CSE 252D: Advanced Computer Vision

Manmohan Chandraker

Lecture 16: Object Detection 2

Virtual classrooms

- Virtual lectures on Zoom
  - Only host shares the screen
  - Keep video off and microphone muted
  - But please do speak up (remember to unmute!)
  - Slides uploaded on webpage just before class
- Virtual interactions on Zoom
  - Ask and answer plenty of questions
  - “Raise hand” feature on Zoom when you wish to speak
  - Post questions on chat window
  - Happy to try other suggestions!
- Lectures recorded and upload on Canvas
  - Available under “My Media” on Canvas

Overall goals for the course

- Introduce the state-of-the-art in computer vision
- Study principles that make them possible
- Get understanding of tools that drive computer vision
- Enable one or all of several such outcomes
  - Pursue higher studies in computer vision
  - Join industry to do cutting-edge work in computer vision
  - Gain appreciation of modern computer vision technologies
- This is a great time to study computer vision!

Papers for Wed, May 26

- R-FCN: Object Detection via Region-based Fully Convolutional Networks
  - https://arxiv.org/abs/1605.06409
- Deformable Convolutional Networks
  - https://arxiv.org/abs/1703.06211
- Feature Pyramid Networks for Object Detection
  - https://arxiv.org/abs/1612.03144
- Training Region-Based Object Detectors with Online Hard Example Mining
  - https://arxiv.org/abs/1604.03540

Papers for Fri, May 28

- You Only Look Once: Unified, Real-Time Object Detection
  - https://arxiv.org/abs/1506.02640
- Focal Loss for Dense Object Detection
  - https://arxiv.org/abs/1708.02002
- Single-Shot Refinement Neural Network for Object Detection
  - https://arxiv.org/abs/1711.06897
- CornerNet: Detecting Objects as Paired Keypoints
  - https://arxiv.org/abs/1808.01244

Papers for Wed, Jun 2

- Playing for Benchmarks
  - https://arxiv.org/abs/1709.07322
- Domain Adaptive Faster R-CNN for Object Detection in the Wild
  - https://arxiv.org/abs/1803.03243
- Fully Convolutional Adaptation Networks for Semantic Segmentation
  - https://arxiv.org/abs/1804.08286
- Label Efficient Learning of Transferable Representations across Domains and Tasks
  - https://arxiv.org/abs/1712.00123

