Dense Prediction with CNNs

Danna Gurari
University of Colorado Boulder
Spring 2025

https://dannagurari.colorado.edu/course/neural-networks-and-deep-learning-spring-2025/

Review

- Last lecture:
  - Dataset challenges: before versus after 2012
  - Hardware: before versus after 2012
  - Programming tutorial

- Assignments (Canvas)
  - Lab assignment 1 due later today (at 10pm)
  - Problem set 3 due in 1 week

- Questions?

Today's Topics

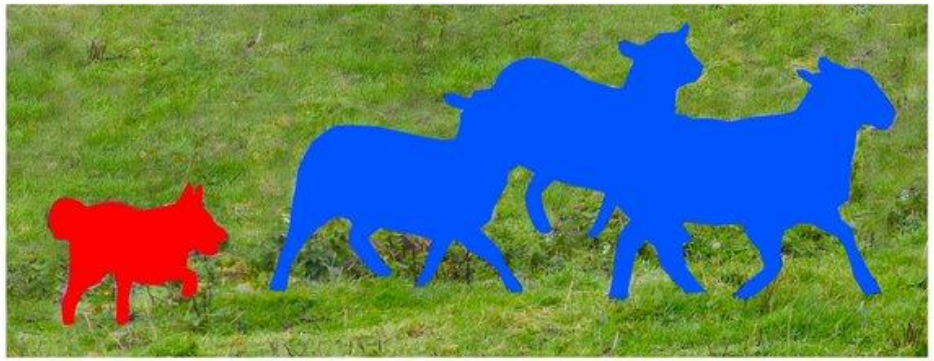
- Motivation: training large capacity, deep models to locate content
- Semantic segmentation: classifying pixels
- Object detection: locating objects with bounding rectangles
- Instance segmentation: demarcating objects with detailed outlines
- Programming tutorial

Today's Topics

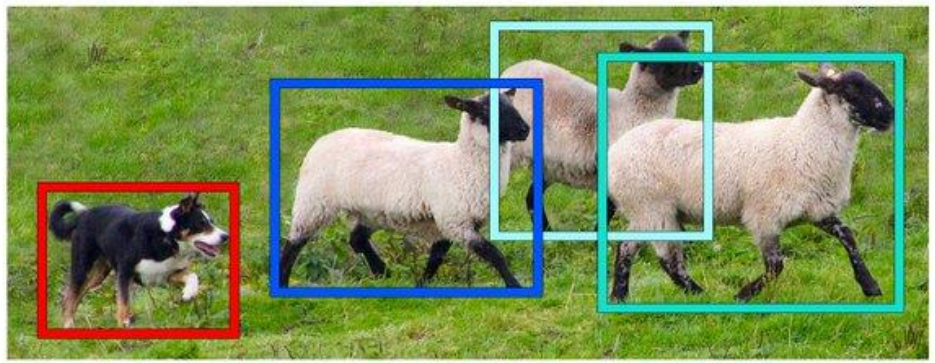
- Motivation: training large capacity, deep models to locate content
- Semantic segmentation: classifying pixels
- Object detection: locating objects with bounding rectangles
- Instance segmentation: demarcating objects with detailed outlines
- Programming tutorial

Object Recognition vs Dense Prediction

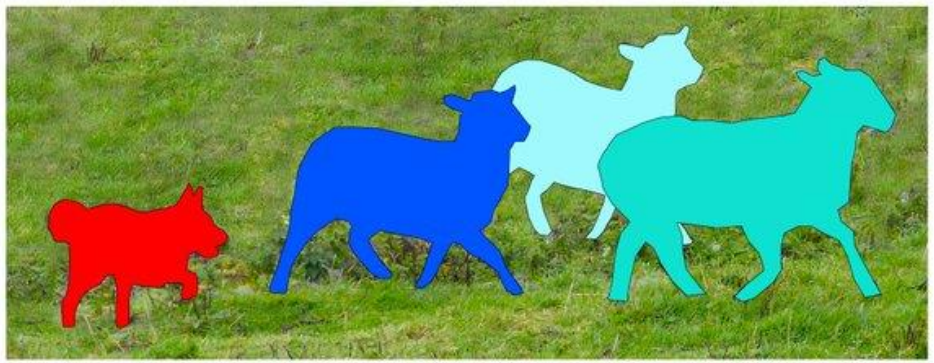
Image Recognition



Semantic Segmentation



Object Detection

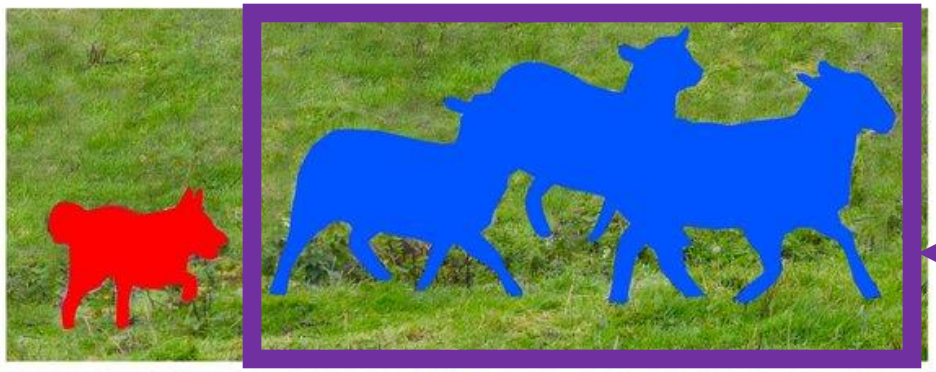


Instance Segmentation

https://ai-pool.com/d/could-you-explain-me-how-instance-segmentation-works

Dense Prediction Tasks

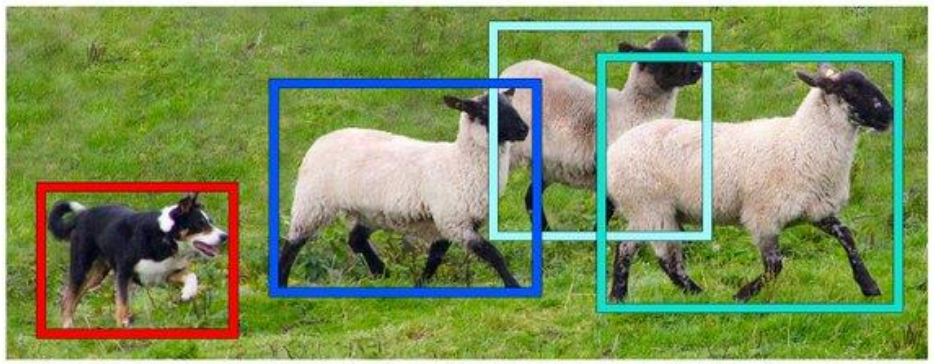
Locate all pixels belonging to pre-specified categories



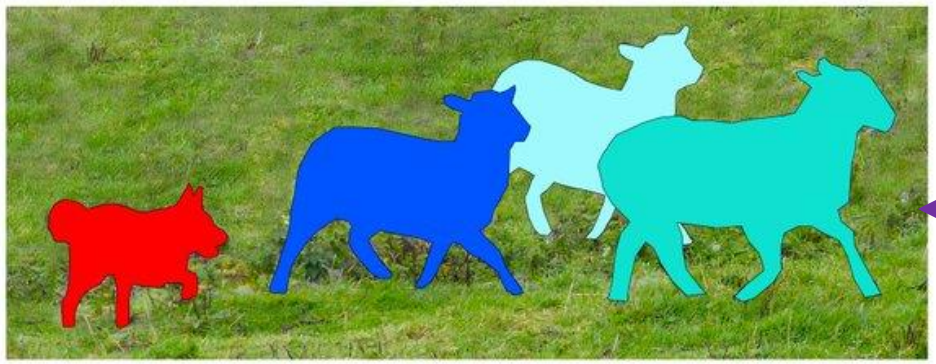
Semantic Segmentation

Note: instances of same category NOT separated

Locate all instances belonging to pre-specified categories with rectangles (aka, bounding boxes)



Object Detection



Instance Segmentation

Unifies semantic segmentation and object detection

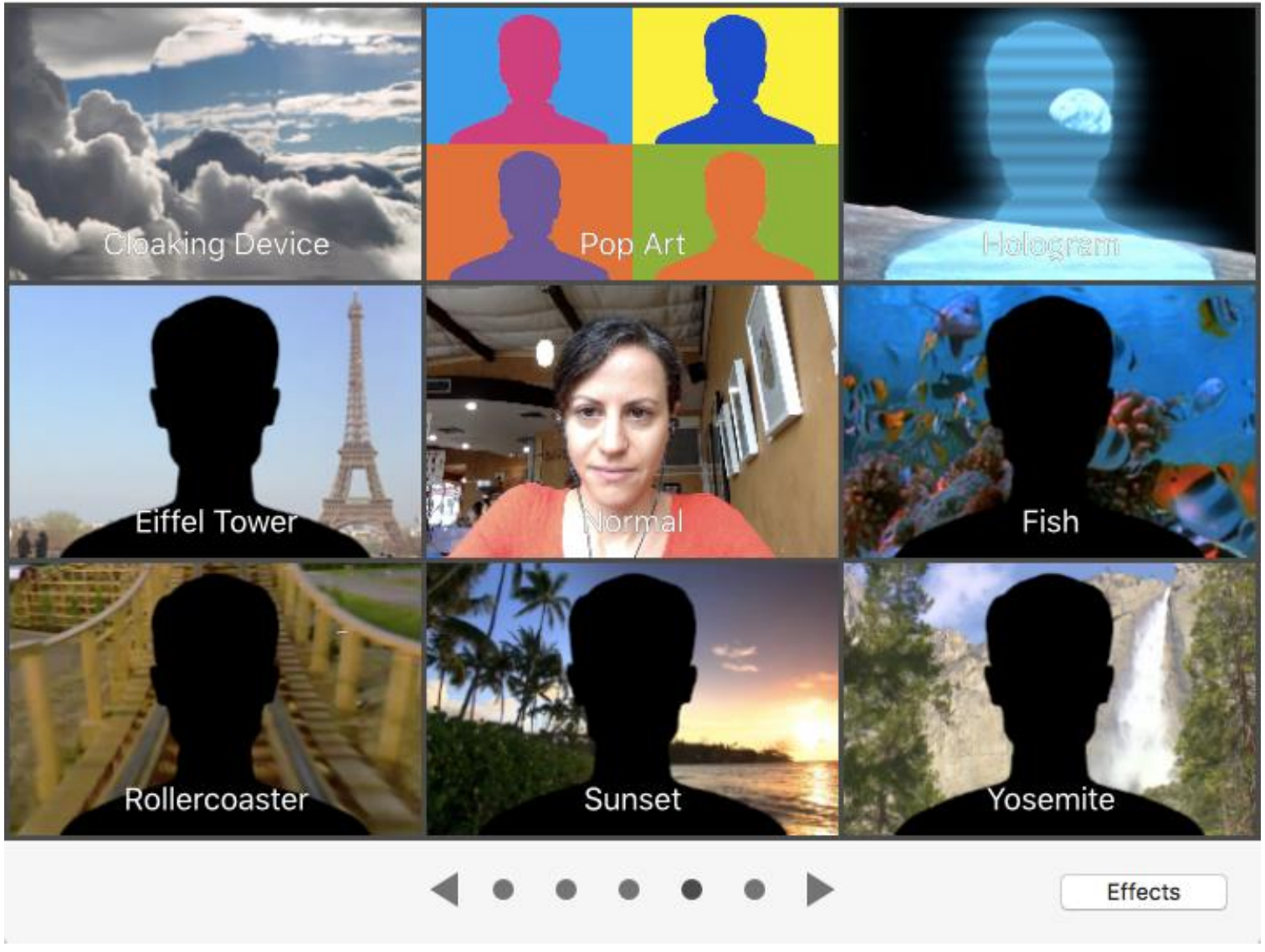
https://ai-pool.com/d/could-you-explain-me-how-instance-segmentation-works

Object Recognition vs Dense Prediction

How do localization tasks differ from object recognition? e.g.,

Must learn objects' appearances rather than only image context;
e.g., giraffes are often photographed in savannah-like landscapes

Applications: Rotoscoping



https://www.starnow.co.uk/ahmedmohammed1/photos/4650871/before-and-after-rotoscopinggreen-screening

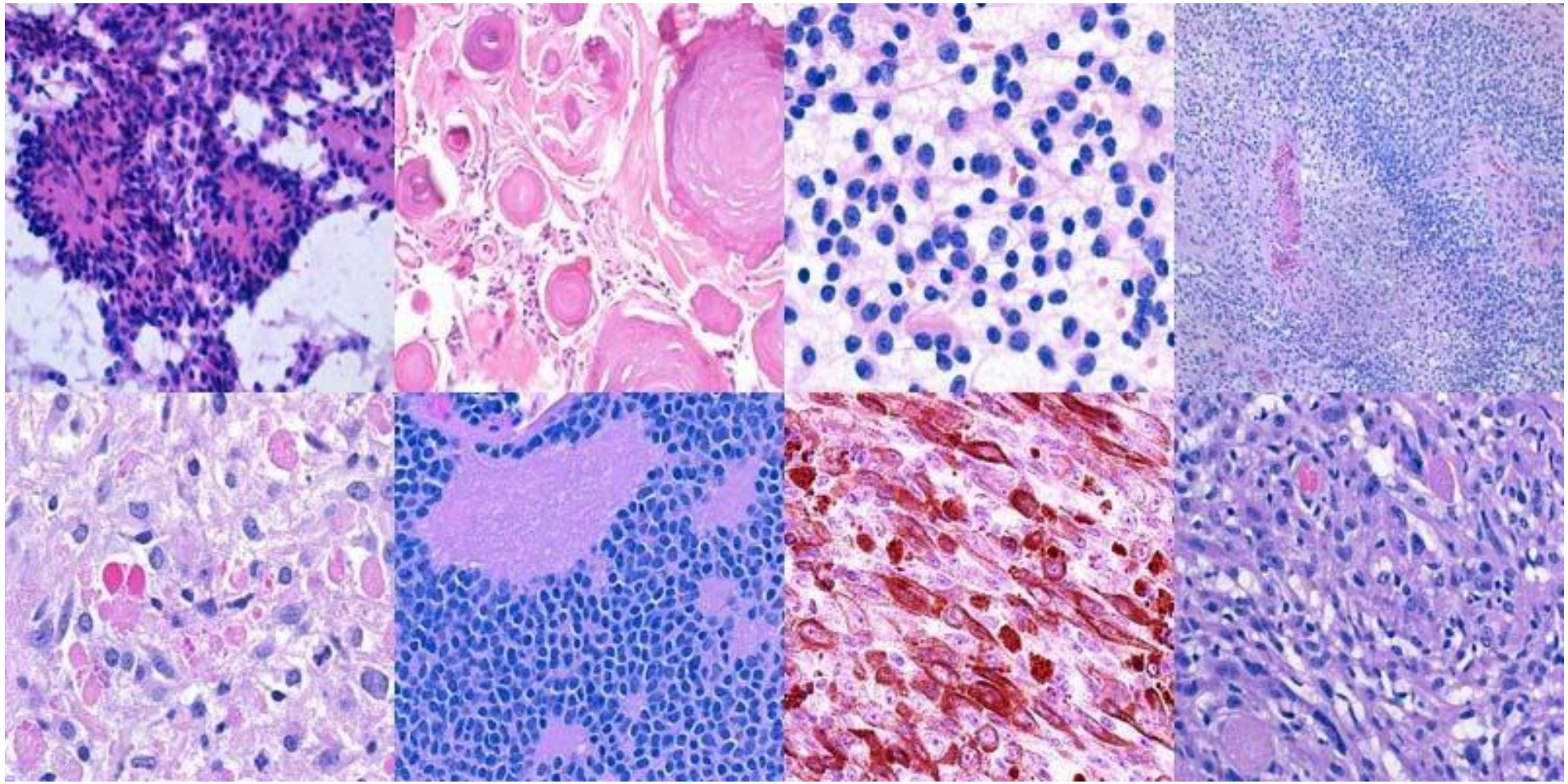
Applications: Remodeling Inspiration

(a) Target photo

(b) Retextured

Bell et al; SIGGRAPH; 2013

Applications: Disease Diagnosis; e.g., PathAI

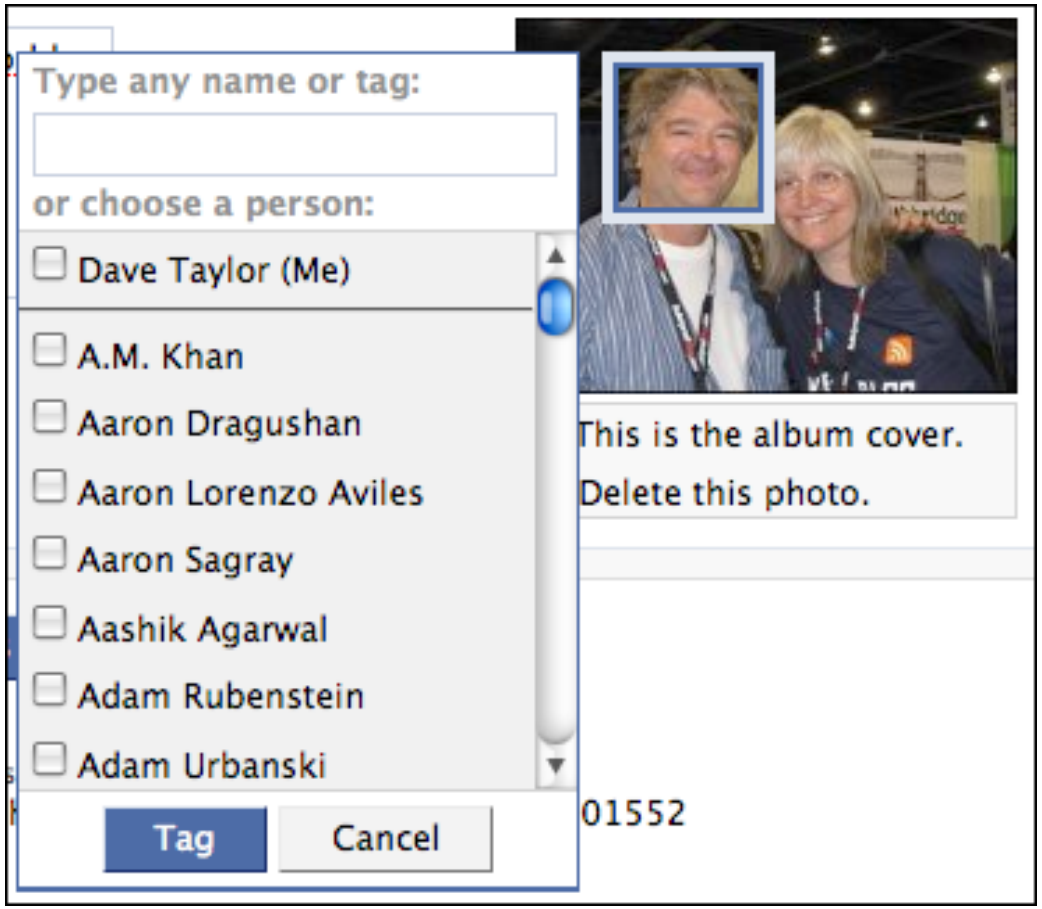


https://pathology.jhu.edu/brain-tumor/grading-classification

Applications: Self-Driving Vehicles

https://www.inc.com/kevin-j-ryan/self-driving-cars-powered-by-people-playing-games-mighty-ai.html

Applications: Social Media



Face detection
(e.g., Facebook)

Applications: Banking

Mobile check deposit
(e.g., Bank of America)

Applications: Transportation

License Plate Detection (e.g., AllGoVision)

Applications: Counting

Counting Fish (e.g., SalmonSoft)

http://www.wecountfish.com/?page_id=143

Business Traffic Analytics

What are other applications that dense prediction can help with?

Recall: Large Capacity Model Necessary

So much complexity for even just one object category:

Illumination

Object pose

Clutter

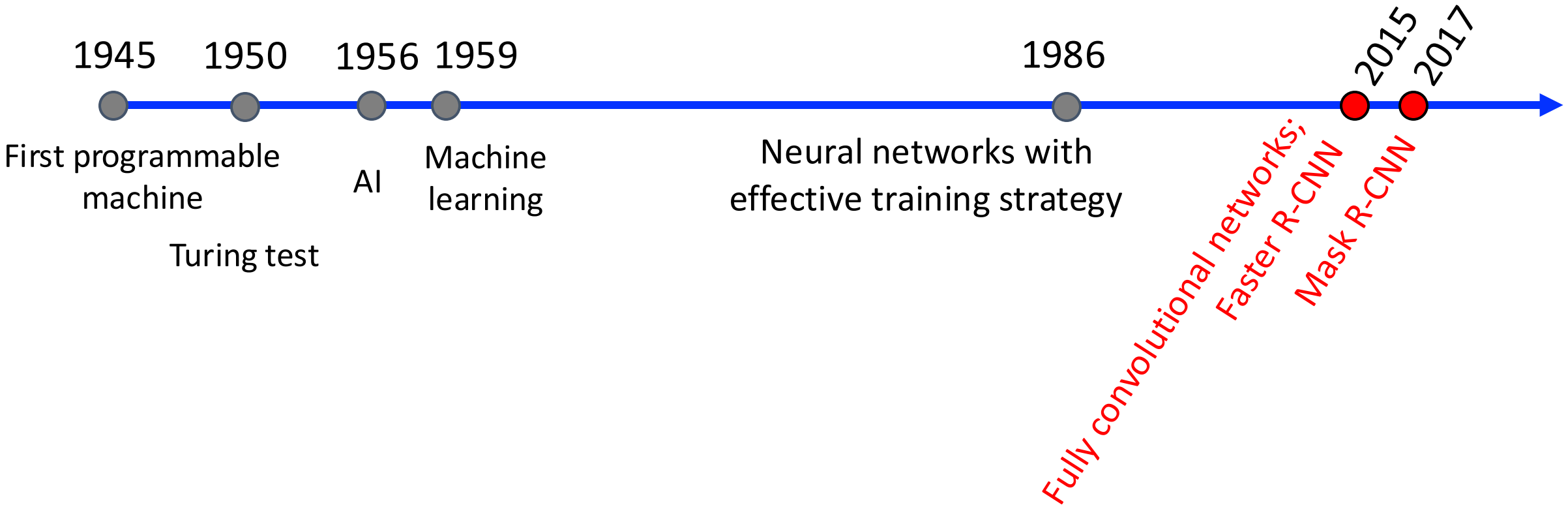
Occlusions

Intra-class appearance

Viewpoint

Source: Kristen Grauman

How to Develop Large Capacity Models?



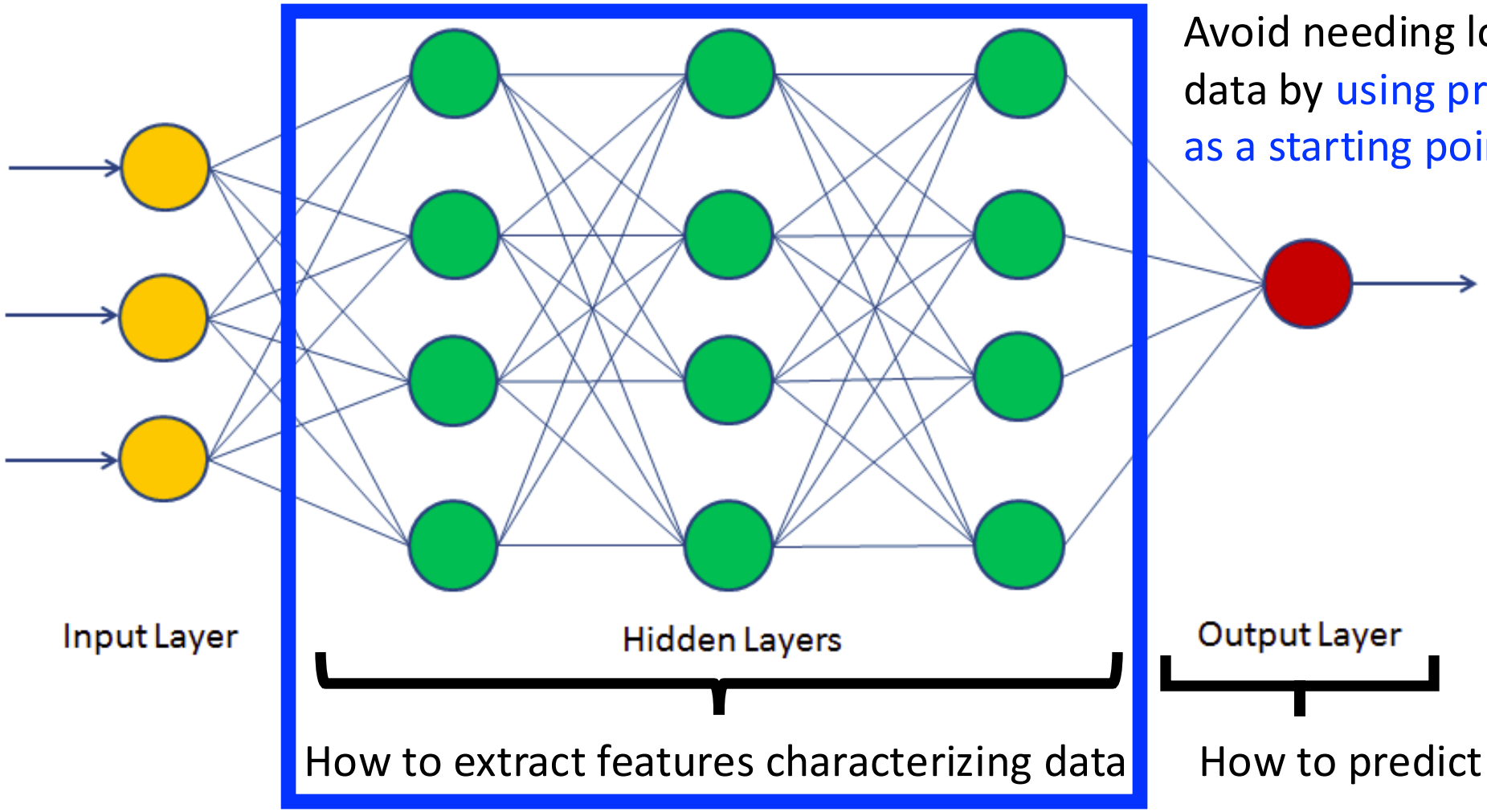
Key Challenge: Train Large Capacity Models

- VERY challenging to collect large-scale annotated datasets
- How long do you think it would take to draw a:
  - bounding box? (a) 5-15 seconds, (b) 16-45 seconds, (c) 46-75 seconds
  - segmentation?

