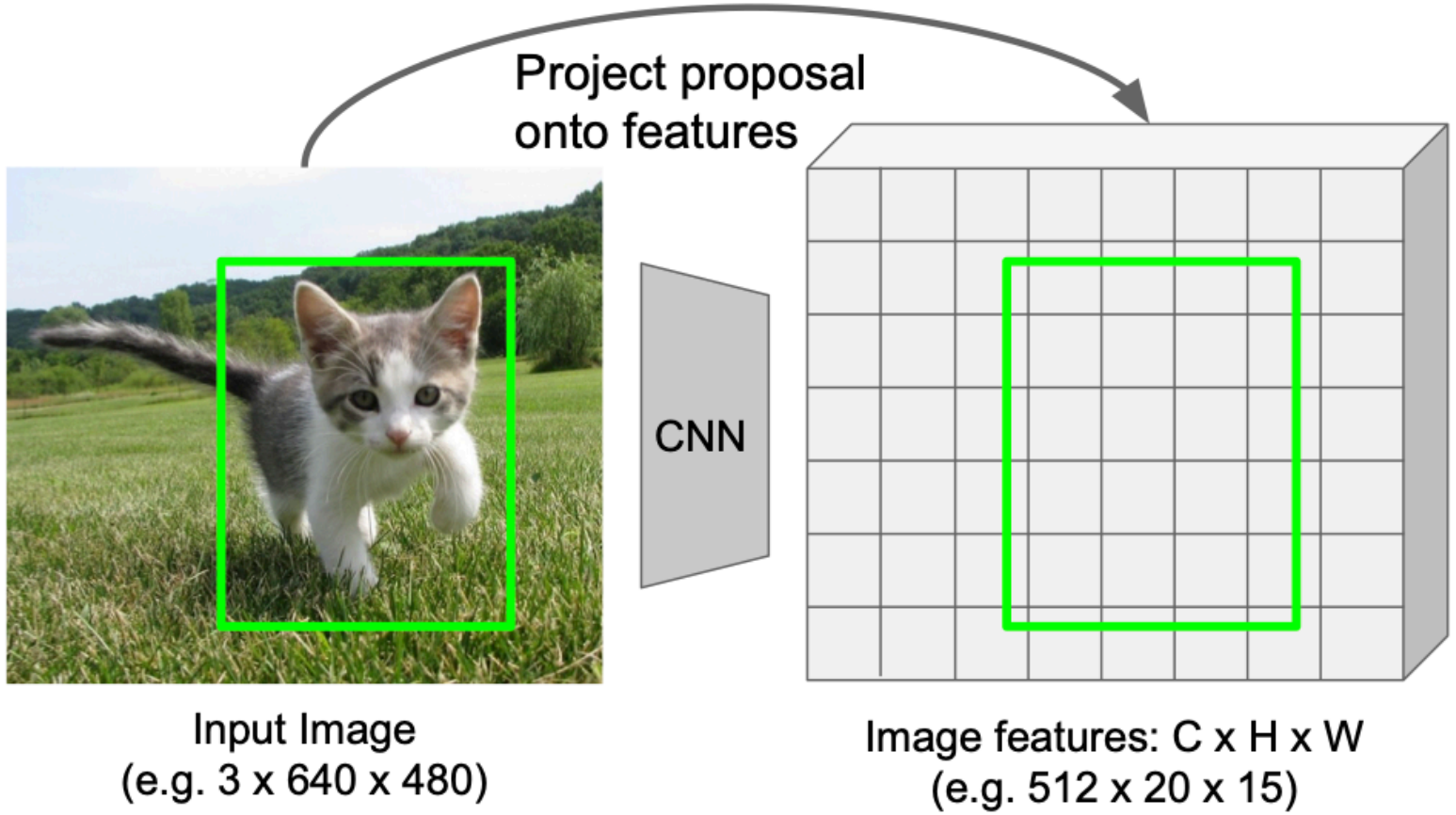


Input Image
(e.g. 3 x 640 x 480)

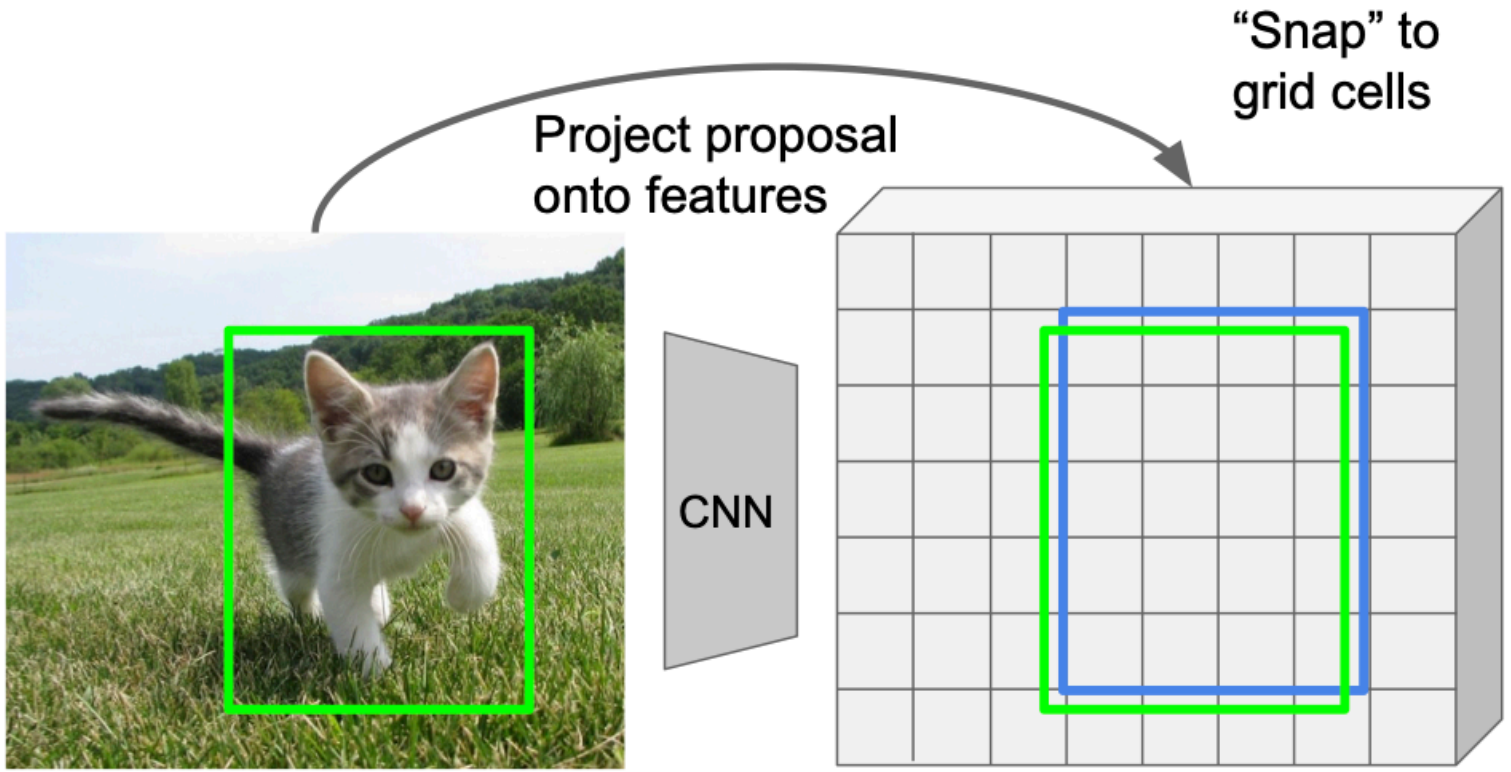
Image features: C x H x W
(e.g. 512 x 20 x 15)

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

Cropping Features: RoI Pool

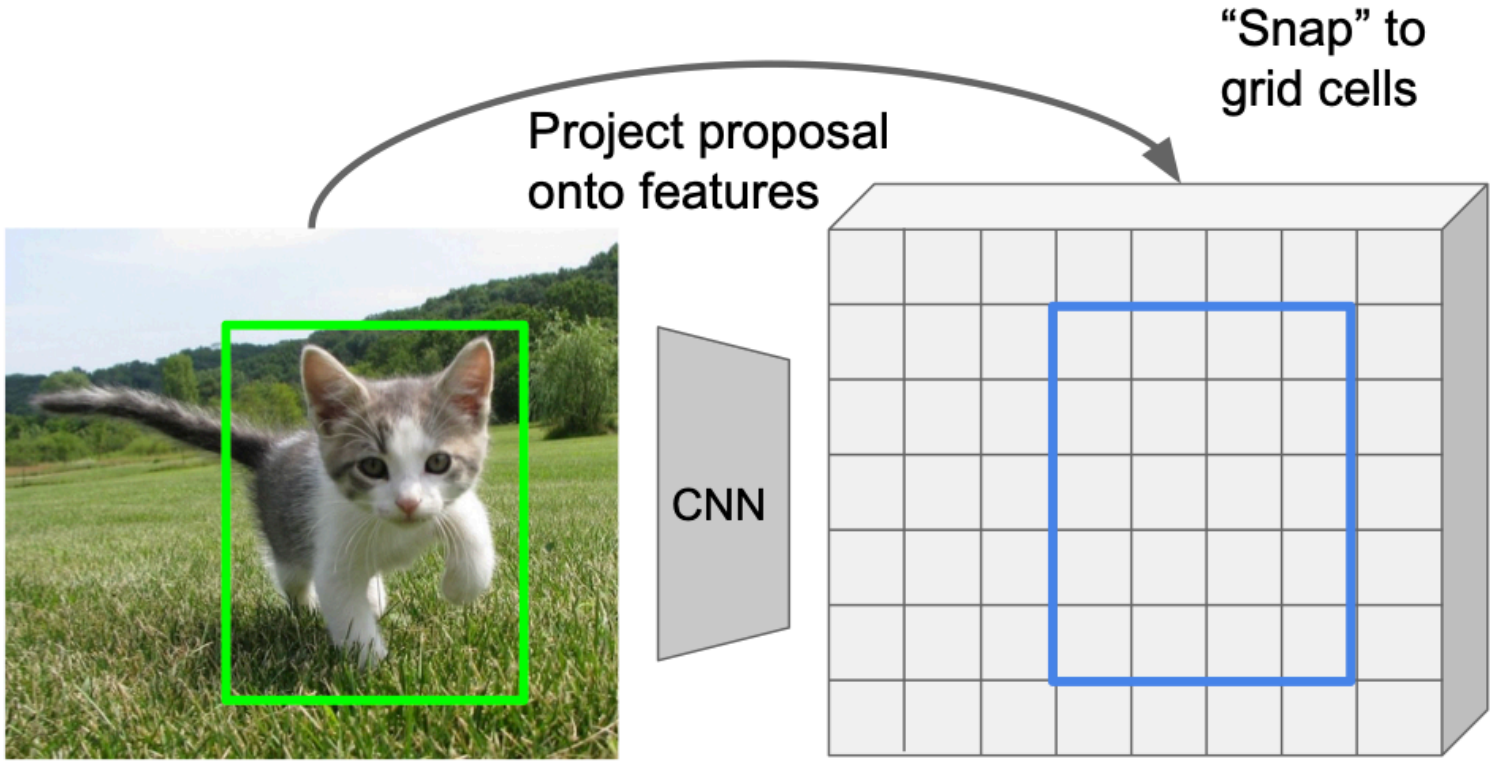


Cropping Features: RoI Pool



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

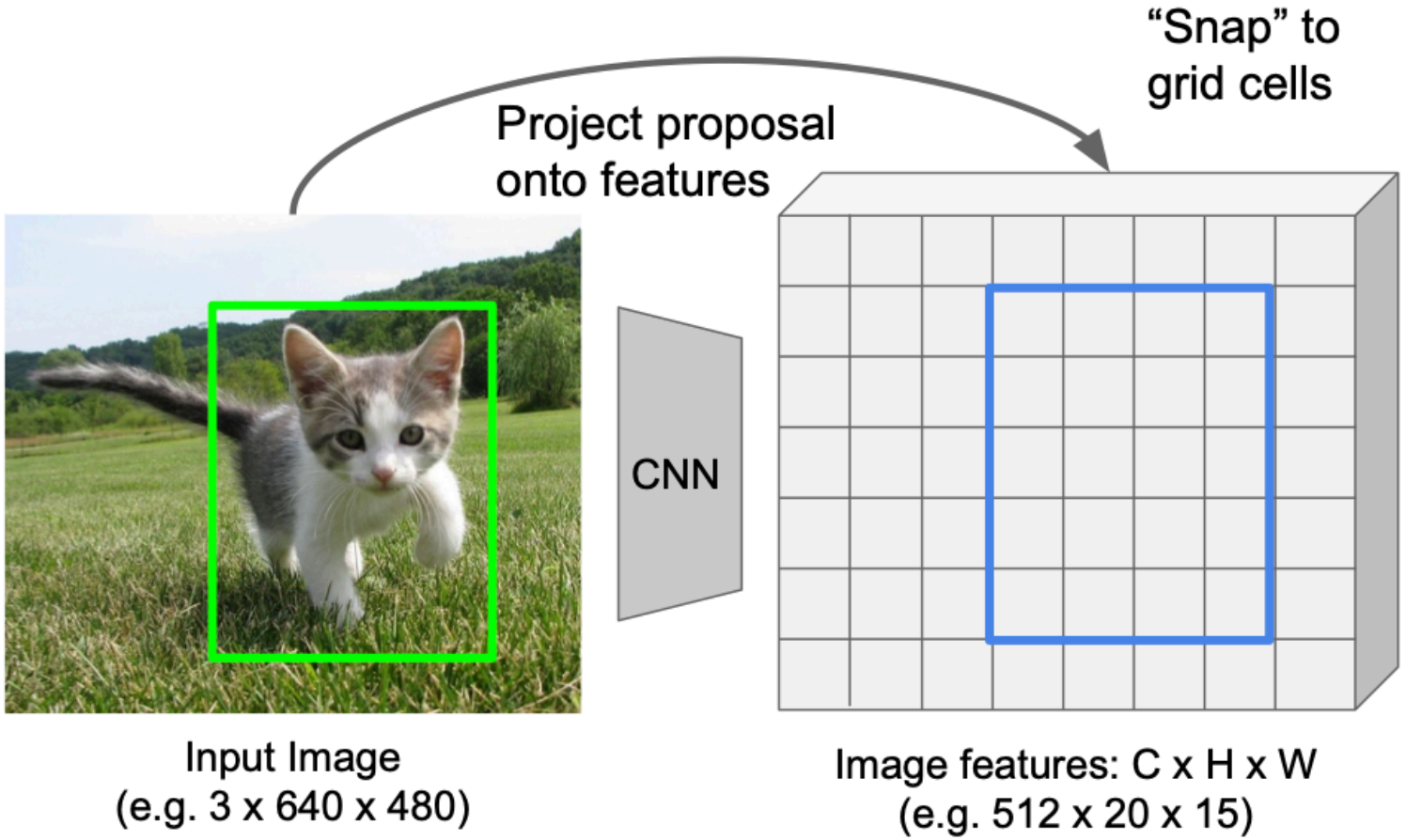
Cropping Features: RoI Pool



Q: how do we resize the 512 x 20 x 15 region to, e.g., a 512 x 2 x 2 tensor?.

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

Cropping Features: RoI Pool

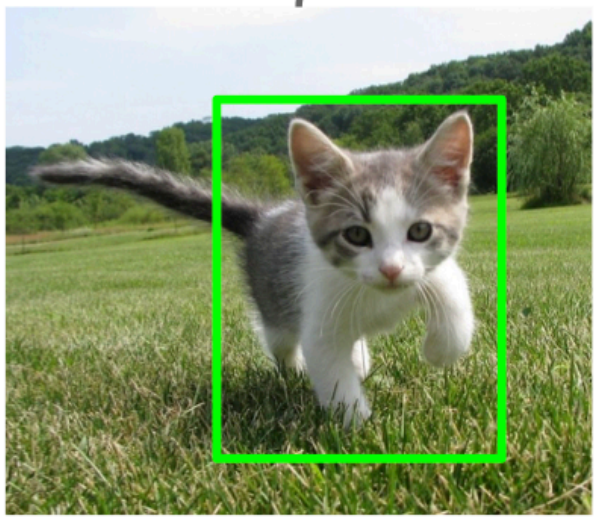


Divide into 2x2 grid of (roughly) equal subregions

Q: how do we resize the 512 x 20 x 15 region to, e.g., a 512 x 2 x 2 tensor?.

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

Cropping Features: RoI Pool



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

Project proposal onto features

CNN

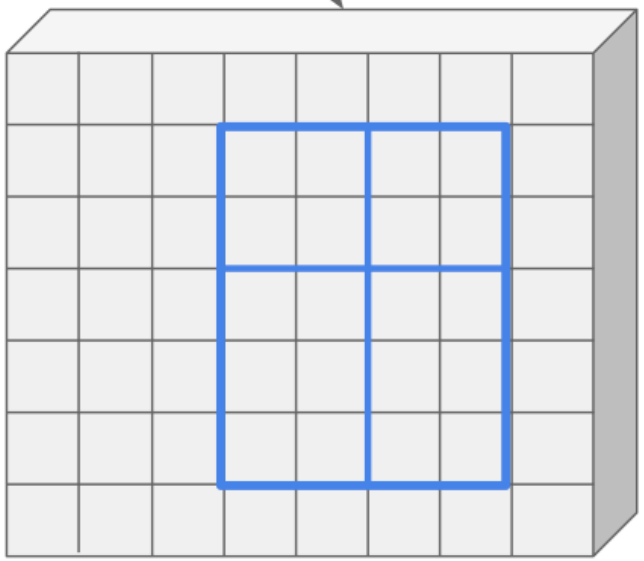
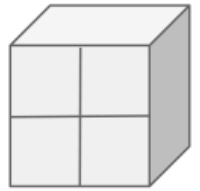


Image features: C x H x W
(e.g. 512 x 20 x 15)

"Snap" to grid cells

Divide into 2x2 grid of (roughly) equal subregions

Max-pool within each subregion

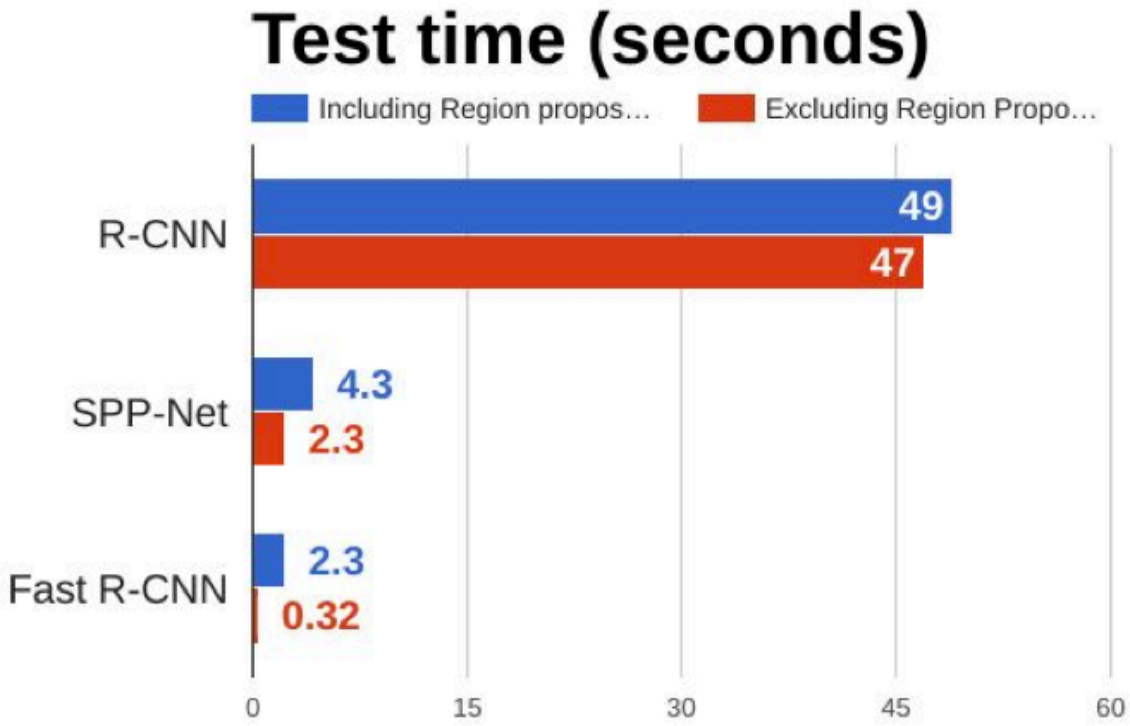


Region features
(here 512 x 2 x 2;
In practice e.g 512 x 7 x 7)

Region features always the same size even if input regions have different sizes!

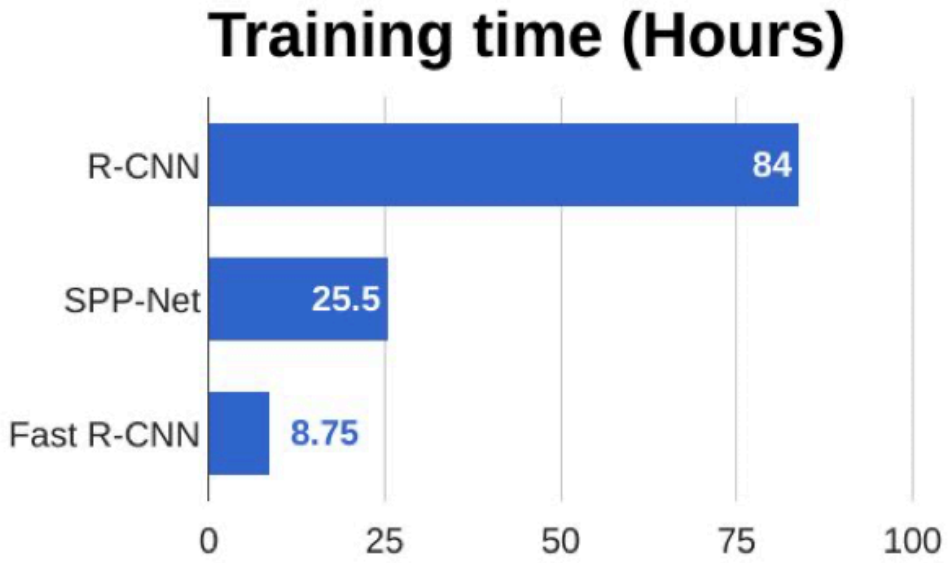
Girshick, "Fast R-CNN", ICCV 2015.