Recap

Object Detection

Object 1: (x_1, y_1, w_1, h_1), dog

Object 2: (x_2, y_2, w_2, h_2), cat

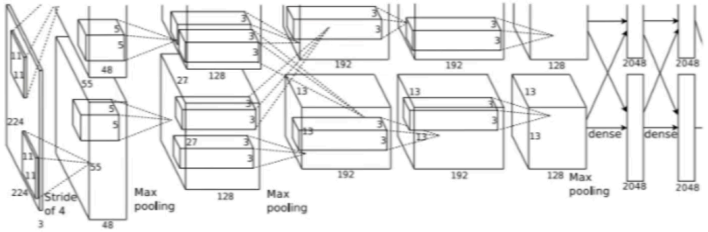
Object 3: (x_3, y_3, w_3, h_3), dog

Object 4: (x_4, y_4, w_4, h_4), cat

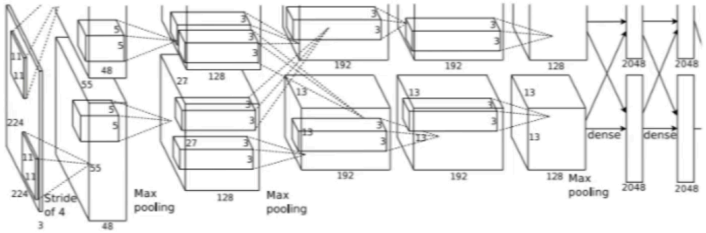
Object 5: (x_5, y_5, w_5, h_5), dog

Need to handle outputs of variable lengths

Sliding Windows



Dog? YES
Cat? NO
Background? NO

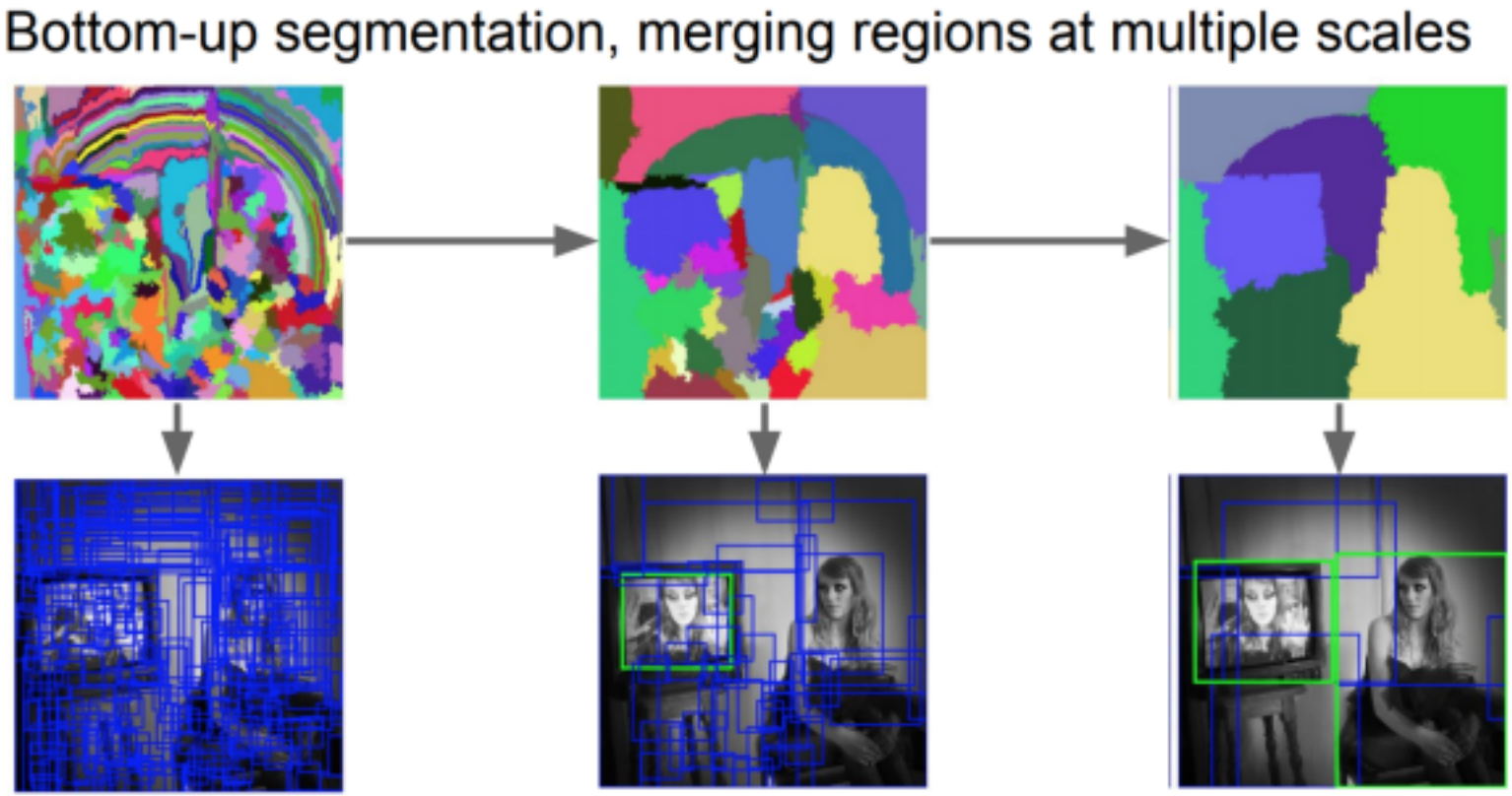


Dog? NO
Cat? NO
Background? YES

- Need to consider windows at several different positions and scales
- Either use a simple feature extractor, or evaluate only few candidates