Avg time from 101 annotators: 7 seconds per bounding box 54 seconds per segmentation

Jain and Grauman. Predicting Sufficient Annotation Strength for Interactive Foreground Segmentation. ICCV 2013

Key Idea: Transfer Learning

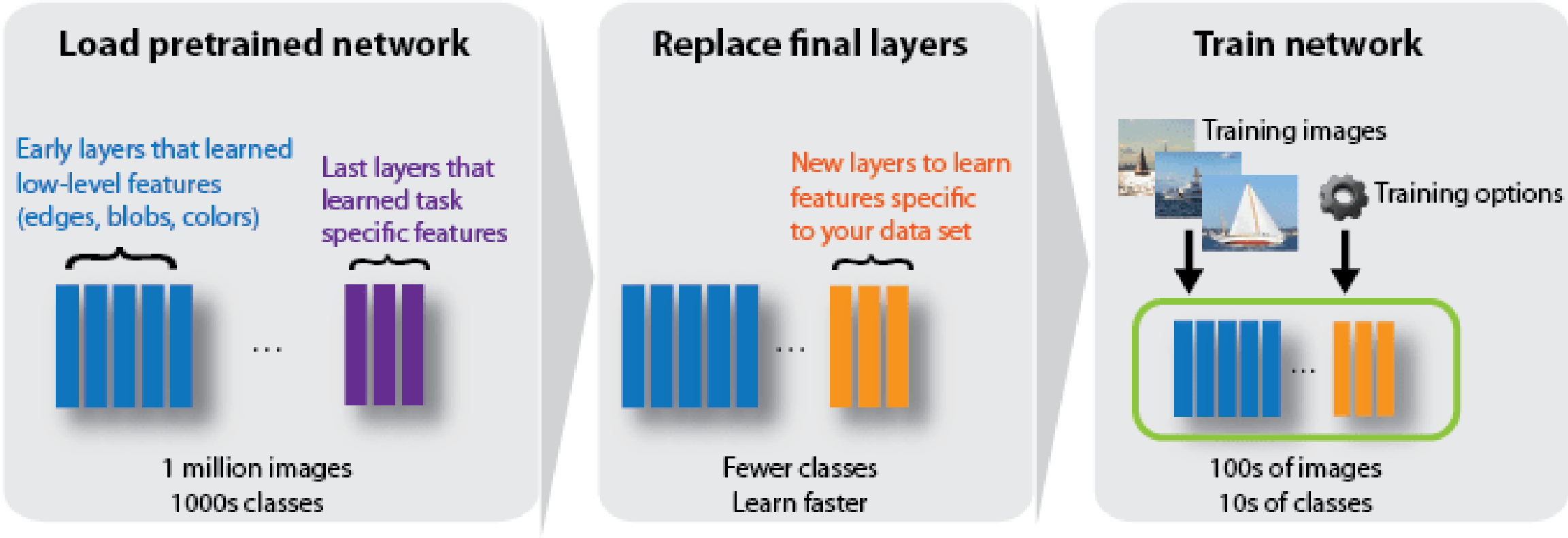


Avoid needing lots of annotated data by using pretrained network as a starting point for target task

https://www.datacamp.com/community/tutorials/neural-network-models-r

Key Idea: Transfer Learning

Use pretrained network as a starting point to train for a different dataset and/or task; e.g.,

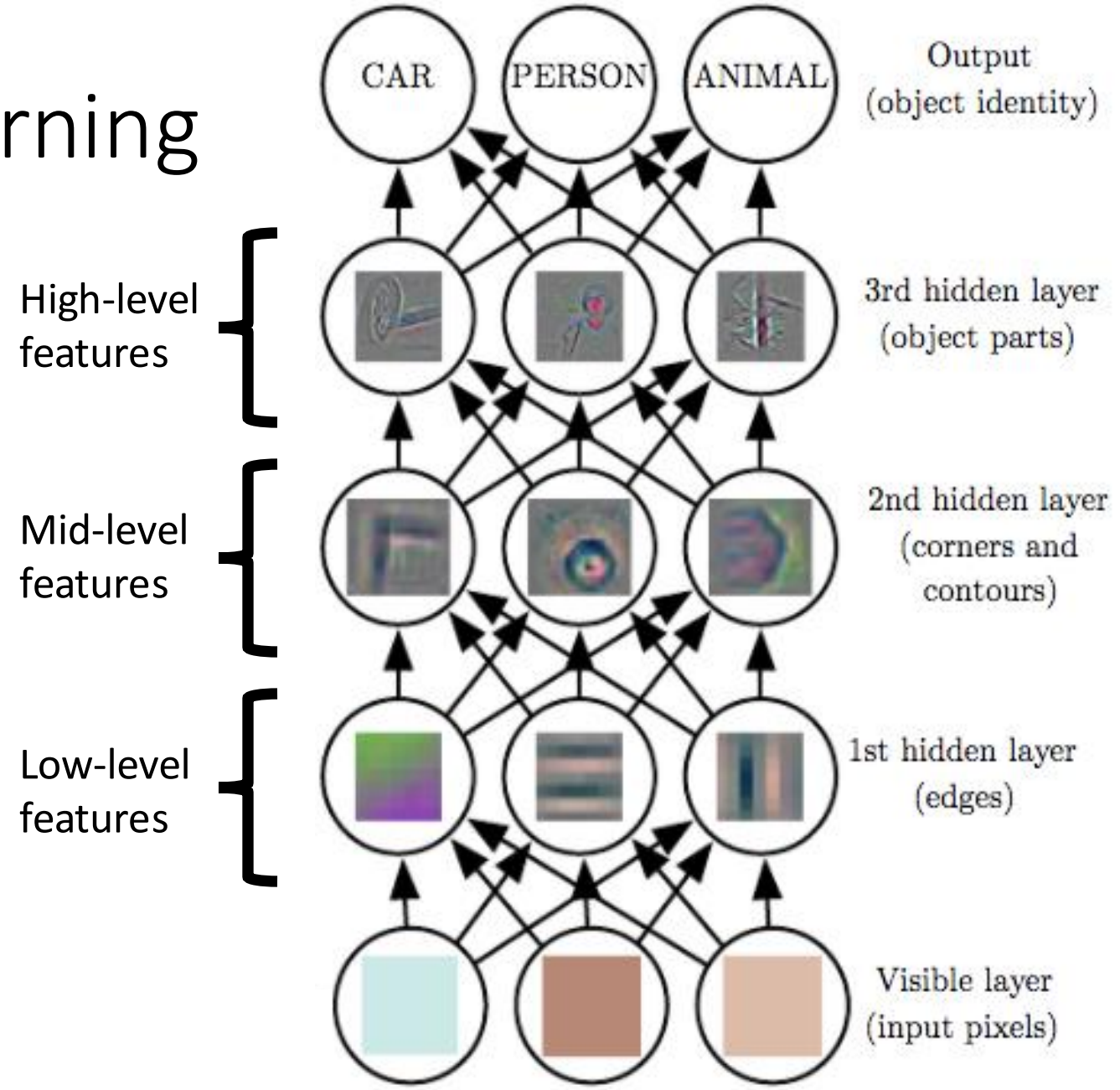


https://www.mathworks.com/help/deeplearning/ug/transfer-learning-using-alexnet.html

Key Idea: Transfer Learning

Key Questions:

1. Which pretrained network?
2. Which layers from the network?
3. Freeze vs train transferred layers (latter is called “fine-tuning”)



Deep Learning, Ian Goodfellow, Yoshua Bengio, and Aaron Courville

Today's Topics

- Motivation: training large capacity, deep models to locate content
- Semantic segmentation: classifying pixels
- Object detection: locating objects with bounding rectangles
- Instance segmentation: demarcating objects with detailed outlines
- Programming tutorial

Fully Convolutional Network

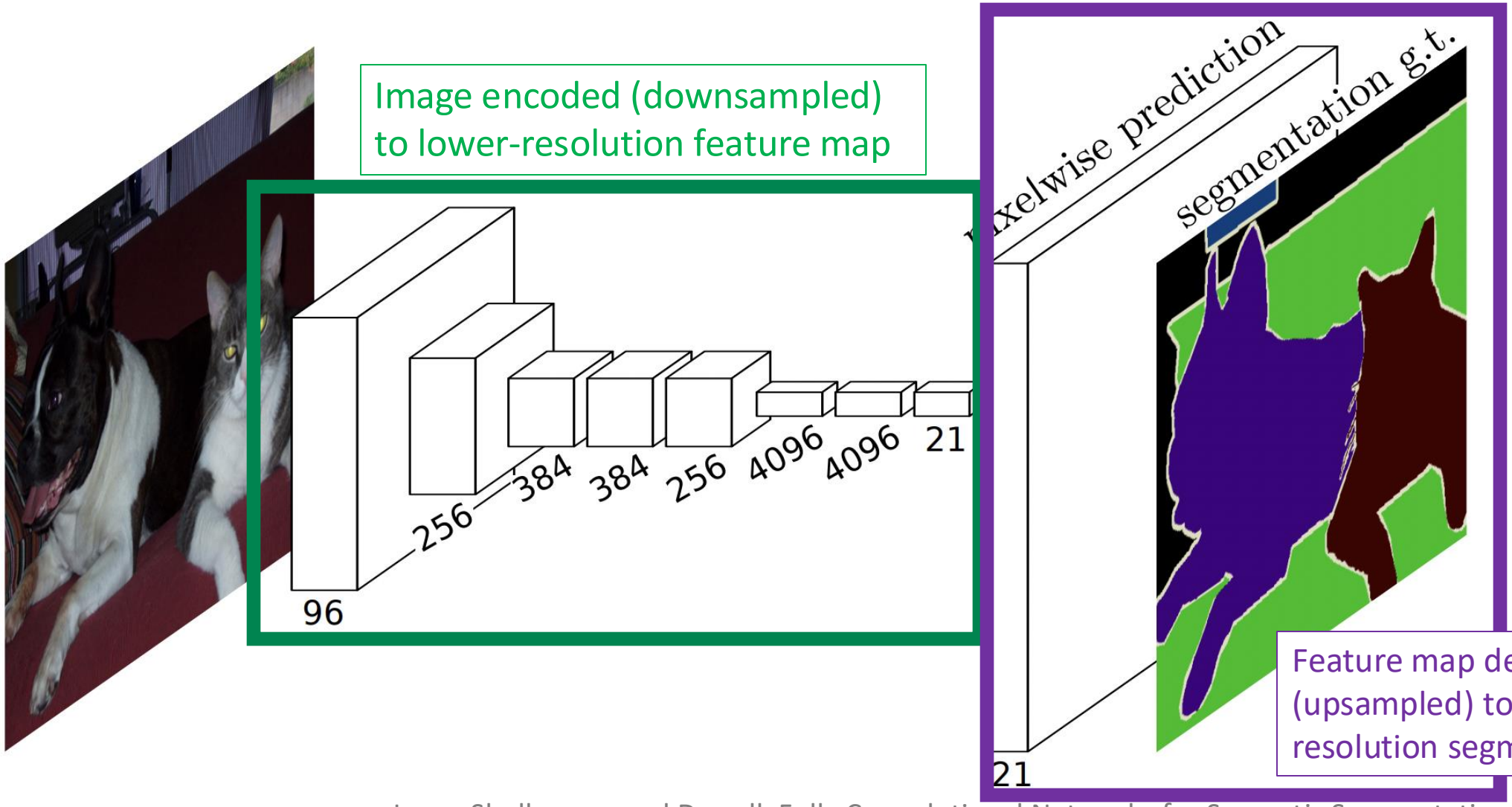
Named after the proposed technique that excludes fully connected layers:

> Jonathon Long, Evan Shelhamer, and Trevor Darrell. “Fully Convolutional Networks for Semantic Segmentation.” CVPR 2015.

First work for pixelwise prediction to:

1. Train fully convolutional networks end-to-end
2. Use supervised pre-training (recall, R-CNN paper showed this can be a great idea when there is a scarce amount of annotated data)

Architecture: Encoder Decoder Architecture



Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

Key Ideas

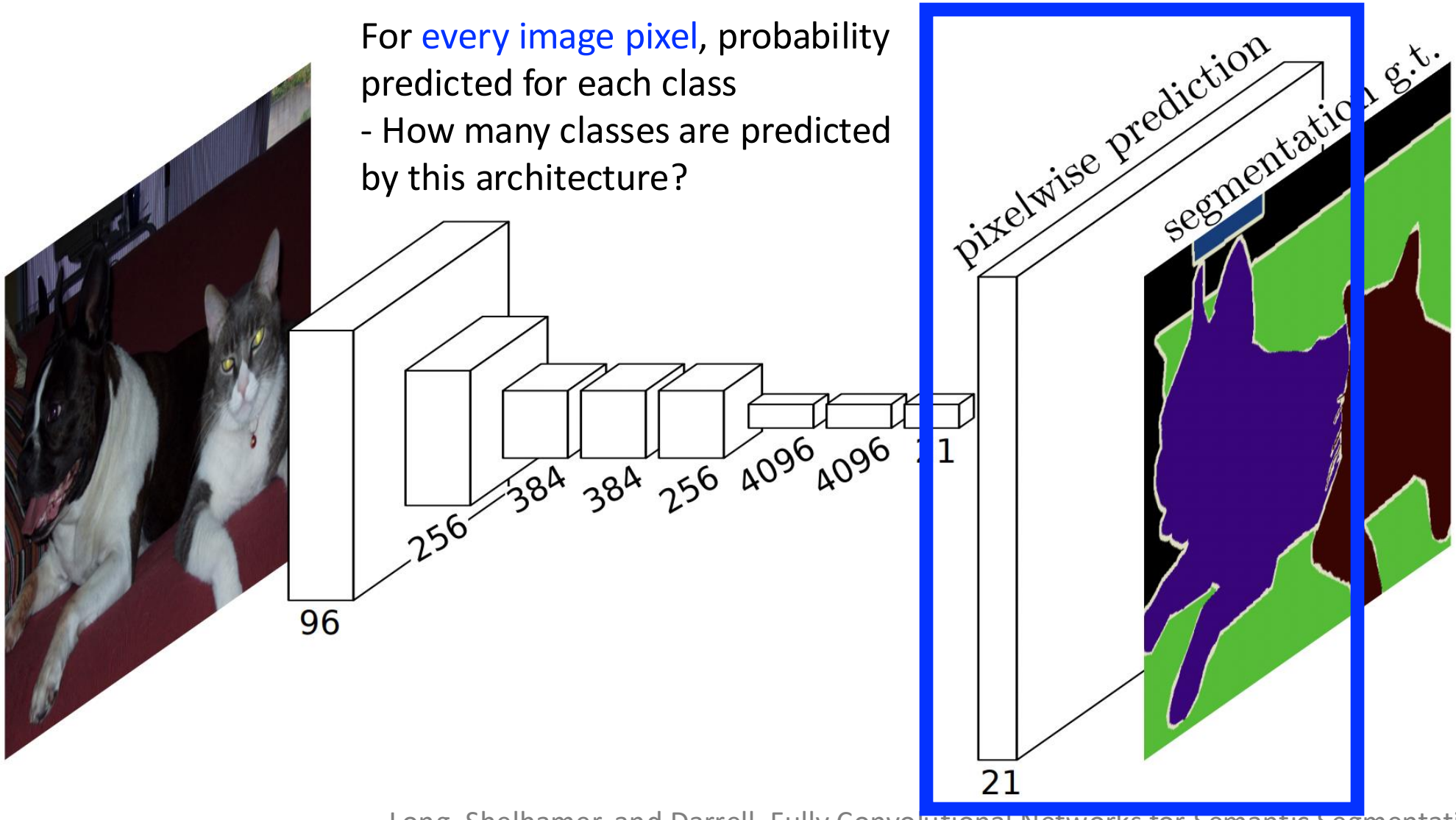
- Enable dense output: per pixel classification output layer
- Support any input image dimension: fully convolutional network
- Generate high-quality segmentations: upsampling and skip connections
- Make learning feasible: supervised pre-training

Key Ideas

- Enable dense output: per pixel classification output layer
- Support any input image dimension: fully convolutional network
- Generate high-quality segmentations: upsampling and skip connections
- Make learning feasible: supervised pre-training

Architecture: Output Layer

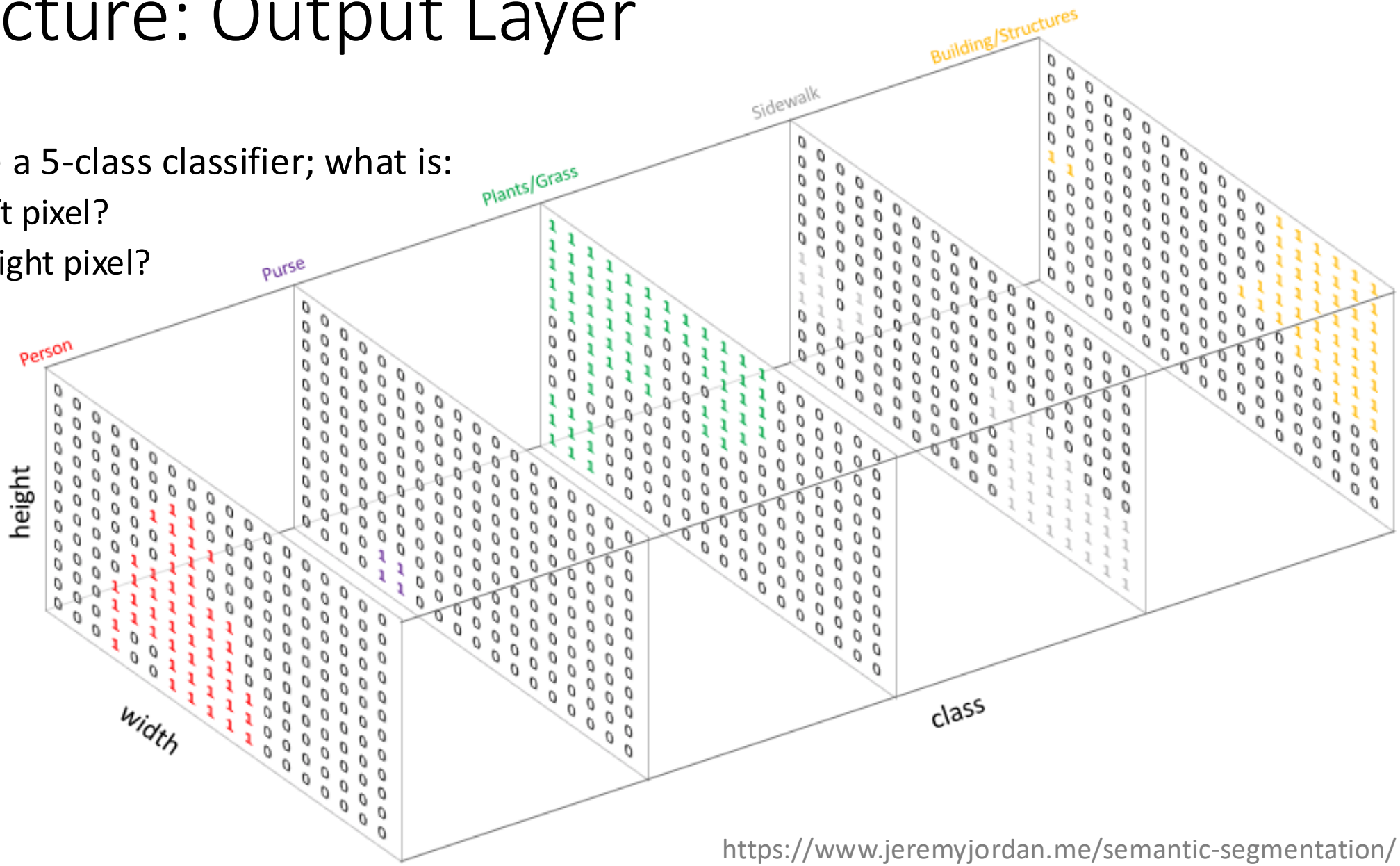
For every image pixel, probability predicted for each class
- How many classes are predicted by this architecture?



Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

Architecture: Output Layer

- e.g., assume a 5-class classifier; what is:
  - upper left pixel?
  - bottom right pixel?



https://www.jeremyjordan.me/semantic-segmentation/

Architecture: Output Layer

- e.g., assume a 5-class classifier; output 1-hot encoding collapsed into single mask image

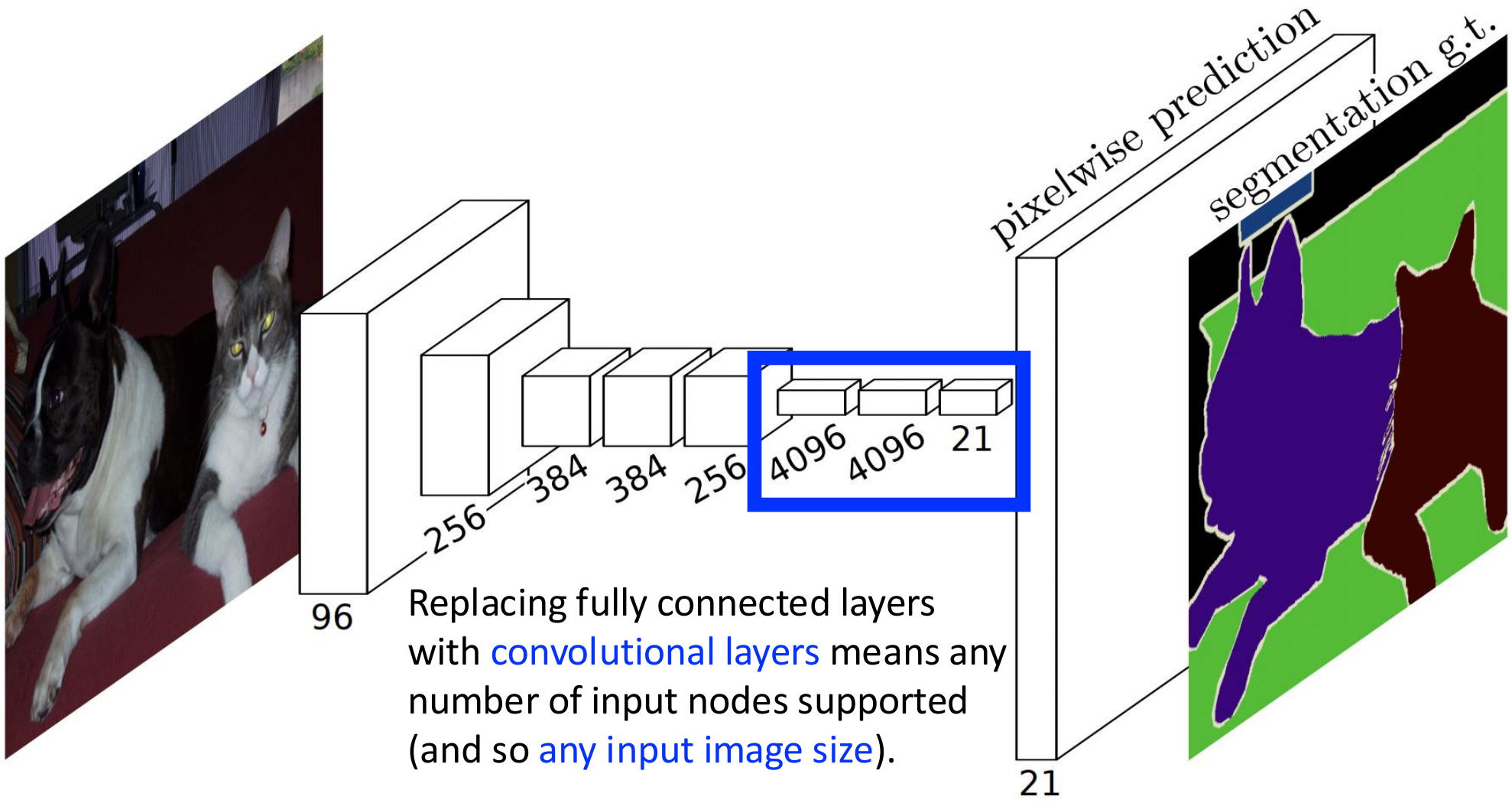
0: Background/Unknown
1: Person
2: Purse
3: Plants/Grass
4: Sidewalk
5: Building/Structures

Source: https://www.jeremyjordan.me/semantic-segmentation/

Key Ideas

- Enable dense output: per pixel classification output layer
- Support any input image dimension: fully convolutional network
- Generate high-quality segmentations: upsampling and skip connections
- Make learning feasible: supervised pre-training

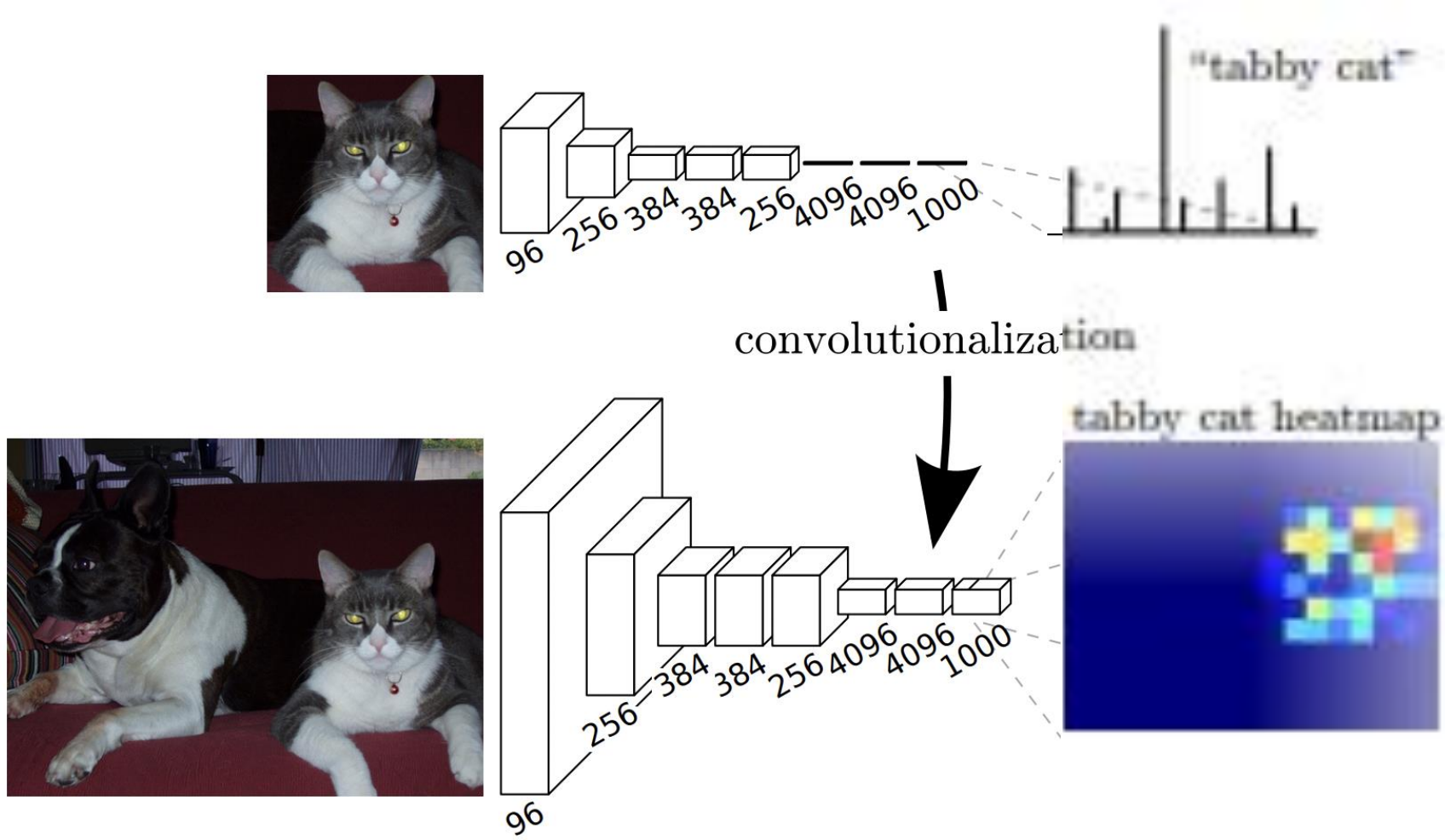
Architecture: All Convolutional Layers



Replacing fully connected layers with convolutional layers means any number of input nodes supported (and so any input image size).

Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

Architecture: Fully vs Convolution Layers

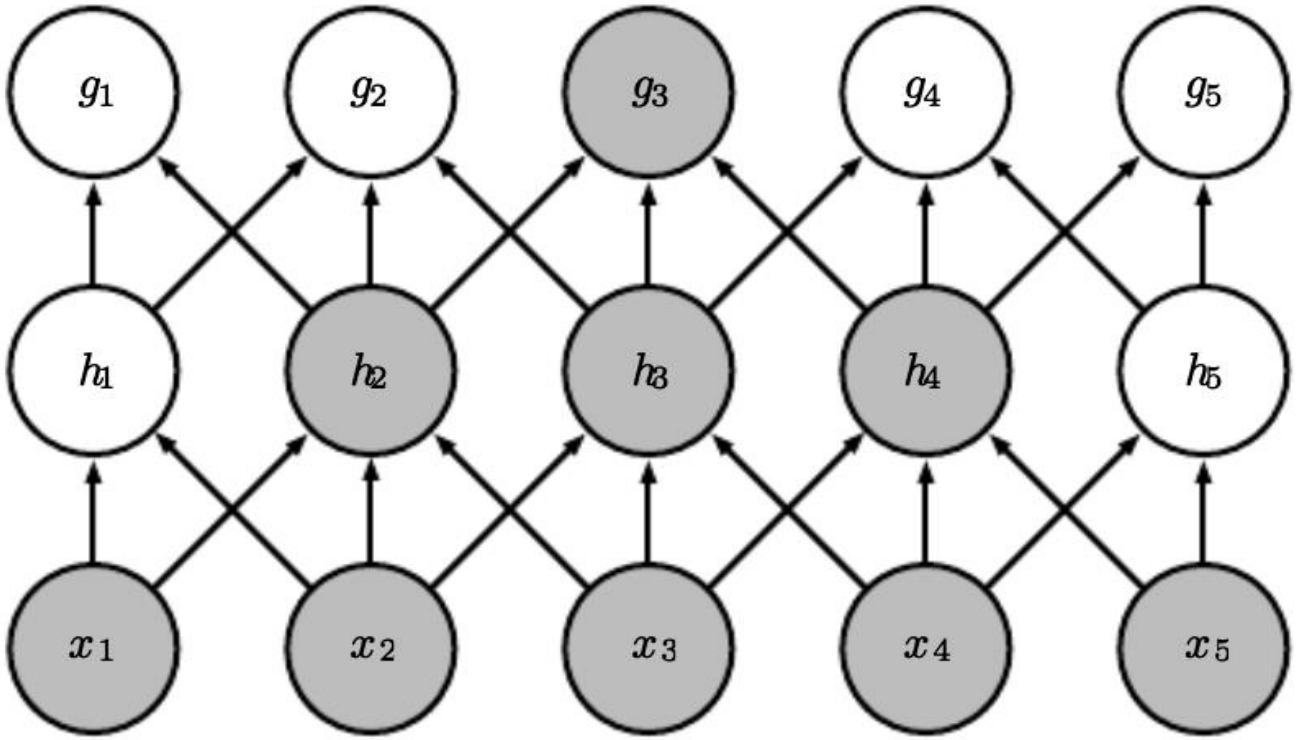


Changes output from a single classification to a slice/heatmap per class, with each slice's value indicating whether a coarse image region belongs to that class

Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

Architecture: Why Coarse Region Classification?

Stacking convolutional layers leads to learning patterns in increasingly **larger regions of the input (e.g., pixel) space.**



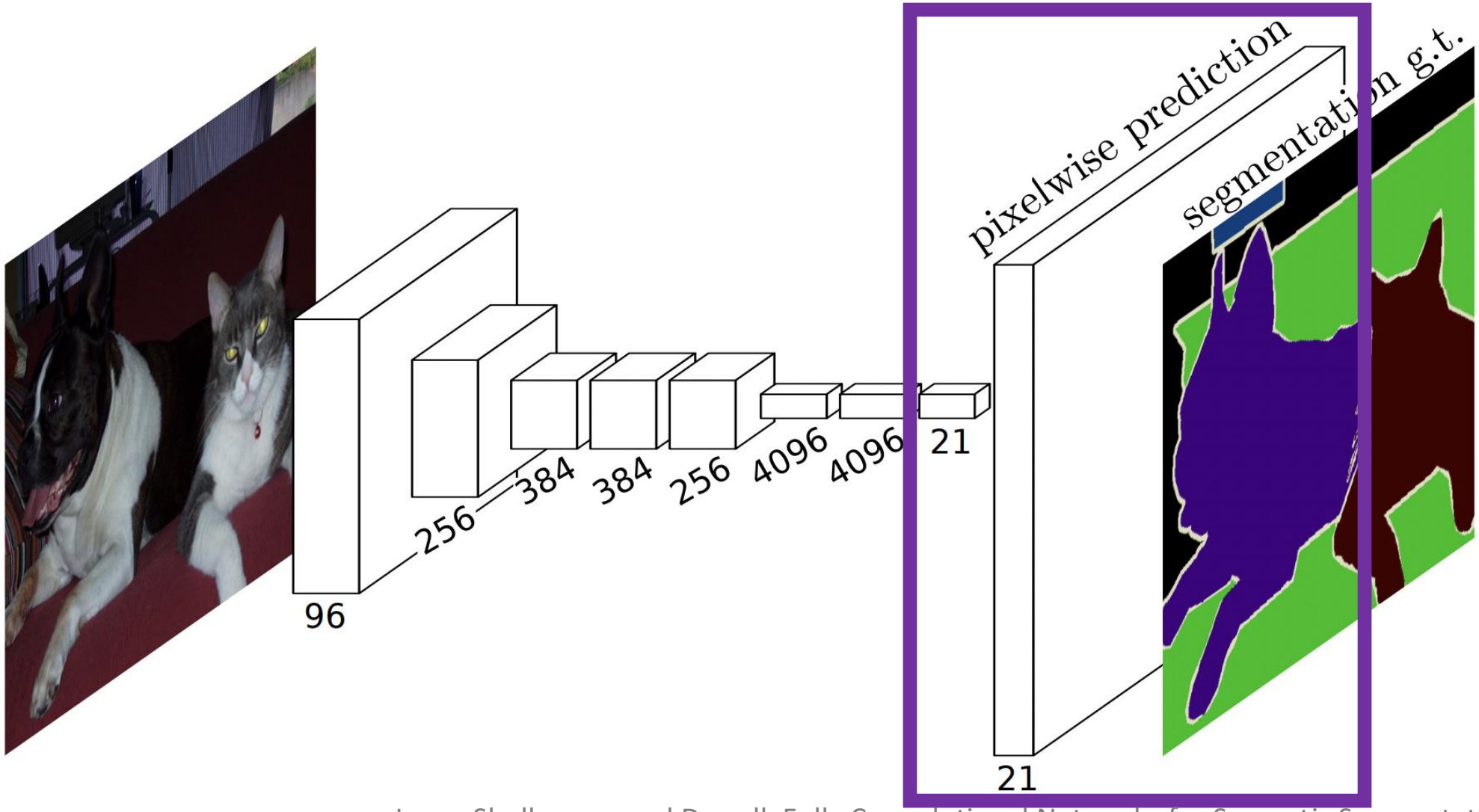
https://www.deeplearningbook.org/contents/convnets.html

Key Ideas

- Enable dense output: per pixel classification output layer
- Support any input image dimension: fully convolutional network
- Generate high-quality segmentations: upsampling and skip connections
- Make learning feasible: supervised pre-training

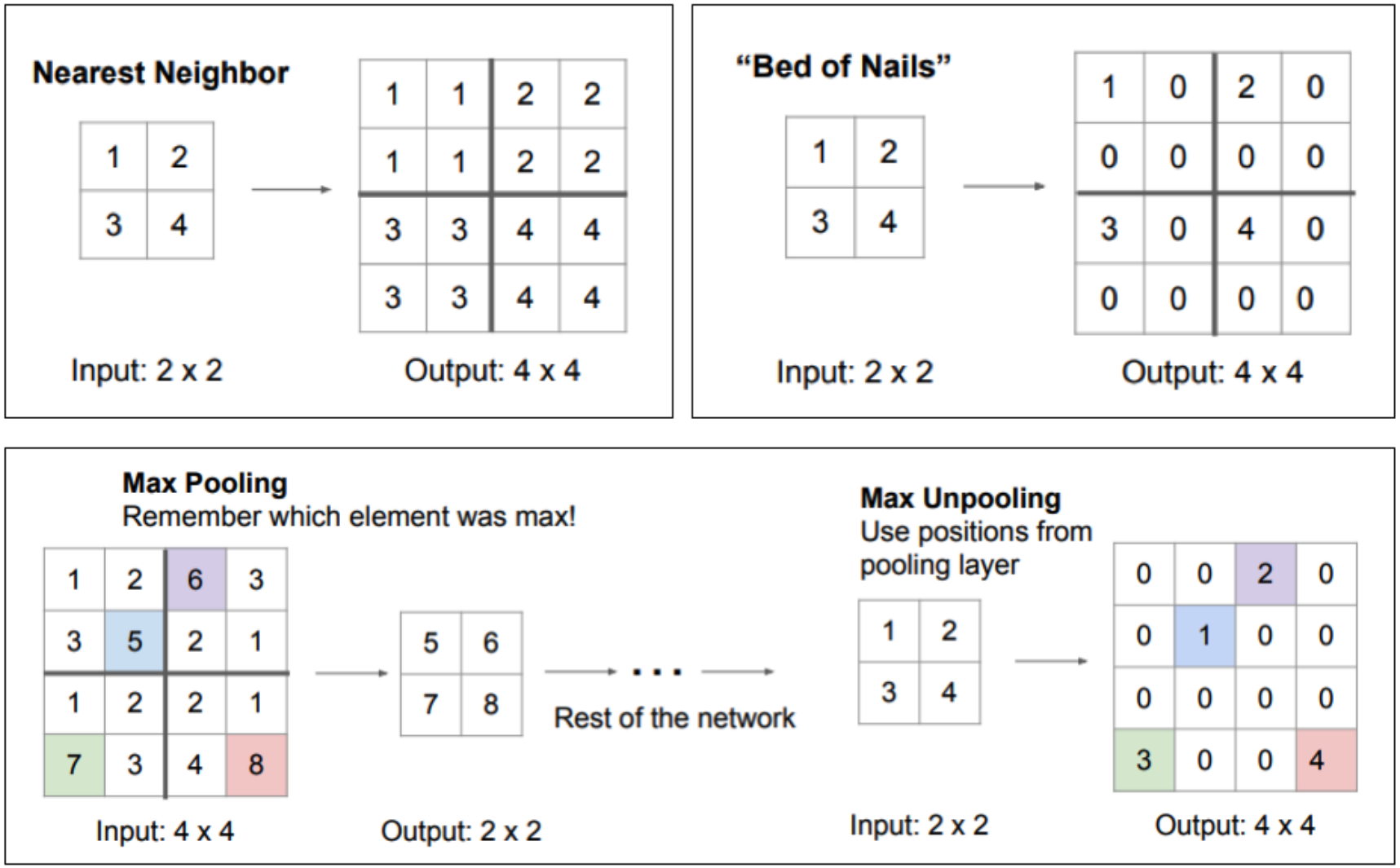
Architecture

How to decode coarse region classifications to per-pixel classification?



Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

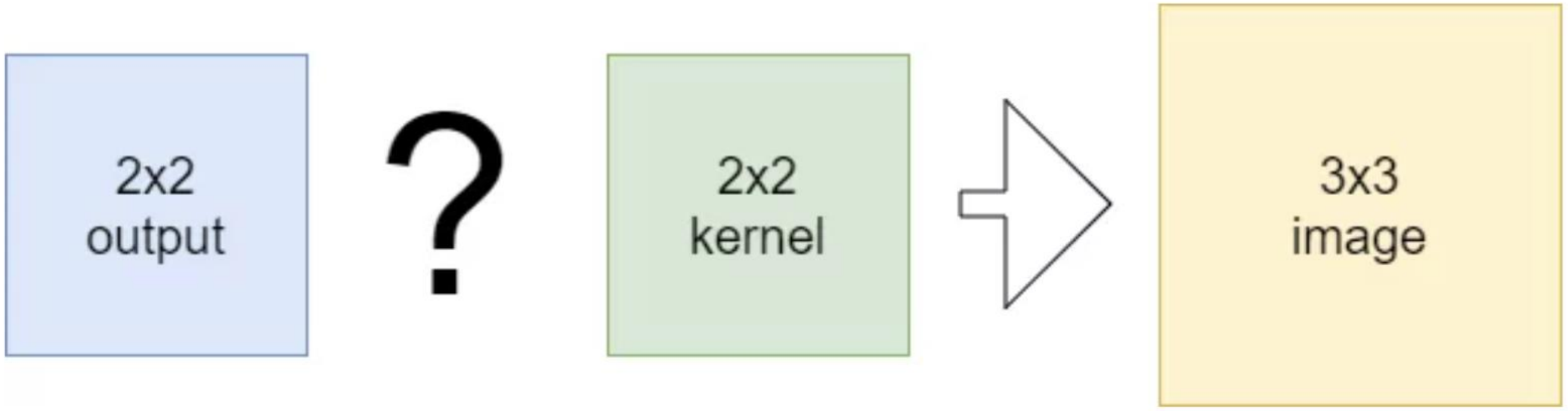
Architecture: Upsampling (Many Approaches)



http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf

Architecture: Upsampling (Transposed Convolutional Layer)

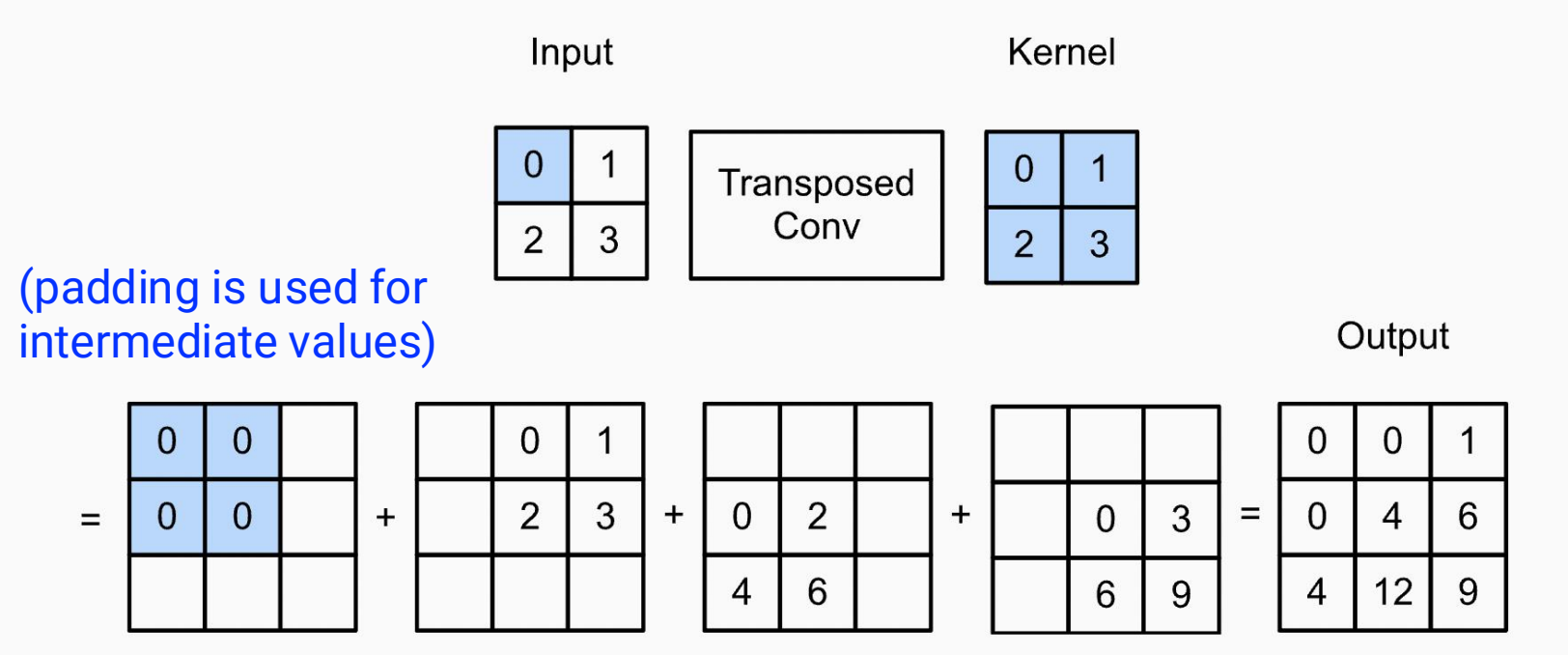
- **Idea**: learn convolutional filters to upsample the coarse image with “fractional” sized steps
- Also called “fractional convolutional layer”, “backward convolution”, and, incorrectly, ”deconvolution layer”, there are many implementations



https://www.machinecurve.com/index.php/2019/09/29/understanding-transposed-convolutions/#the-goal-reconstructing-the-original-input

Architecture: Upsampling (Transposed Convolutional Layer)

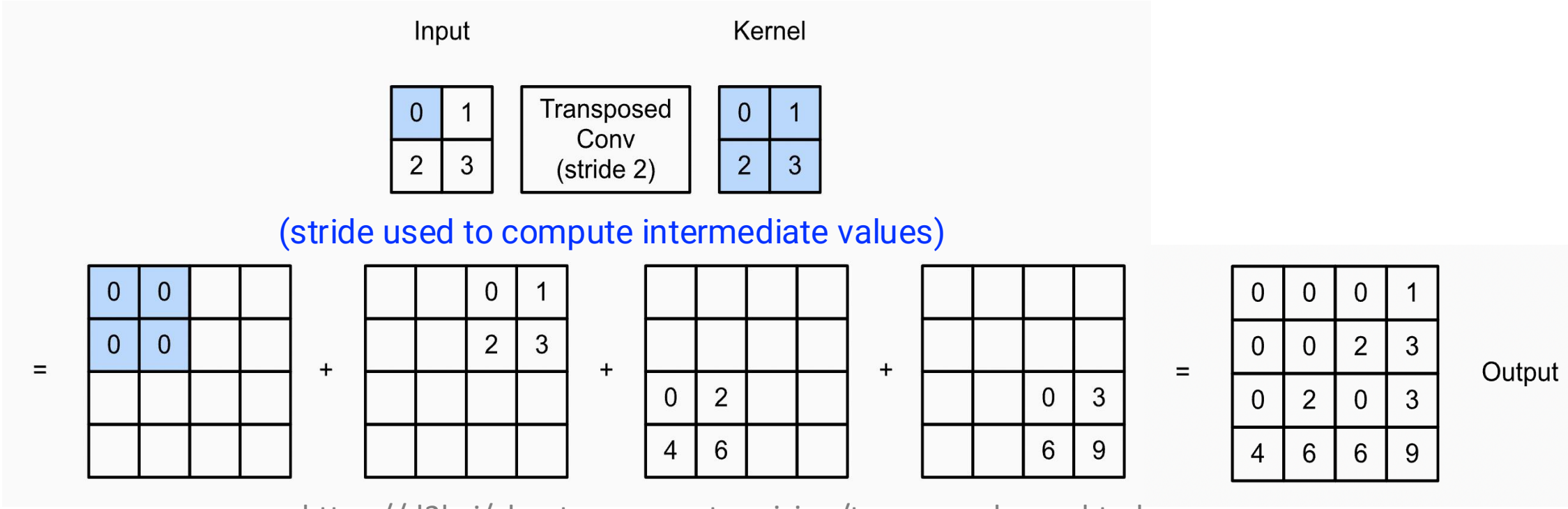
- **Idea**: learn convolutional filters to upsample the coarse image with “fractional” sized steps
- Also called “fractional convolutional layer”, “backward convolution”, and, incorrectly, ”deconvolution layer”, there are many implementations



https://d2l.ai/chapter_computer-vision/transposed-conv.html

Architecture: Upsampling (Transposed Convolutional Layer)

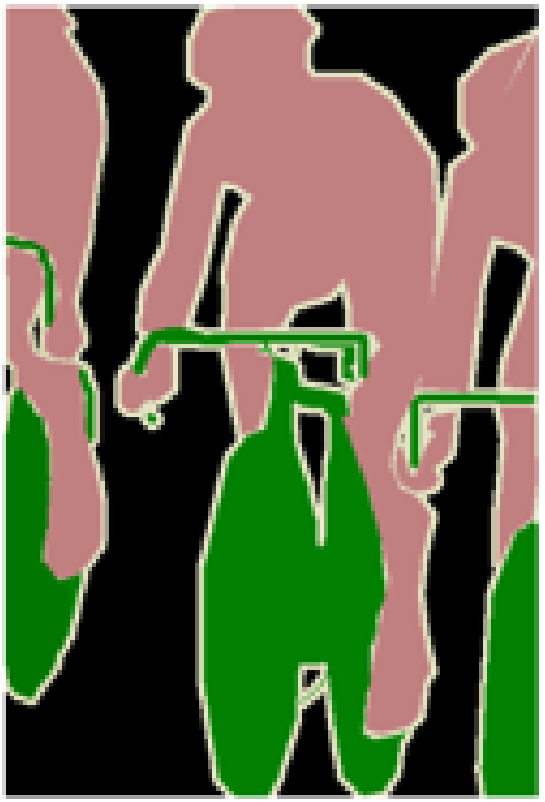
- **Idea**: learn convolutional filters to upsample the coarse image with “fractional” sized steps
- Also called “fractional convolutional layer”, “backward convolution”, and, incorrectly, ”deconvolution layer”, there are many implementations



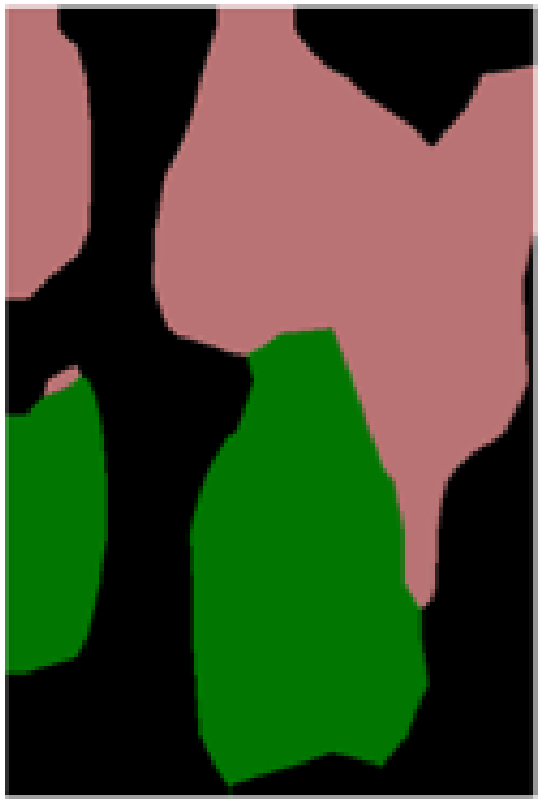
https://d2l.ai/chapter_computer-vision/transposed-conv.html

Problem: We Still Get Coarse Segmentations

Ground truth target

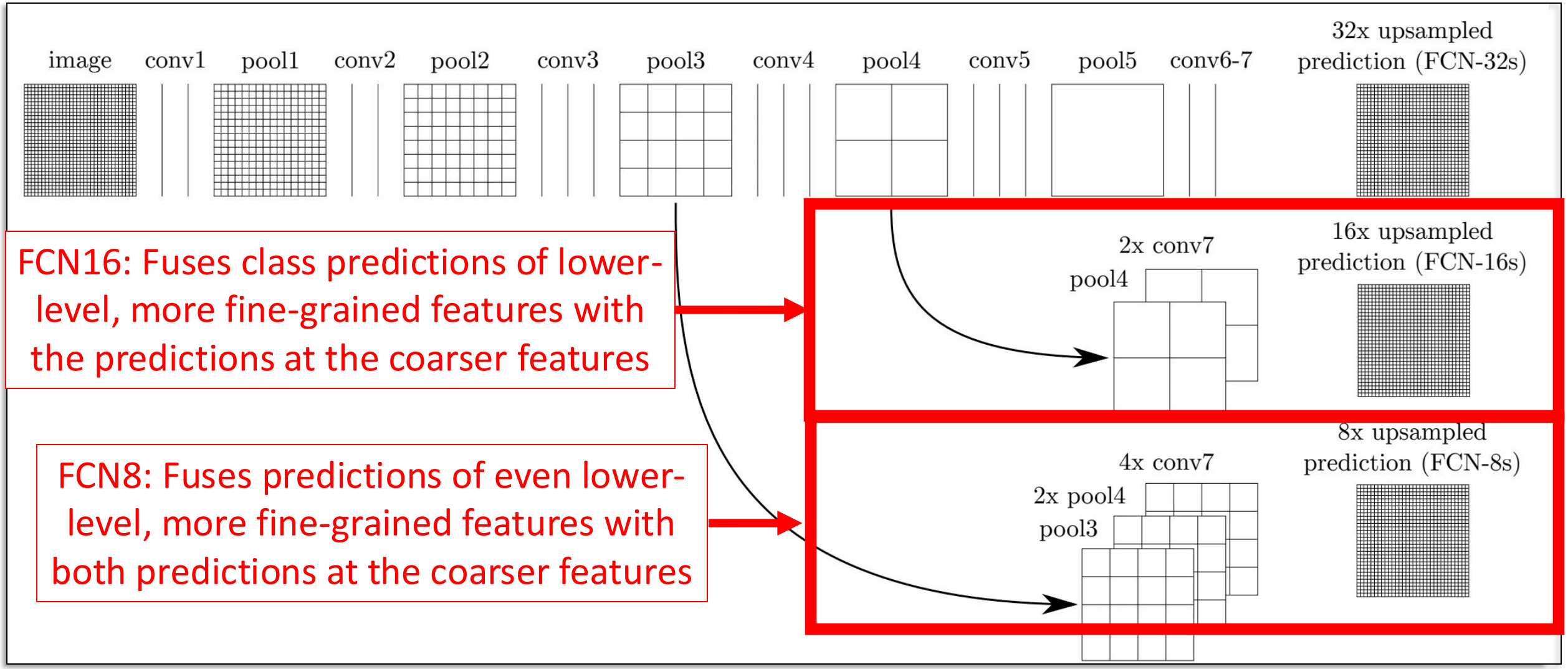


Predicted segmentation

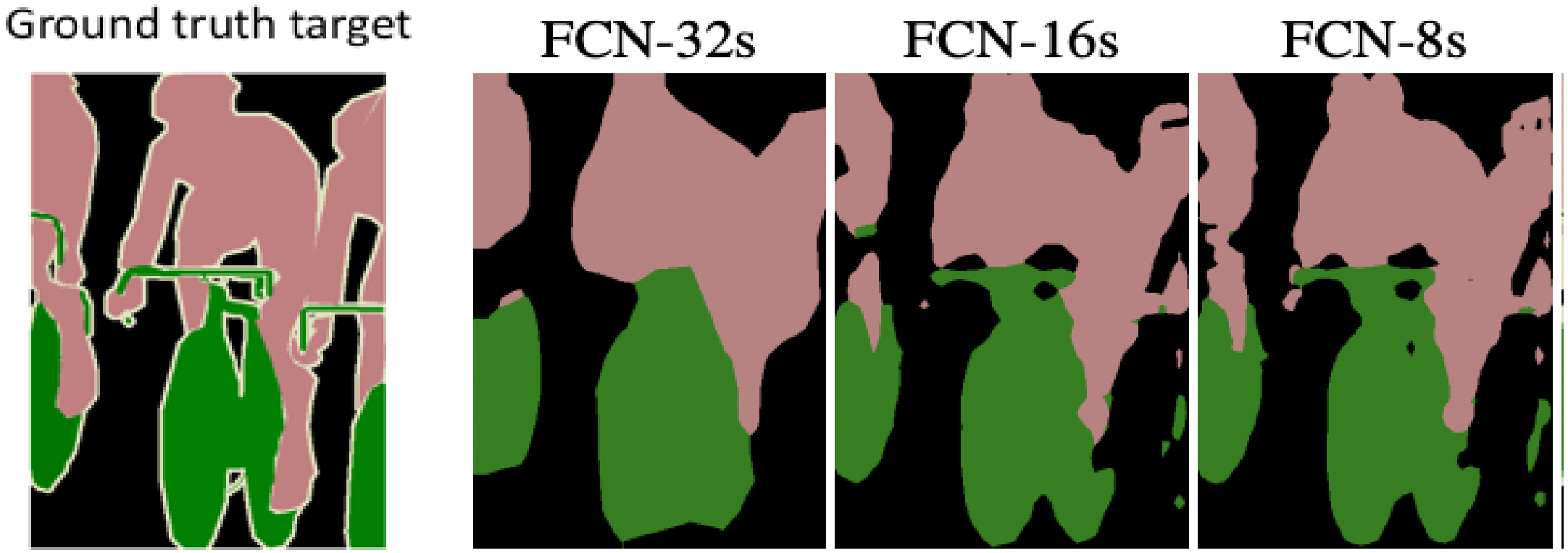


Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

Architecture: Update to Use Skip Connections



Architecture Results



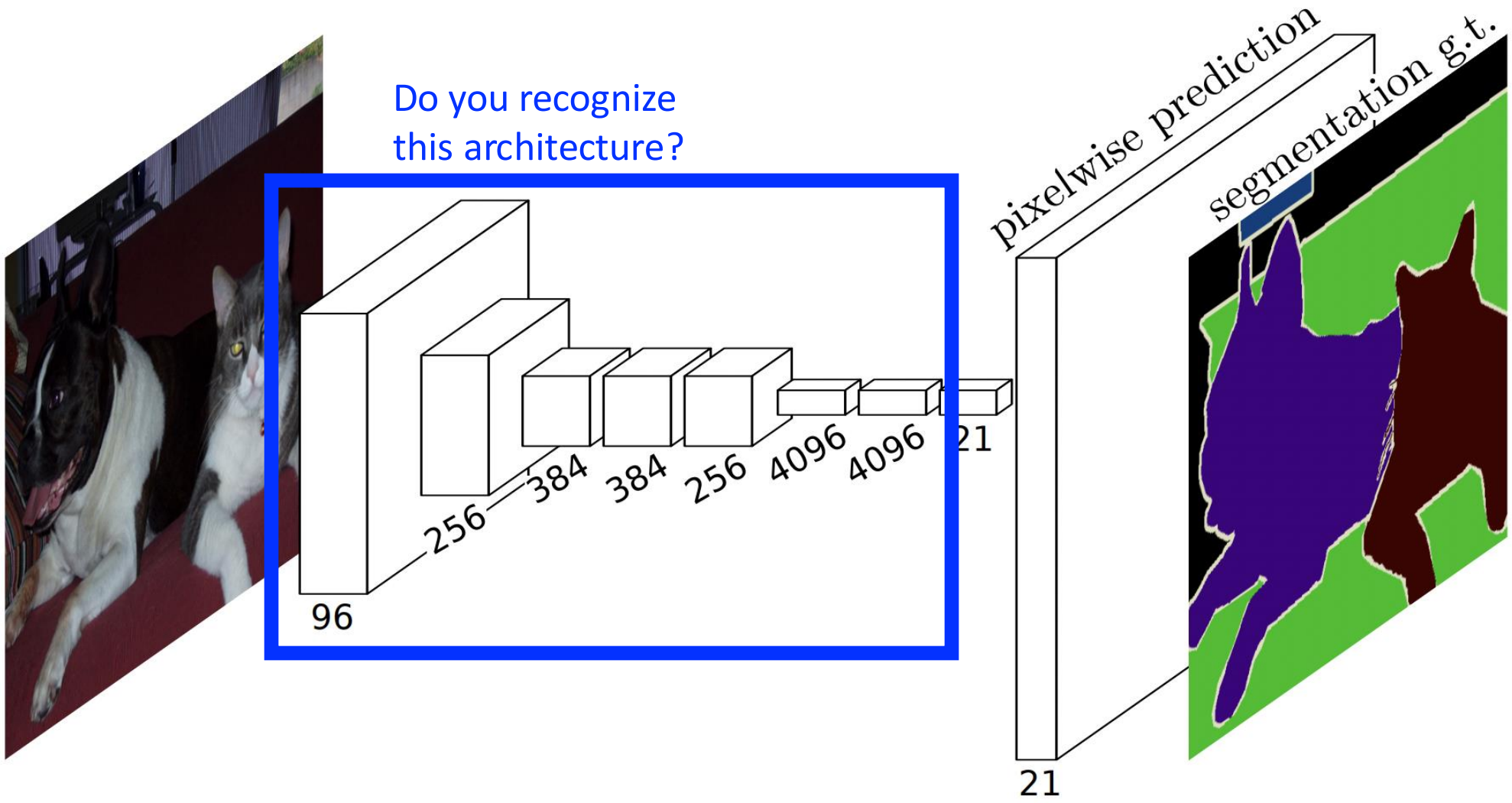
Skip connections support capturing finer-grained details while retaining correct semantic information!

Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

Key Ideas

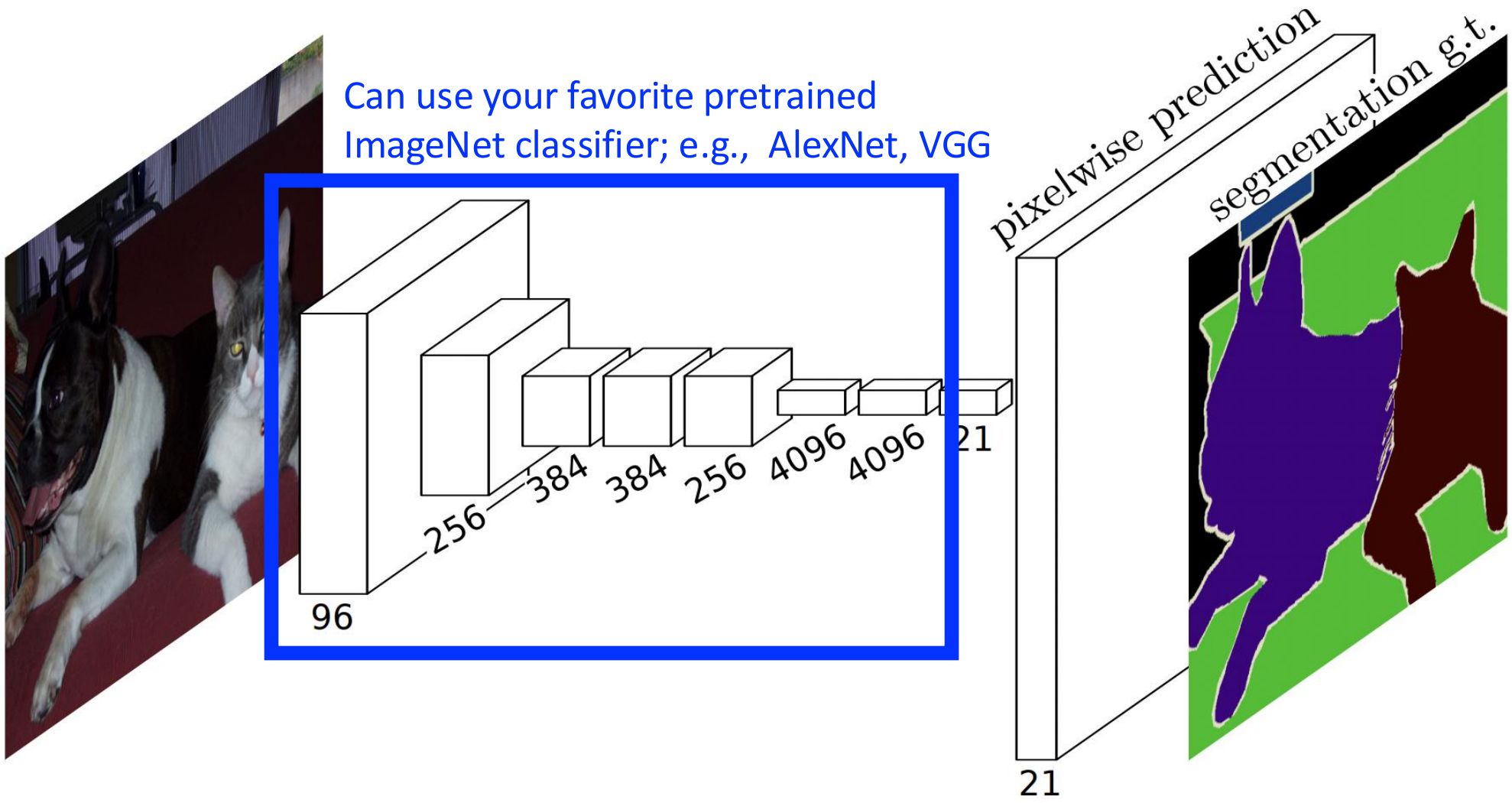
- Enable dense output: per pixel classification output layer
- Support any input image dimension: fully convolutional network
- Generate high-quality segmentations: upsampling and skip connections
- Make learning feasible: supervised pre-training