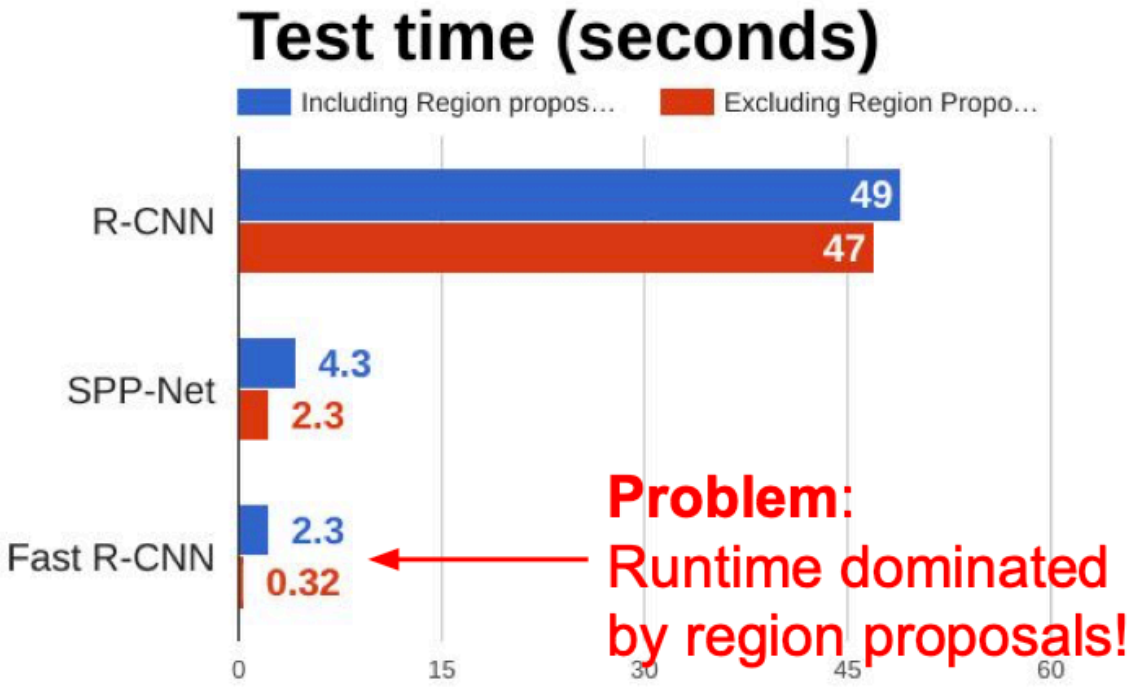
R-CNN vs. Fast R-CNN



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. Fast R-CNN





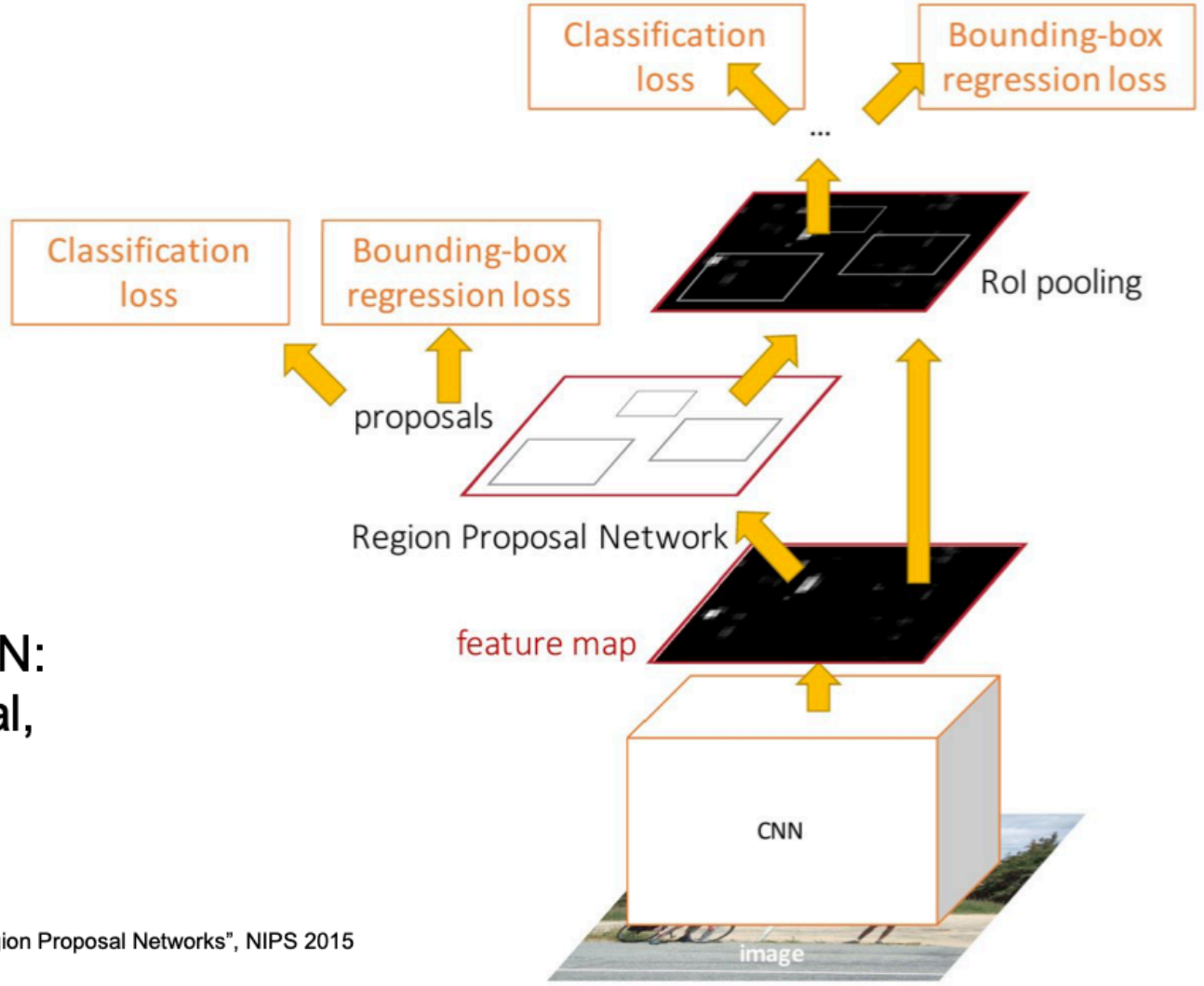
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
He et al, "Spatial pyramid pooling in deep convolutional networks for visual recognition", ECCV 2014
Girshick, "Fast R-CNN", ICCV 2015

Faster R-CNN

Make CNN do proposals!

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

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



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

Region Proposal Network

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

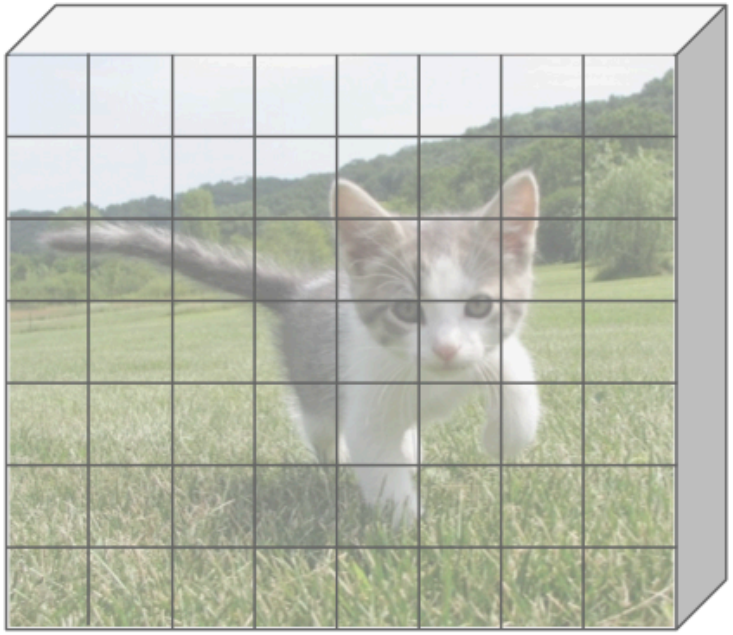


Image features
(e.g. 512 x 20 x 15)

Region Proposal Network

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

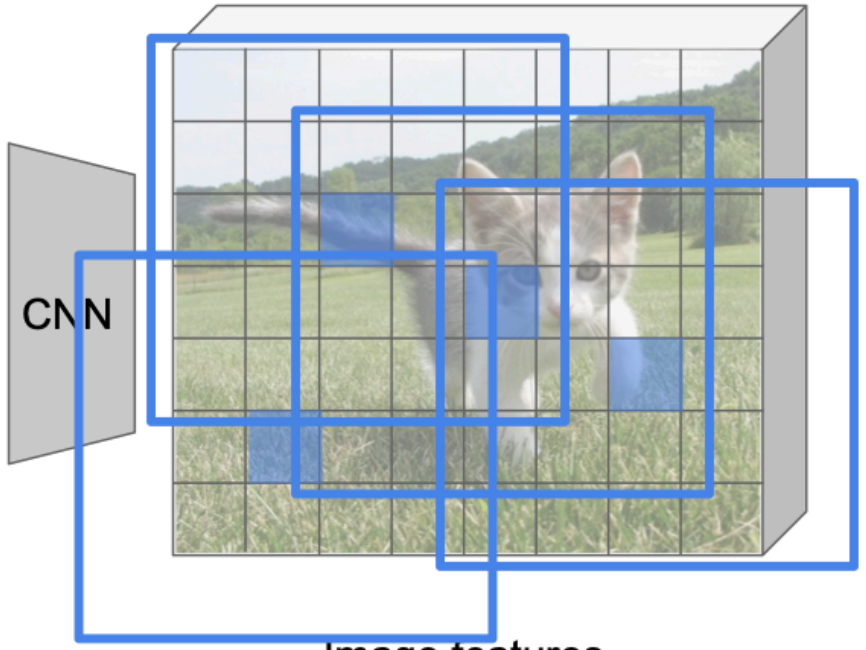


Image features
(e.g. 512 x 20 x 15)

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

Region Proposal Network

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

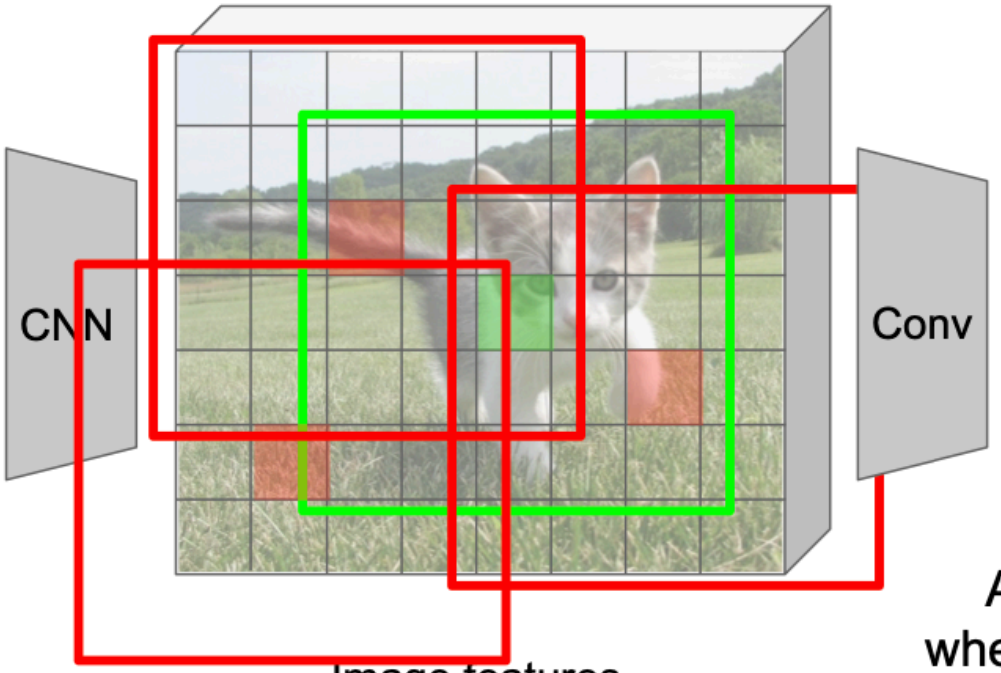


Image features
(e.g. 512 x 20 x 15)

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

Anchor is an object?
1 x 20 x 15

At each point, predict whether the corresponding anchor contains an object (binary classification)

Region Proposal Network

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

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

CNN

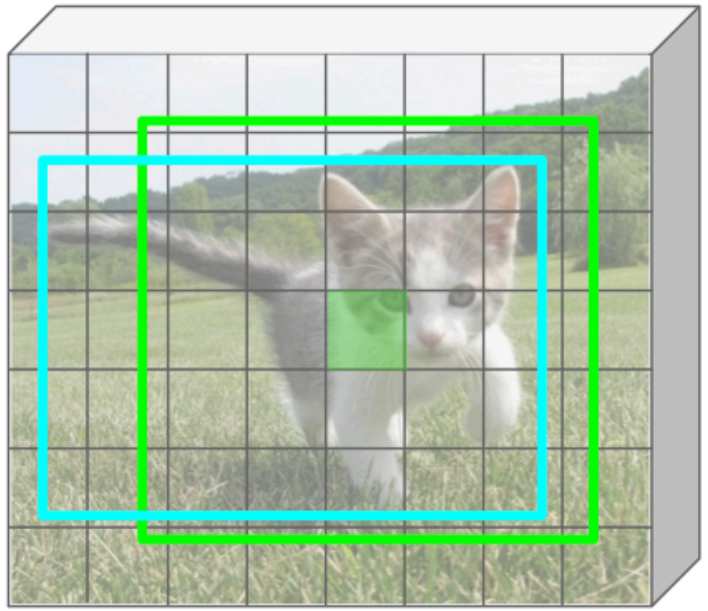


Image features
(e.g. 512 x 20 x 15)

Conv

Anchor is an object?
1 x 20 x 15

Box corrections
4 x 20 x 15

For positive boxes, also predict a corrections from the anchor to the ground-truth box (regress 4 numbers per pixel)

Region Proposal Network

In practice use K different anchor boxes of different size / scale at each point

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

CNN

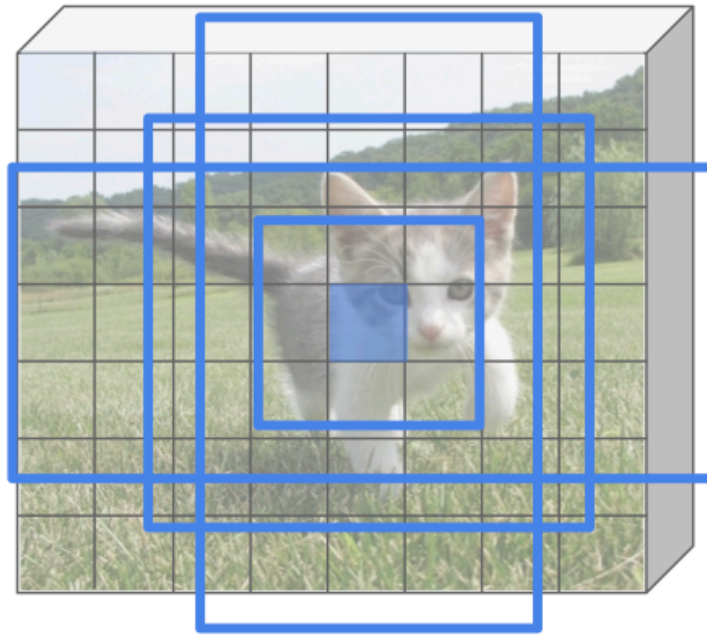


Image features
(e.g. 512 x 20 x 15)

Conv

Anchor is an object?
K x 20 x 15

Box transforms
4K x 20 x 15

Region Proposal Network

In practice use K different anchor boxes of different size / scale at each point

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

CNN

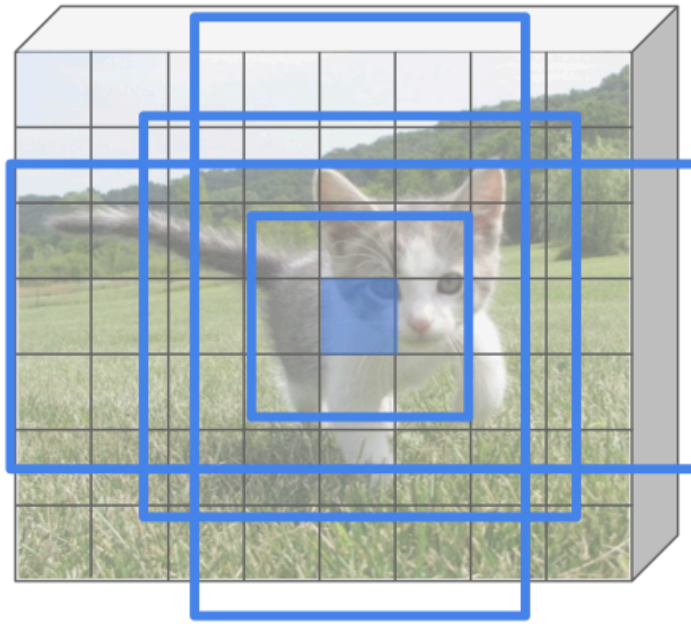


Image features
(e.g. 512 x 20 x 15)

Conv

Anchor is an object?
K x 20 x 15

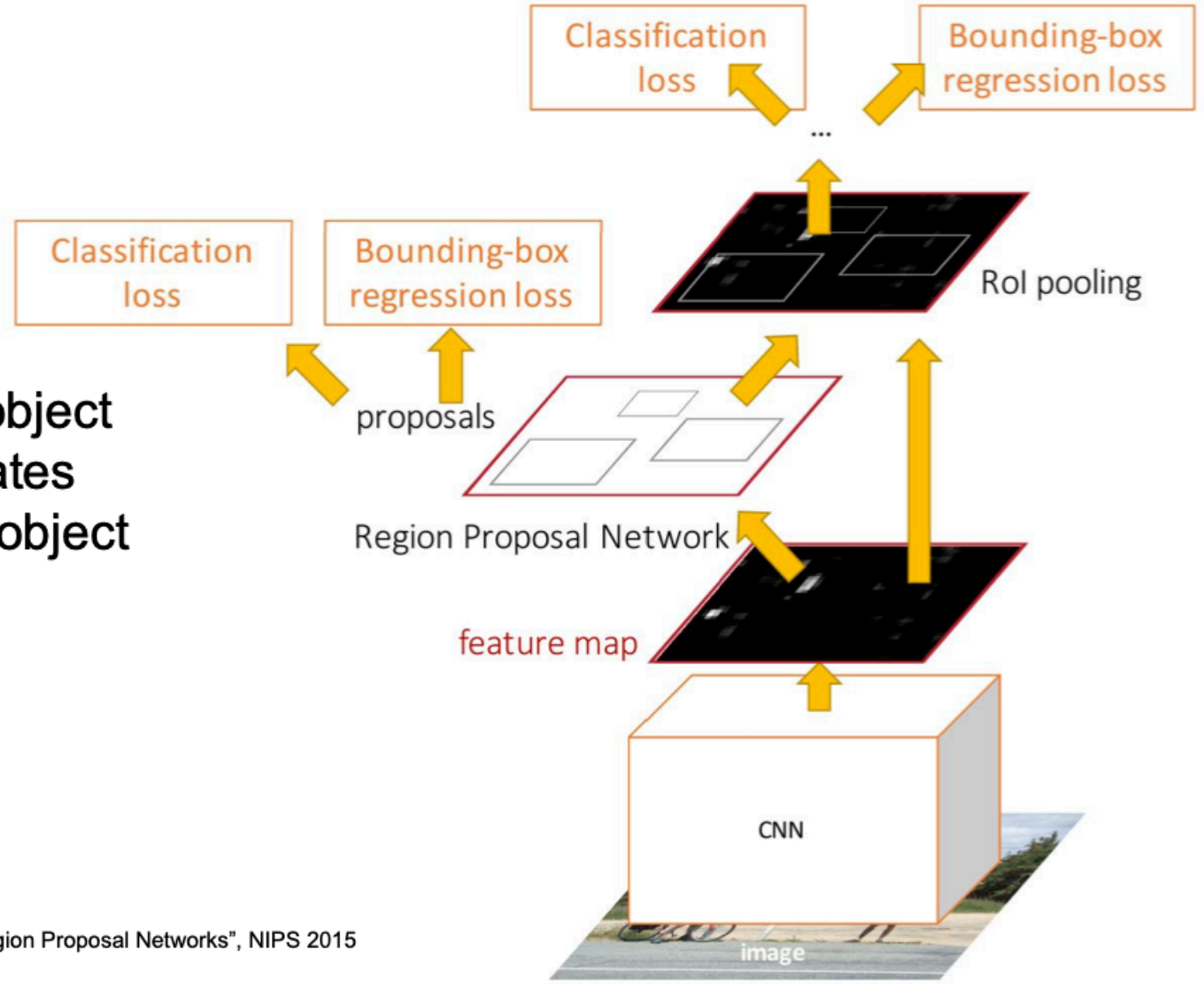
Box transforms
4K x 20 x 15

Sort the K*20*15 boxes by their "objectness" score, take top ~300 as our proposals