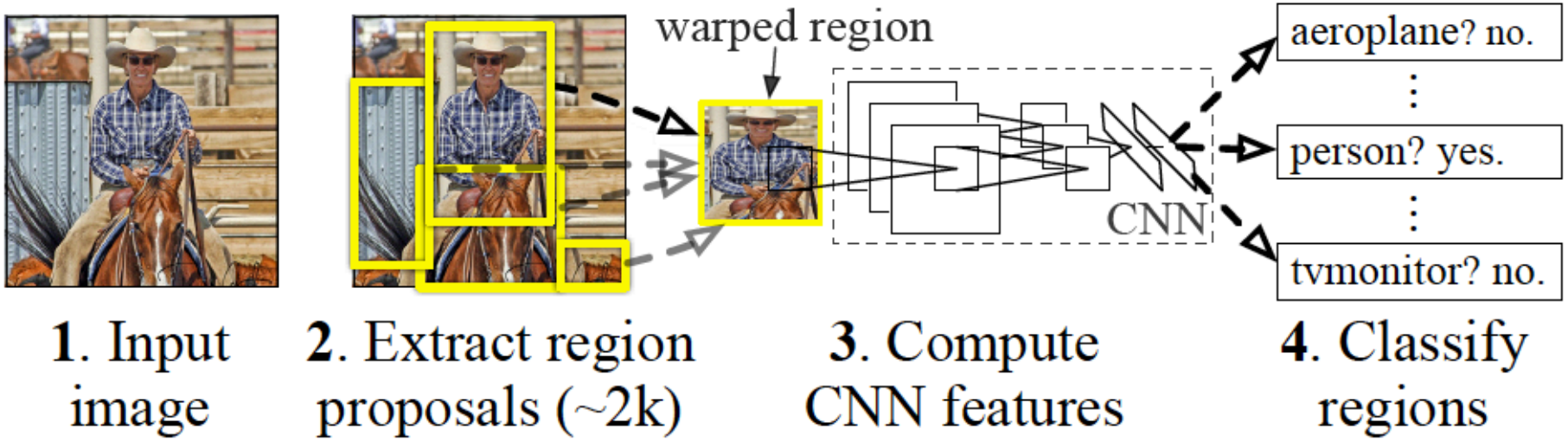
Region Proposals

- Find blob-like regions of image that might be objects
- Do not consider object type and tolerate high rate of false positives



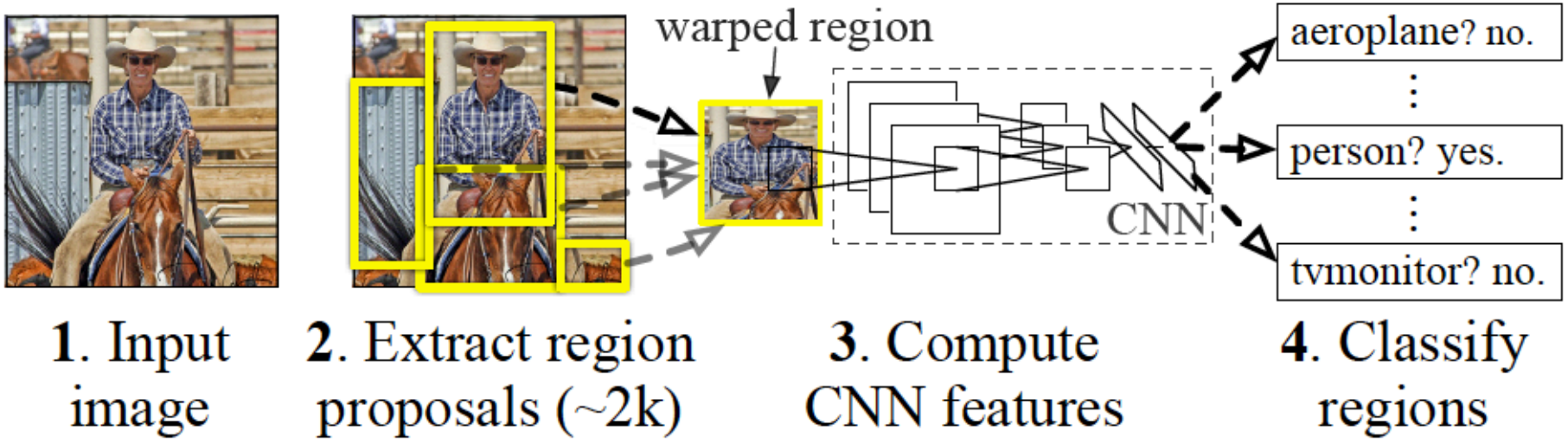
[Uijlings et al., Selective search for object recognition]