Architecture



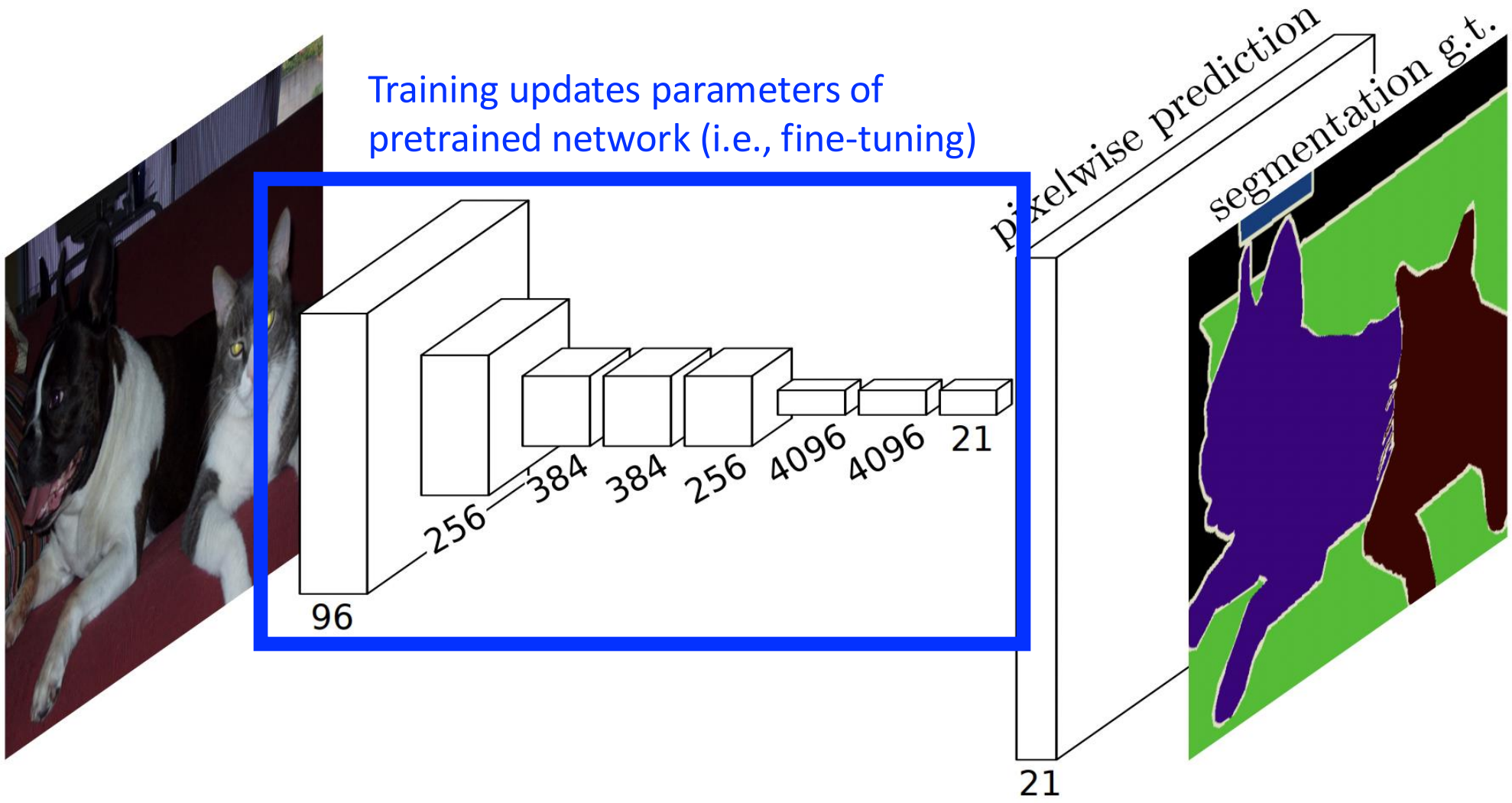
Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

Architecture



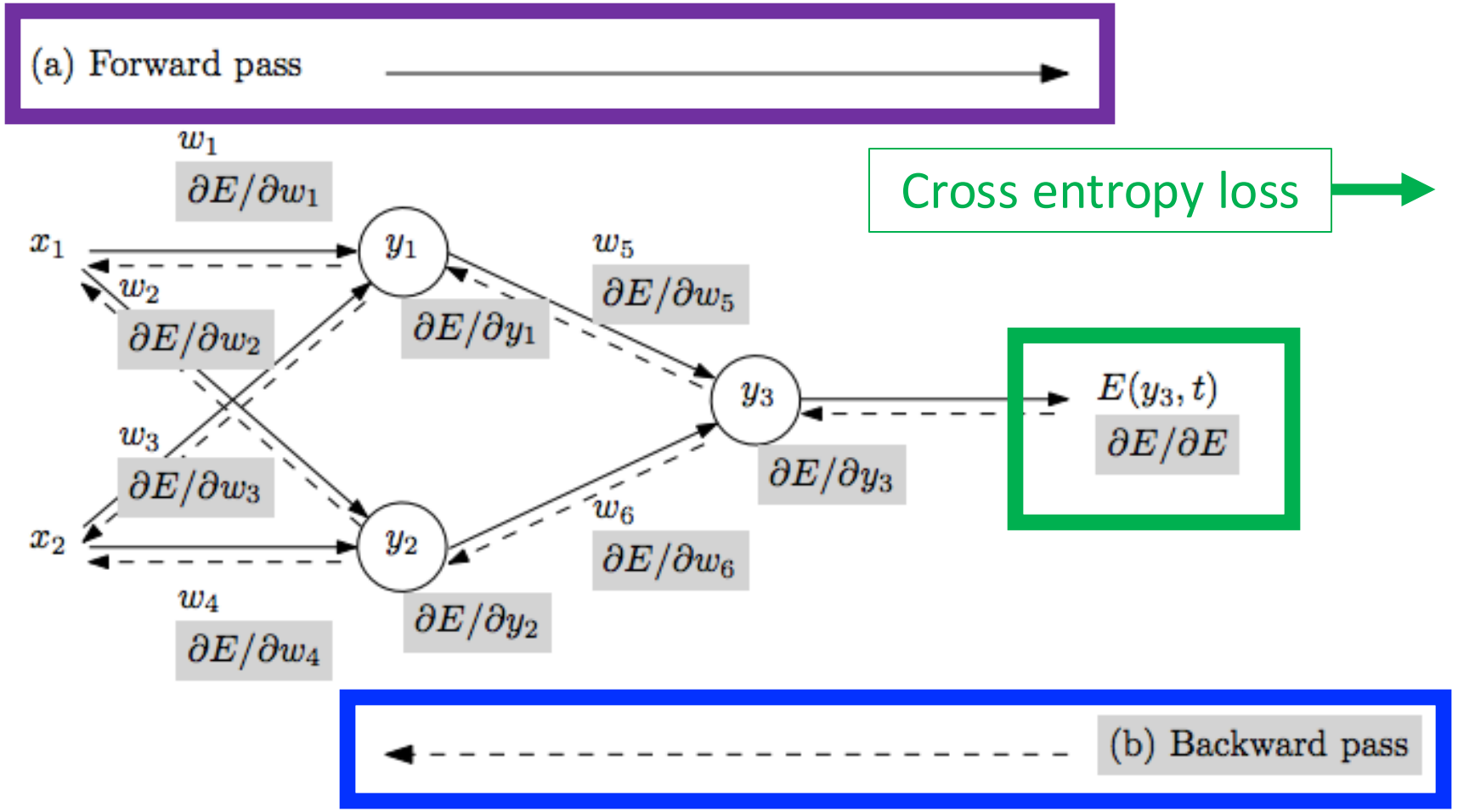
Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

Architecture Fine-Tuning



Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

Training: Took 3 days on 1 GPU



Cross entropy loss

Repeat until stopping criterion met:

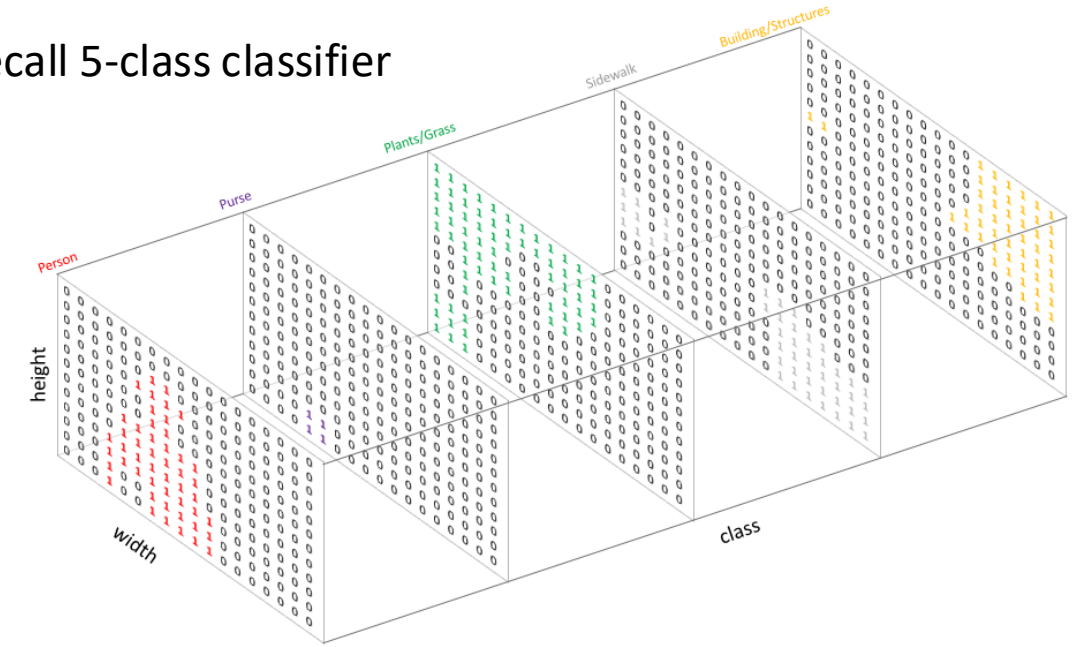
1. **Forward pass**: propagate training data through model to make predictions
2. **Error quantification**: measure error of the model's predictions on training data using a loss function
3. **Backward pass**: calculate gradients to determine how each model parameter contributed to model error
4. Account for weight sharing by using average of all connections for a parameter
5. Update each parameter using calculated gradients

Baydin et al. Automatic Differentiation in Machine Learning: a Survey. 2018

Training: Took 3 days on 1 GPU

Sums across all pixels distance between predicted and true distributions using cross entropy loss

e.g., recall 5-class classifier



Sum of gradients for all pixels (acts like a minibatch)

Repeat until stopping criterion met:

1. **Forward pass**: propagate training data through model to make predictions
2. **Error quantification**: measure error of the model's predictions on training data using a loss function
3. **Backward pass**: calculate gradients to determine how each model parameter contributed to model error
4. Account for weight sharing by using average of all connections for a parameter
5. Update each parameter using calculated gradients

https://www.jeremyjordan.me/semantic-segmentation/

Performance Analysis

	mean IU VOC2011 test	mean IU VOC2012 test	inference time
R-CNN [12]	47.9	-	-
SDS [16]	52.6	51.6	$\sim$ 50 s
FCN-8s	**62.7**	**62.2**	$\sim$ **175 ms**

Compared to existing methods at the time, the model produced better results at a faster speed!

Long, Shelhamer, and Darrell. Fully Convolutional Networks for Semantic Segmentation. CVPR 2015.

Recap: Tricks for Semantic Segmentation

- Enable dense output: per pixel classification output layer
- Support any input image dimension: fully convolutional network
- Generate high-quality segmentations: upsampling and skip connections
- Make learning feasible: supervised pre-training

Today's Topics

- Motivation: training large capacity, deep models to locate content
- Semantic segmentation: classifying pixels
- Object detection: locating objects with bounding rectangles
- Instance segmentation: demarcating objects with detailed outlines
- Programming tutorial

Faster R-CNN

Named after the proposed technique: **R**egion proposals with **CNN** features

> Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.” Neurips 2015.

Idea: test for a “manageable” number of image regions with diverse properties (e.g., scales, aspect ratios) whether target object types are there

Key Ideas

- Enable dense output: multi-task learning
- Support any input image dimension: ROI pooling
- Generate high-quality candidate regions: region proposals
- Make learning feasible: use supervised pre-training

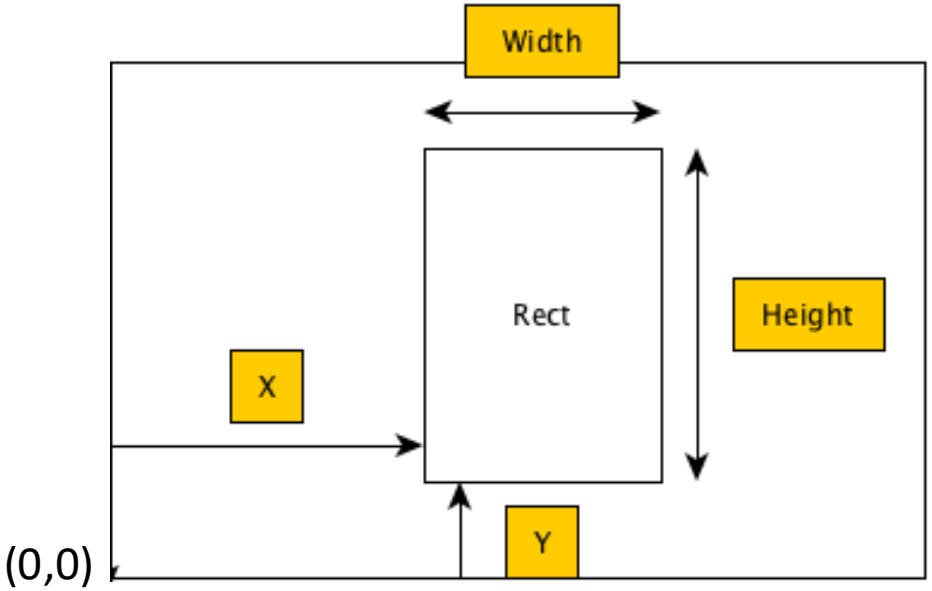
Key Ideas

- Enable dense output: multi-task learning
- Support any input image dimension: ROI pooling
- Generate high-quality candidate regions: region proposals
- Make learning feasible: use supervised pre-training

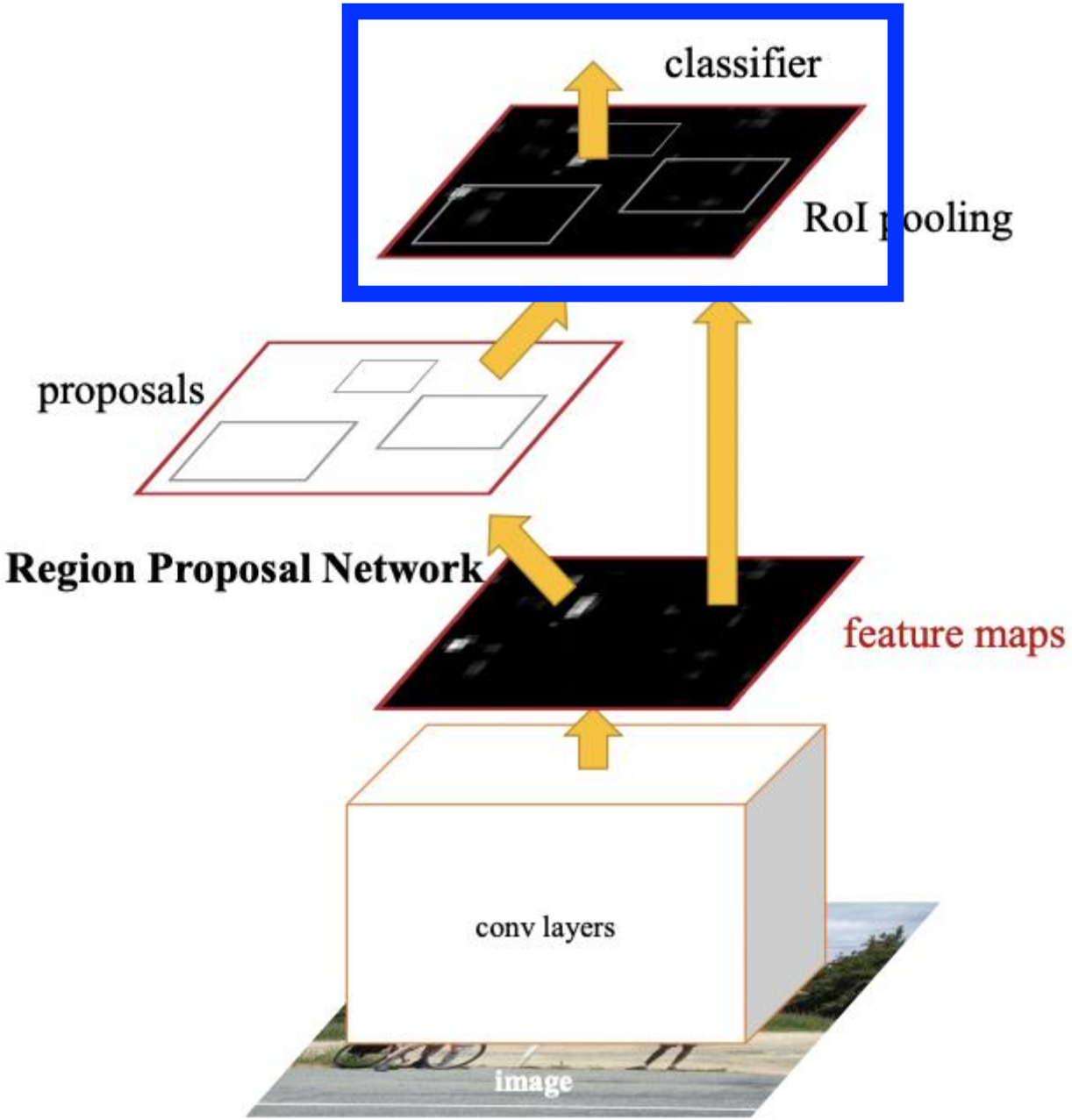
Architecture: Multi-Task Learning

Single model performs two tasks:

1. detects each object with four coordinates (x, y, w, h) and then



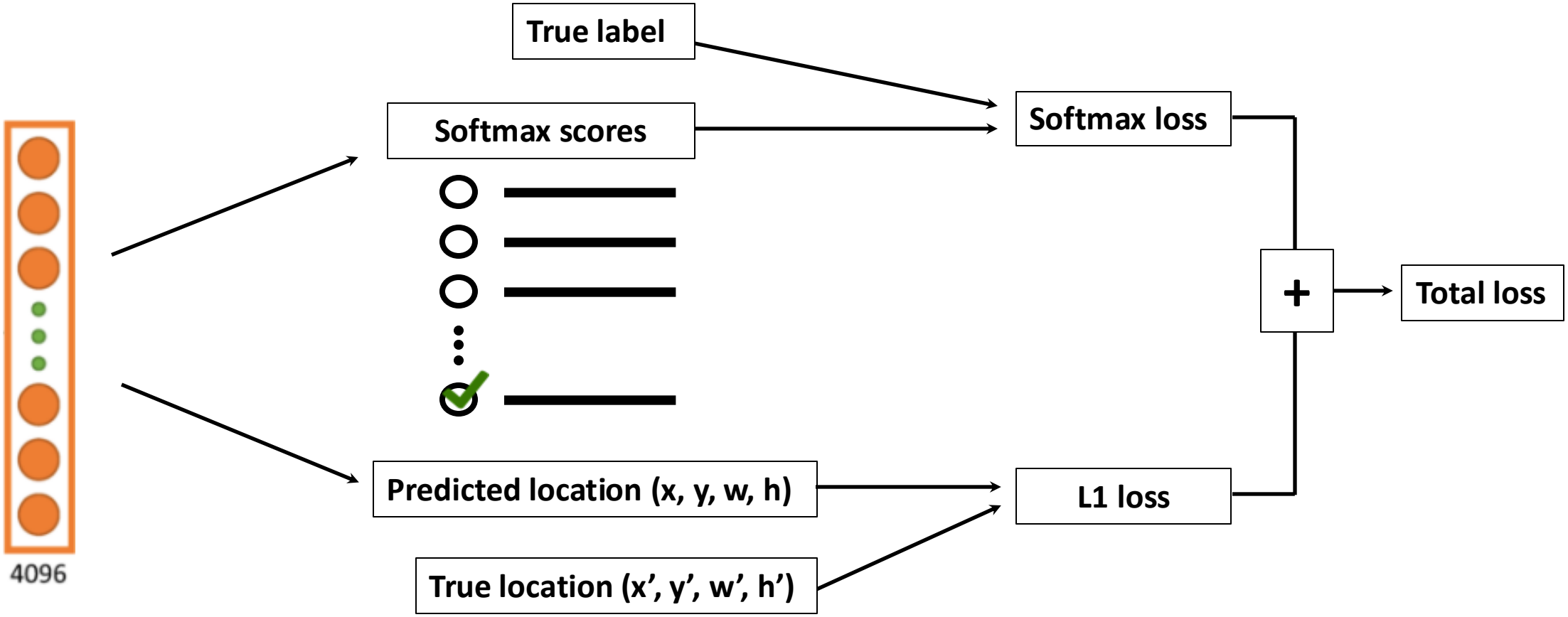
2. classifies category per region



Ren et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Neurips 2015.

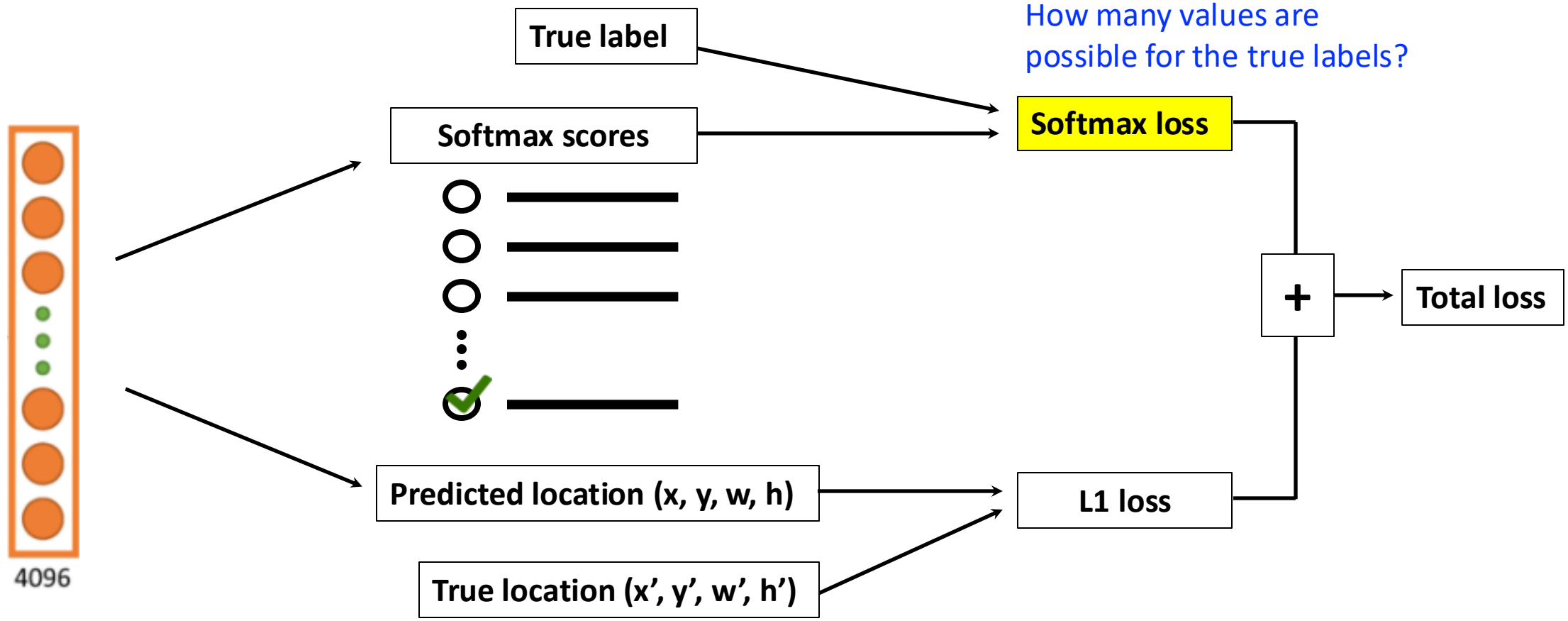
Objective Function: Multi-task Loss

Sums classification and localization losses for each region proposal:



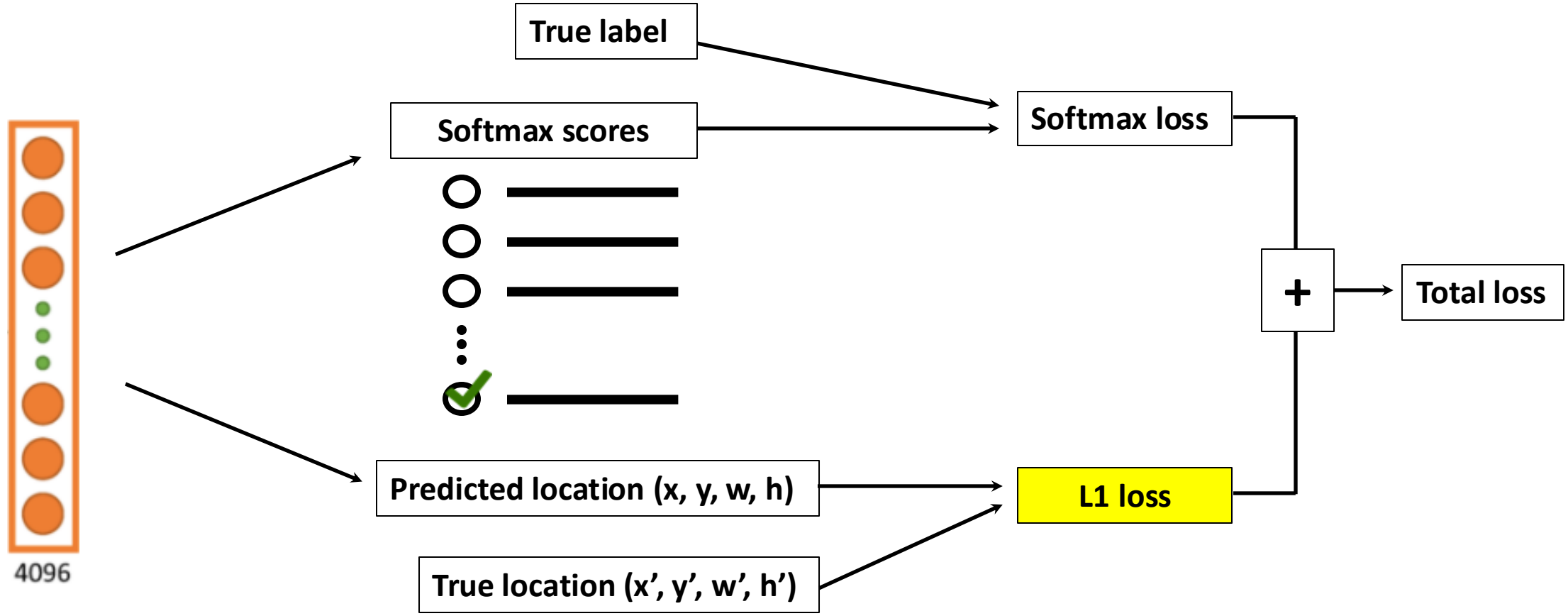
Objective Function: Multi-task Loss

Sums classification and localization losses for each region proposal:



Objective Function: Multi-task Loss

Sums classification and localization losses for each region proposal:



Training: Region Proposal Multi-task Loss

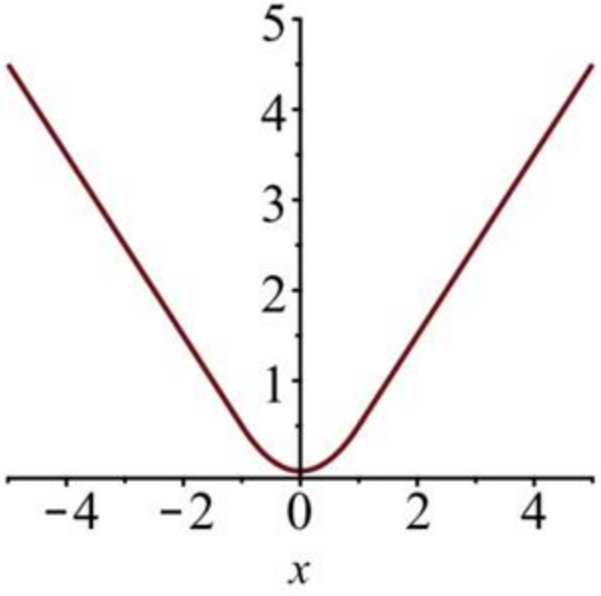
$$\mathcal{L}_{\text{box}}(t^u, v) = \sum_{i \in \{x,y,w,h\}} L_1^{\text{smooth}}(t_i^u - v_i)$$

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

True location for true class "u"

Predicted location for class u

Less sensitive to outliers than SSE

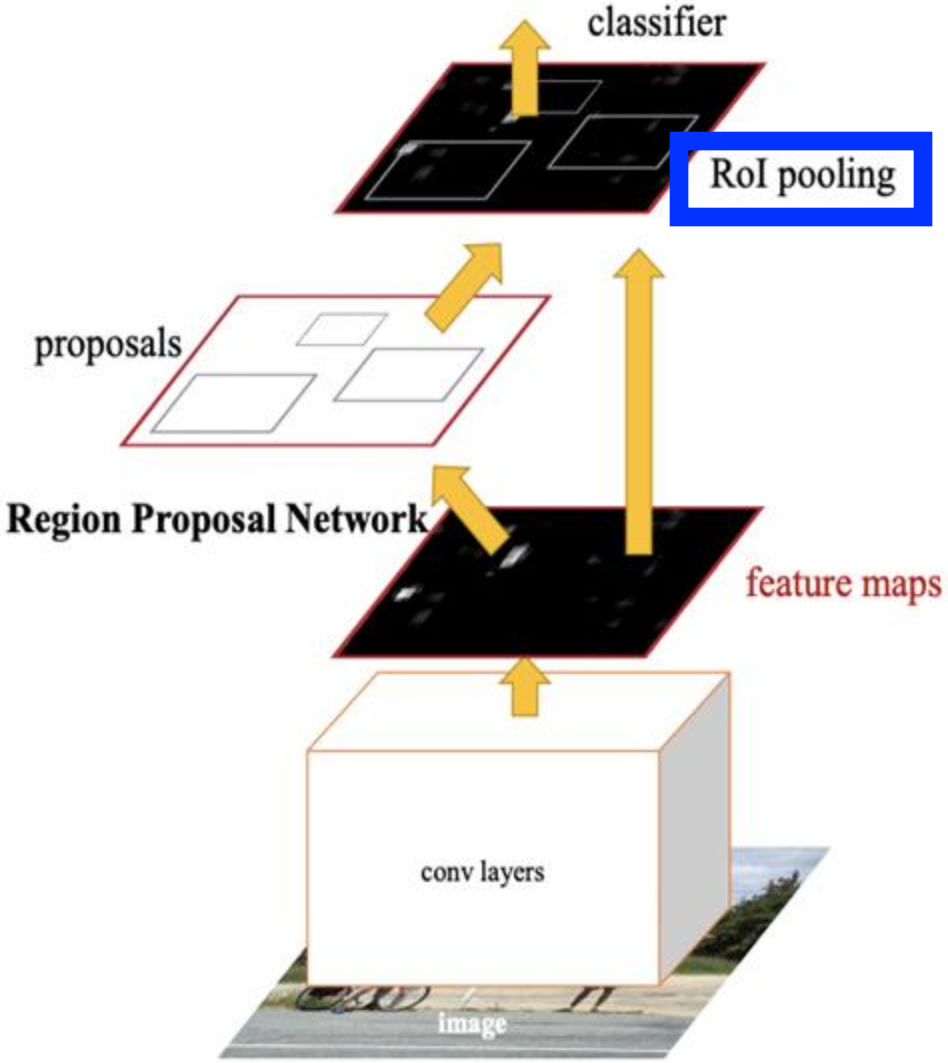


https://lilianweng.github.io/lil-log/2017/12/31/object-recognition-for-dummies-part-3.html#bounding-box-regression