Training Faster RCNN

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



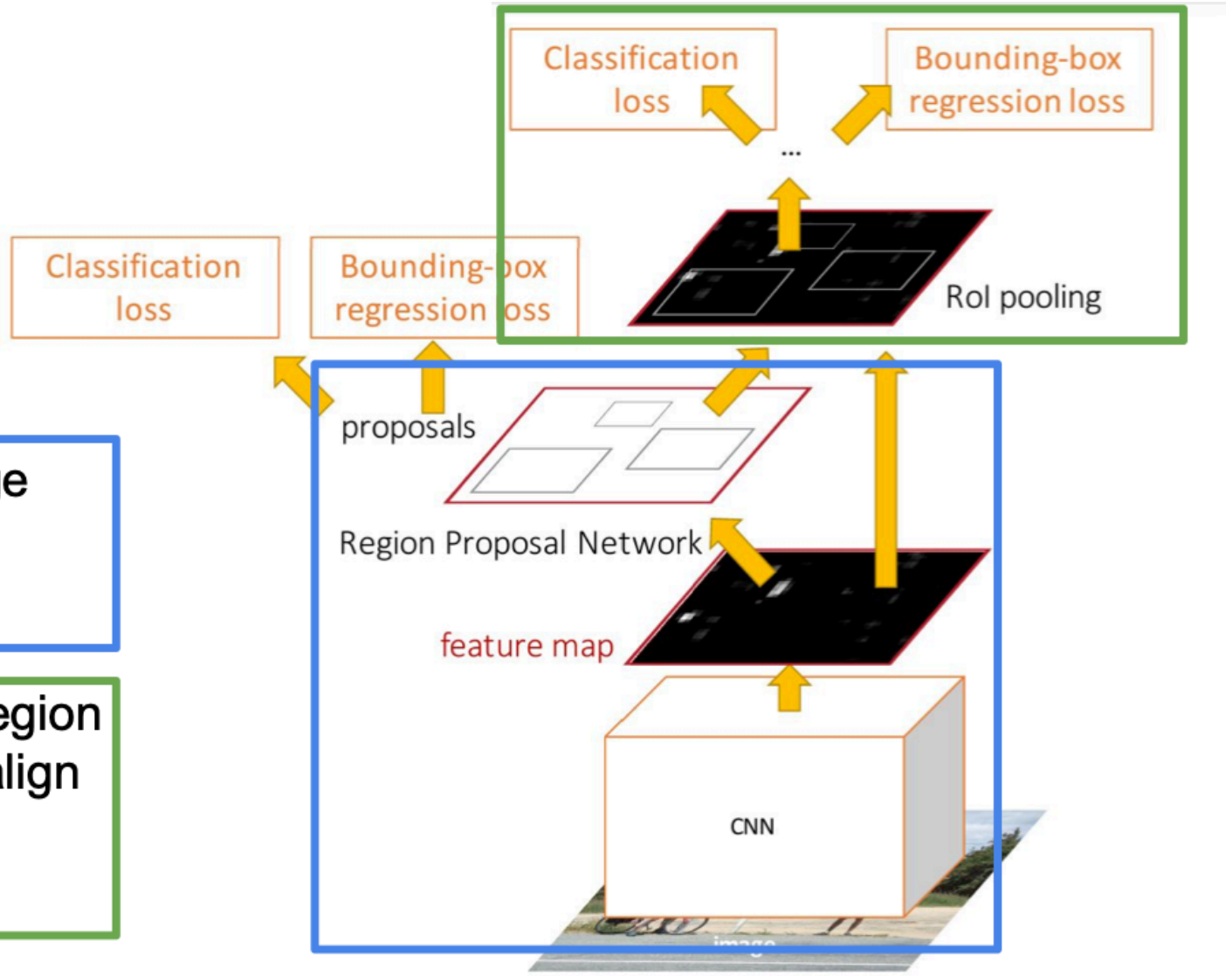
Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission

Inference Time: Two-Stage Detector

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

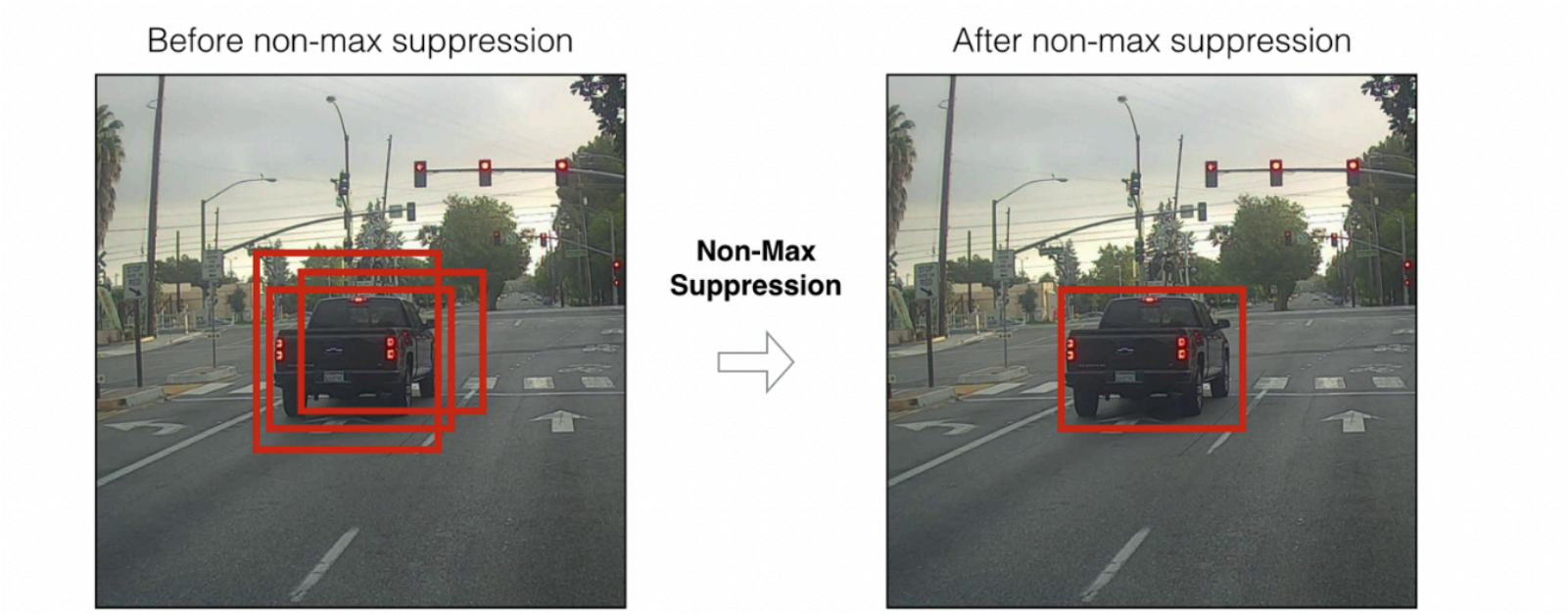


Inference Time

- First stage:
  - Use backbone to extract features
  - Use RPN to generate ~ 300 proposals
- Second stage:
  - For each proposal, predict class label and bbox refinement
  - Perform confidence thresholding to remove low-confidence bbox predictions
  - Perform non-maximal suppression (NMS) for deduplication

Non-Maximal Suppression (NMS)

Input: A list of Proposal boxes B, corresponding confidence scores S (in Faster RCNN, simply the classification score) and IoU threshold τ.
Output: A list of detected bounding boxes D.



https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c

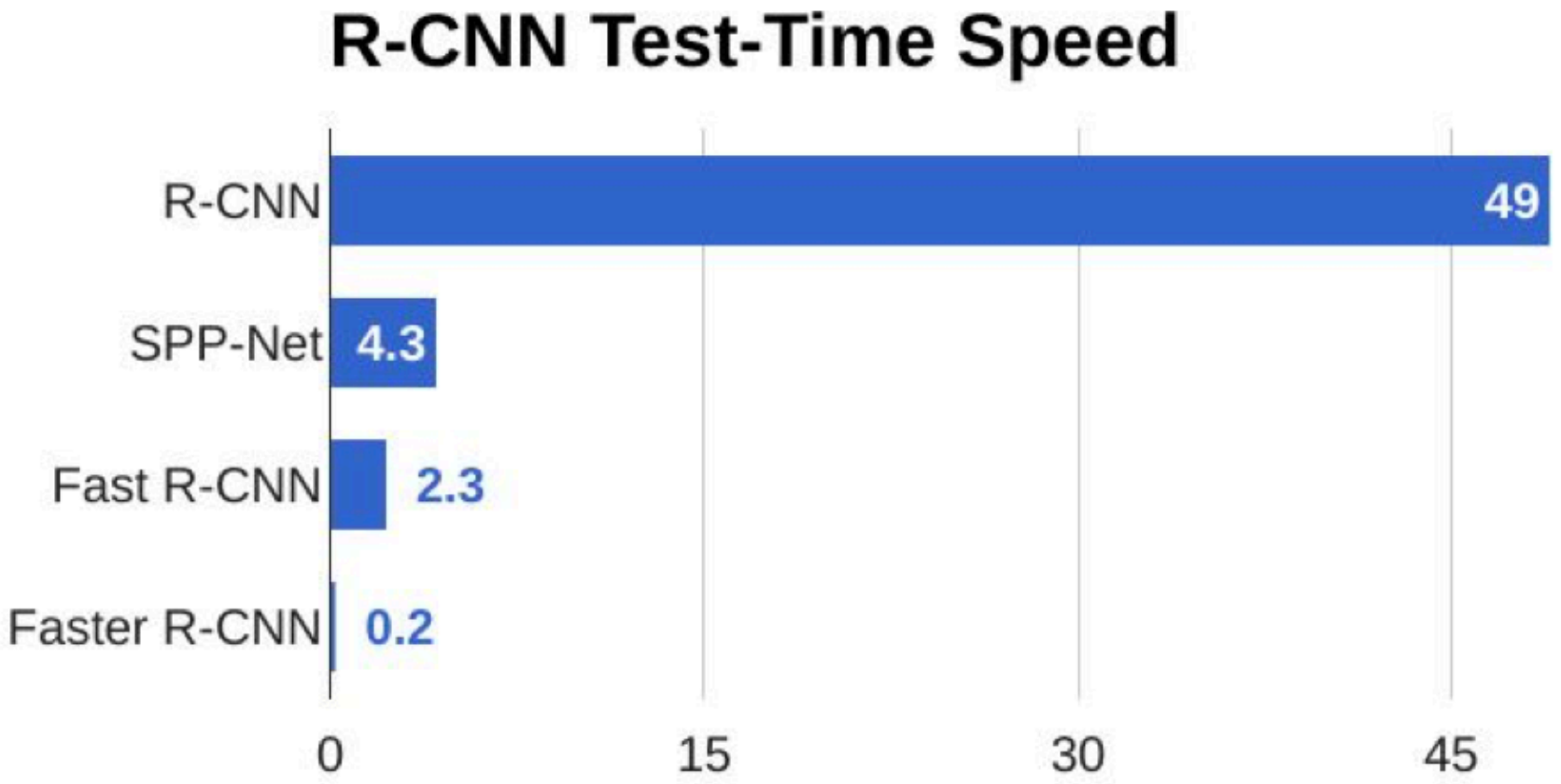
NMS

Algorithm:

- Initially D is empty
- Select the proposal with highest confidence score, remove it from B and add it to the final detection list D.
- Now compare this proposal with all the proposals — calculate the IoU of this proposal with every other proposal. If the IOU is greater than the threshold τ, remove that proposal from B.
- Again take the proposal with the highest confidence from the remaining proposals in B and remove it from B and add it to D.
- Once again calculate the IOU of this proposal with all the proposals in B and eliminate the boxes which have a IoU higher than τ.
- This process is repeated until there are no more proposals left in B.

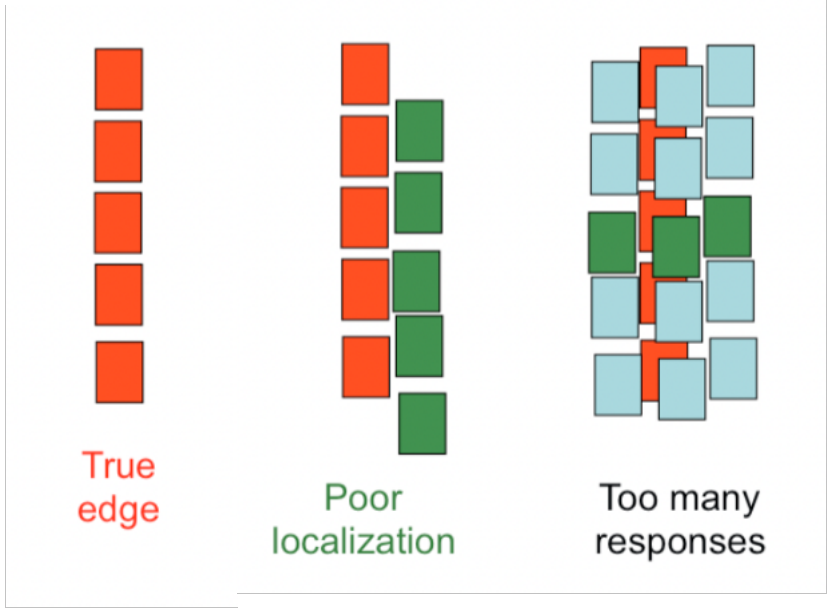
https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c

Speed Comparison



How to Evaluate Detection?

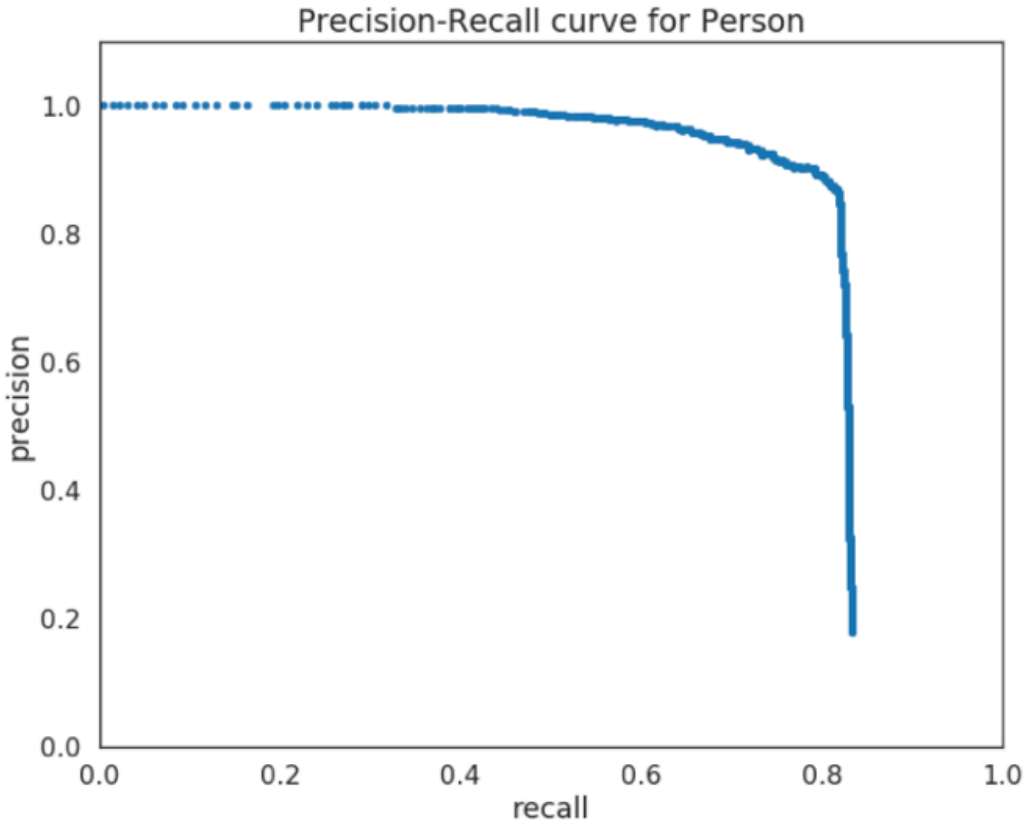
Recall from optimal edge detection, Lecture 02



- Good accuracy (precision): minimize false positive
- Good localization (precision): maximize IoU
- Single response constraint (precision): minimize redundant responses
- Good coverage (recall): make sure all edges are detected.

Evaluation Metric: AP (Average Precision)

- Per category rank the output bounding boxes according to the confidence (classification score) in a descending order.
- Select top n outputs and compute recall.
- Precision: the ratio of bboxes that satisfy IoU > x% threshold
- Compute the area under precision-recall curve (approximate by 11 points).

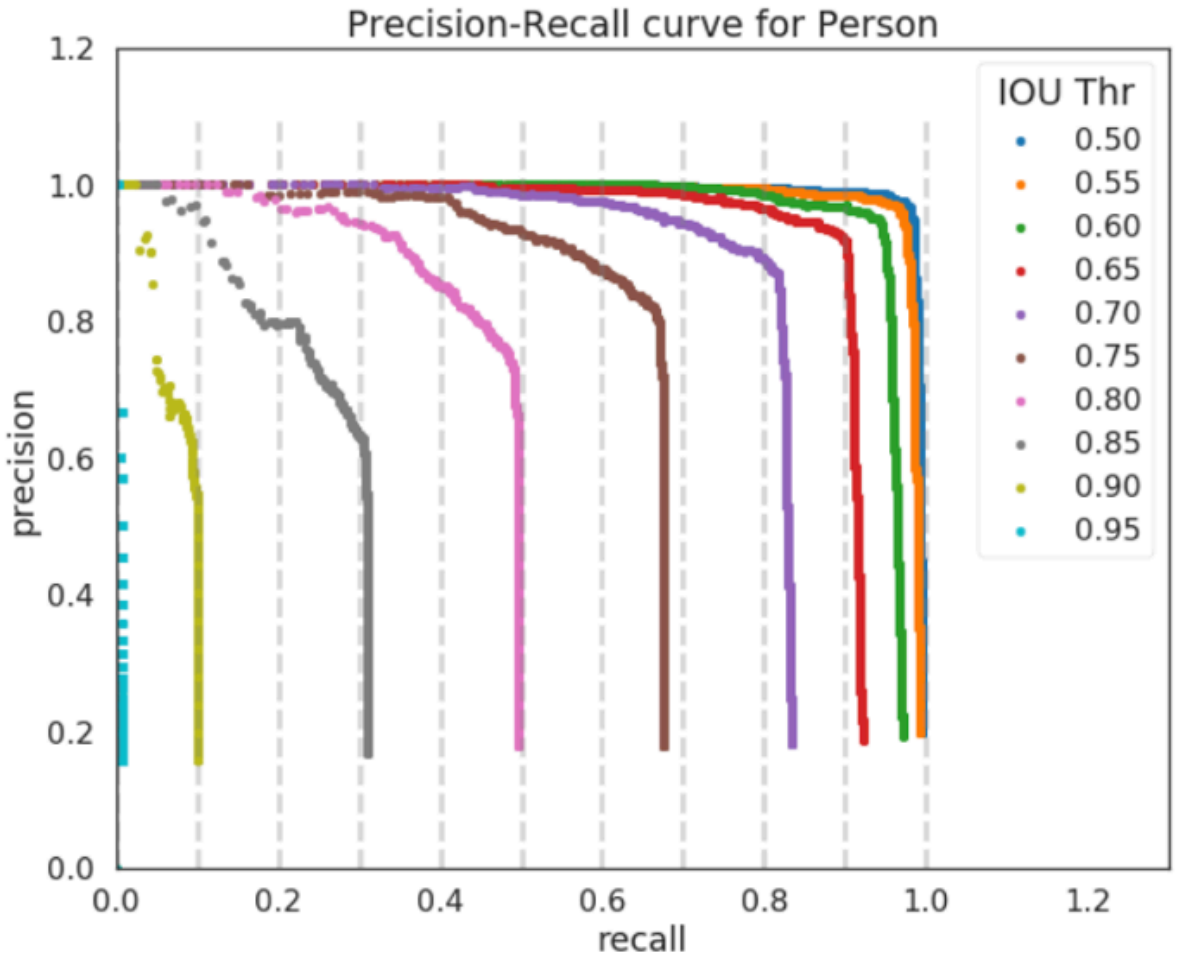


$$AP = \frac{1}{11} \sum_{\text{Recall}_i} \text{Precision}(\text{Recall}_i)$$

Recall_i = [0, 0.1, 0.2, …, 1.0].

https://medium.com/@timothycarlen/understanding-the-map-evaluation-metric-for-object-detection-a07fe6962cf3

Evaluation Metric: AP at Different IoU Thres.