R-CNN

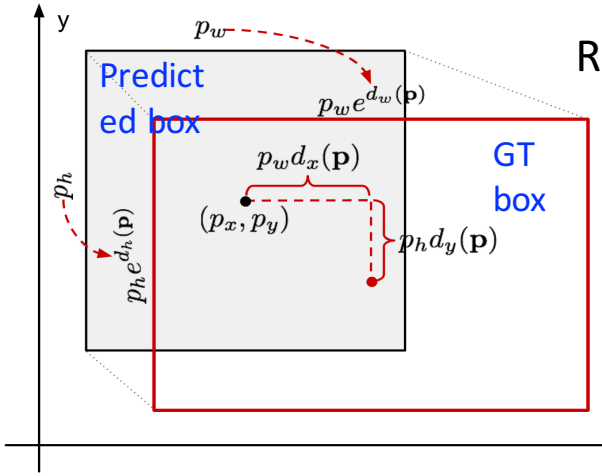


[Girshick et al., Rich feature hierarchies]

R-CNN



Step 5 (bbox regression): For each class, train a linear regression model to map from cached features to offsets to GT boxes to make up for "slightly wrong" proposals

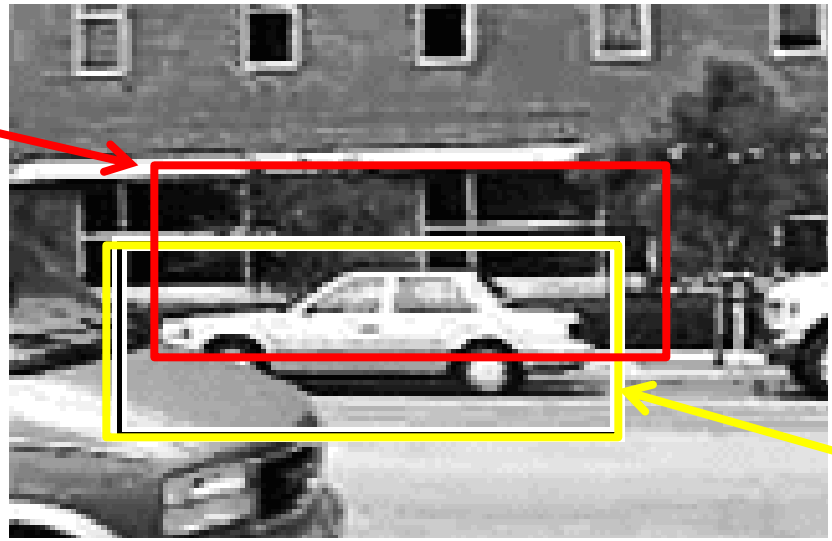


Regression targets:

$$t_x = (g_x - p_x)/p_w$$
$$t_y = (g_y - p_y)/p_h$$
$$t_w = \log(g_w/p_w)$$
$$t_h = \log(g_h/p_h)$$

[Girshick et al., Rich feature hierarchies]