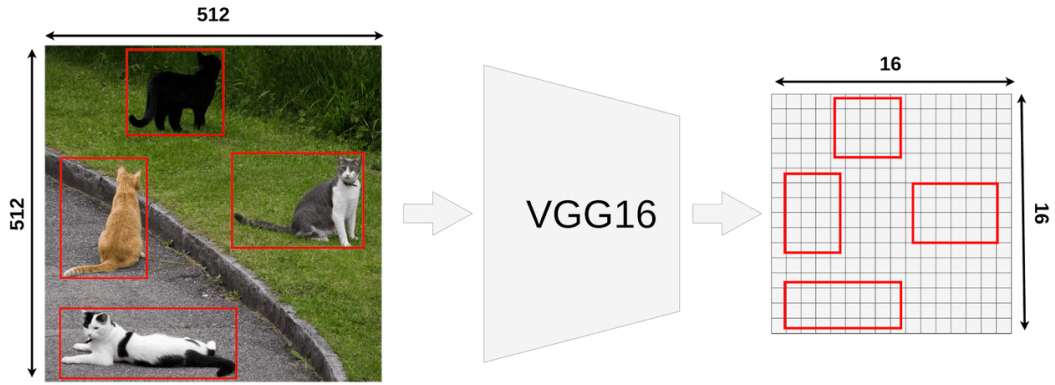
Key Ideas

- Enable dense output: multi-task learning
- Support any input image dimension: ROI pooling
- Generate high-quality candidate regions: region proposals
- Make learning feasible: use supervised pre-training

ROI Pooling: Quantization + Pooling



1. Quantization by downsizing feature map; e.g., 1/32 of 512 is 16

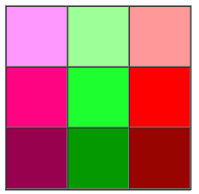


2. Pool quantized region into fixed size; e.g.,

4x6 RoI

0.1	0.2	0.3	0.4	0.5	0.6
1	0.7	0.2	0.6	0.1	0.9
0.9	0.8	0.7	0.3	0.5	0.2

3x3 RoI Pooling



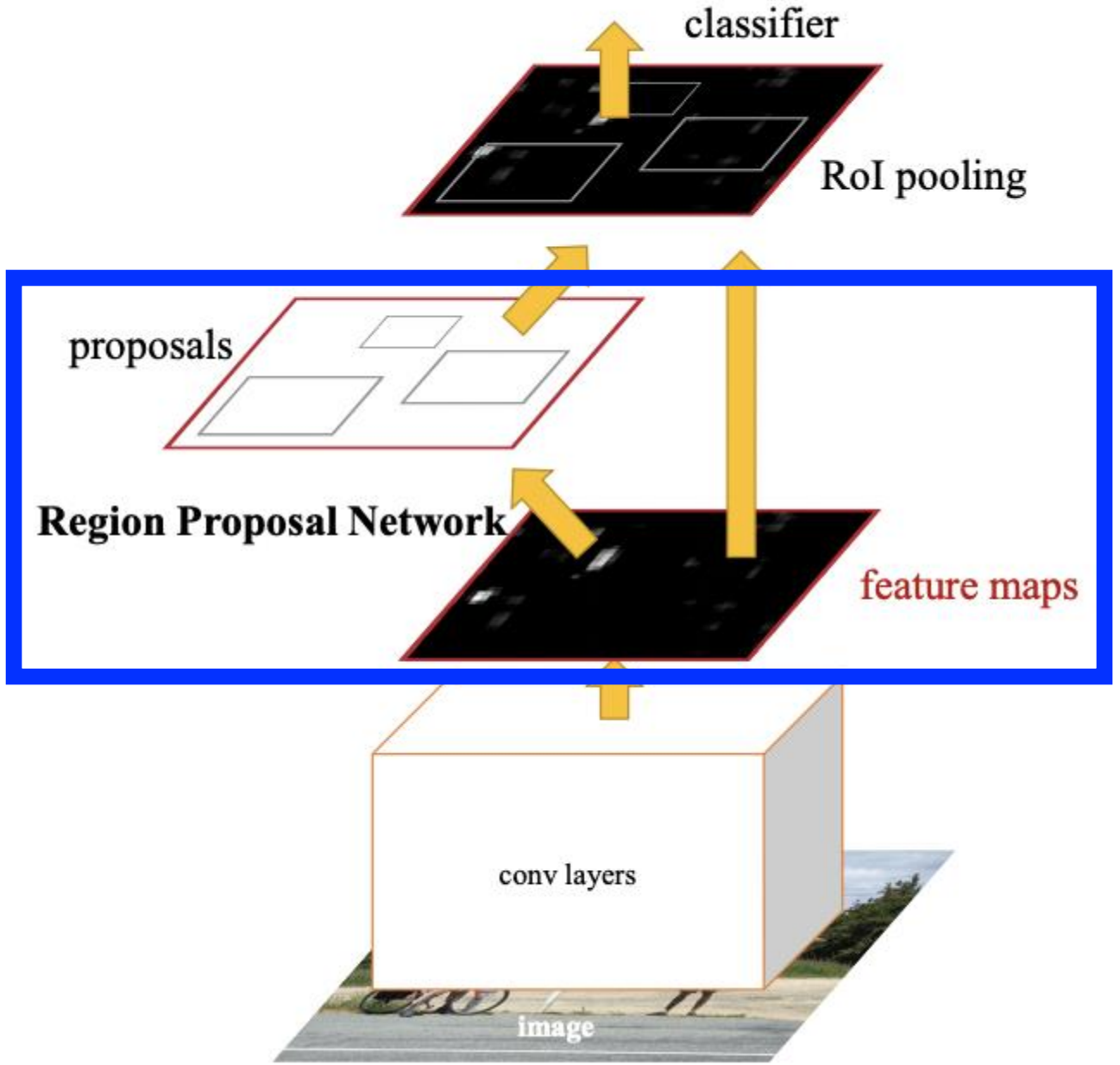
https://erdem.pl/2020/02/understanding-region-of-interest-ro-i-pooling

Ren et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Neurips 2015.

Key Ideas

- Enable dense output: multi-task learning
- Support any input image dimension: ROI pooling
- Generate high-quality candidate regions: region proposals
- Make learning feasible: use supervised pre-training

Architecture



Ren et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Neurips 2015.

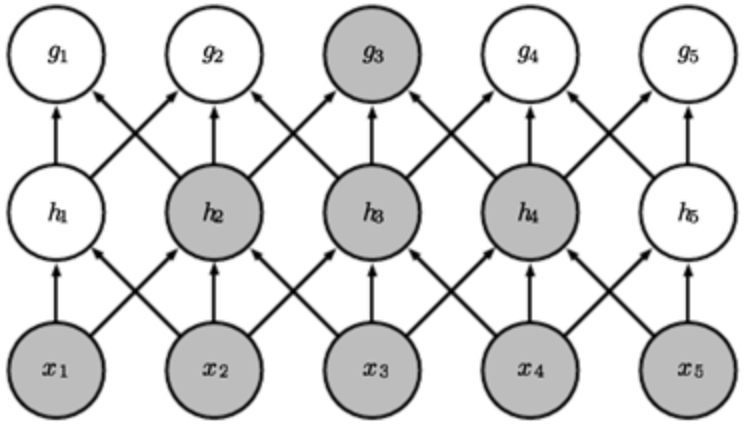
Idea: Be More Efficient than Sliding Window

https://yourboulder.com/boulder-neighborhood-downtown/

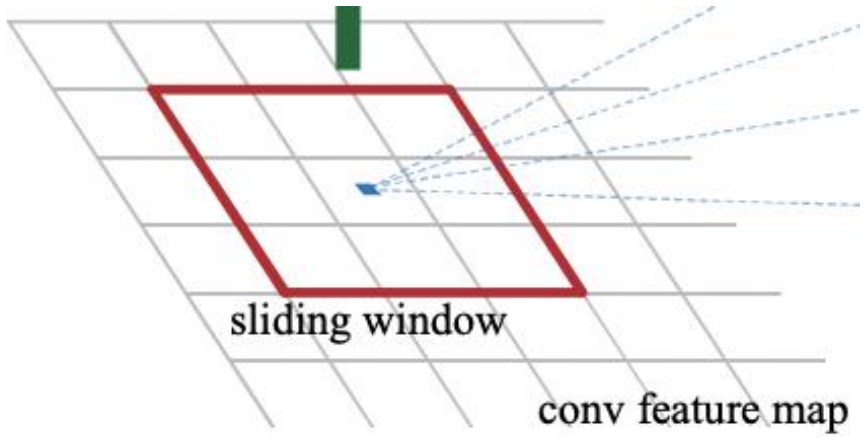
Architecture: Region Proposal Network

Input: convolutional feature map from pretrained model

Step 1: 3 x 3 convolutional filter applied to identify candidate proposals (recall, filter in the middle of an architecture maps to a larger input space, aka receptive field)



https://www.deeplearningbook.org/contents/convnets.html

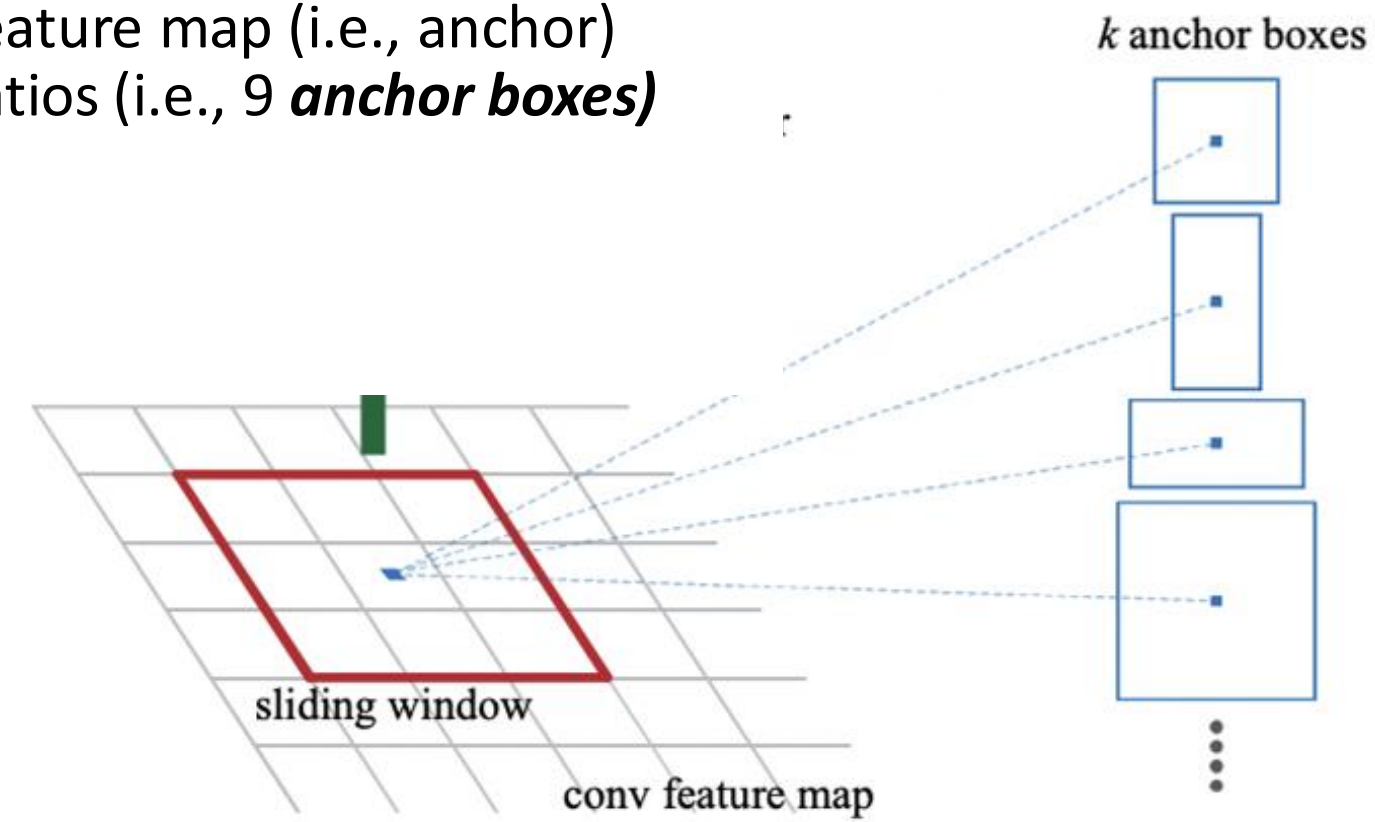


Ren et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Neurips 2015.

Architecture: Region Proposal Network

Step 2: multiple scales are efficiently supported by generating for each point on the feature map (i.e., anchor) boxes with 3 scales and 3 aspect ratios (i.e., 9 ***anchor boxes)***

Each anchor box specializes in a particular shape and size (centered on each pixel)

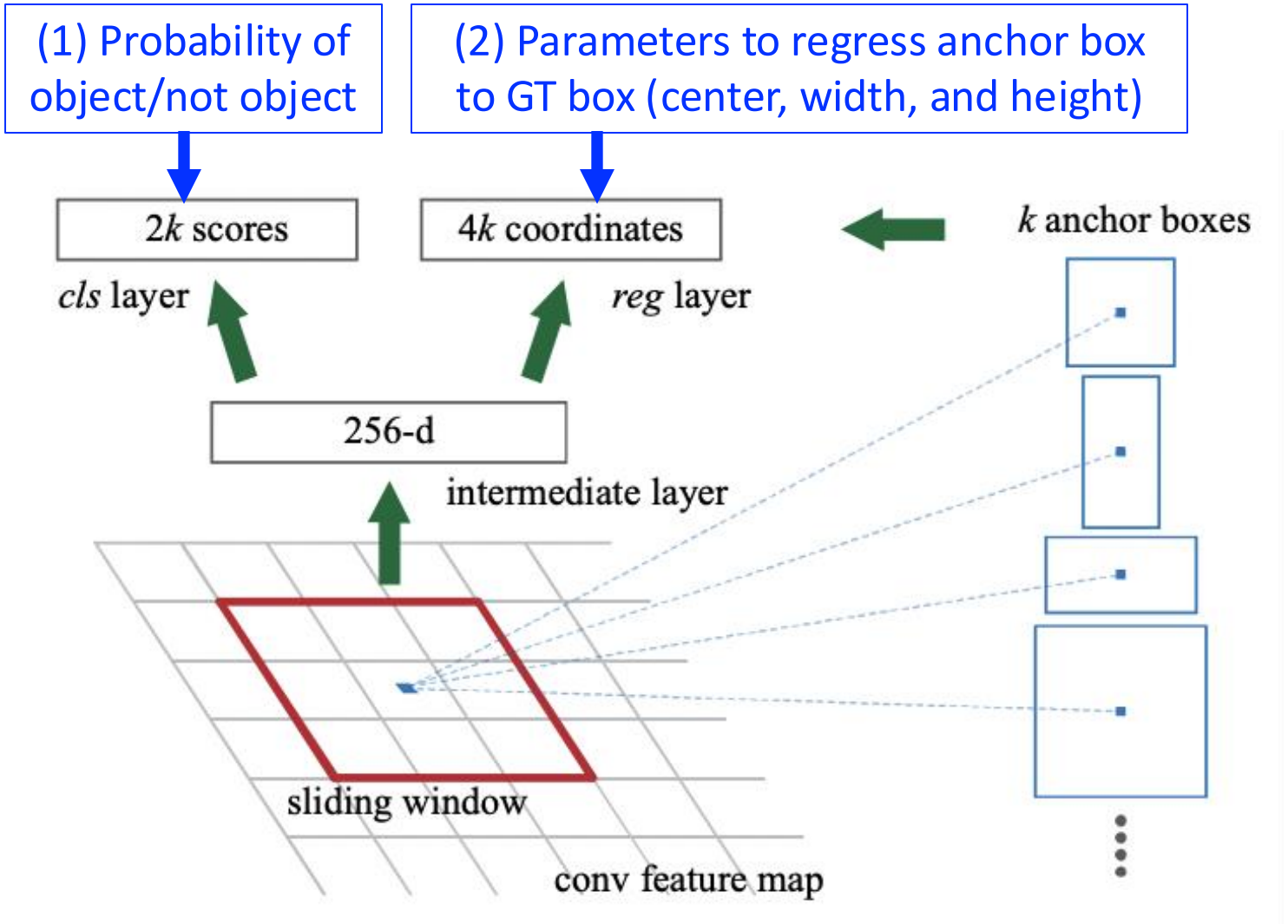


Ren et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Neurips 2015.

Architecture: Region Proposal Network

(*k* independent regressors learned to support *k* anchor box dimensions)

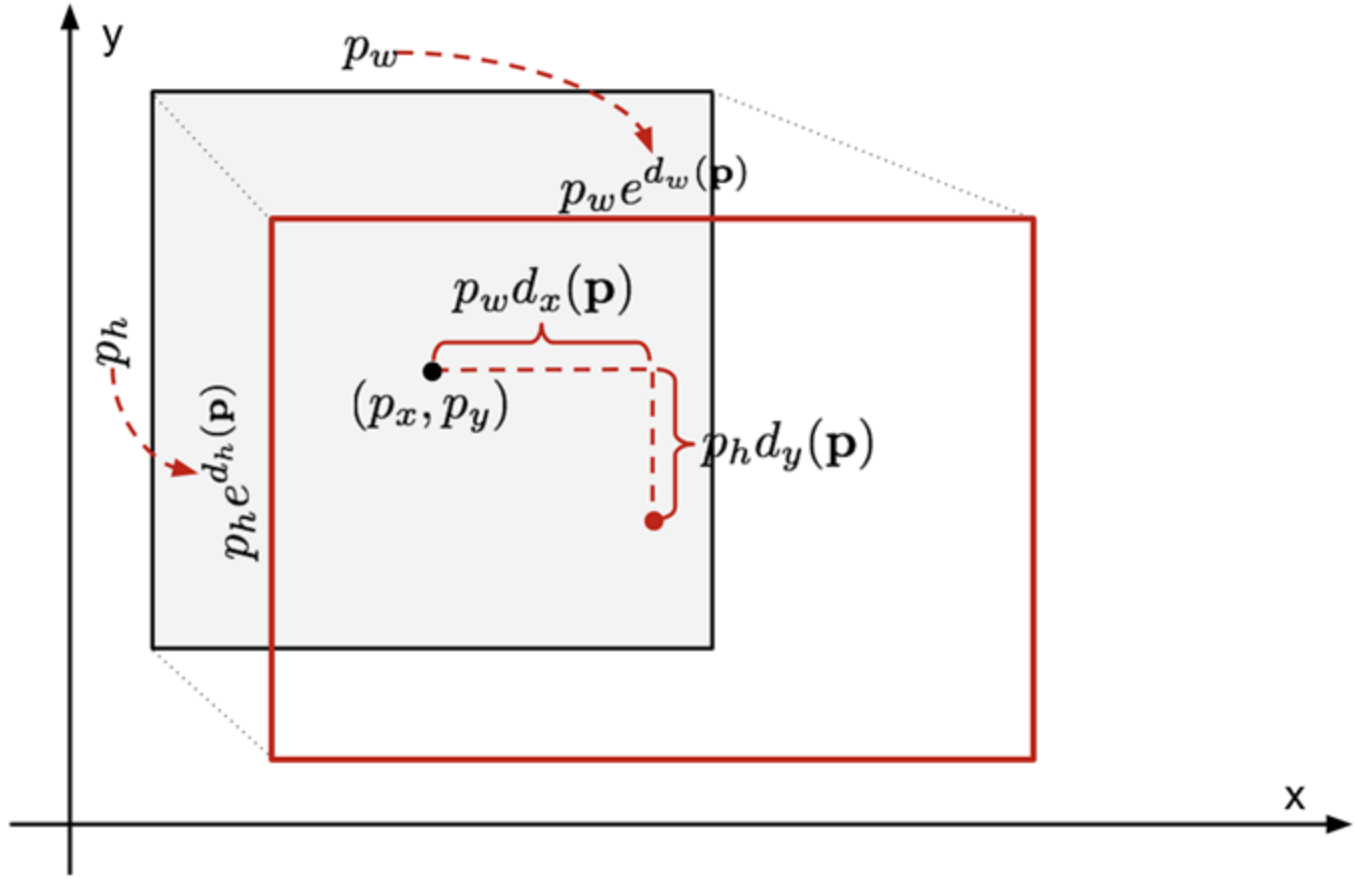
Step 3: For each region, then predict:



Ren et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Neurips 2015.

Architecture: Region Proposal Refinement

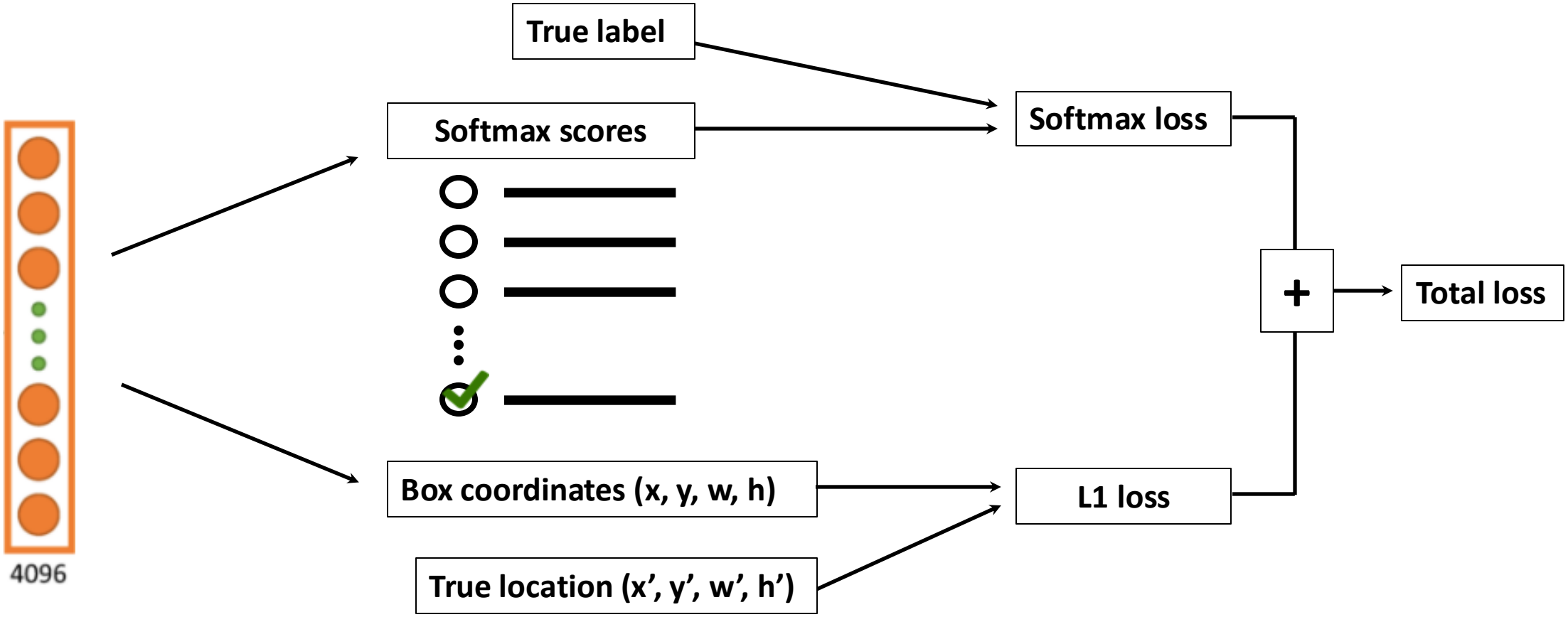
Parameters to regress original region proposal with center (p_x, p_y), width (p_w), and height (p_h) to the ground truth location: d_x, d_y, d_w, d_h



https://lilianweng.github.io/lil-log/2017/12/31/object-recognition-for-dummies-part-3.html#bounding-box-regression

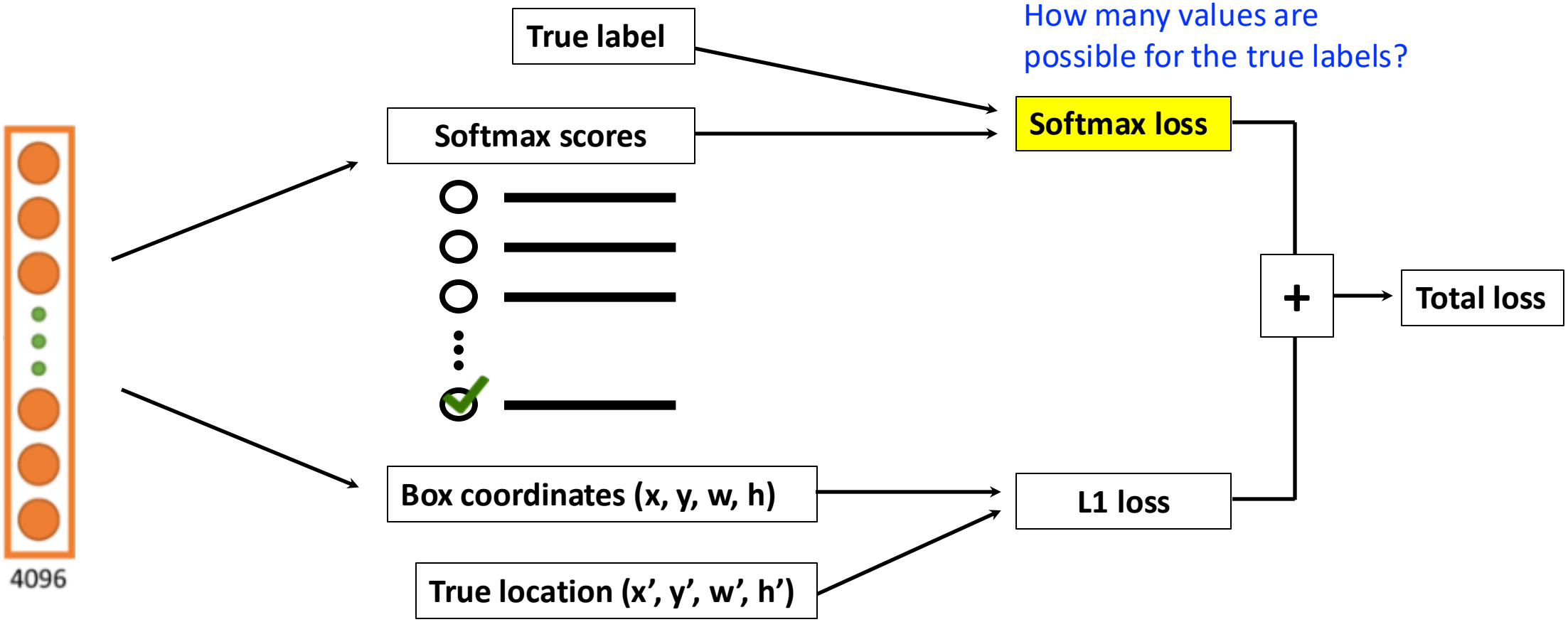
Region Proposal Multi-task Loss

Sum classification and (sometimes) localization losses for each region proposal



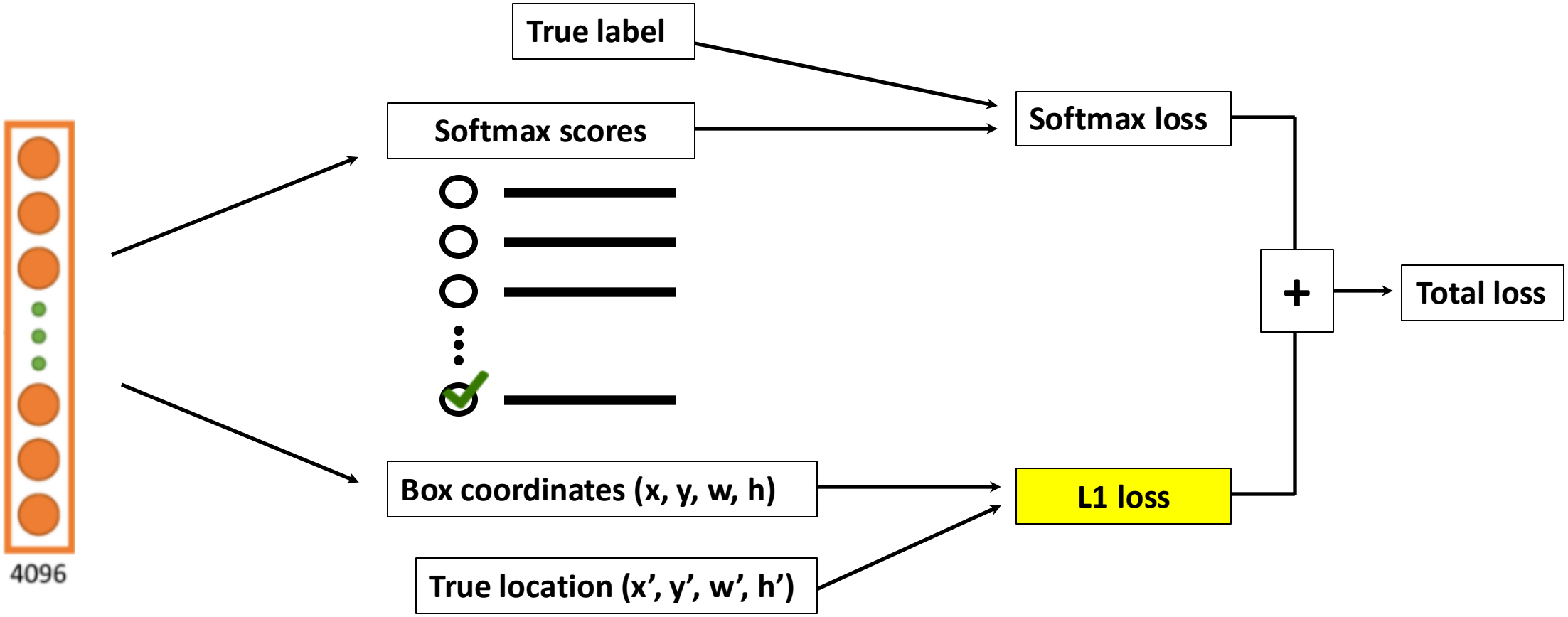
Region Proposal Multi-task Loss

Sum classification and (sometimes) localization losses for each region proposal



Region Proposal Multi-task Loss

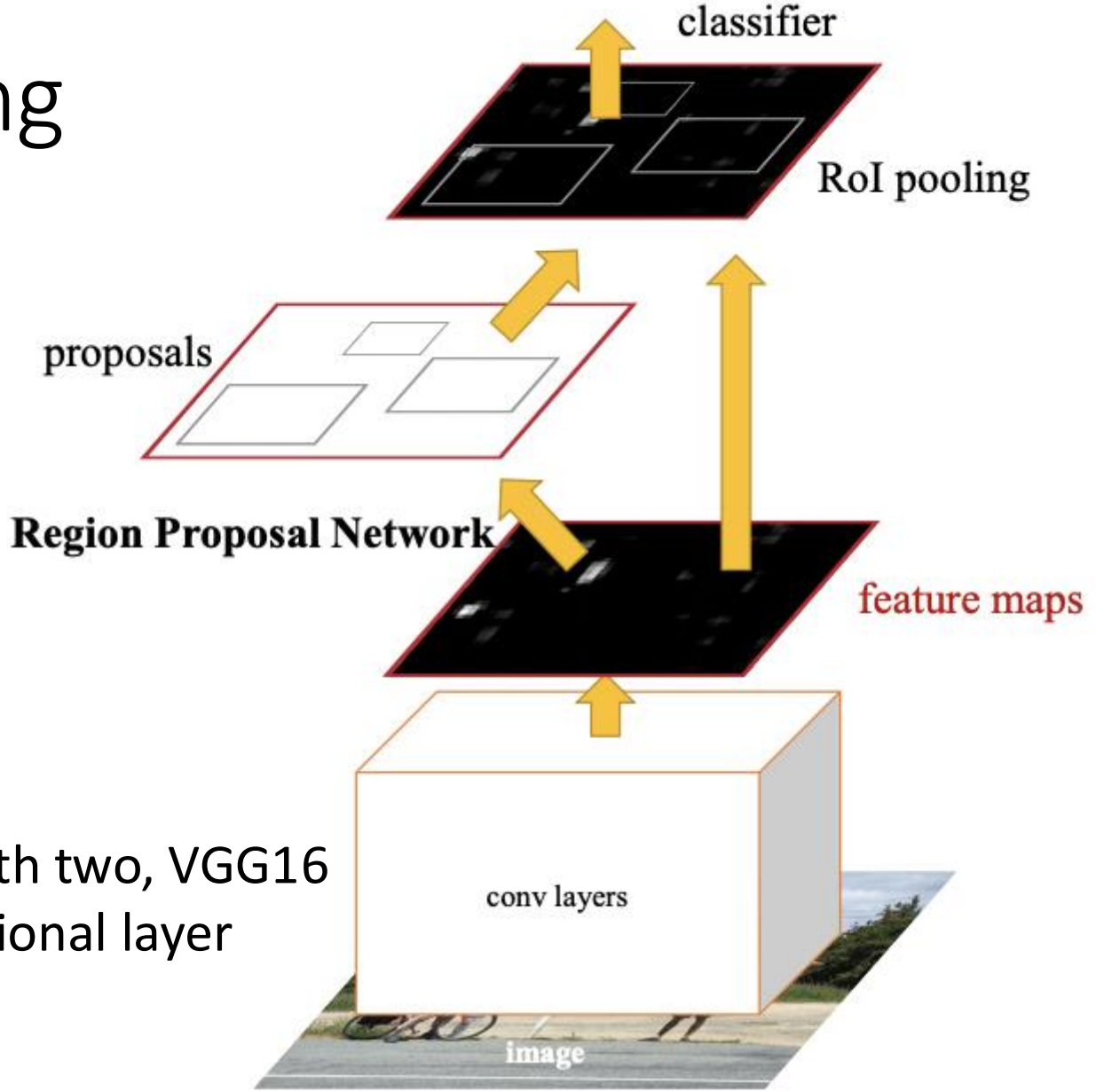
Sum classification and (sometimes) localization losses for each region proposal