Precision-Recall curves calculated at various IoU thresholds, according to the COCO challenge. Dashed lines correspond to equally spaced recall values where the AP is calculated.

https://medium.com/@timothycarlen/understanding-the-map-evaluation-metric-for-object-detection-a07fe6962cf3

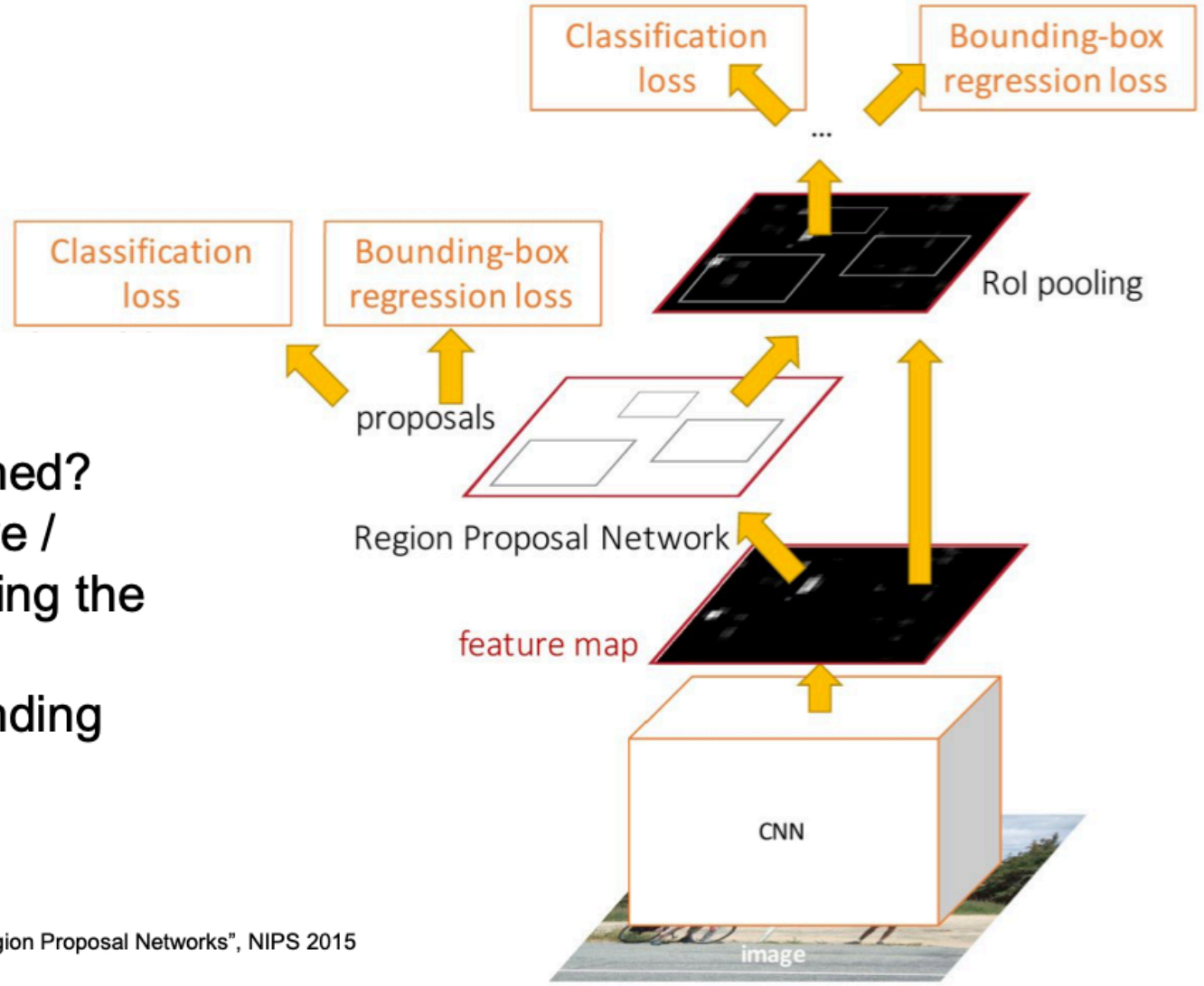
Evaluation Metric: mAP

- mAP is the mean of AP across different categories and/or IoU thresholds. Sometimes m is ignored.
- Examples when evaluating on MS COCO:
  - AP
  - AP_{50}

Faster RCNN

Glossing over many details:

- How are anchors determined?
- How do we sample positive / negative samples for training the RPN?
- How to parameterize bounding box regression?



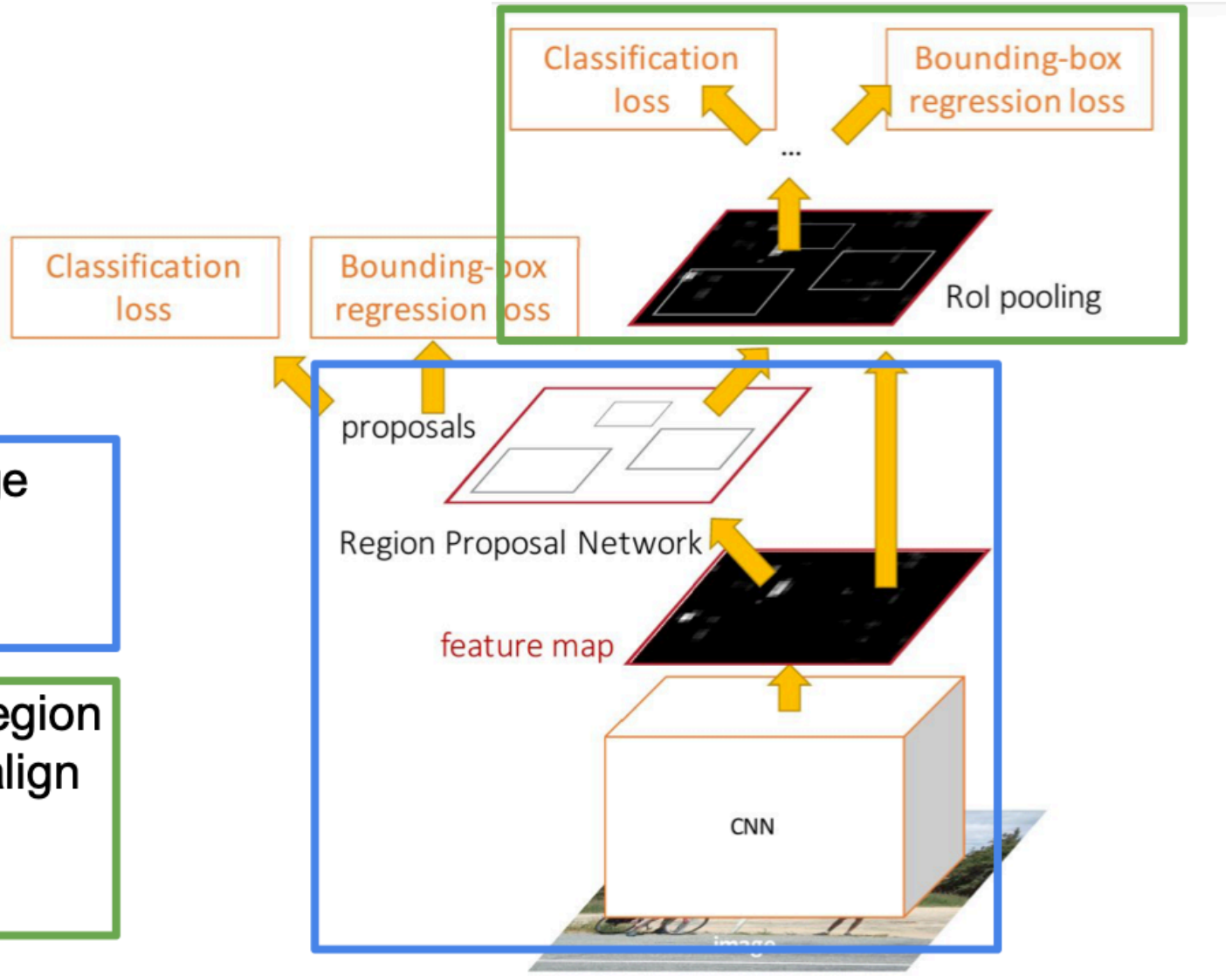
Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission

Two-Stage Detector

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



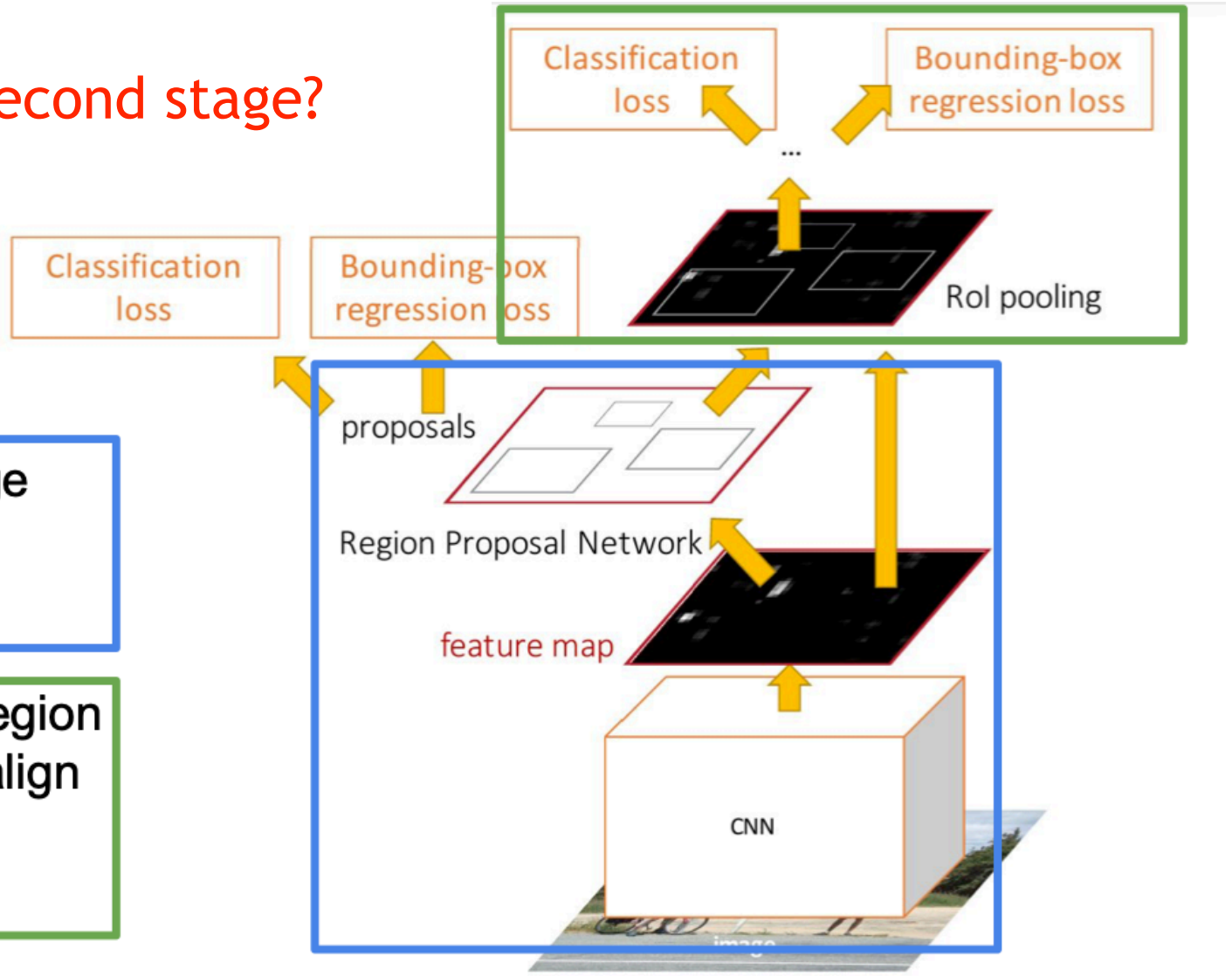
Two-Stage Detector

Do we really need the second stage?

Faster R-CNN is a
Two-stage object detector

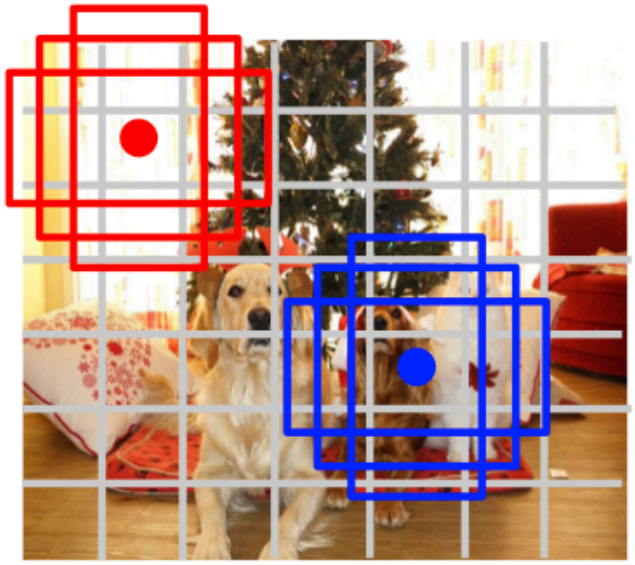
First stage: Run once per image
- Backbone network
- Region proposal network

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



Single-Stage Detectors: YOLO/SSD/RetinaNet

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)
- Looks a lot like RPN, but category-specific!

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
Lin et al, "Focal Loss for Dense Object Detection", ICCV 2017

Object Detection: Lots of Variables

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

"Meta-Architecture"
Two-stage: Faster R-CNN
Single-stage: YOLO / SSD
Hybrid: R-FCN

Image Size
 # Region Proposals
...

Takeaways
Faster R-CNN is slower but more accurate

SSD is much faster but not as accurate

Bigger / Deeper backbones work better

Huang et al, "Speed/accuracy trade-offs for modern convolutional object detectors", CVPR 2017
Zou et al, "Object Detection in 20 Years: A Survey", arXiv 2019

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

Some slides are borrowed from Stanford CS231N.

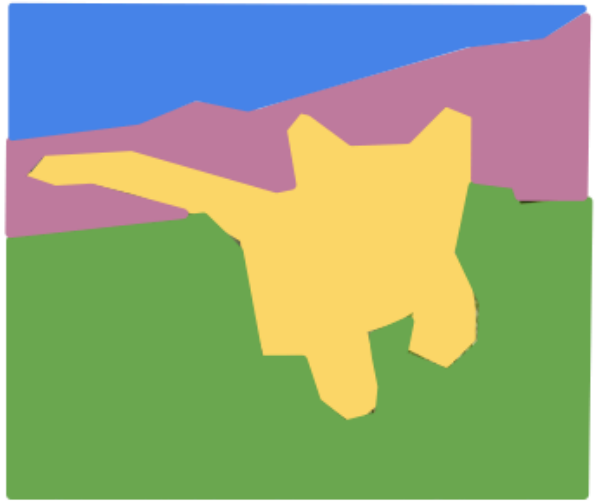
Computer Vision Tasks

Classification

CAT

No spatial extent

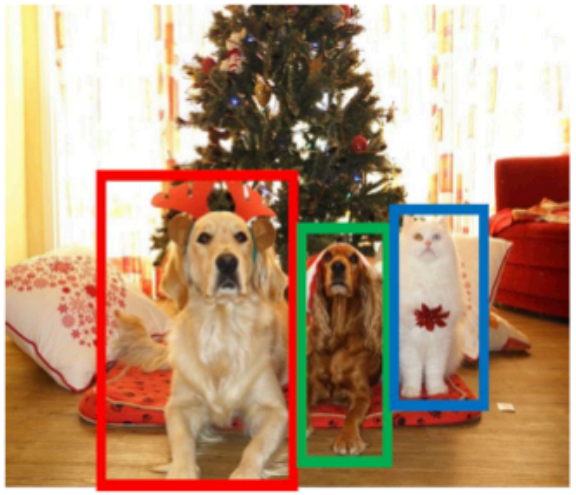
Semantic Segmentation



GRASS, CAT, TREE, SKY

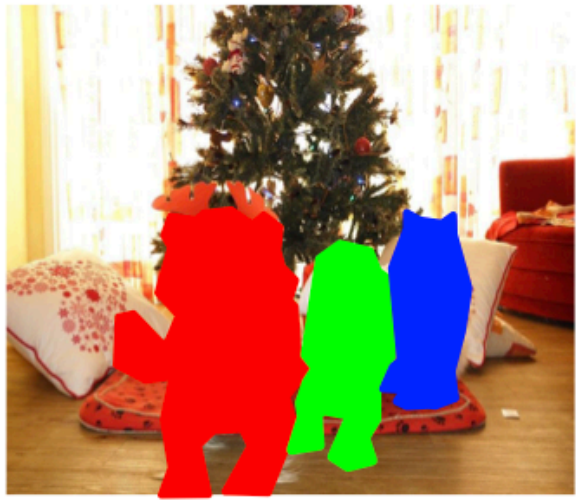
No objects, just pixels

Object Detection



DOG, DOG, CAT

Instance Segmentation

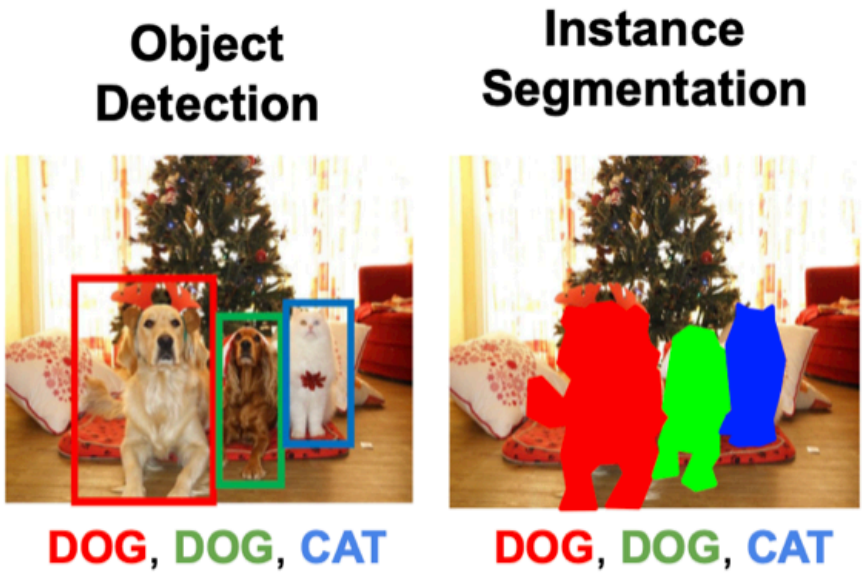


DOG, DOG, CAT

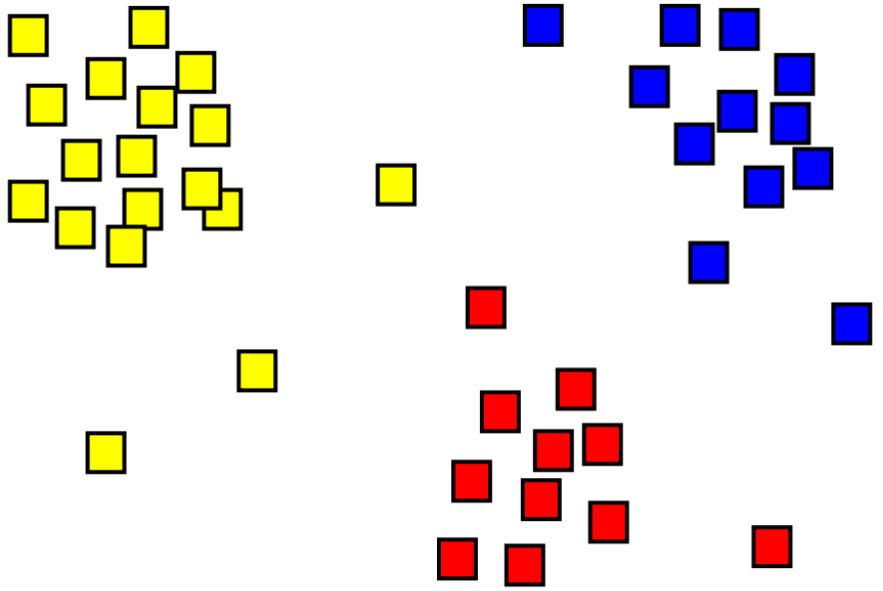
Multiple Object

Different Approaches for Instance Segmentation

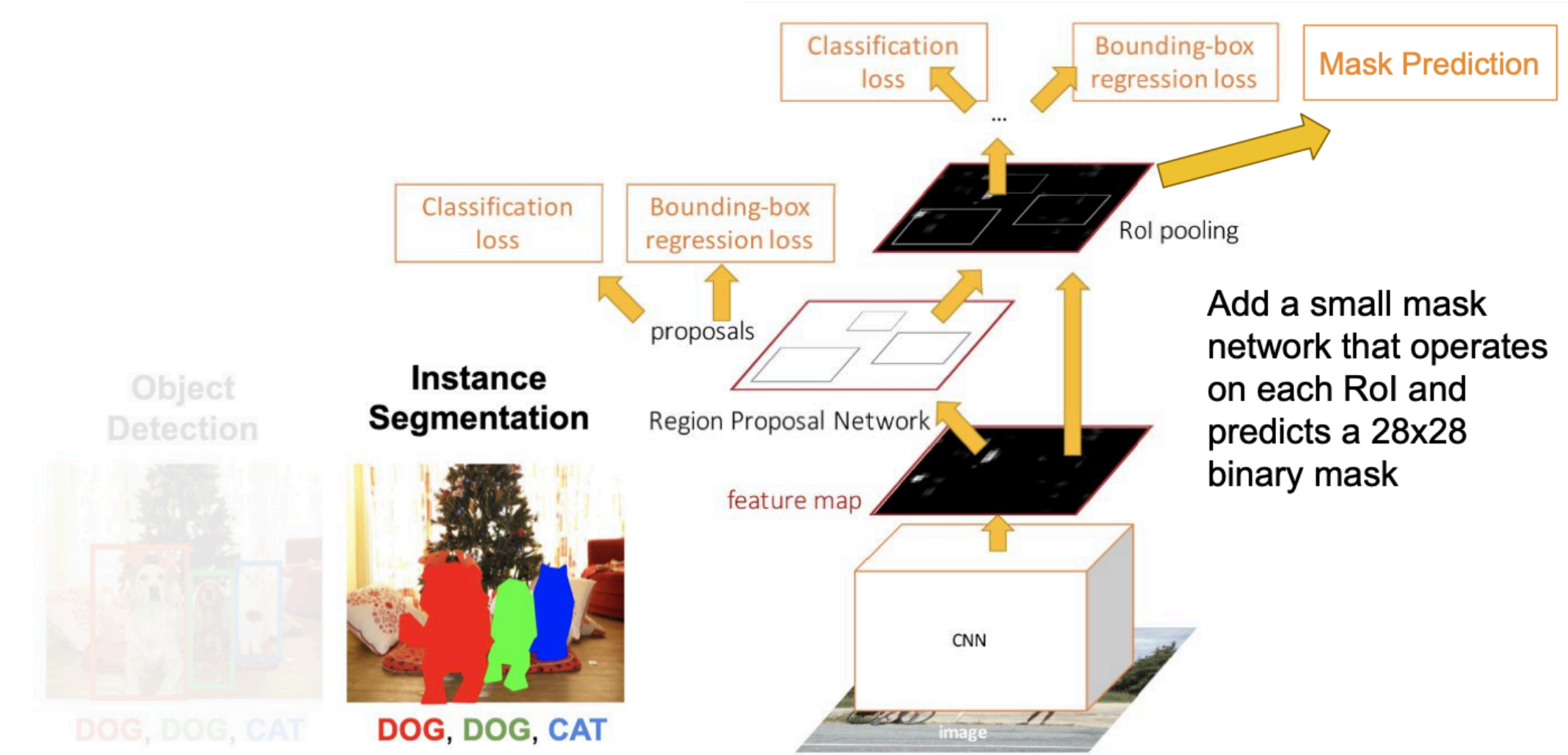
- Top-down approach:object detection and then further find a binary mask inside the bounding box