Key Ideas

- Enable dense output: multi-task learning
- Support any input image dimension: ROI pooling
- Generate high-quality candidate regions: region proposals
- Make learning feasible: use supervised pre-training

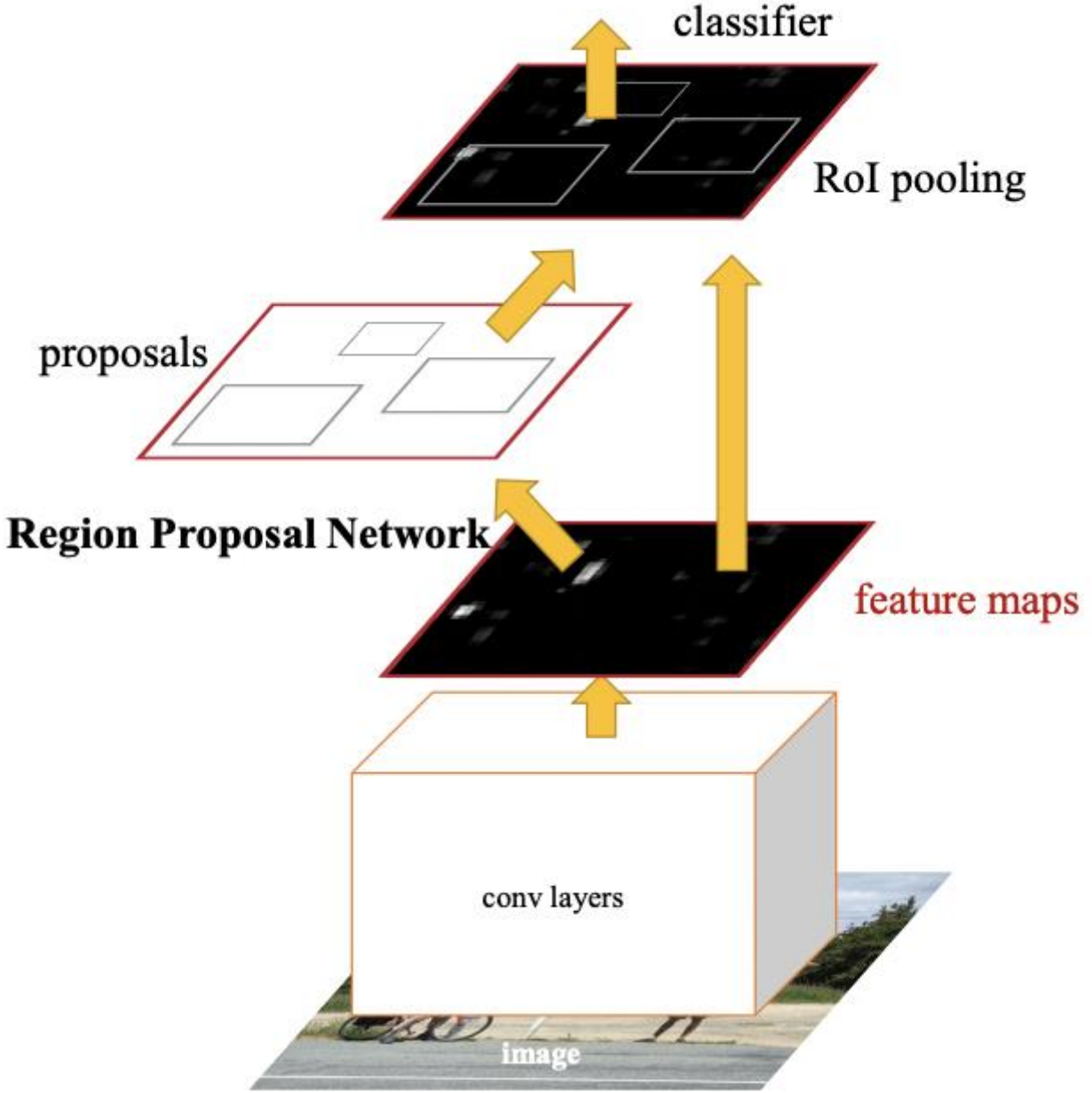
Supervised Pre-Training



ImageNet trained architectures: tested with two, VGG16 and ZF model, using input to last convolutional layer

Ren et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Neurips 2015.

Training Faster R-CNN



1. Train RPN (fine-tuning conv layers)
2. Train Faster R-CNN using proposals from pretrained RPN
3. Fine-tune layers unique to RPN
4. Fine-tune the fully connected layers of Faster R-CNN

Ren et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Neurips 2015.

Recap: Tricks for Object Detection

- Enable dense output: multi-task learning
- Support any input image dimension: ROI pooling
- Generate high-quality candidate regions: region proposals
- Make learning feasible: use supervised pre-training

Today's Topics

- Motivation: training large capacity, deep models to locate content
- Semantic segmentation: classifying pixels
- Object detection: locating objects with bounding rectangles
- Instance segmentation: demarcating objects with detailed outlines
- Programming tutorial

Why Mask R-CNN?

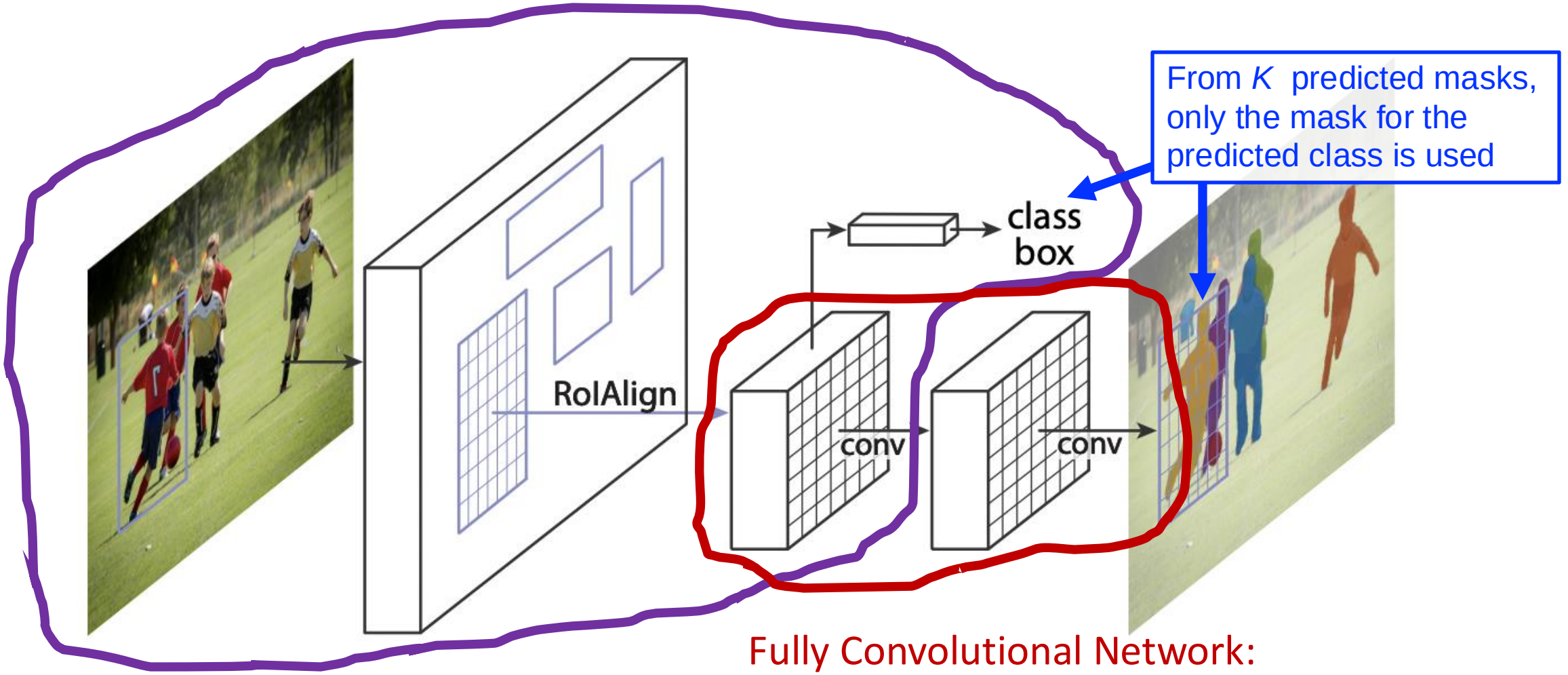
Named after the approach of adapting Faster R-CNN to also predict **masks**:

> Kaiming He, Georgia Gkioxari, Piotr Dollar, & Ross Girshick. “Mask R-CNN.” ICCV 2017.

Key contributions of the method:

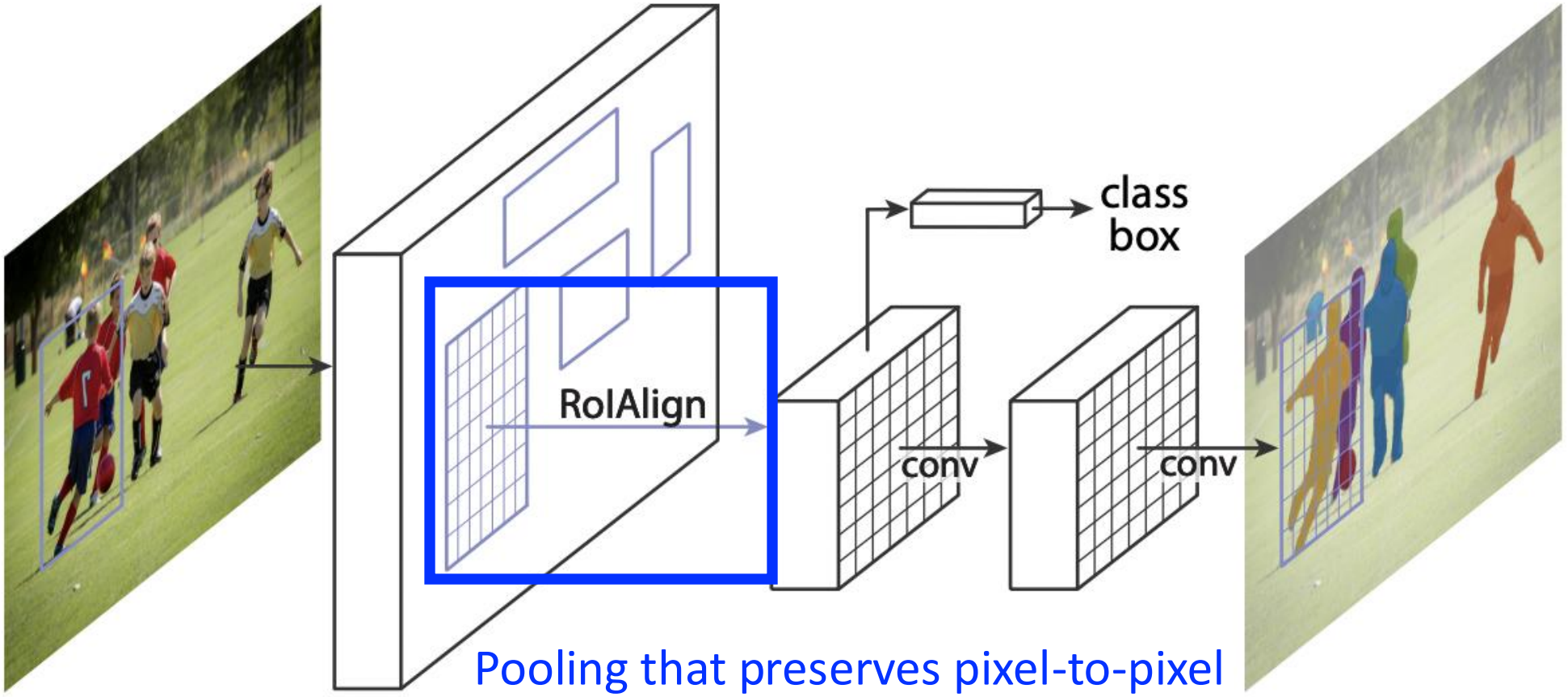
1. Pooling method that preserves the pixel-to-pixel alignment between the model’s input and output when downsampling
2. State-of-the-art performance

Architecture: Extends Faster R-CNN by Also Predicting in Parallel a Mask Per Region



He et al. Mask R-CNN. ICCV 2017

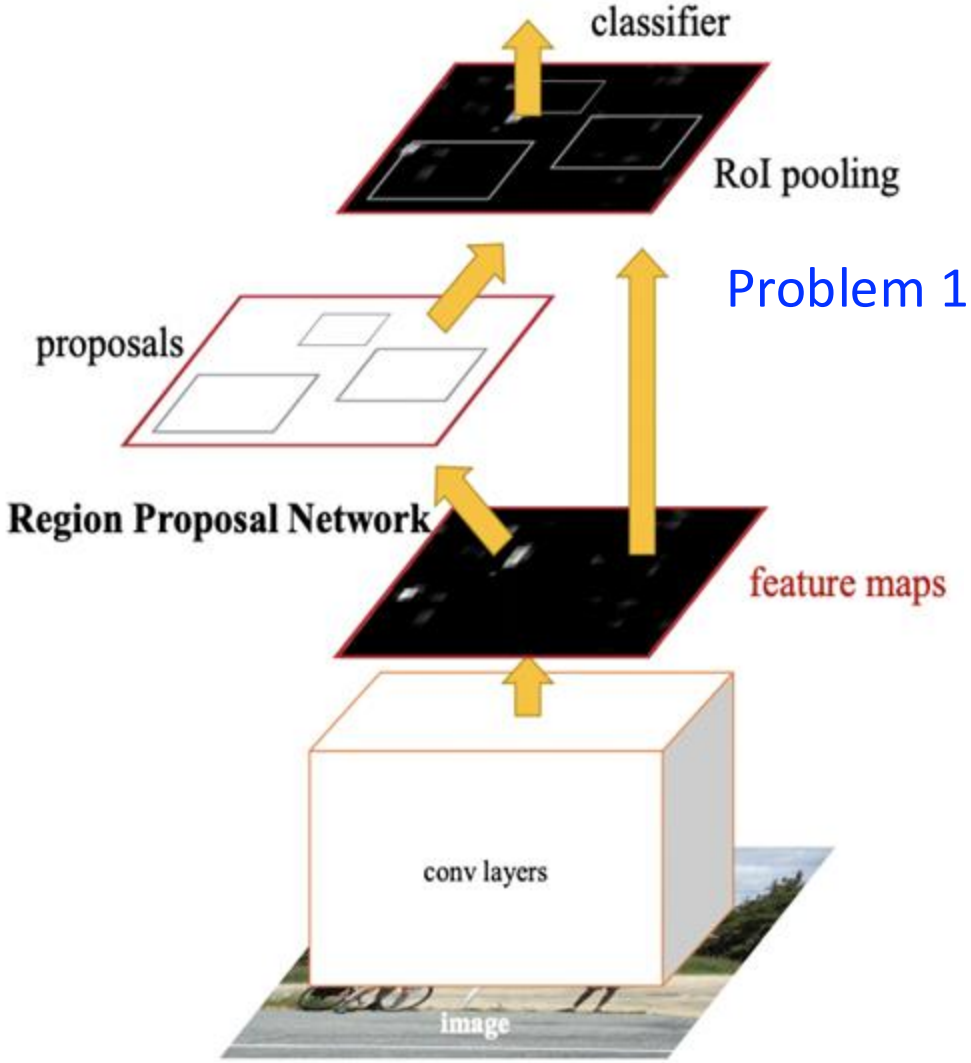
Architecture: Key Idea



Pooling that preserves pixel-to-pixel alignment between model's input and output

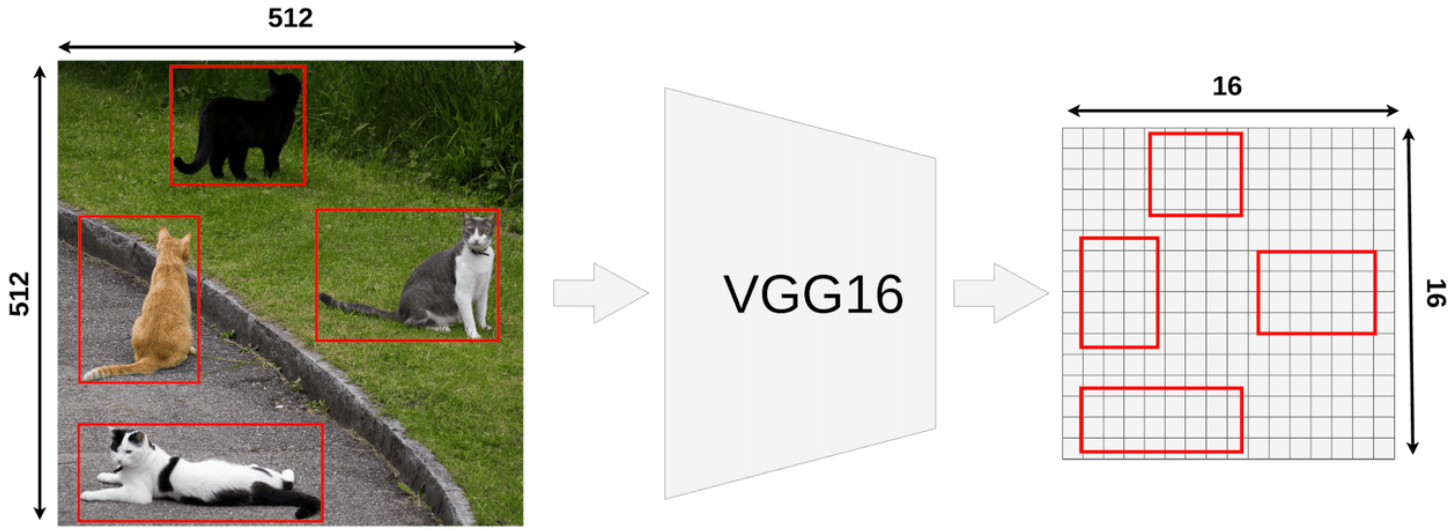
He et al. Mask R-CNN. ICCV 2017

ROIAlign Motivation: Revisiting Faster R-CNN



Problem 1: quantization of region proposals in a downsized feature map

e.g., 1/32 of the size (512/32 = 16)

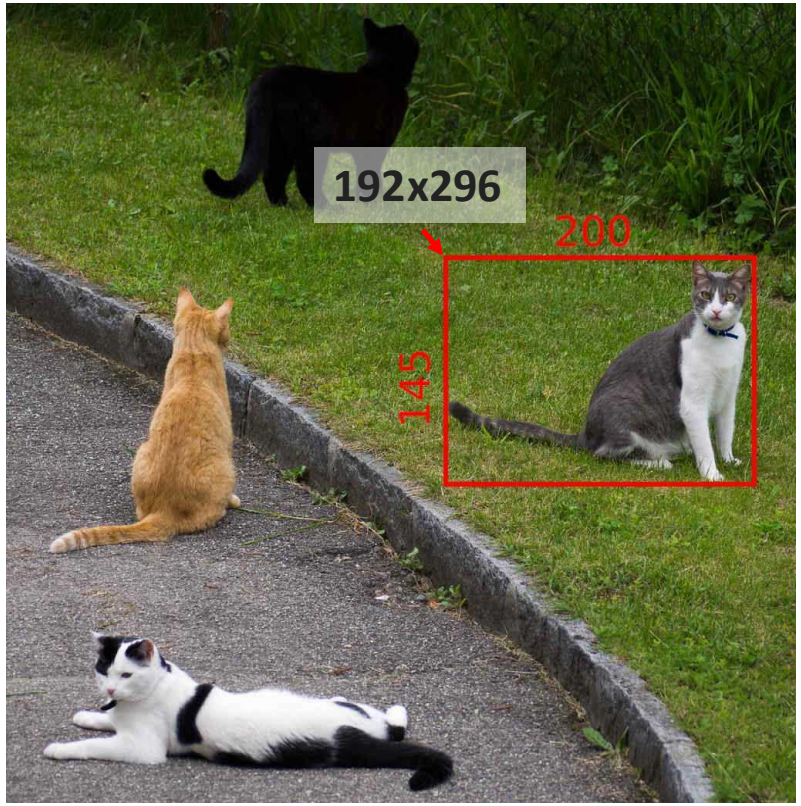


https://erdem.pl/2020/02/understanding-region-of-interest-ro-i-pooling

Ren Shaoqing Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.” Neurips 2015

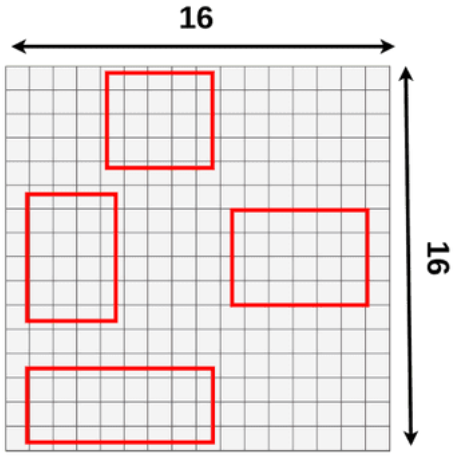
ROIAlign Motivation: Revisiting Faster R-CNN

What are the values for the region in the original image in the downsampled feature map?



VGG16

(1/32 of original size)

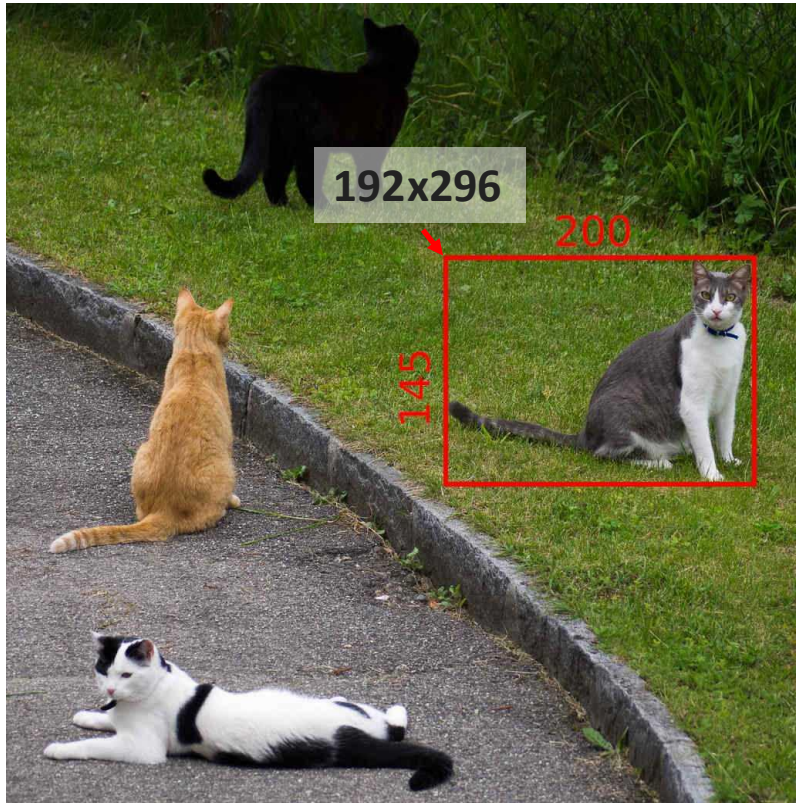


Width: ?
Height: ?
Upper-left X: ?
Upper-left Y: ?

https://erdem.pl/2020/02/understanding-region-of-interest-ro-i-pooling

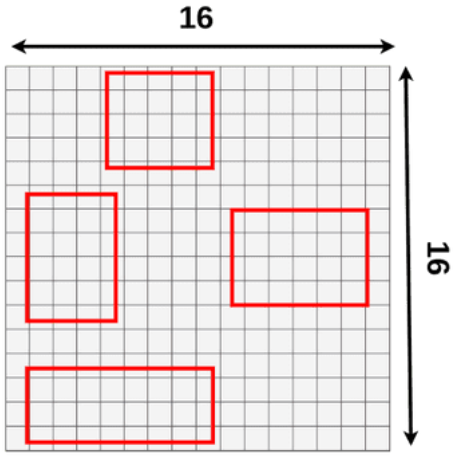
ROIAlign Motivation: Revisiting Faster R-CNN

What are the values for the region in the original image in the downsampled feature map?



VGG16

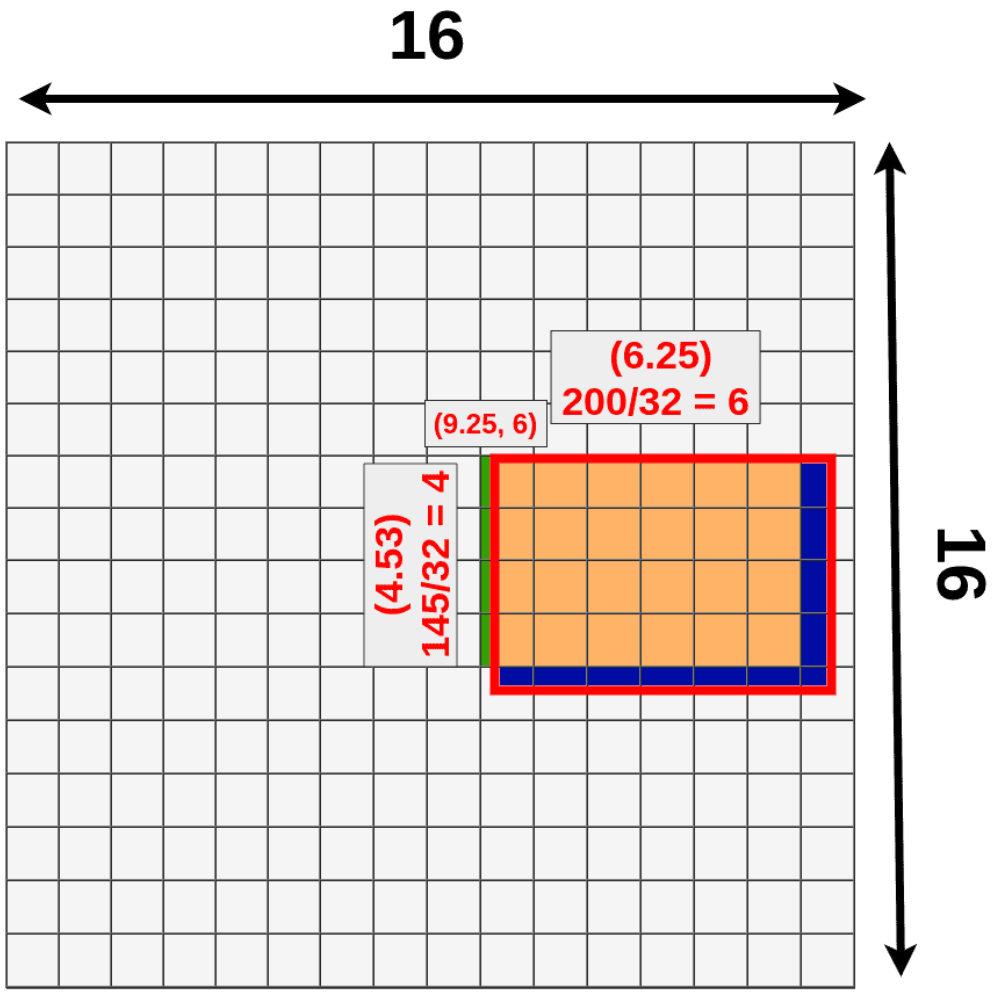
(1/32 of original size)



Width: 200/32 = 6.25
Height: 145/32 = ~4.53
Upper-left X: 192/32 = 9.25
Upper-left Y: 145/32 = 6

https://erdem.pl/2020/02/understanding-region-of-interest-ro-i-pooling

ROIAlign Motivation: Revisiting Faster R-CNN



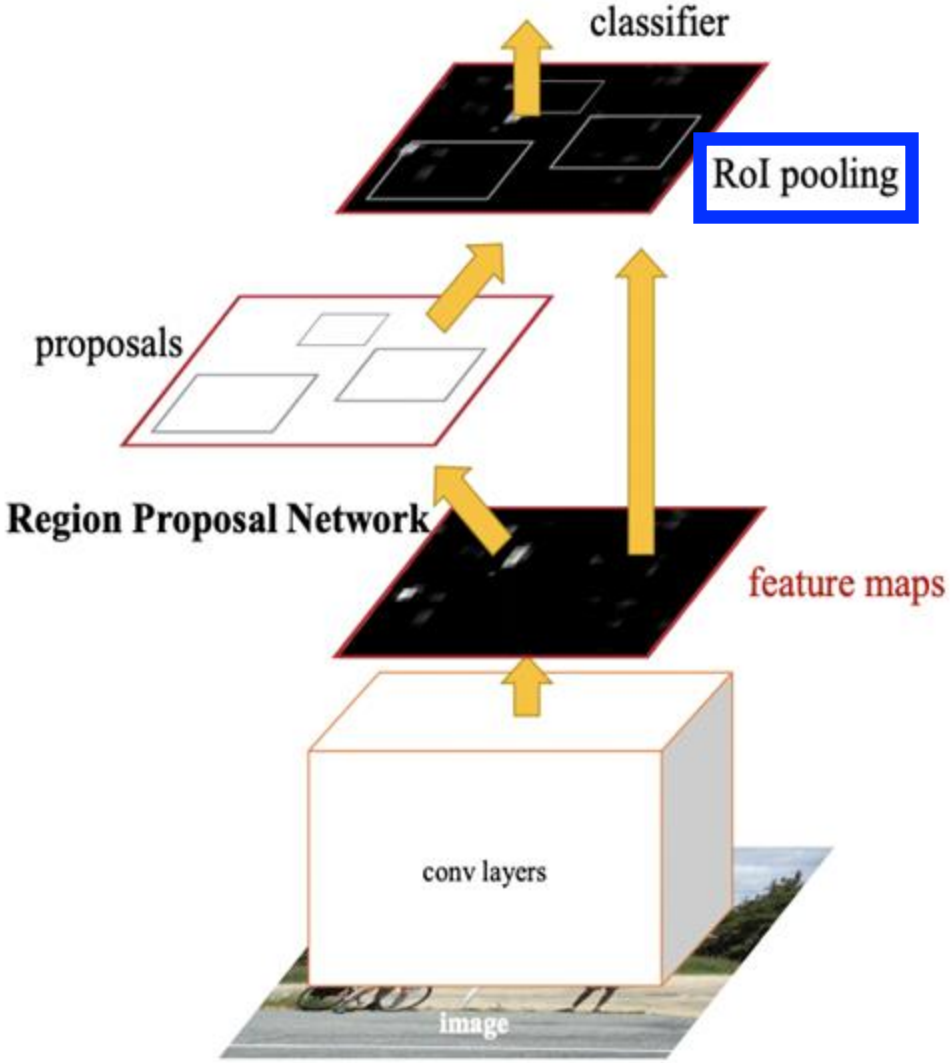
Original region on feature map

Quantized variant: values rounded down to only include a discrete set of integers to match the grid
- Original information preserved
- Information added
- Information lost

Quantization changes the information utilized from the original image, losing information about the object and adding extra image context (recall, the original image is orders of magnitude larger than the feature map!)

https://erdem.pl/2020/02/understanding-region-of-interest-ro-i-pooling

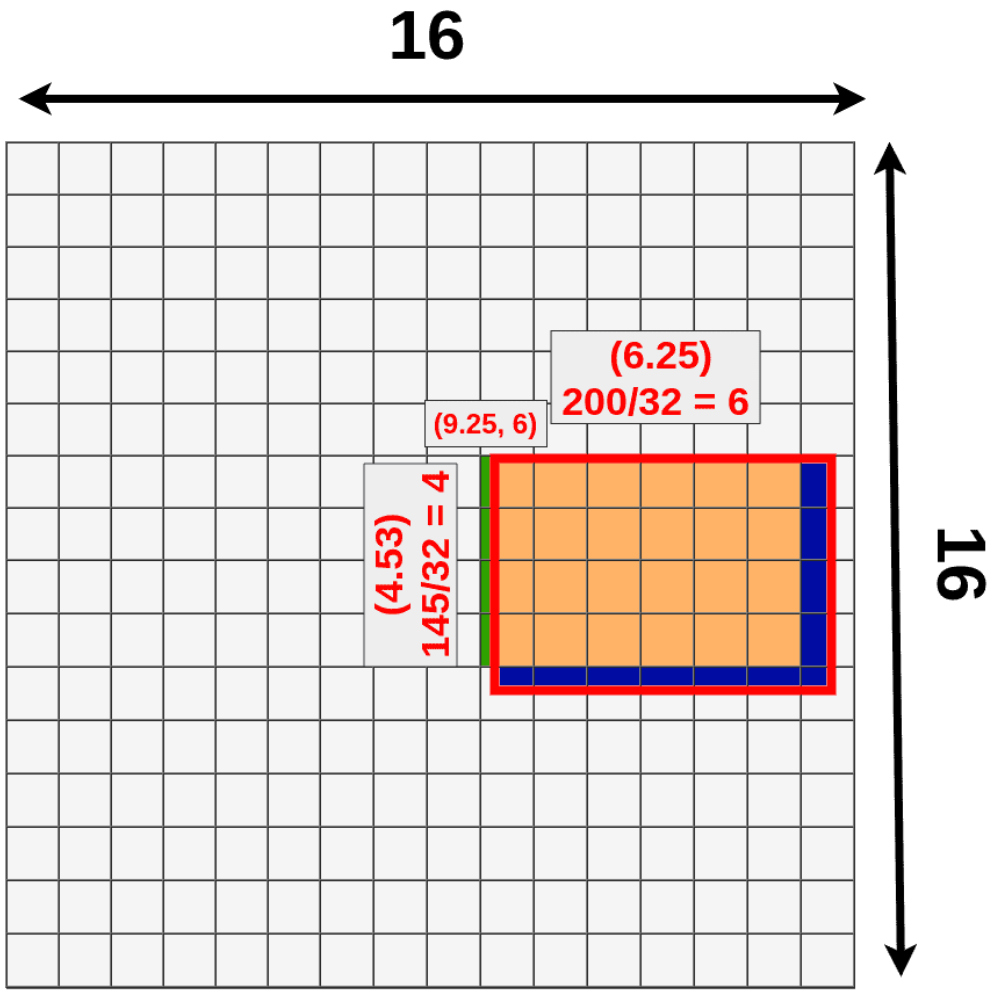
ROIAlign Motivation: Revisiting Faster R-CNN



Problem 2: Quantization when pooling region proposals of various sizes to the fixed size required by the fully connected layer

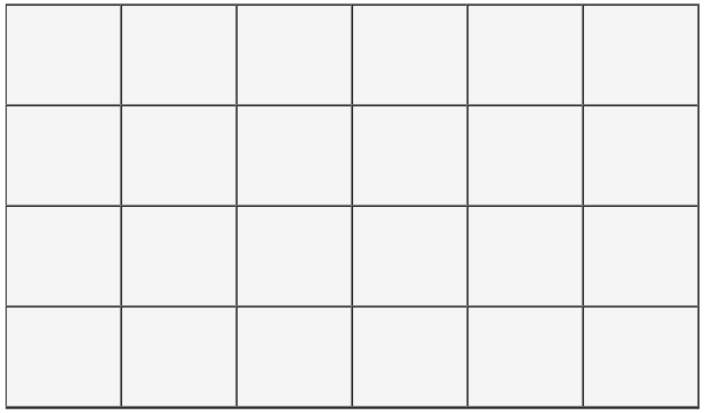
Ren Shaoqing Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.” Neurips 2015

ROIAlign Motivation: Revisiting Faster R-CNN

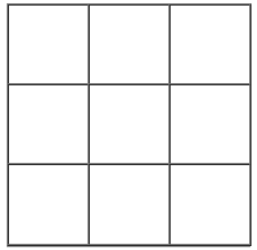


e.g., convert quantized 4x6 region into a 3x3 feature

4x6 RoI



3x3 RoI Pooling

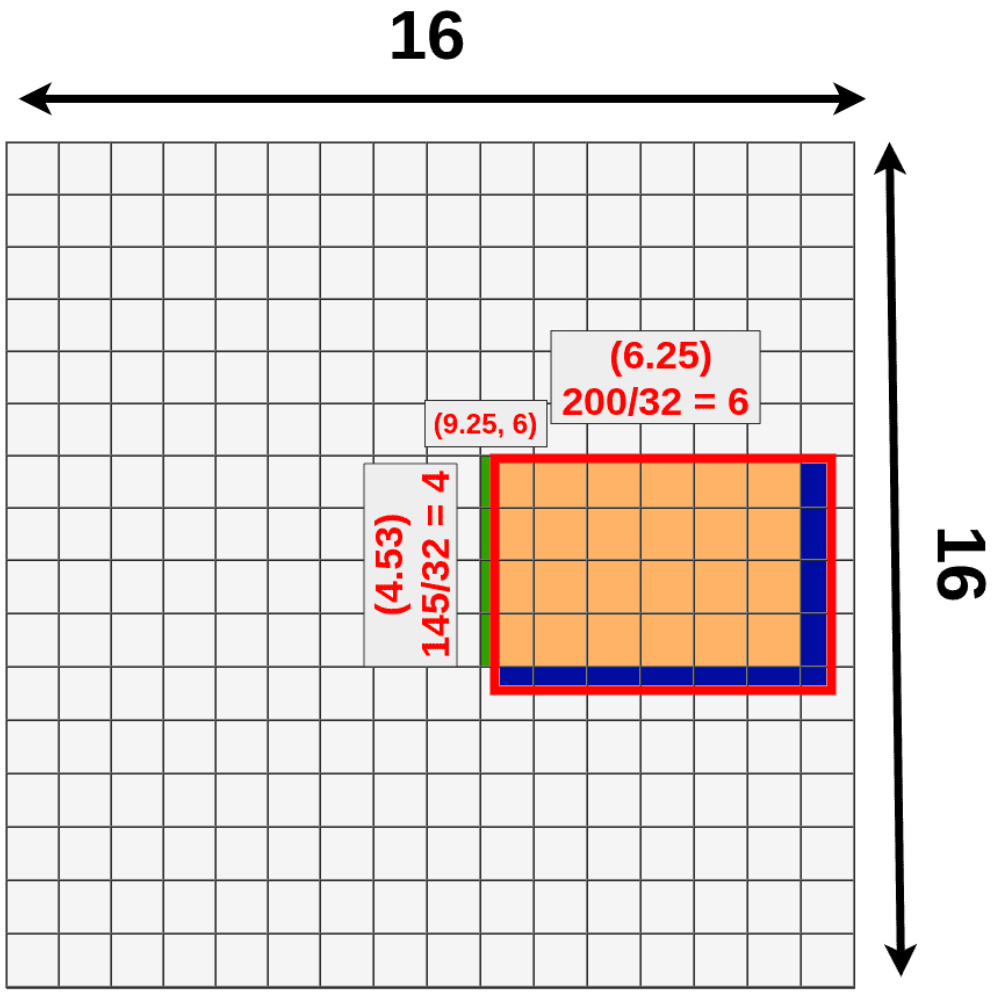


Quantized approach: identify discrete integers for pooling to result in the target size

e.g., 4/3 = 1.3 -> 1 and 6/3 = 2

https://erdem.pl/2020/02/understanding-region-of-interest-ro-i-pooling

ROIAlign Motivation: Revisiting Faster R-CNN

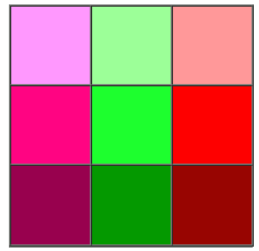


e.g., convert quantized 4x6 region into a 3x3 feature

4x6 RoI

0.1	0.2	0.3	0.4	0.5	0.6
1	0.7	0.2	0.6	0.1	0.9
0.9	0.8	0.7	0.3	0.5	0.2

3x3 RoI Pooling

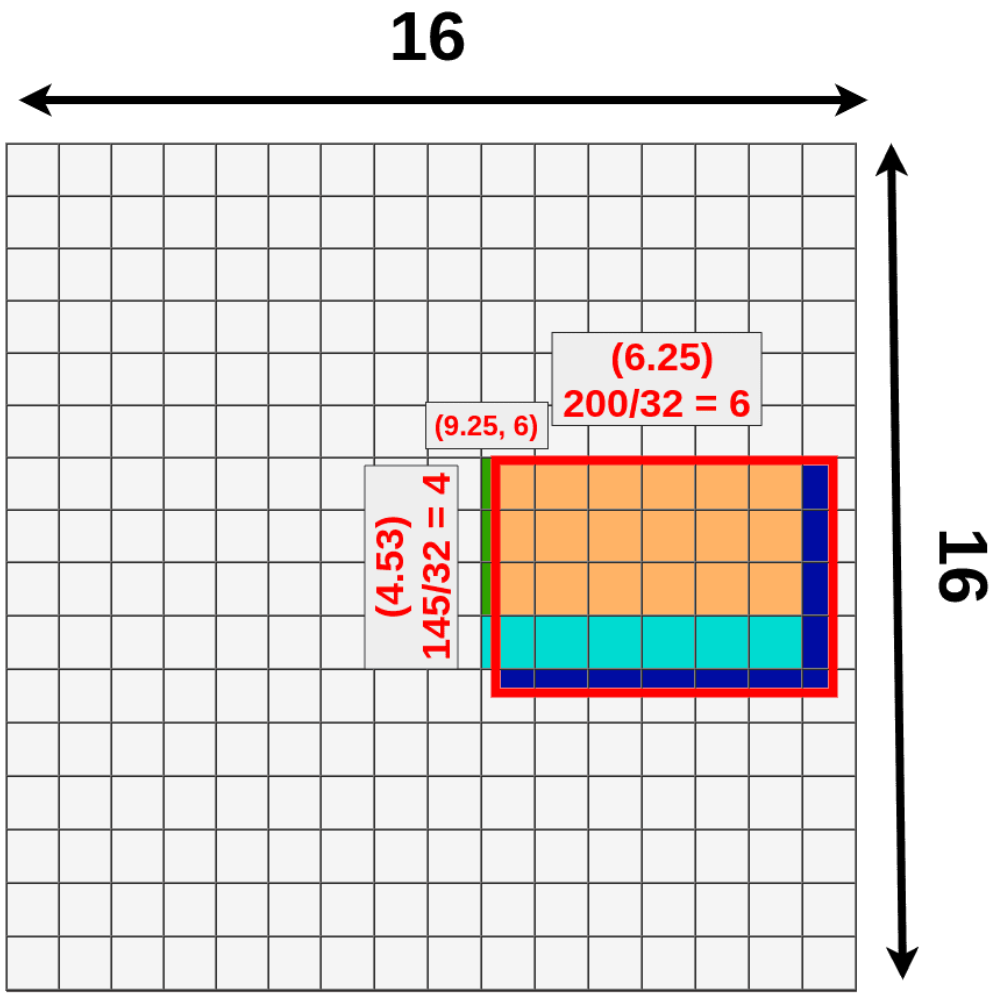


Quantized approach: identify discrete integers for pooling to result in the target size

e.g., 1x2 vector using max pooling

https://erdem.pl/2020/02/understanding-region-of-interest-ro-i-pooling

ROIAlign Motivation: Revisiting Faster R-CNN



e.g., convert quantized 4x6 region into a 3x3 feature

4x6 RoI

0.1	0.2	0.3	0.4	0.5	0.6
1	0.7	0.2	0.6	0.1	0.9
0.9	0.8	0.7	0.3	0.5	0.2
0.2	0.5	1	0.7	0.1	0.1

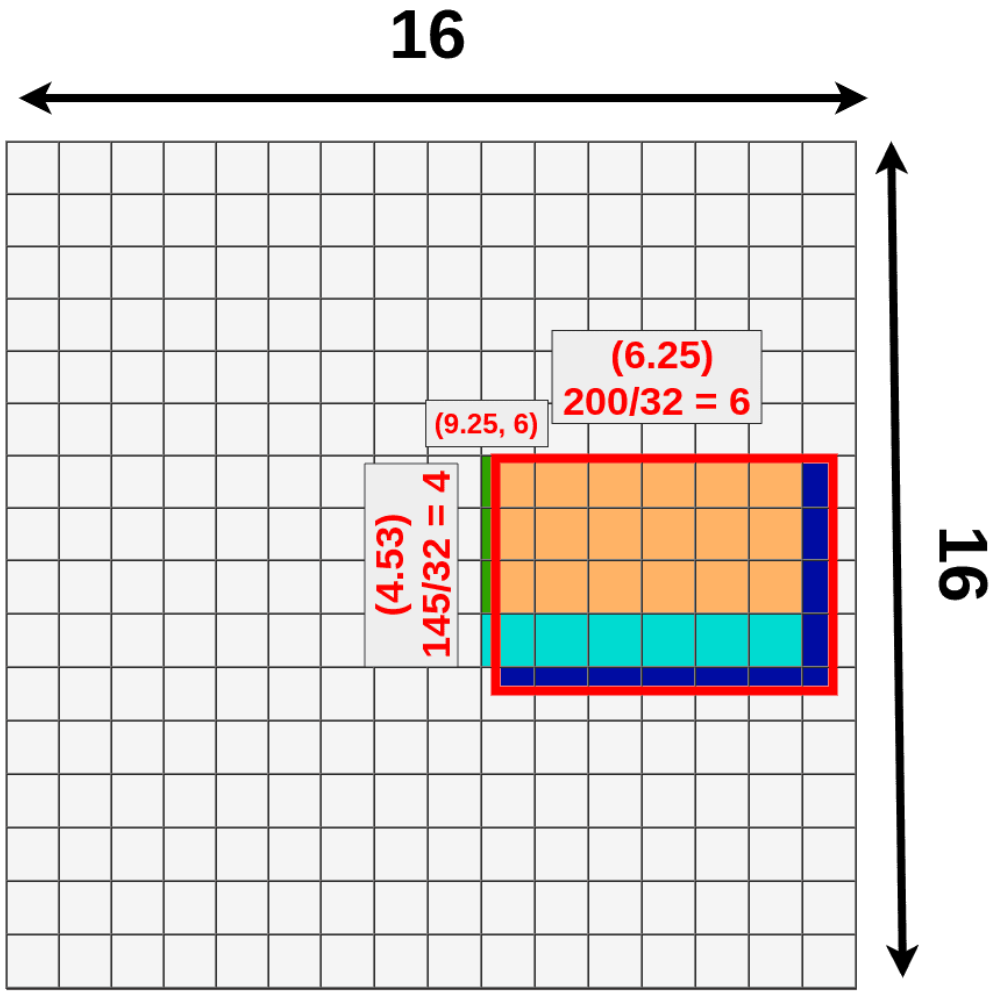
Again, quantization discards information about the object from the original image (recall, the original image is orders of magnitude larger than the feature map!)

Quantized approach: identify discrete integers for pooling to result in the target size

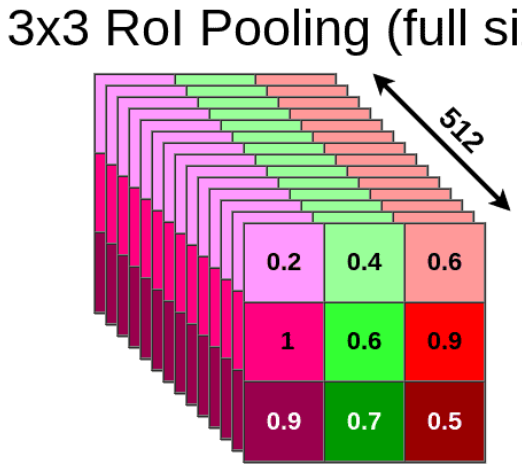
e.g., 1x2 vector using max pooling

https://erdem.pl/2020/02/understanding-region-of-interest-ro-i-pooling

ROIAlign Motivation: Revisiting Faster R-CNN



e.g., convert quantized 4x6 region into a 3x3 feature



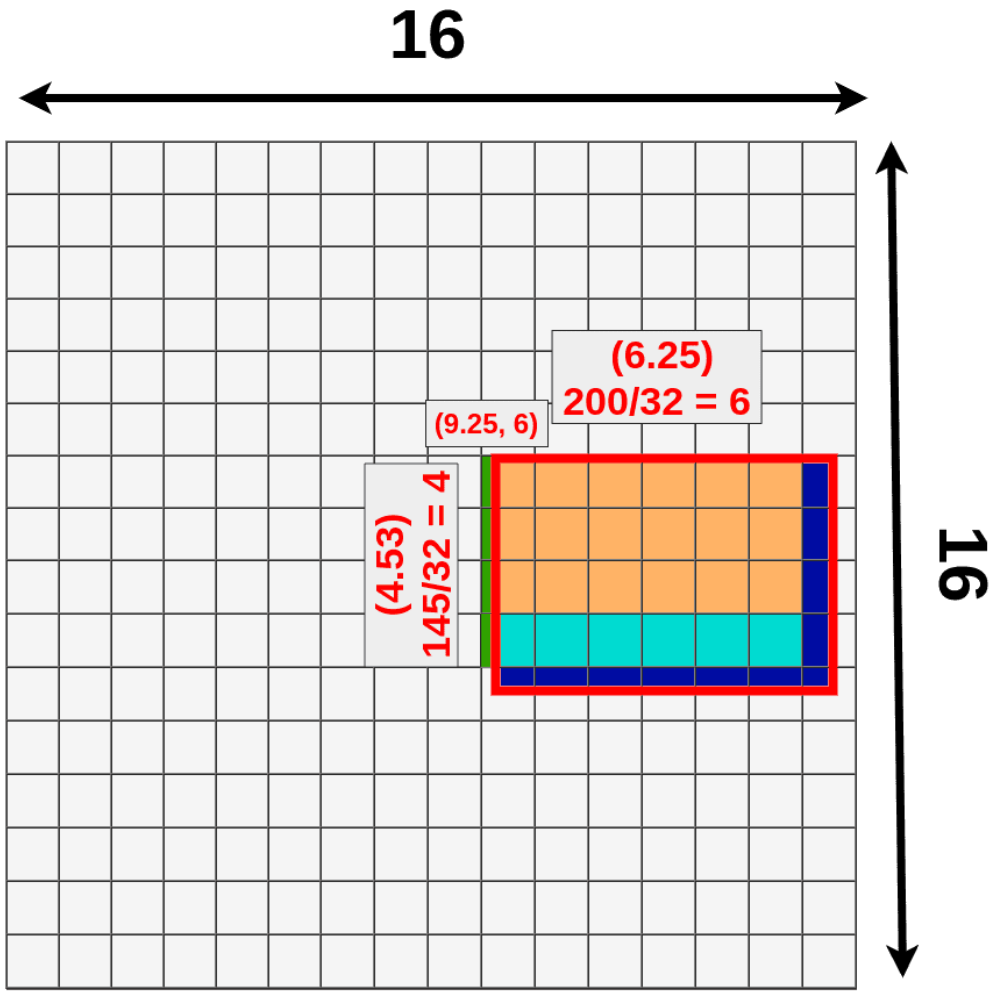
Information is lost for *all* channels for *every* region proposal (each of which is used to predict a class and bounding box)!

Quantized approach: identify discrete integers for pooling to result in the target size

e.g., 1x2 vector using max pooling

https://erdem.pl/2020/02/understanding-region-of-interest-ro-i-pooling

ROIAlign Motivation: Summary

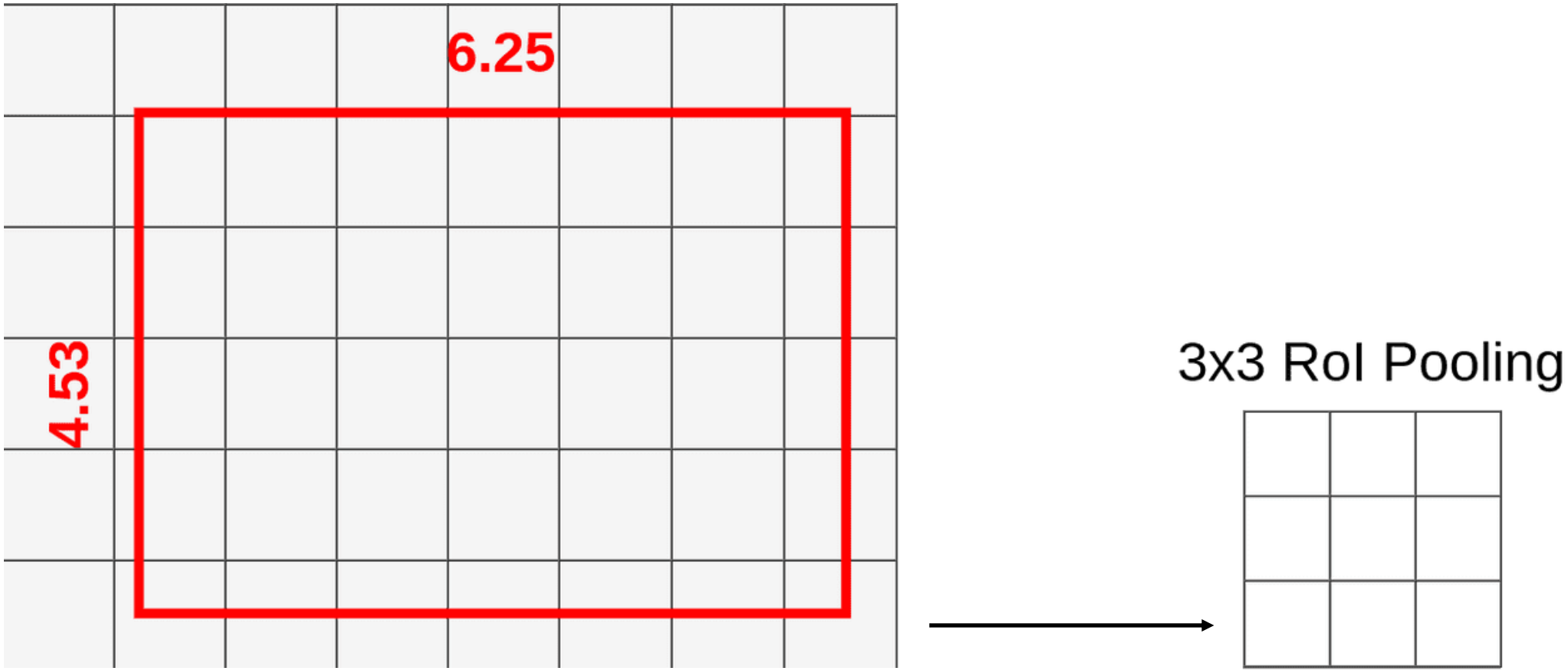


Original region on feature map

Quantization changes the information utilized from the original image, losing information about the object and adding extra image context (recall, the original image is orders of magnitude larger than the feature map!)

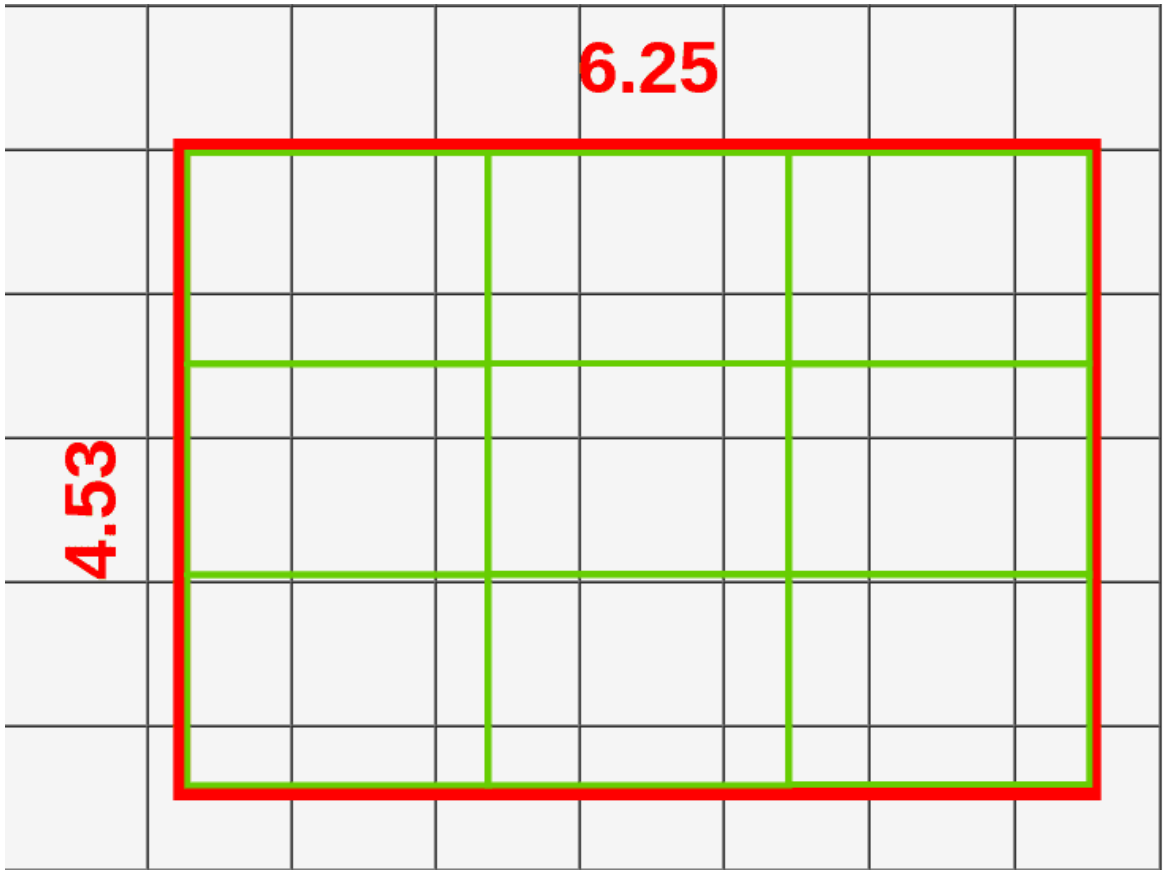
https://erdem.pl/2020/02/understanding-region-of-interest-ro-i-pooling

ROIAlign: Pooling ***Without*** Quantization



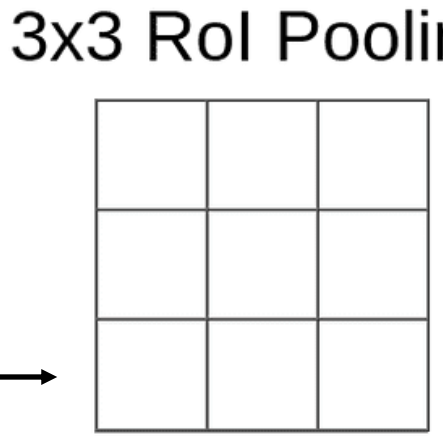
https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

ROIAlign: Pooling ***Without*** Quantization



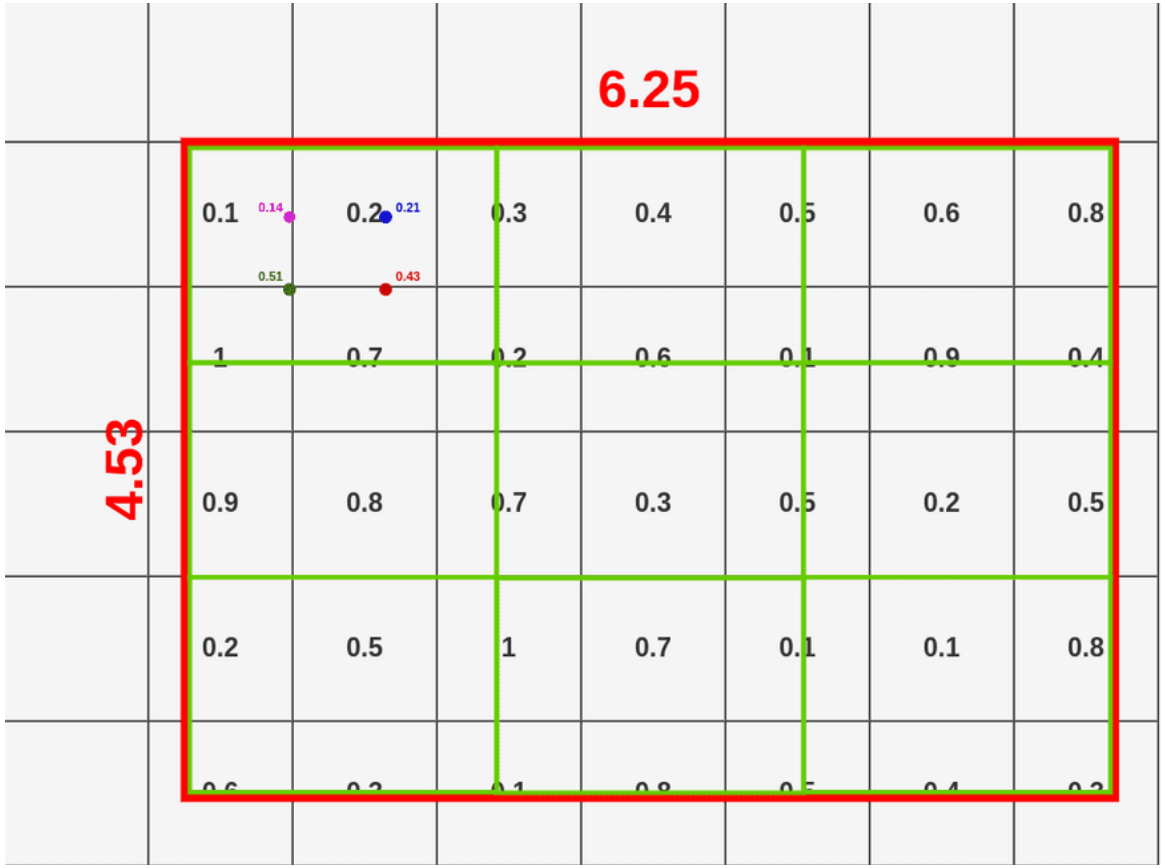
Divide region into 9 equal sized boxes; what is the size of each box?