- Bottom-up approach:grouping and then classification
  - Grouping: group together similar data points and represents them with a single token

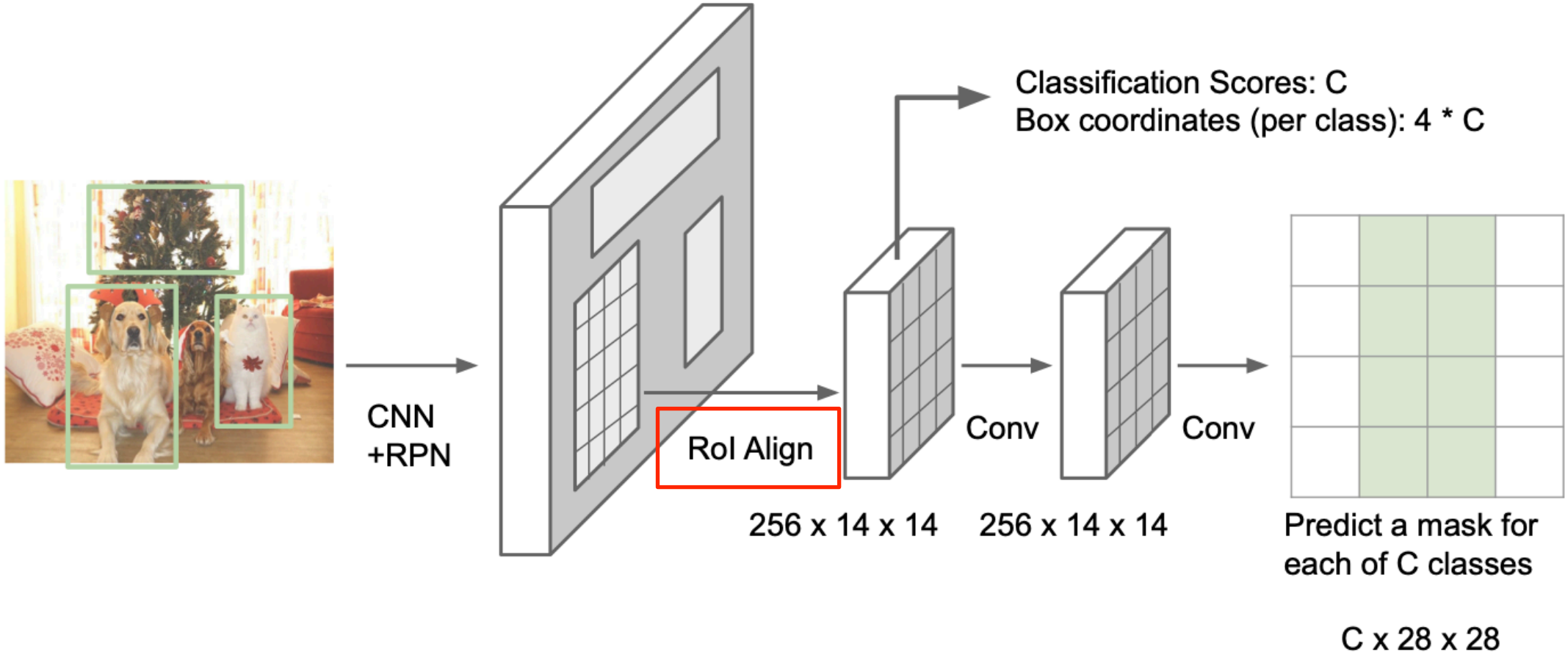


Top-Down Approach: Mask R-CNN



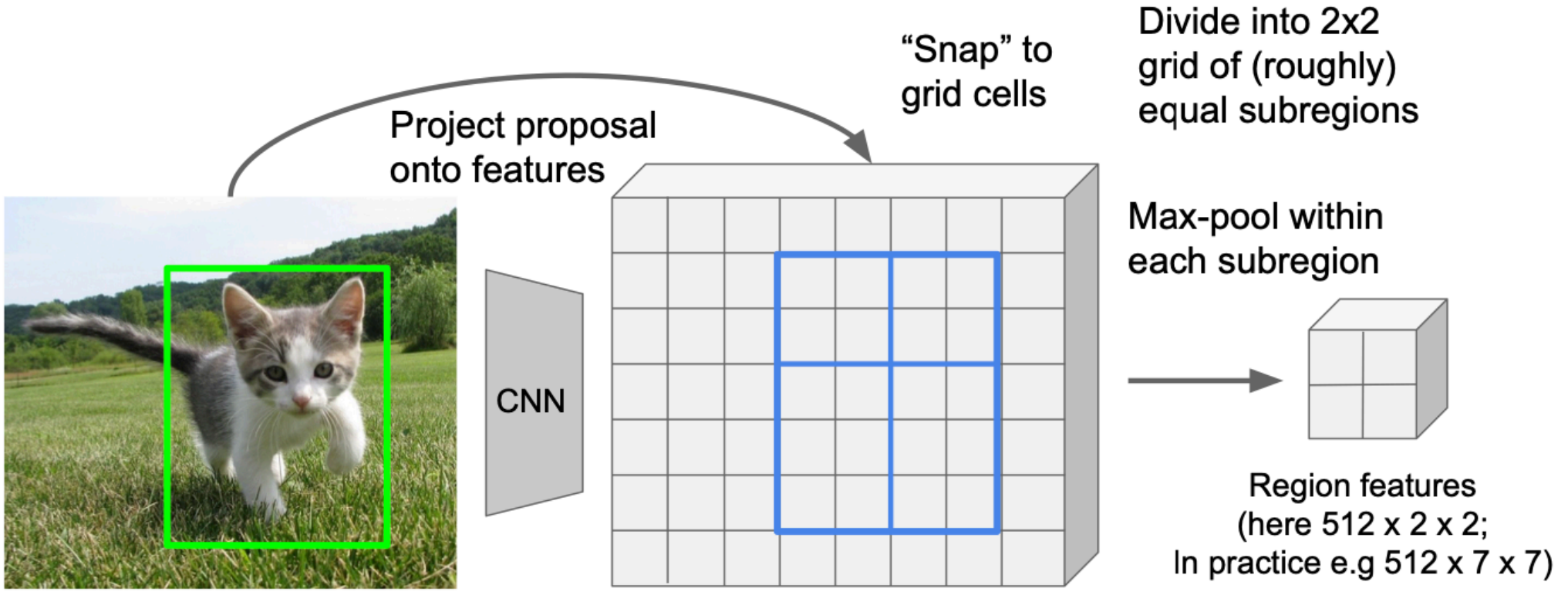
He et al, "Mask R-CNN", ICCV 2017

Mask R-CNN



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

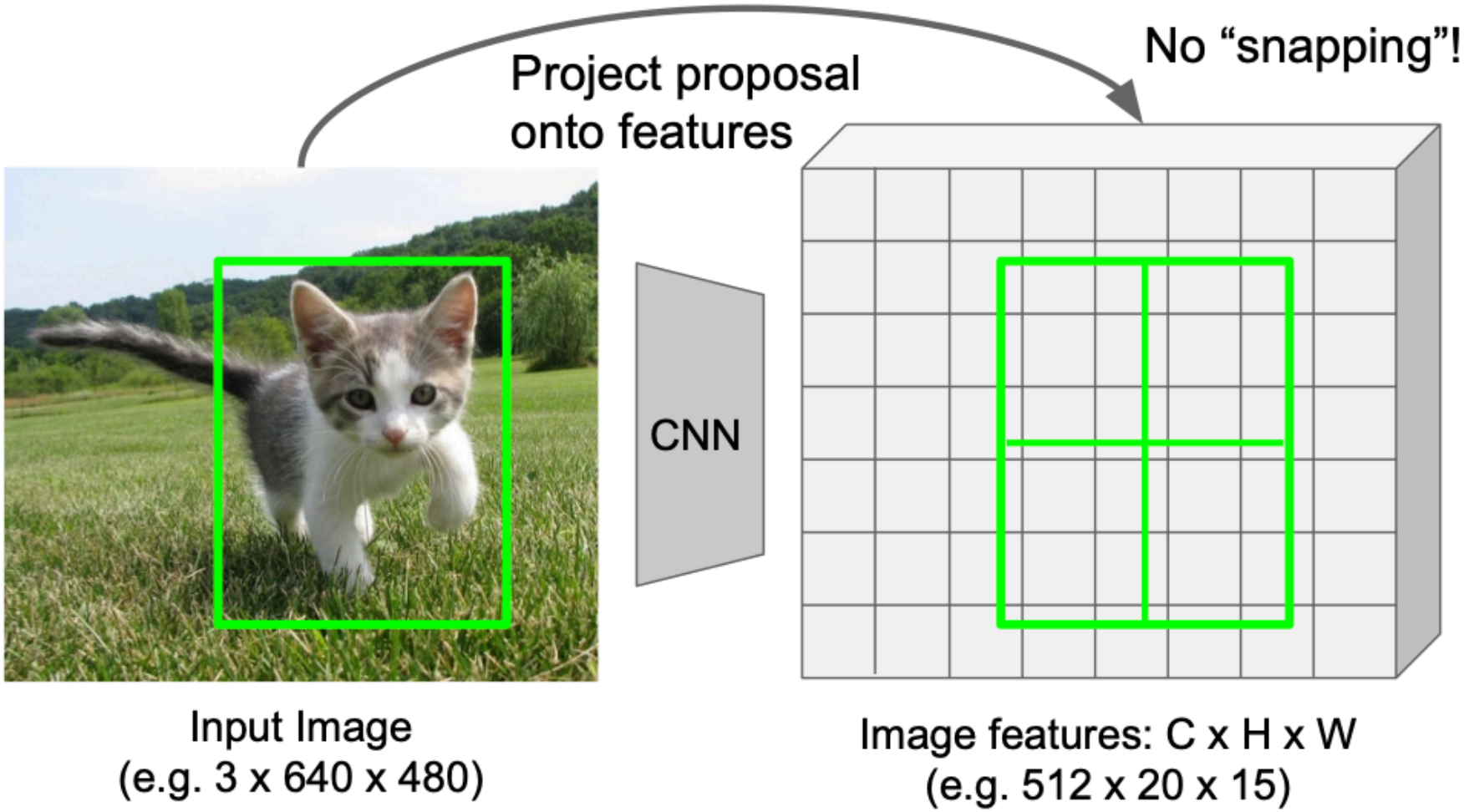
Problems with RoI Pool



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

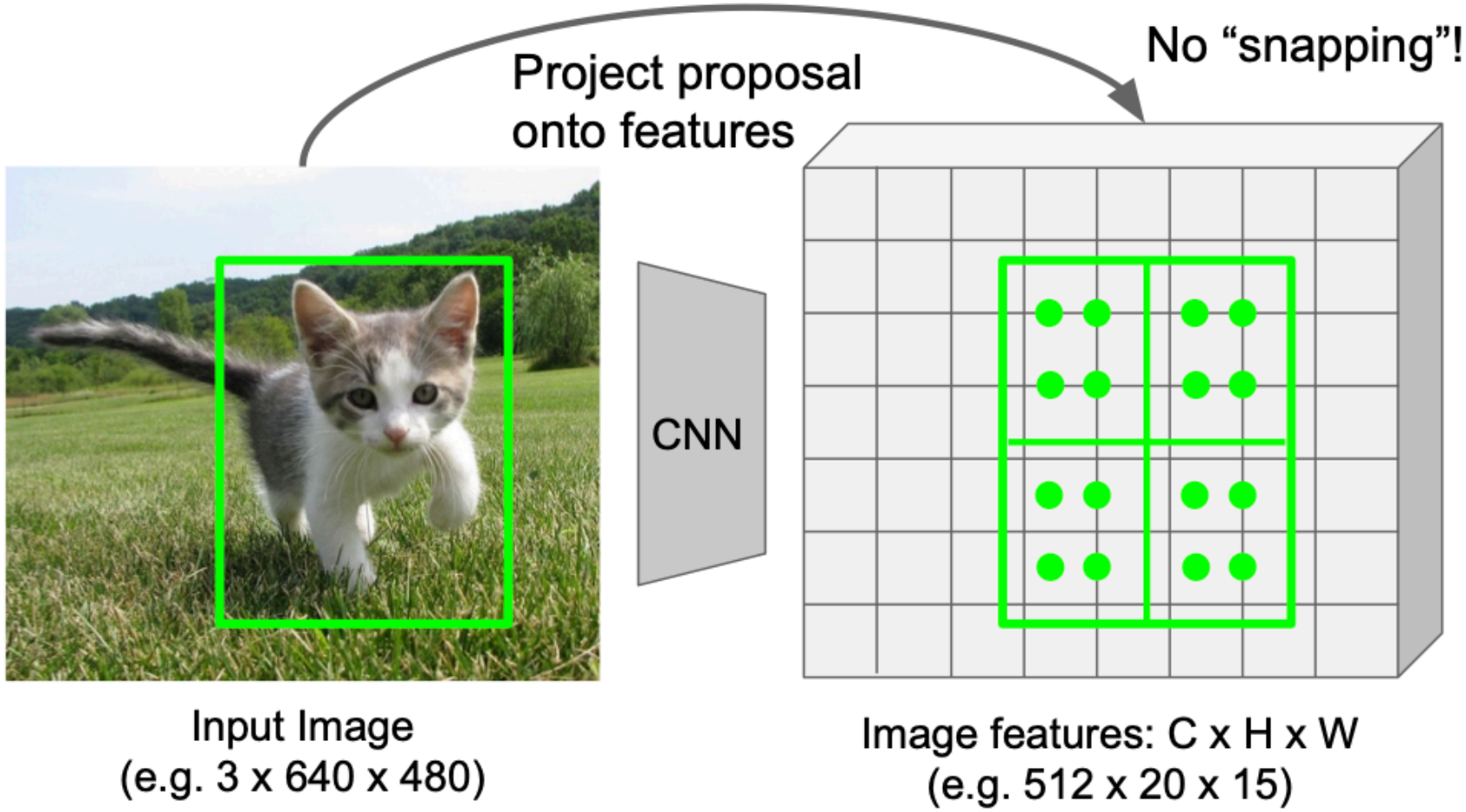
Problem: Region features slightly misaligned

RoI Align



He et al, “Mask R-CNN”, ICCV 2017

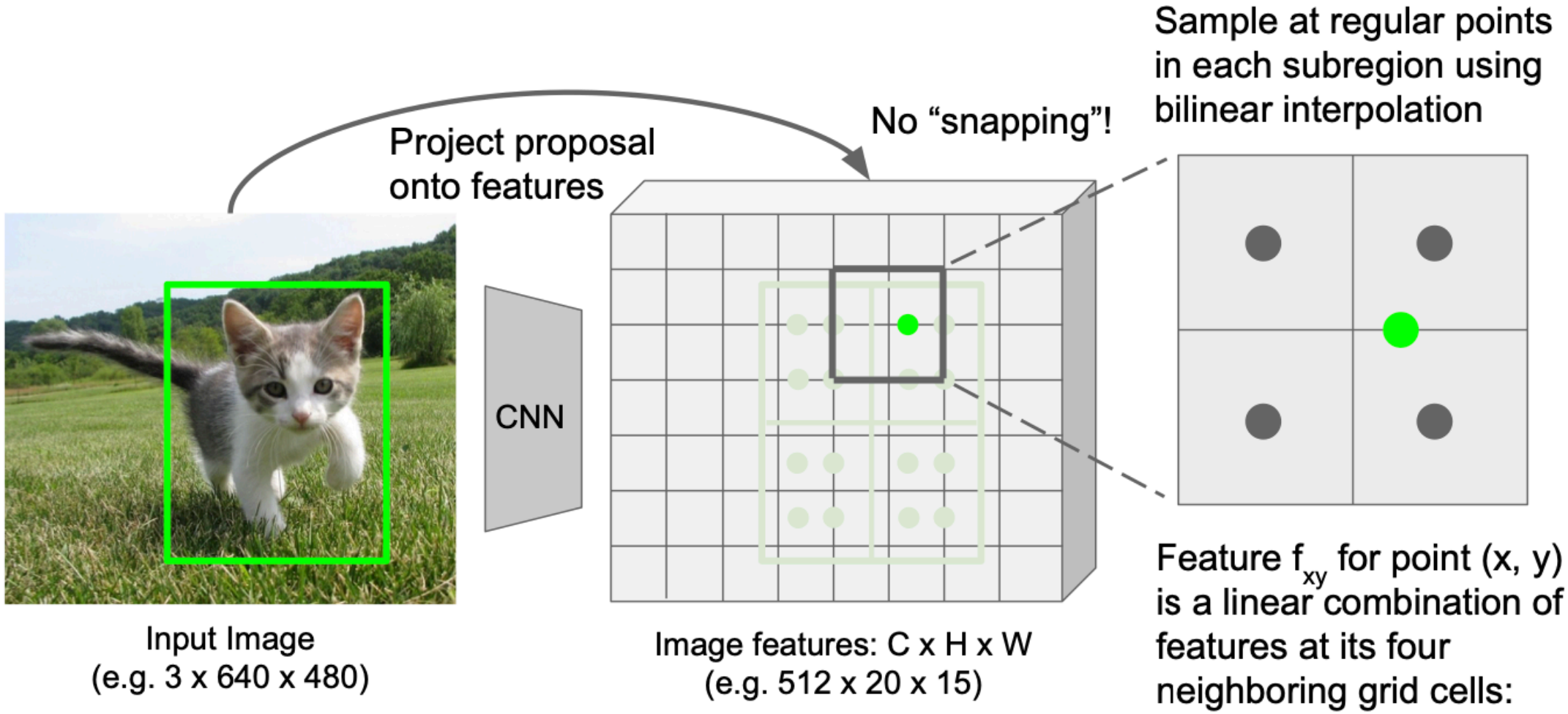
RoI Align



Sample at regular points in each subregion using bilinear interpolation

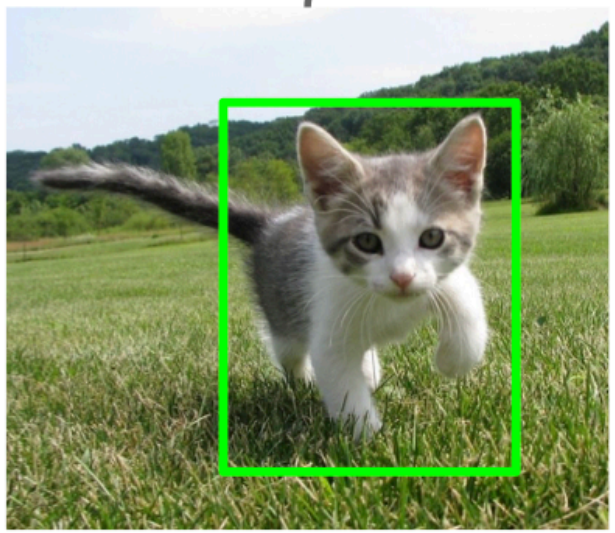
He et al, “Mask R-CNN”, ICCV 2017

RoI Align



He et al, “Mask R-CNN”, ICCV 2017

RoI Align



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

Project proposal onto features

No "snapping"!