- 6.25/3 x 4.53/3 = 2.08 x 1.51



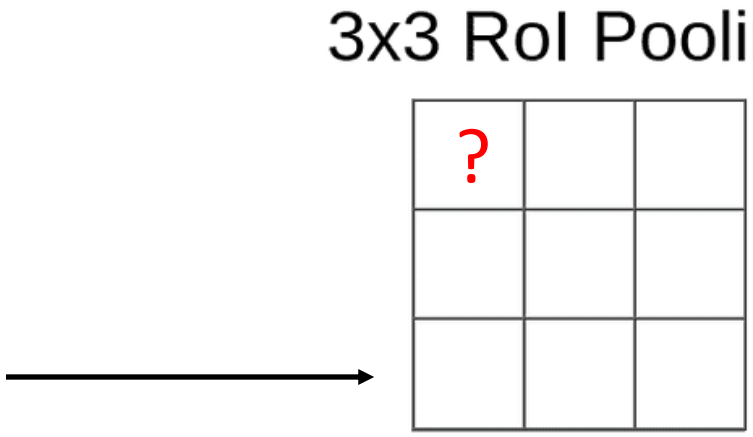
https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

ROIAlign: Pooling ***Without*** Quantization



Perform pooling on sampled values in each box
- e.g., max(0.14, 0.21, 0.51, 0.43) = ?

How do we find the four sample values?



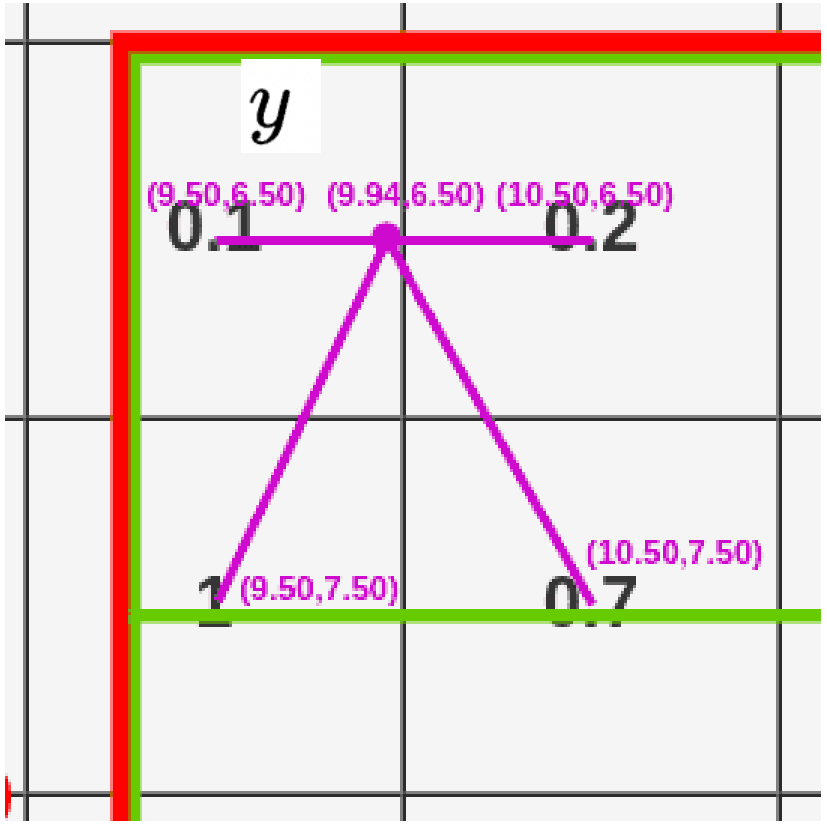
https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

ROIAlign: Pooling ***Without*** Quantization

Compute each sample value with interpolation between 4 points

https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

ROIAlign: Pooling ***Without*** Quantization



Compute each sample value with interpolation between 4 points:

1. Identify sample location

$$y \begin{cases} \text{X = X\_box + (width/3) * 1 = 9.25 + (2.08/3) = 9.94} \\ \text{Y = Y\_box + (height/3) * 1 = 6 + (1.51/3) = 6.50} \end{cases}$$

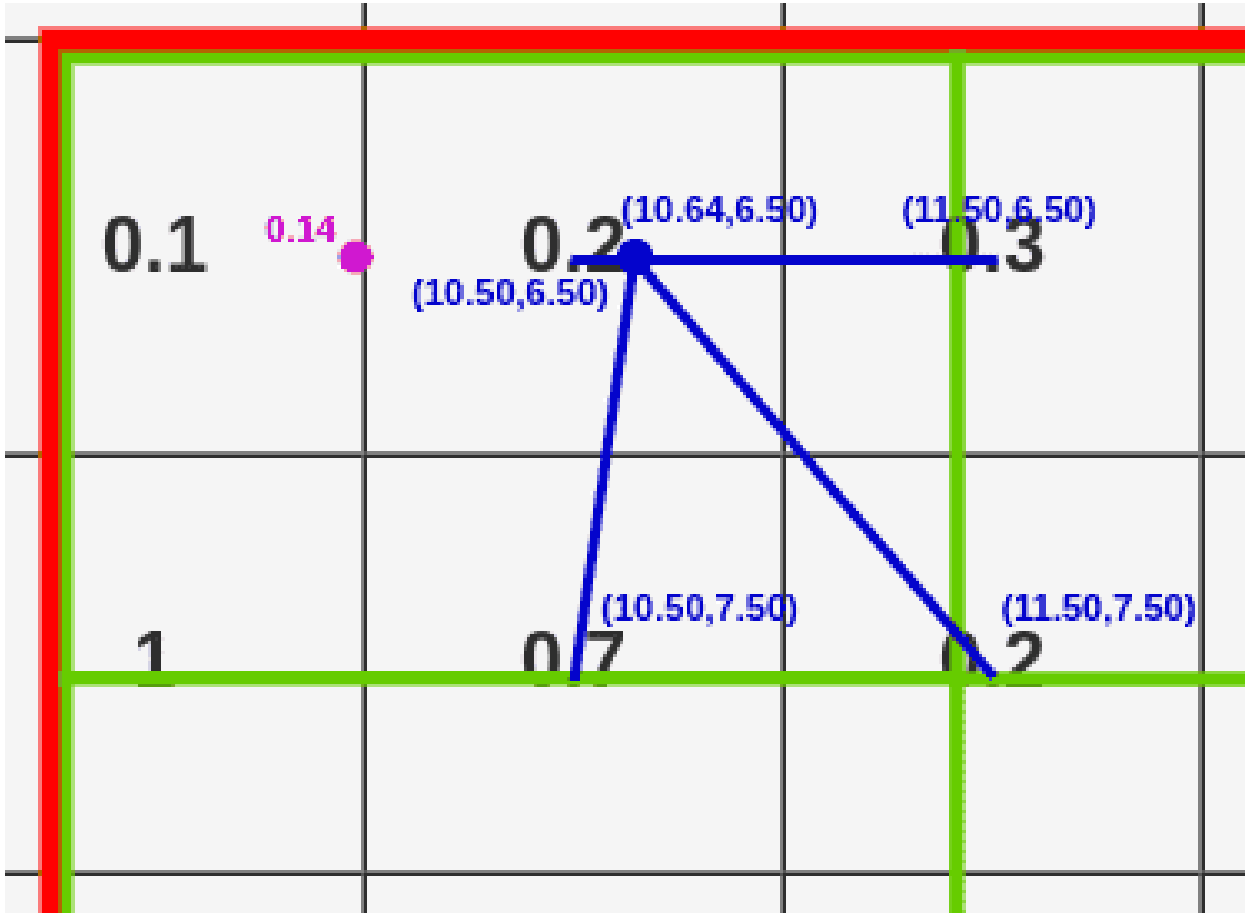
2. Identify 4 points for interpolation, using the middle of each closest neighboring box in each direction
3. Calculate value using bilinear interpolation (= 0.14)

$$P \approx \frac{y_2 - y}{y_2 - y_1}\left(\frac{x_2 - x}{x_2 - x_1}Q_{11} + \frac{x - x_1}{x_2 - x_1}Q_{21}\right) + \frac{y - y_1}{y_2 - y_1}\left(\frac{x_2 - x}{x_2 - x_1}Q_{12} + \frac{x - x_1}{x_2 - x_1}Q_{22}\right)$$

$$\approx \frac{7.5 - 6.5}{7.5 - 6.5}\left(\frac{10.5 - 9.94}{10.5 - 9.5}0.1 + \frac{9.94 - 9.5}{10.5 - 9.5}0.2\right) + \frac{6.5 - 6.5}{7.5 - 6.5}\left(\frac{10.5 - 9.94}{10.5 - 9.5}1 + \frac{9.94 - 9.5}{10.5 - 9.5}0.7\right)$$

https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

ROIAlign: Pooling ***Without*** Quantization

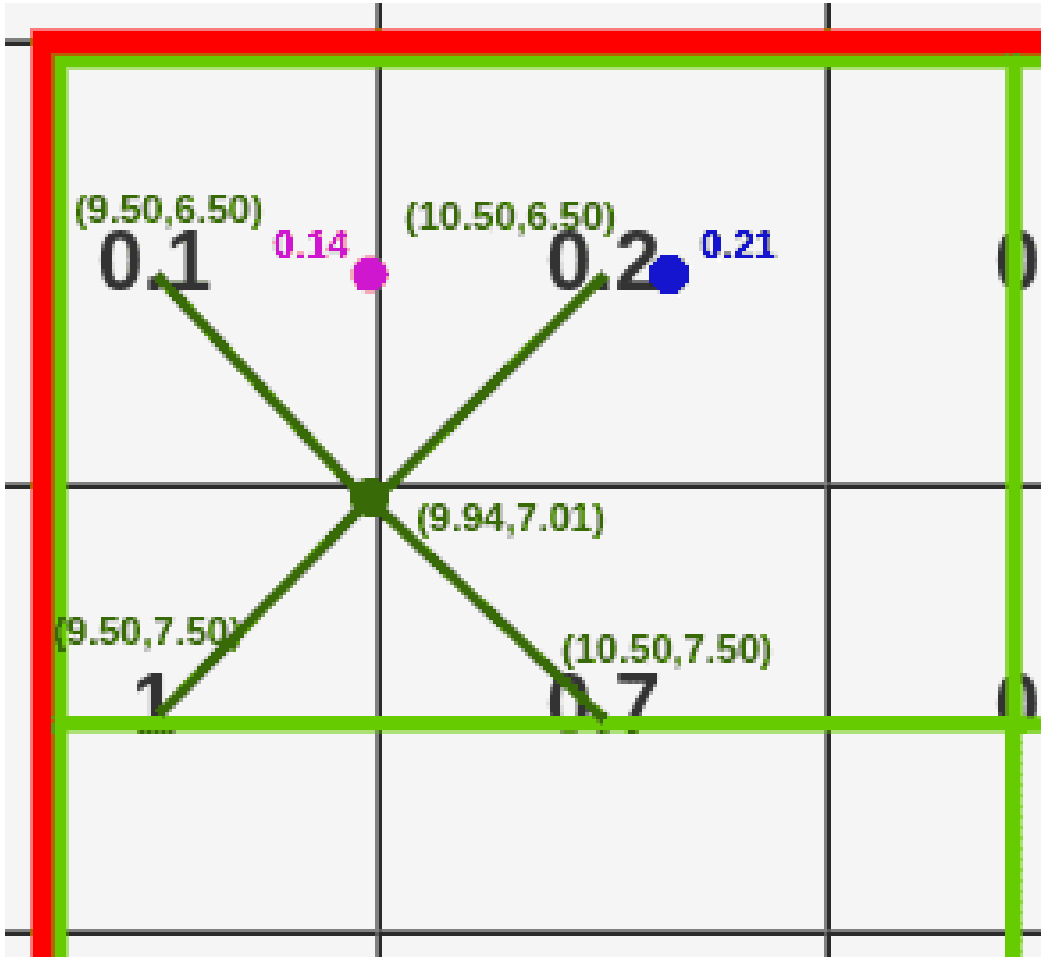


Compute each sample value with interpolation between 4 points:

1. Identify sample location
2. Identify 4 points for interpolation, using the middle of each closest neighboring box in each direction
3. Calculate value using bilinear interpolation (=0.21)

https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

ROIAlign: Pooling ***Without*** Quantization

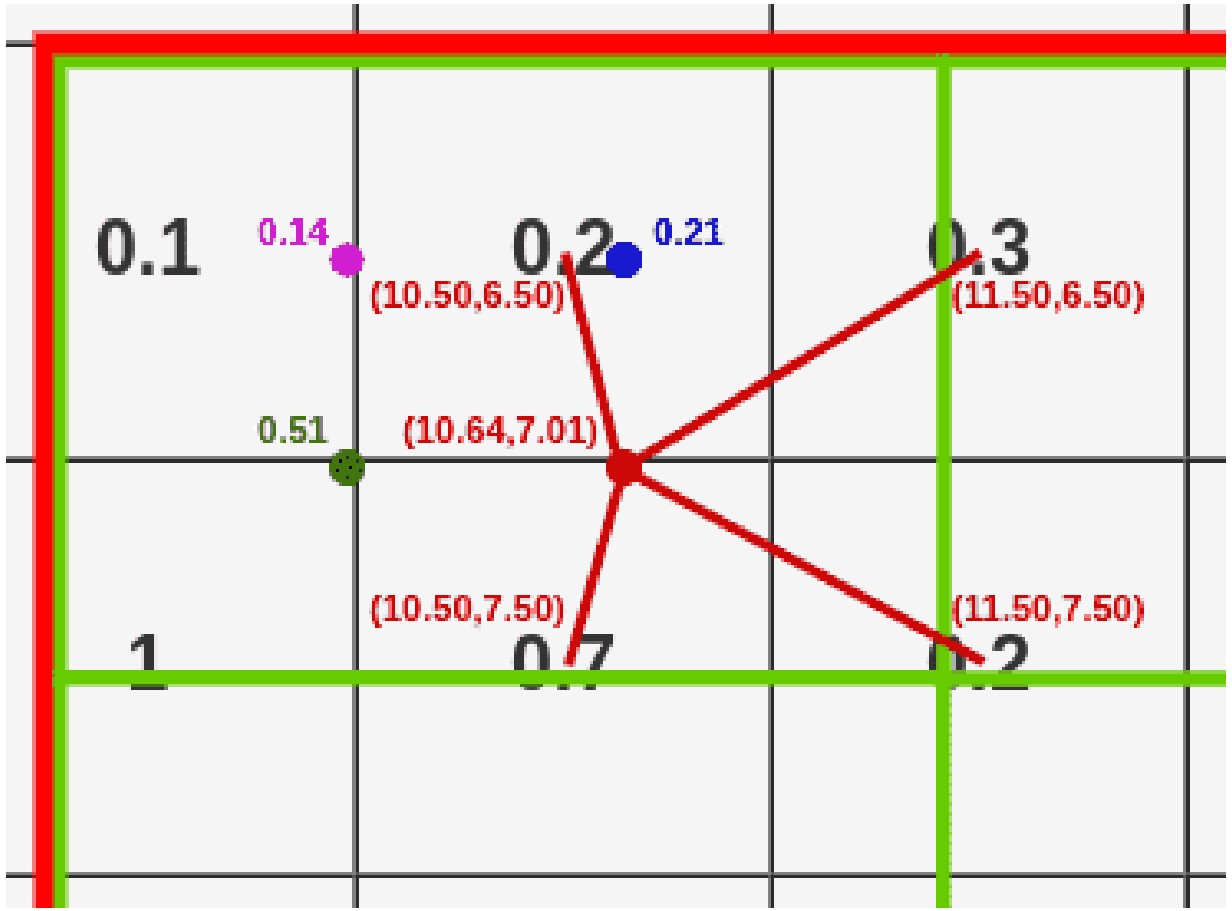


Compute each sample value with interpolation between 4 points:

1. Identify sample location
2. Identify 4 points for interpolation, using the middle of each closest neighboring box in each direction
3. Calculate value using bilinear interpolation (=0.51)

https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

ROIAlign: Pooling ***Without*** Quantization

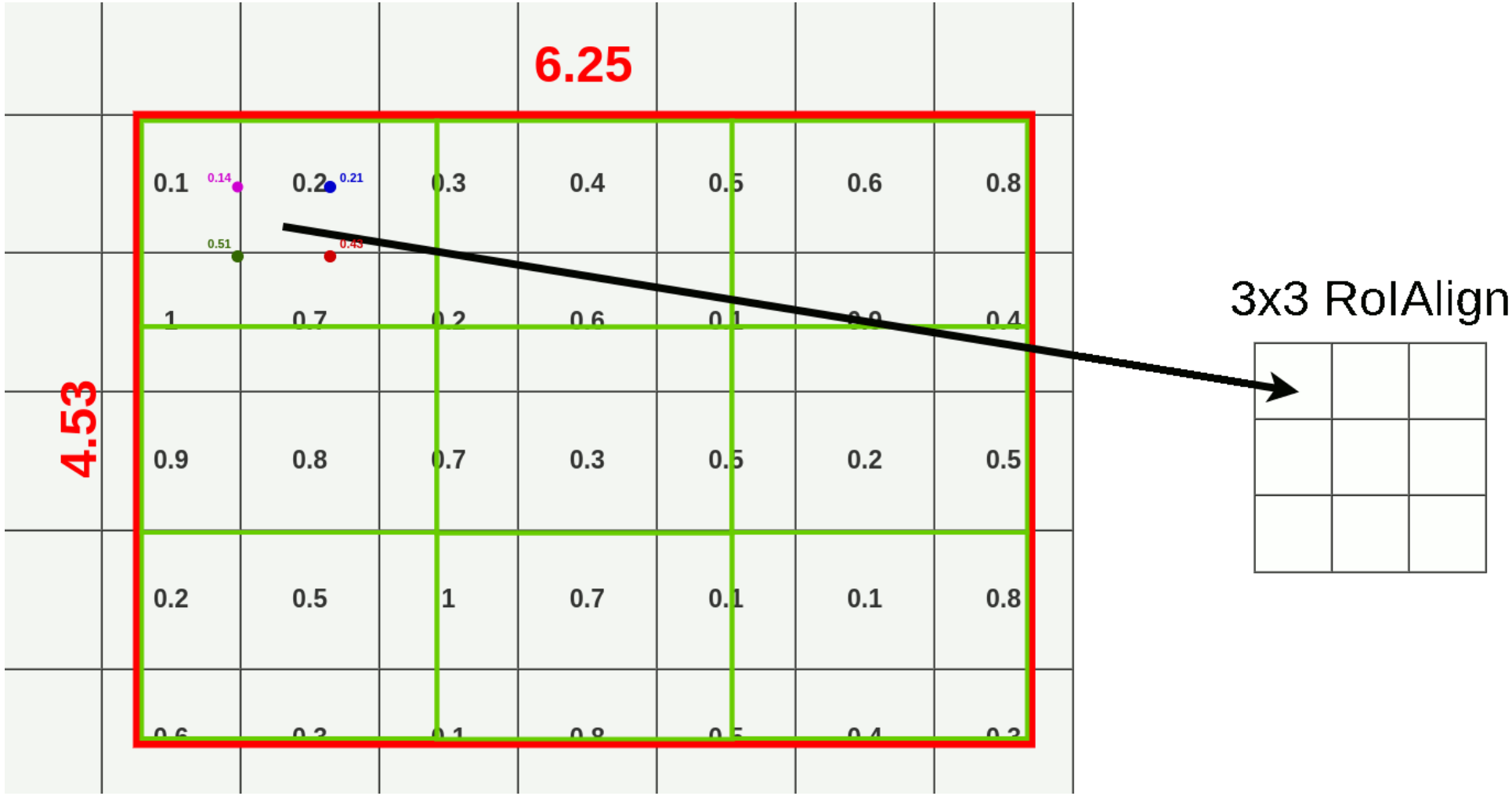


Compute each sample value with interpolation between 4 points:

1. Identify sample location
2. Identify 4 points for interpolation, using the middle of each closest neighboring box in each direction
3. Calculate value using bilinear interpolation (=0.43)

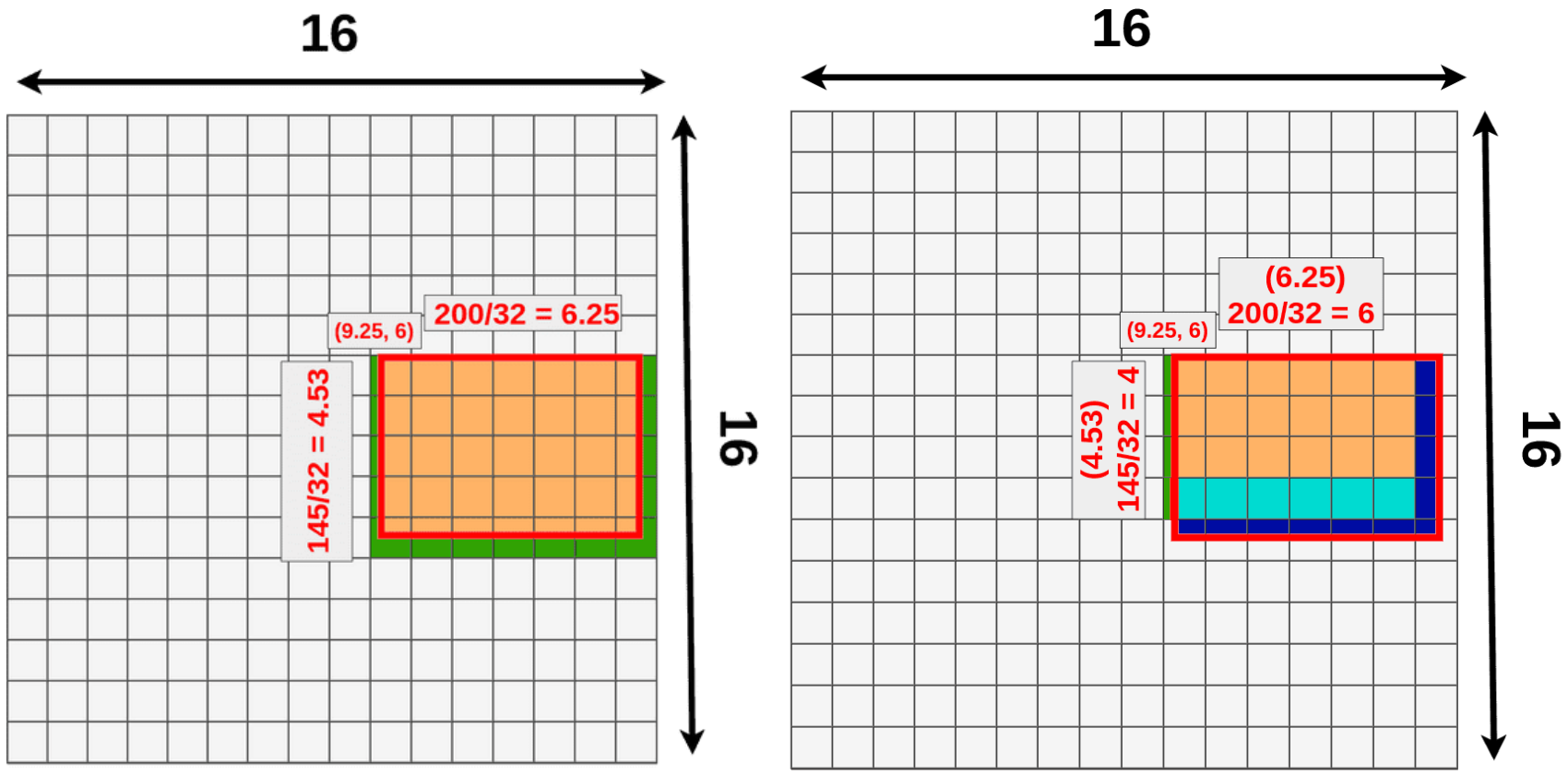
https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

ROIAlign: Pooling ***Without*** Quantization



https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

ROIAlign vs ROI Pooling



Original region on feature map

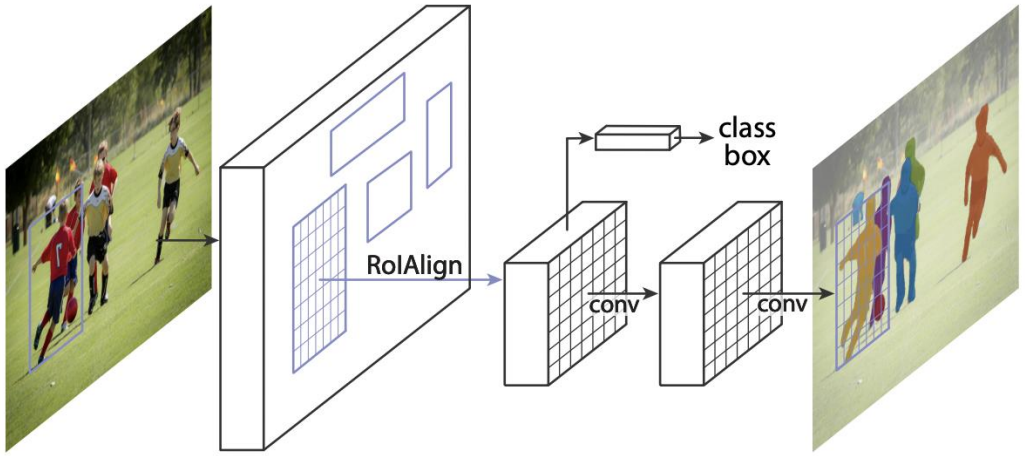
Both methods add extra image context

Only ROI pooling loses information about the object from the original image

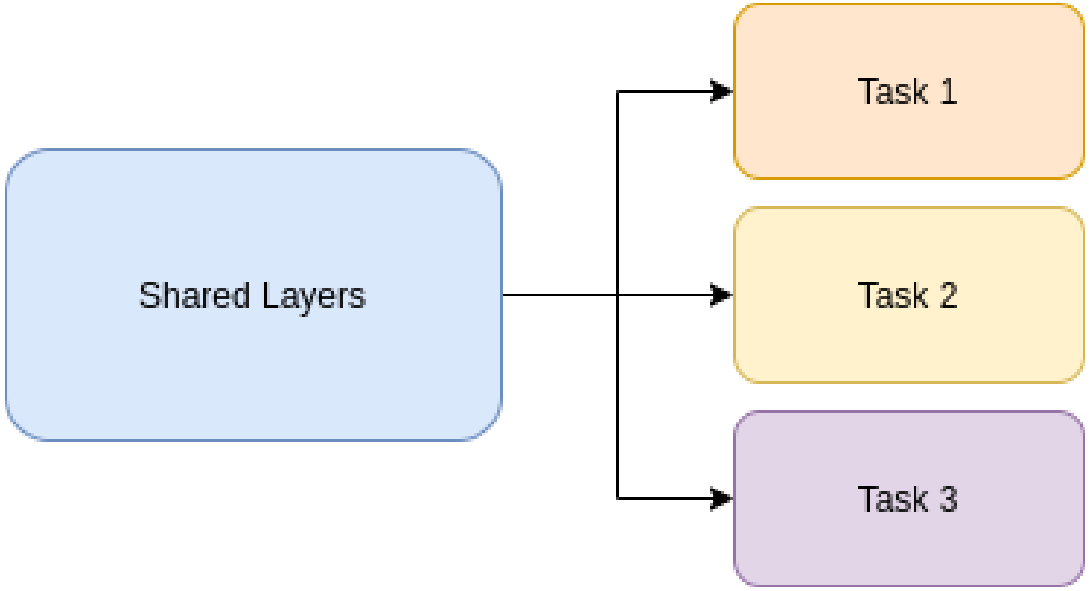
https://erdem.pl/2020/02/understanding-region-of-interest-part-2-ro-i-align

Training: Multi-Task Learning

What are the three tasks (and so types of losses) used during training?



He et al. Mask R-CNN. ICCV 2017



https://towardsdatascience.com/multi-task-learning-with-pytorch-and-fastai-6d10dc7ce855

$$L = L_{class} + L_{box} + L_{mask}$$

Today's Topics

- Motivation: training large capacity, deep models to locate content
- Semantic segmentation: classifying pixels
- Object detection: locating objects with bounding rectangles
- Instance segmentation: demarcating objects with detailed outlines
- Programming tutorial

Today's Topics

- Motivation: training large capacity, deep models to locate content
- Semantic segmentation: classifying pixels
- Object detection: locating objects with bounding rectangles
- Instance segmentation: demarcating objects with detailed outlines
- Programming tutorial


The End