CNN

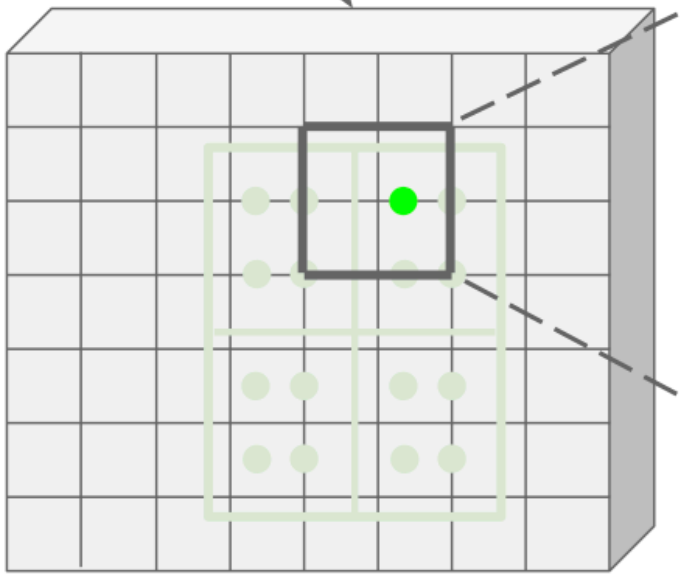
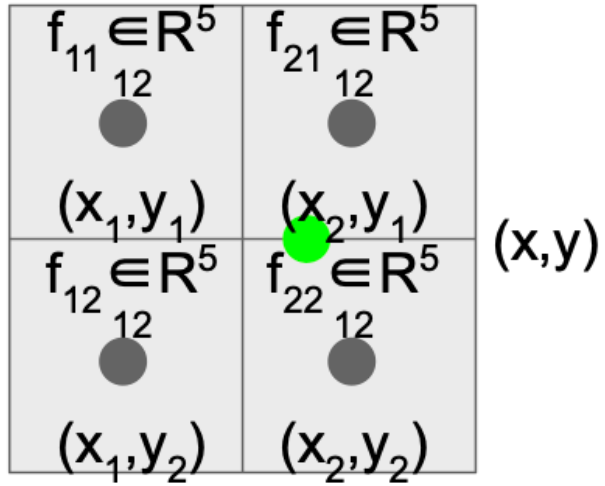


Image features: C x H x W
(e.g. 512 x 20 x 15)

Sample at regular points in each subregion using bilinear interpolation

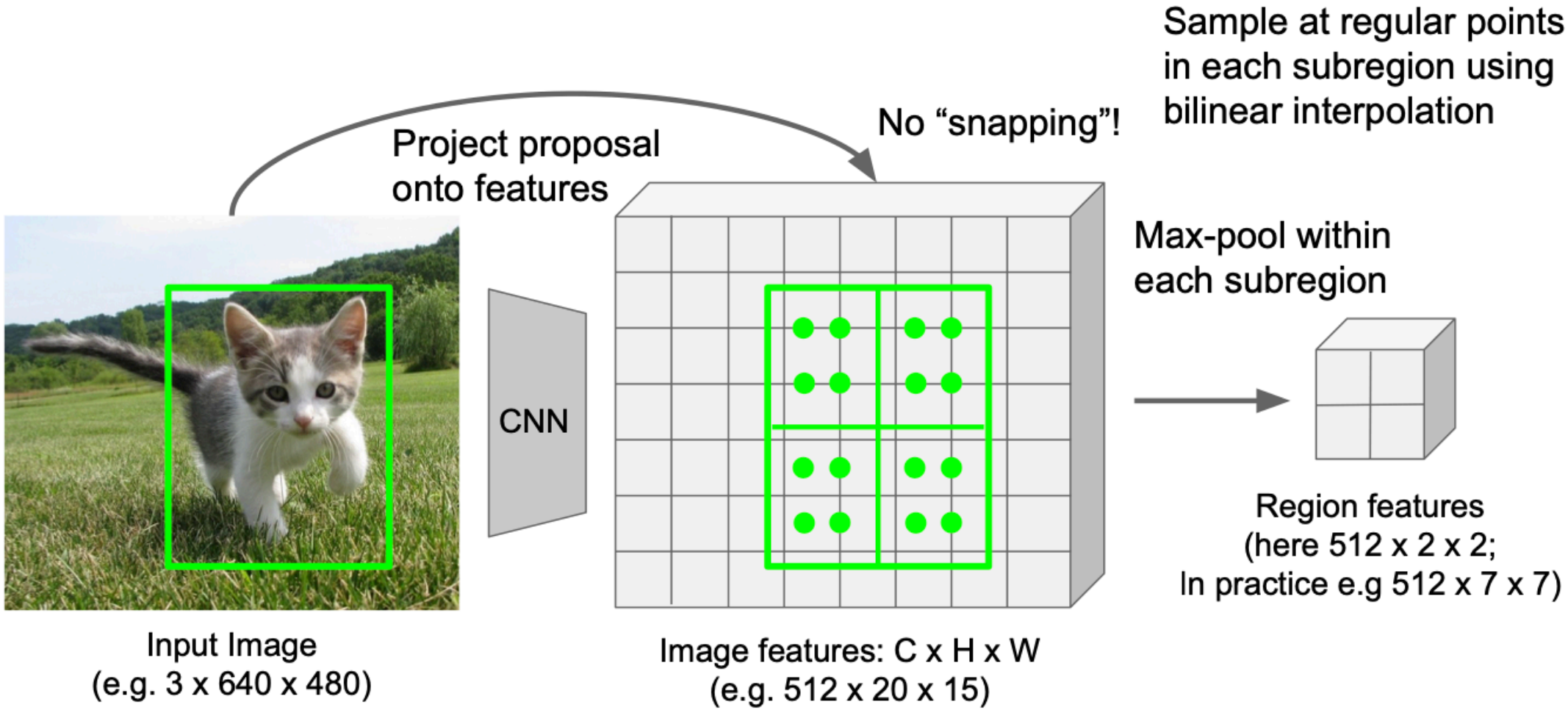


Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

$$f_{xy} = \sum_{i,j=1}^{2} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_j|)$$

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

RoI Align



He et al, “Mask R-CNN”, ICCV 2017

Ablation Study on RoI Align

	AP	AP_{50}	AP_{75}	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}
RoIPool	23.6	46.5	21.6	28.2	52.7	26.9
RoIAlign	**30.9**	**51.8**	**32.1**	**34.0**	**55.3**	**36.4**
	+7.3	*+ 5.3*	*+10.5*	*+5.8*	*+2.6*	*+9.5*

(d) **RoIAlign** (ResNet-50-**C5**, *stride 32*): Mask-level and box-level AP using *large-stride* features. Misalignments are more severe than with stride-16 features (Table 2c), resulting in big accuracy gaps.

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

Class-Specific vs. Class-Agnostic Masks

- Our default instantiation predicts class-specific masks, *i.e.*, one m×m mask per class.
- Mask R-CNN with class-agnostic masks (*i.e.*, predicting a single m×m output regardless of class) is nearly as effective.

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

Multinomial vs. Independent Masks

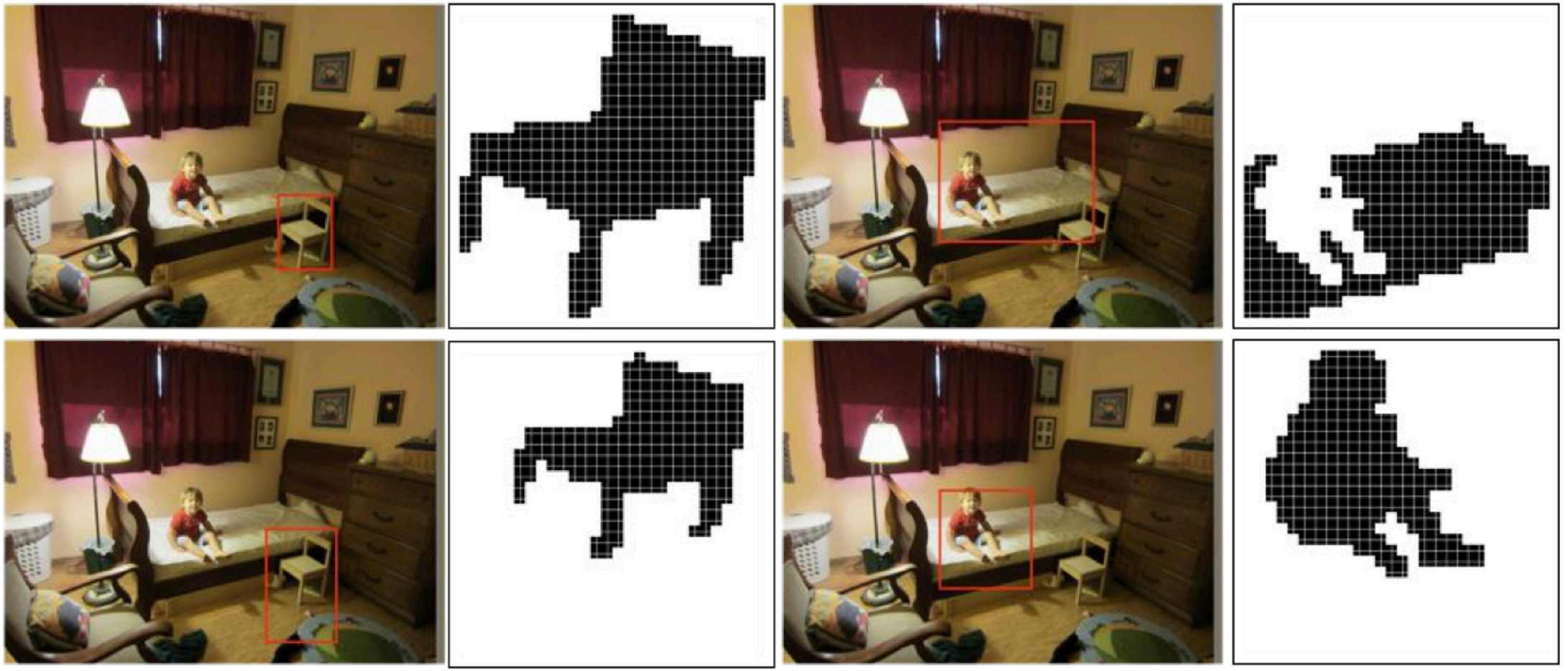
- Decouples mask and class prediction
- Generate a mask for each class without competition among classes (by a per-pixel *sigmoid* and a *binary* loss).

	AP	AP_{50}	AP_{75}
softmax	24.8	44.1	25.1
sigmoid	**30.3**	**51.2**	**31.5**
	+5.5	*+7.1*	*+6.4*

(b) **Multinomial *vs.* Independent Masks** (ResNet-50-C4): *Decoupling* via per-class binary masks (sigmoid) gives large gains over multinomial masks (softmax).

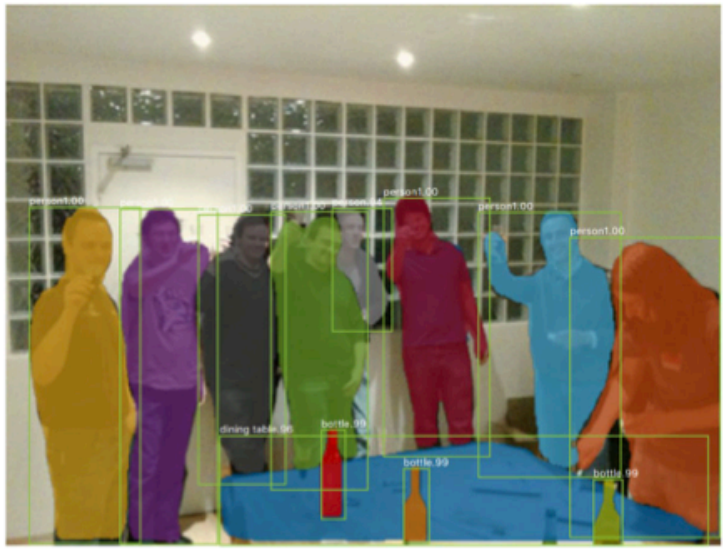
He et al, "Mask R-CNN", ICCV 2017

Mask RCNN: Example Mask Training Target



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

Result Visualization



He et al, “Mask R-CNN”, ICCV 2017

Human Pose Visualization

He et al, “Mask R-CNN”, ICCV 2017

Open Source Framework

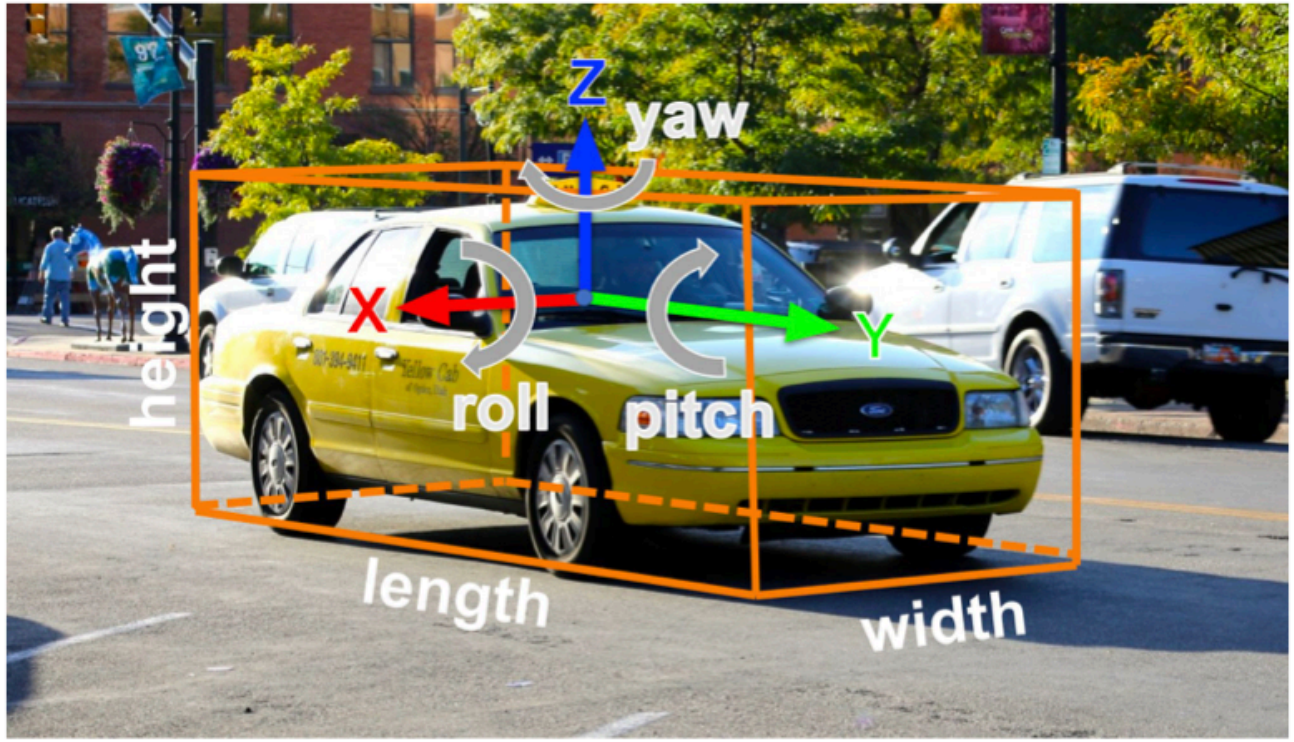
- Lots of good implementations on GitHub!

- TensorFlow Detection API:
  - https://github.com/tensorflow/models/tree/master/research/object_detection
  - Faster RCNN, SSD, RFCN, Mask R-CNN, ...

- Detectron2 (PyTorch) :
  - https://github.com/facebookresearch/detectron2
  - Mask R-CNN, RetinaNet, Faster R-CNN, RPN, Fast R-CNN, R-FCN, ... Finetune on your own dataset with pre-trained models

3D Object Detection and Instance Segmentation

3D Object Detection



2D Object Detection:
2D bounding box
(x, y, w, h)

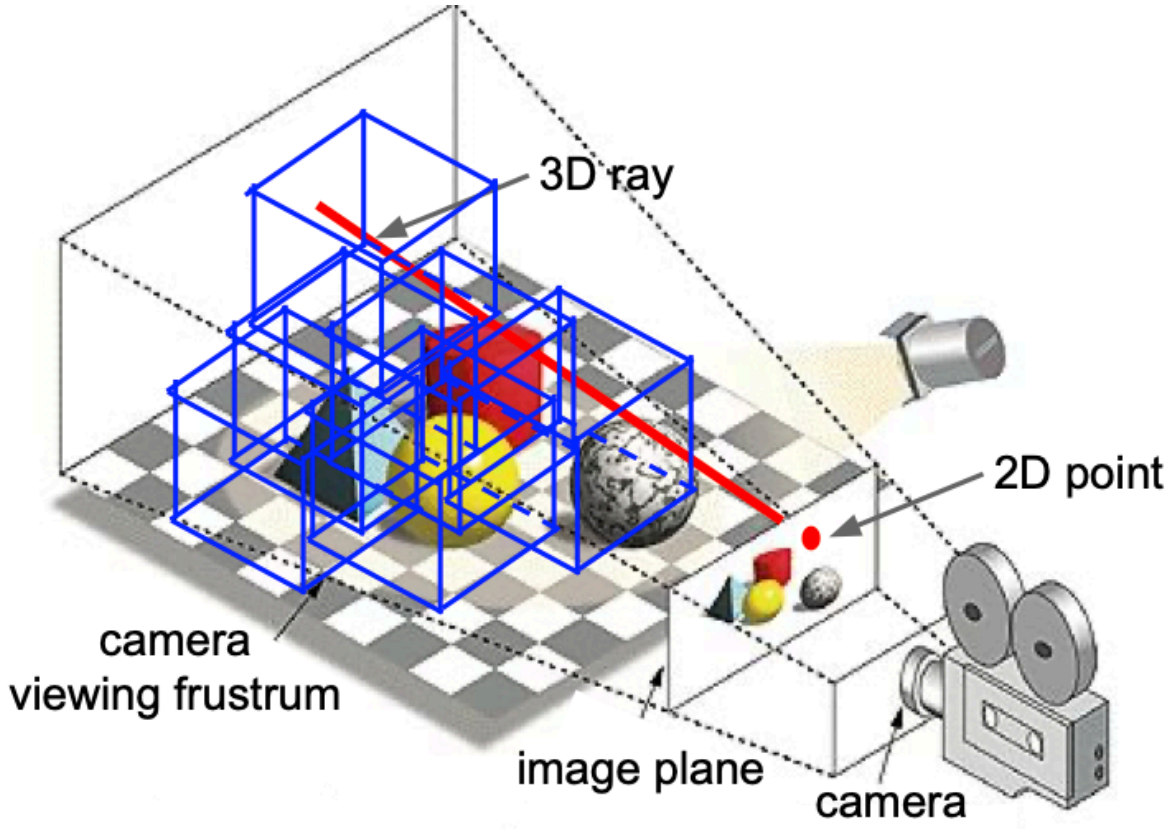
3D Object Detection:
3D oriented bounding box
(x, y, z, w, h, l, r, p, y)

Simplified bbox: no roll & pitch

Much harder problem than 2D object detection!

Slides credit: Stanford CS231N

3D Object Detection



A point on the image plane corresponds to a **ray** in the 3D space

A 2D bounding box on an image is a **frustrum** in the 3D space

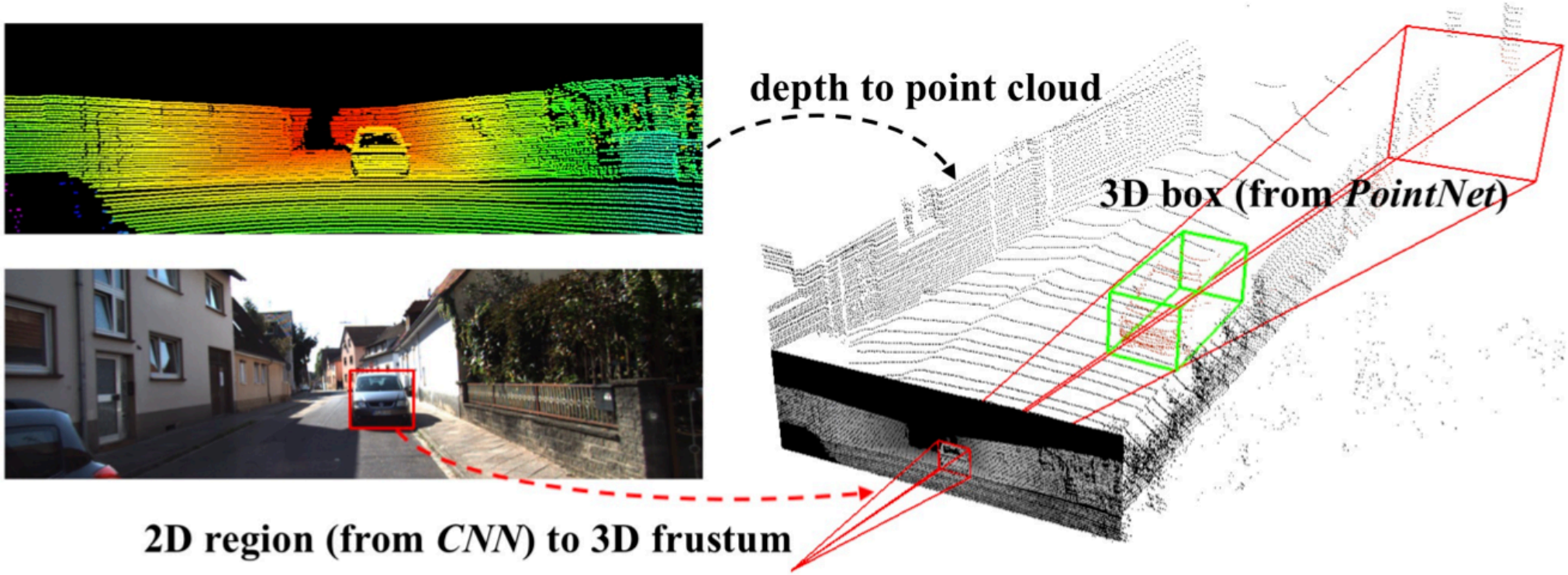
Localize an object in 3D:
The object can be anywhere in the **camera viewing frustrum**!

Image source: https://www.pcmag.com/encyclopedia_images/_FRUSTUM.GIF

Slides credit: Stanford CS231N

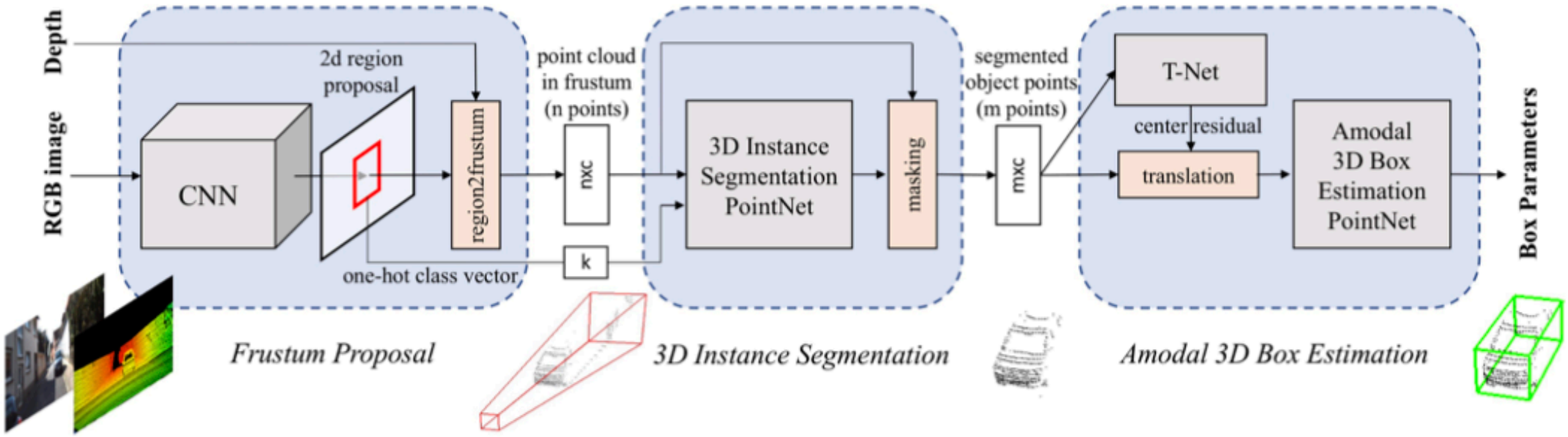
3D Object Detection from RGB-D

Frustum PointNet



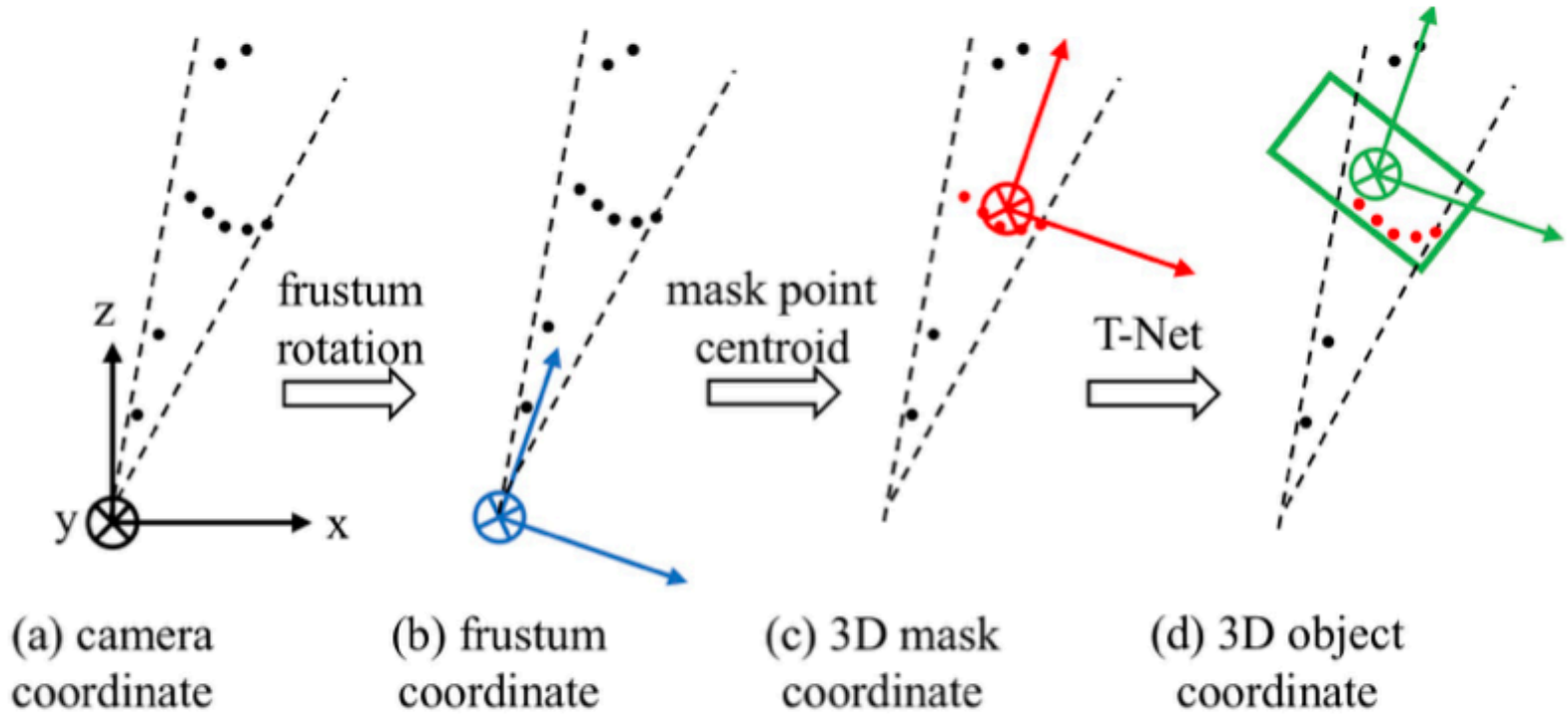
Qi, Charles R., et al. "Frustum pointnets for 3d object detection from rgb-d data." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

Pipeline of Frustum PointNet



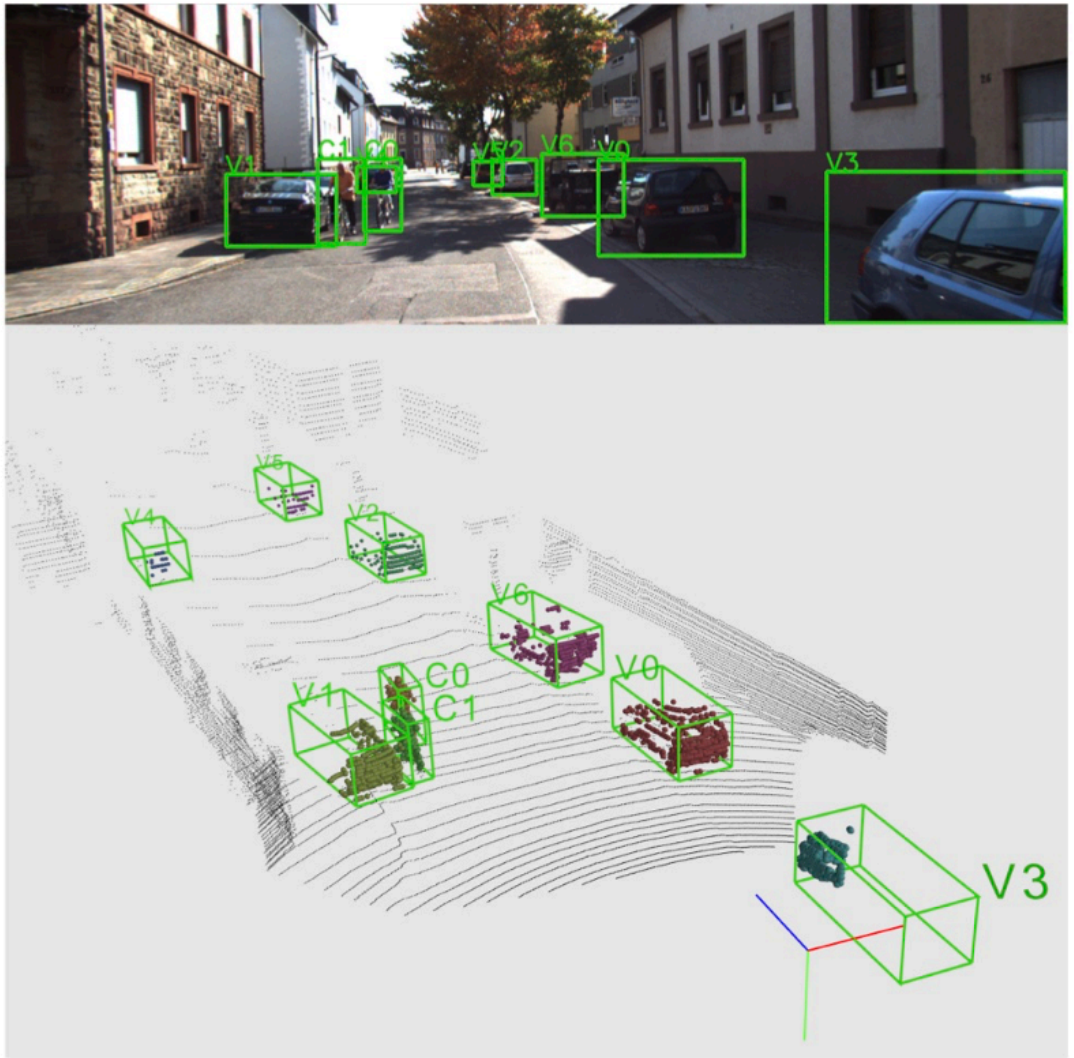
Qi, Charles R., et al. "Frustum pointnets for 3d object detection from rgb-d data." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

Coordinate Systems for Point Cloud

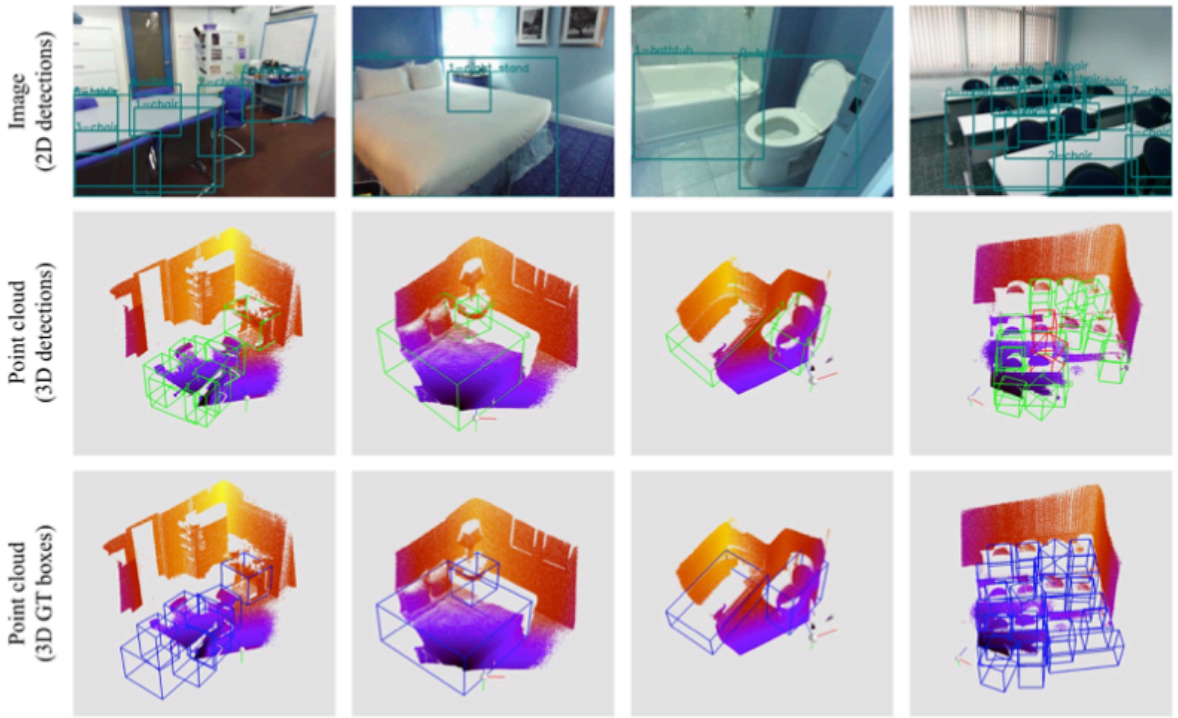


Qi, Charles R., et al. "Frustum pointnets for 3d object detection from rgb-d data." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

Result Visualization

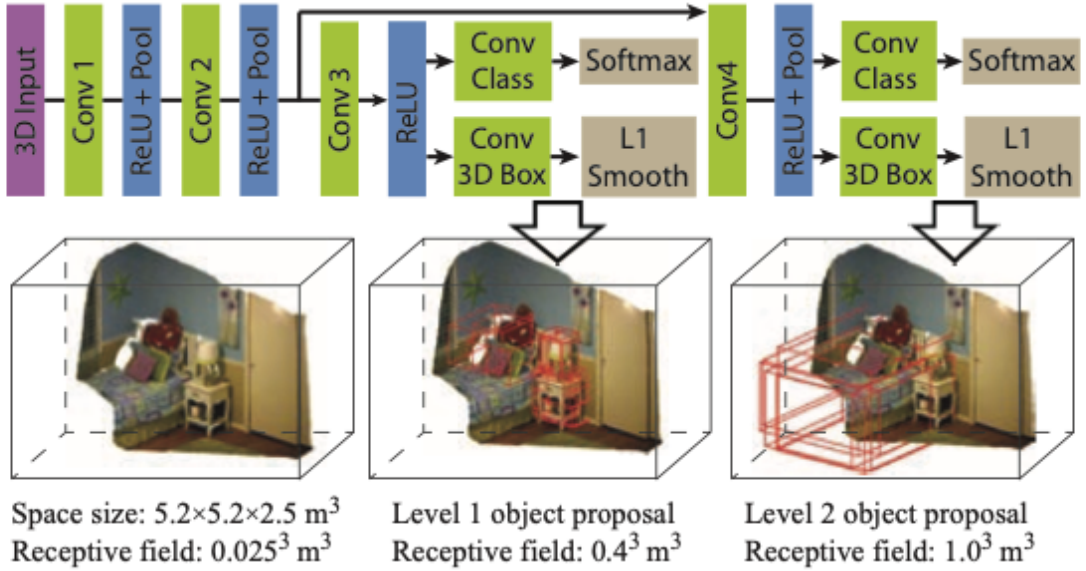


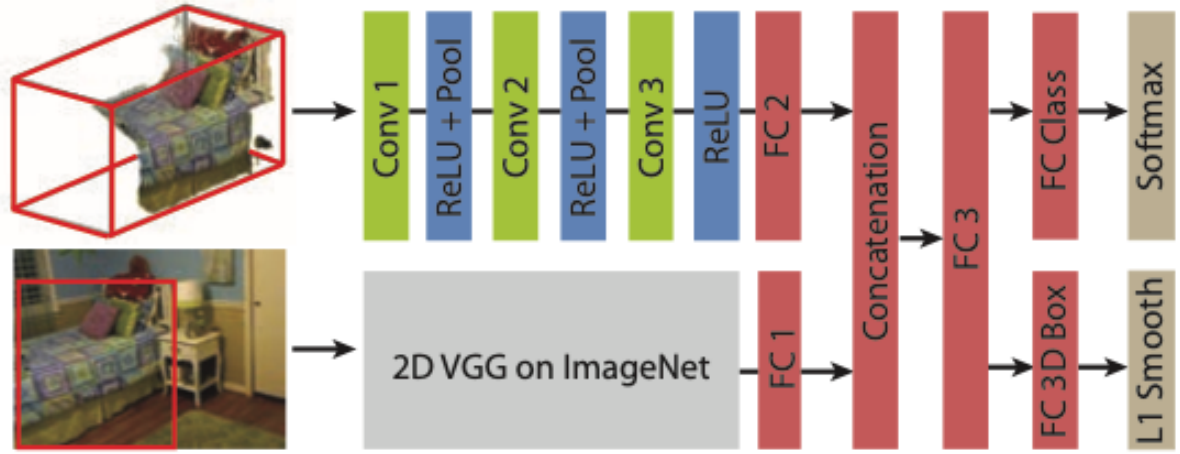
KITTI



SUN-RGBD

Deep Sliding Shape

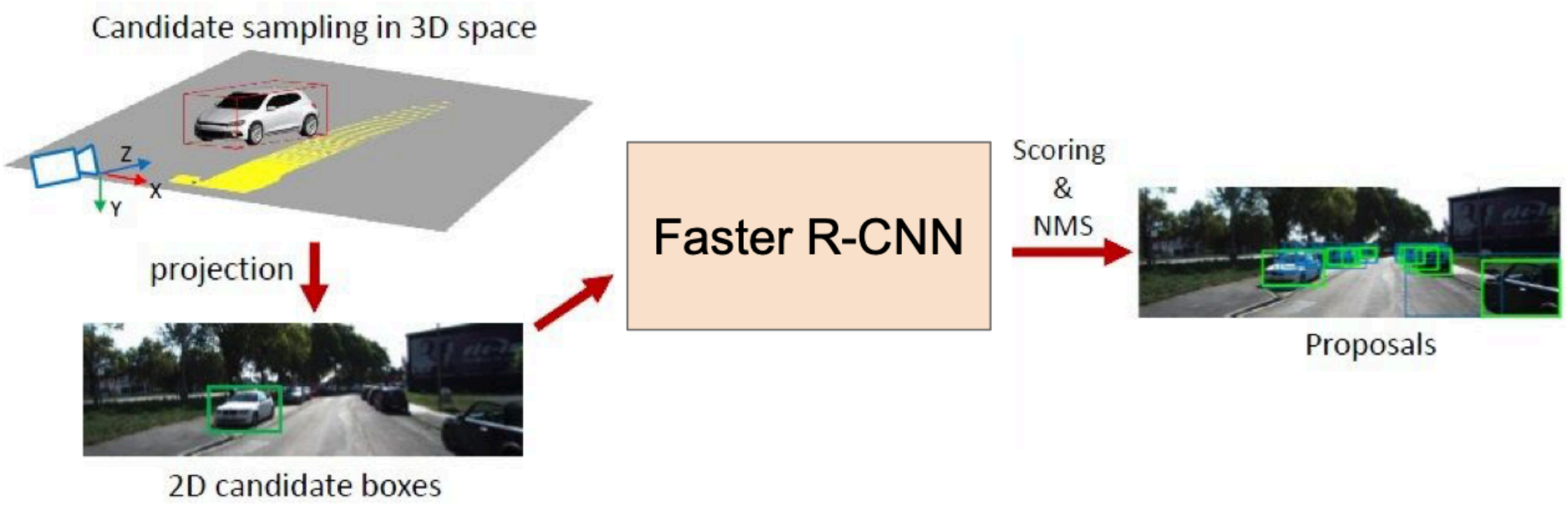




Very expensive to perform sliding windows in 3D!

Song, Shuran, and Jianxiong Xiao. "Deep sliding shapes for amodal 3d object detection in rgb-d images." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

3D Object Detection: Monocular Camera

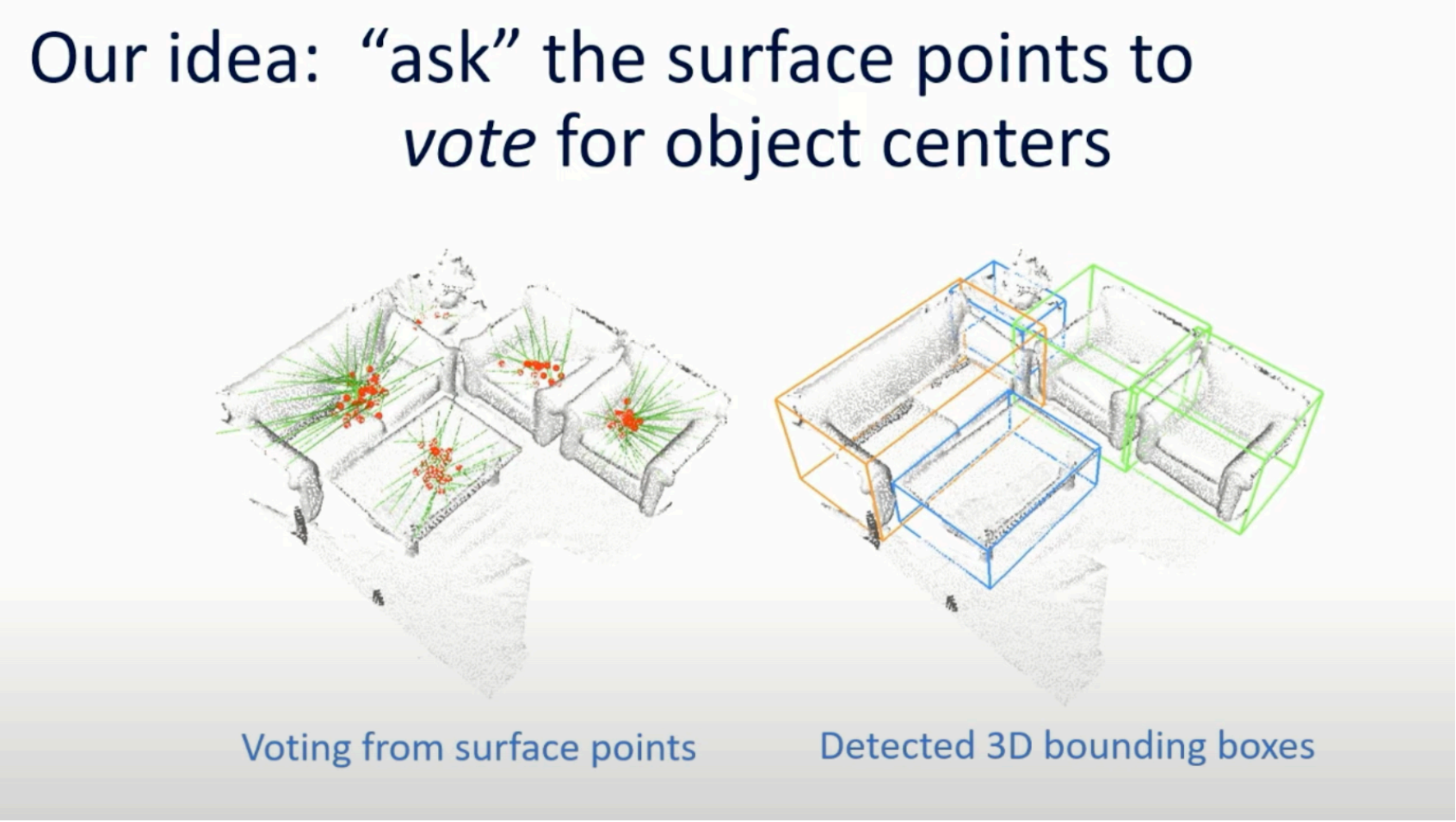


- Same idea as Faster RCNN, but proposals are in 3D
- 3D bounding box proposal, regress 3D box parameters + class score

Chen, Xiaozhi, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. "Monocular 3d object detection for autonomous driving." CVPR 2016.

Slides credit: Stanford CS231N

VoteNet

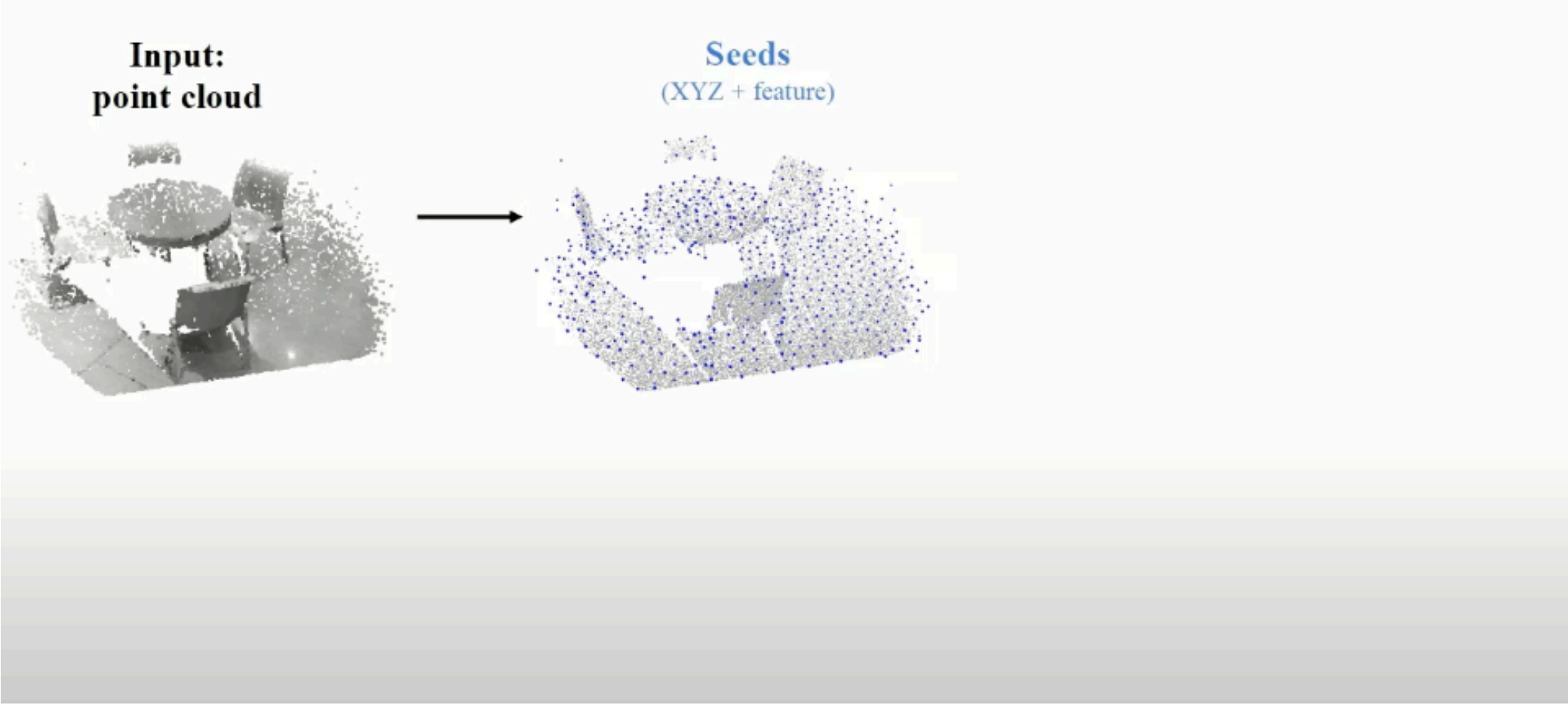


Qi, Charles R., et al. "Deep hough voting for 3d object detection in point clouds." *proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

Deep Hough Voting: Detection Pipeline

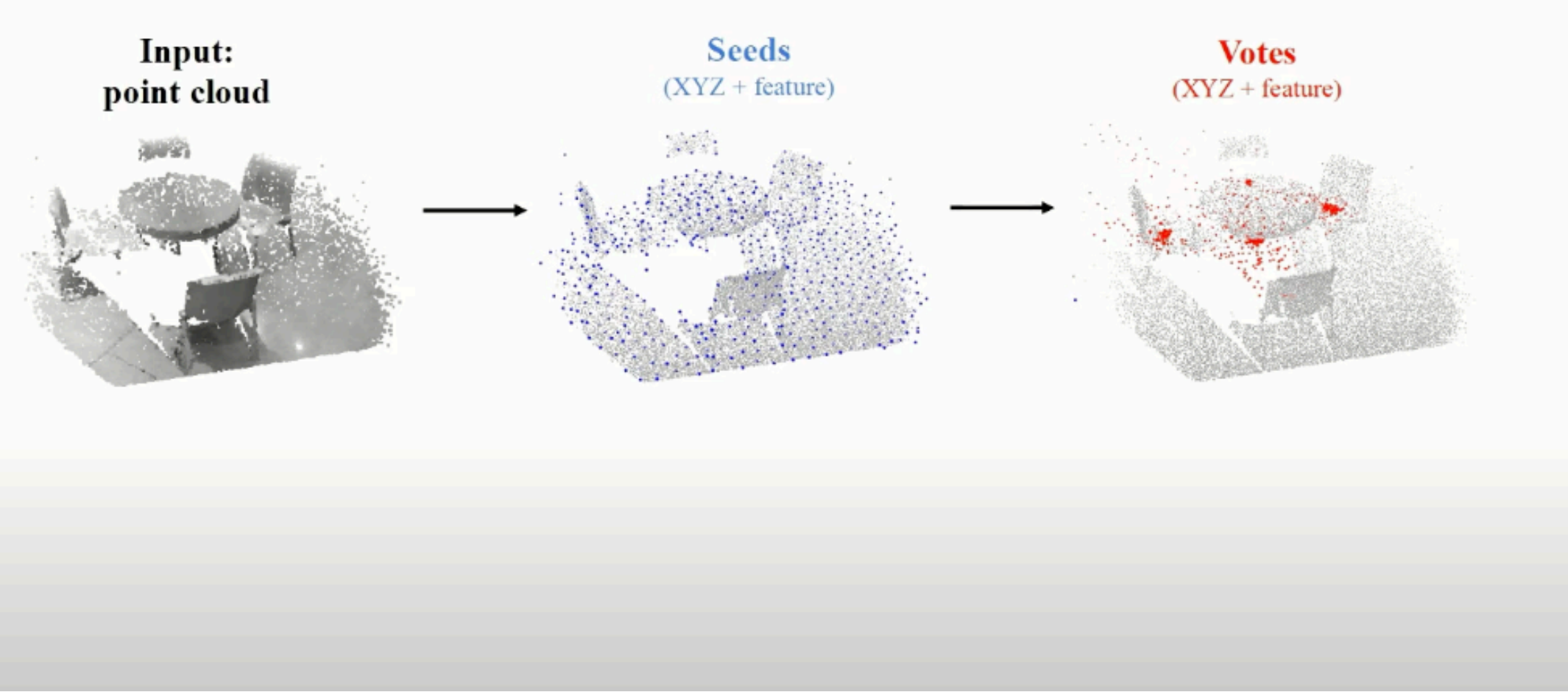
Qi, Charles R., et al. "Deep hough voting for 3d object detection in point clouds." *proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

Deep Hough Voting: Detection Pipeline



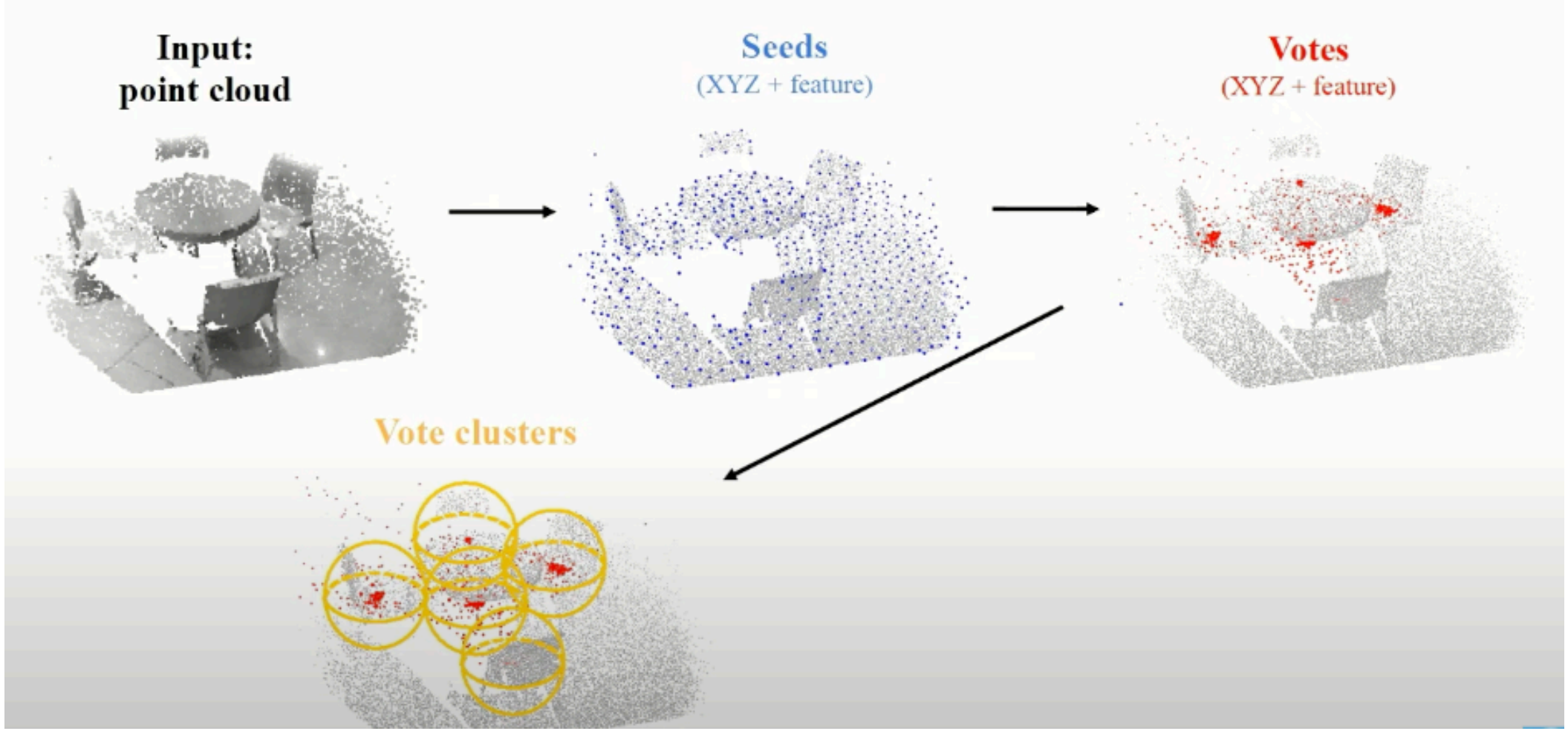
Qi, Charles R., et al. "Deep hough voting for 3d object detection in point clouds." *proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

Deep Hough Voting: Detection Pipeline



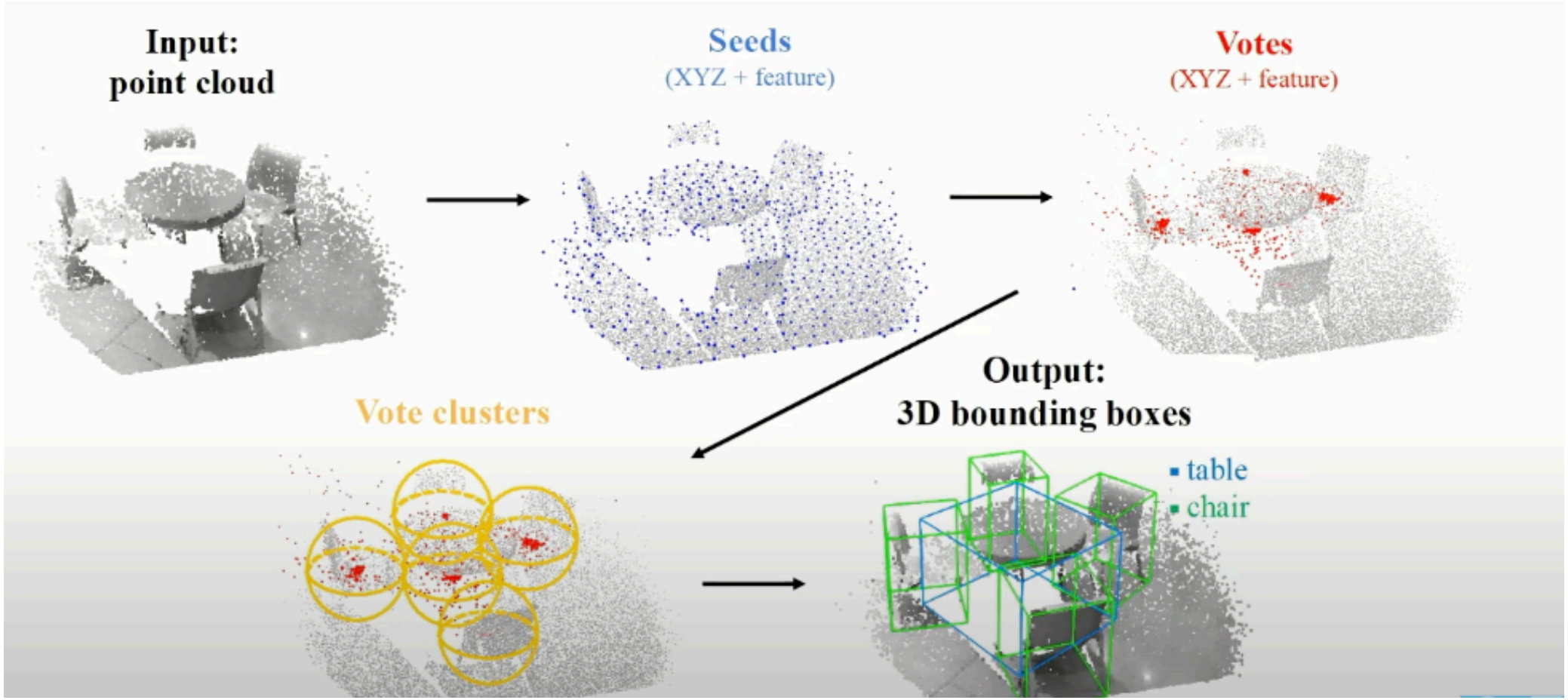
Qi, Charles R., et al. "Deep hough voting for 3d object detection in point clouds." *proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

Deep Hough Voting: Detection Pipeline



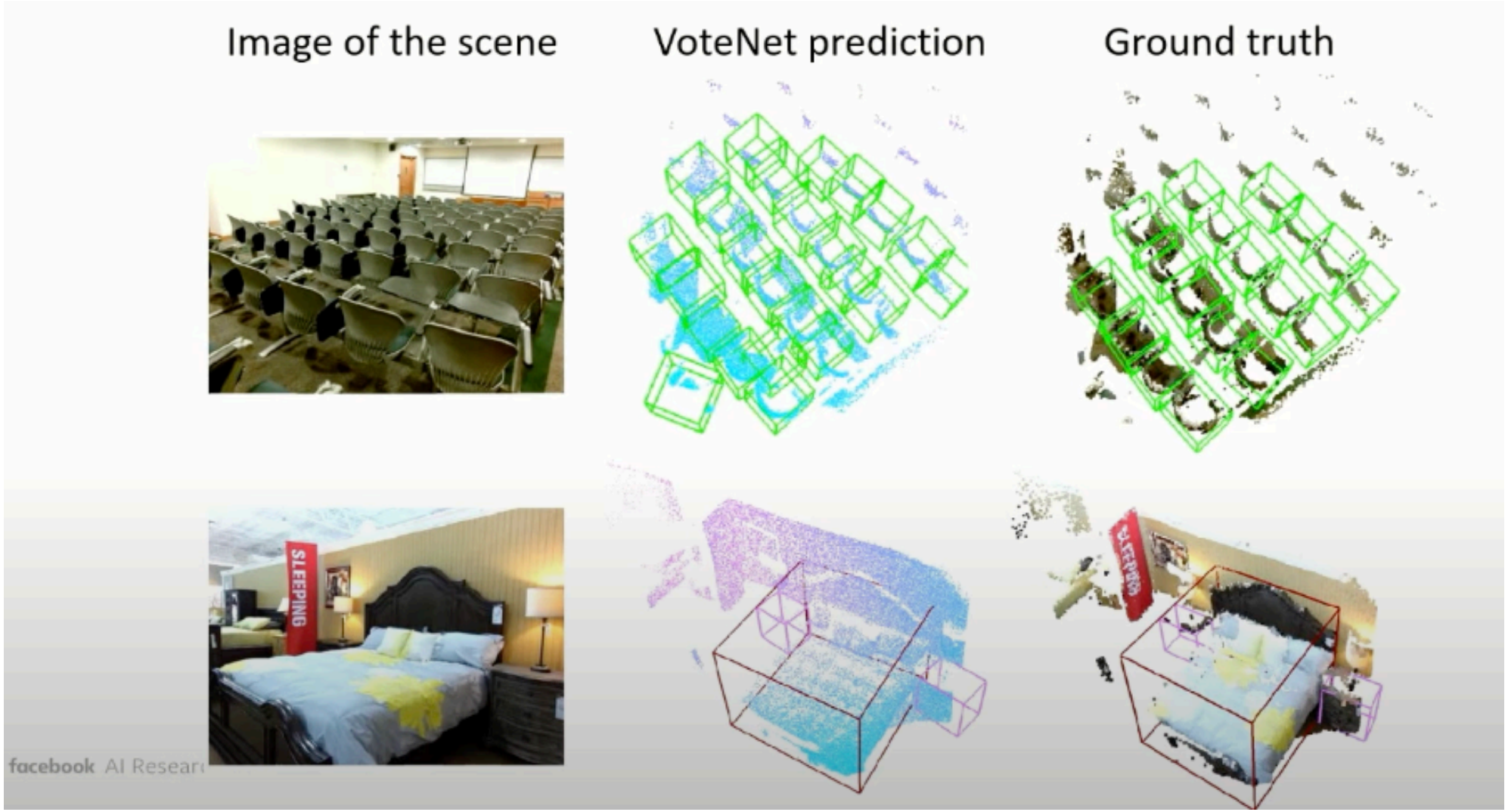
Qi, Charles R., et al. "Deep hough voting for 3d object detection in point clouds." *proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

Deep Hough Voting: Detection Pipeline



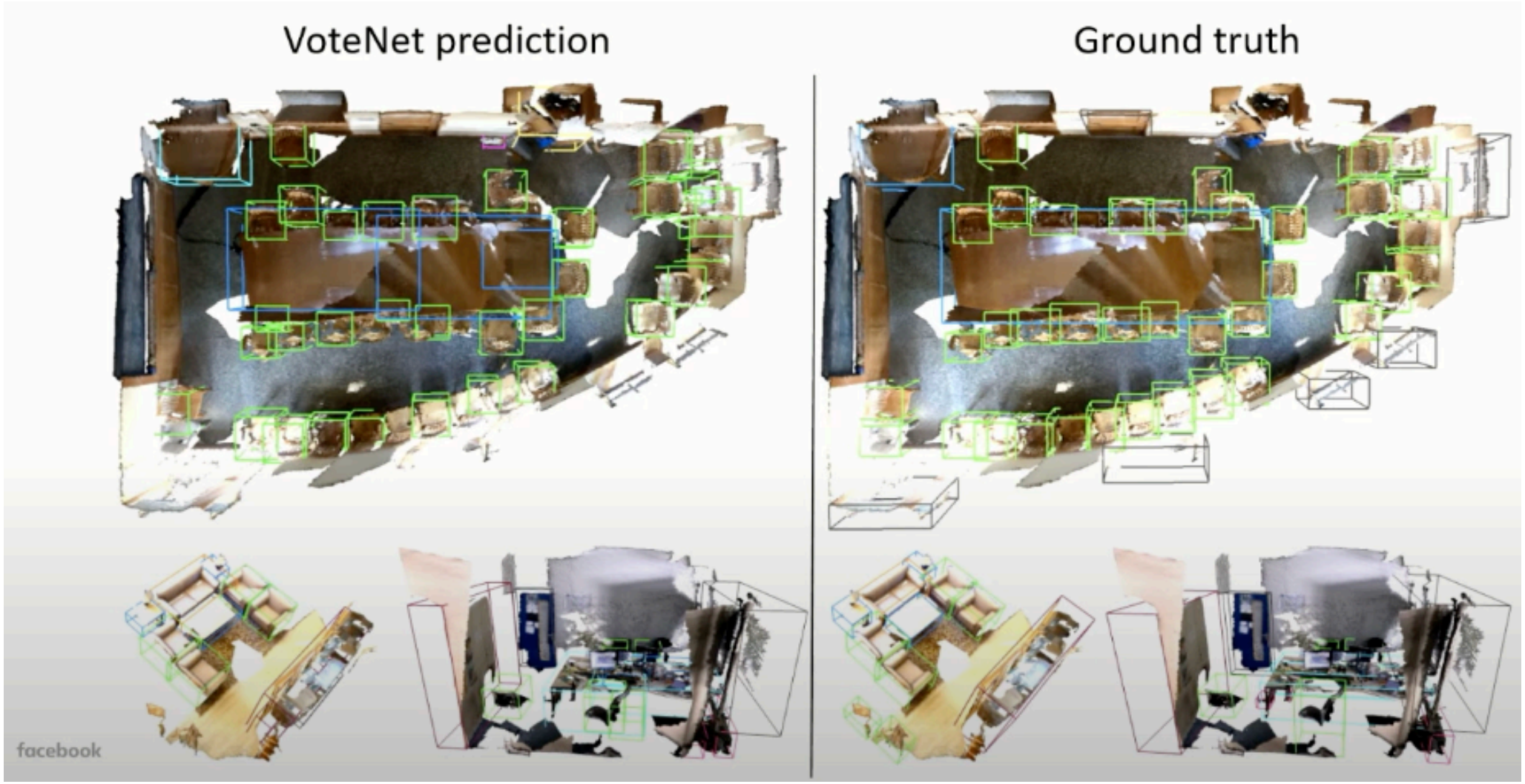
Qi, Charles R., et al. "Deep hough voting for 3d object detection in point clouds." *proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

Results: SUN RGB-D (single depths)



Qi, Charles R., et al. "Deep hough voting for 3d object detection in point clouds." *proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

Results: ScanNet (3D Reconstruction)



Qi, Charles R., et al. "Deep hough voting for 3d object detection in point clouds." *proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.

3D Instance Segmentation

- Top-Down
  - GSPN
- Bottom-Up
  - SGPN
  - PointGroup

Introduction to Computer Vision

Next lecture: Lecture 12,
Generative Model