Object Detection Evaluation: Scoring a Bounding Box



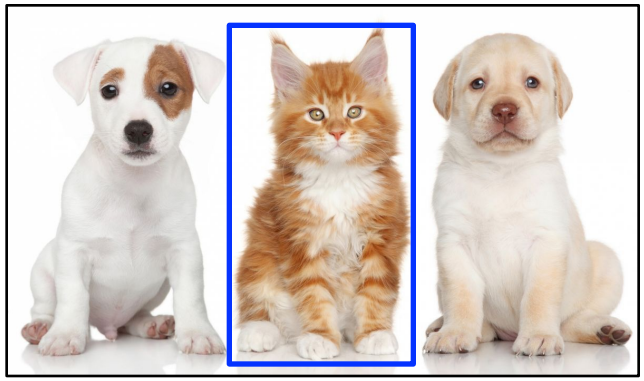
$$a_o = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})}$$

$$a_o > 0.5 \qquad correct$$

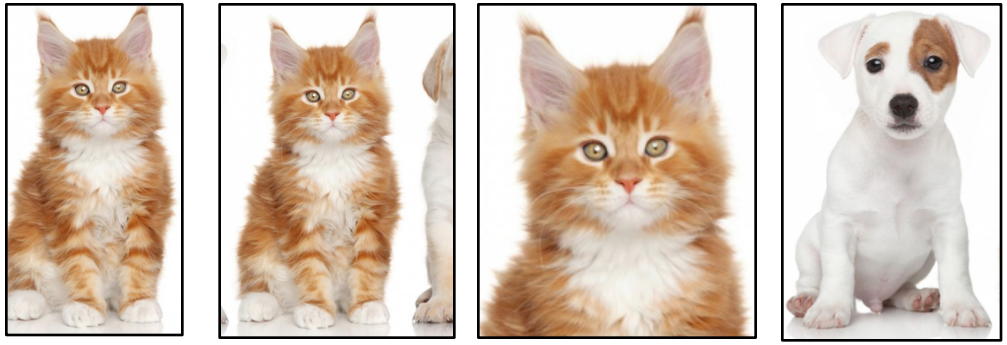
We say the detection is correct (a "true positive") if the intersection of the bounding boxes, divided by their union, is greater than a threshold.

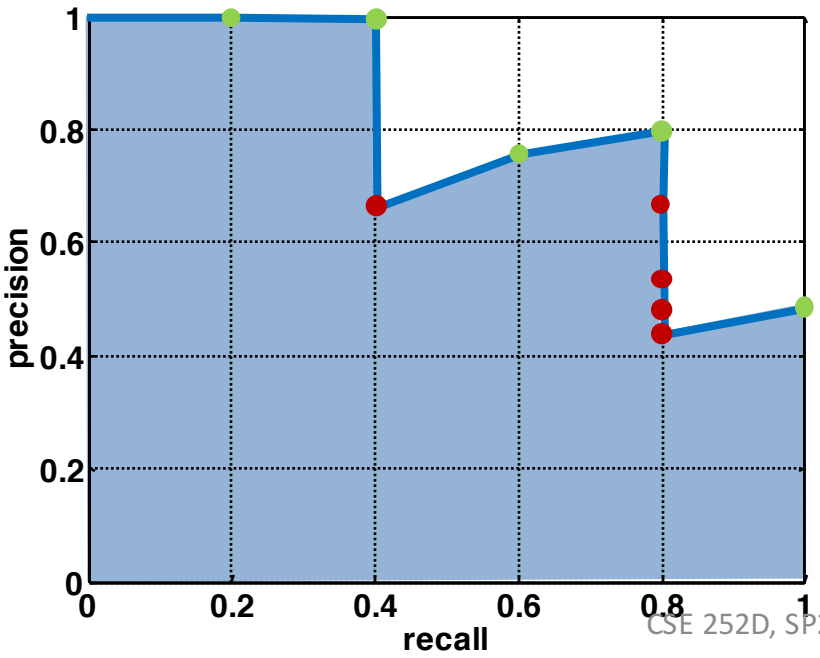
Object Detection Evaluation: Average Precision

Query



Results (ordered):

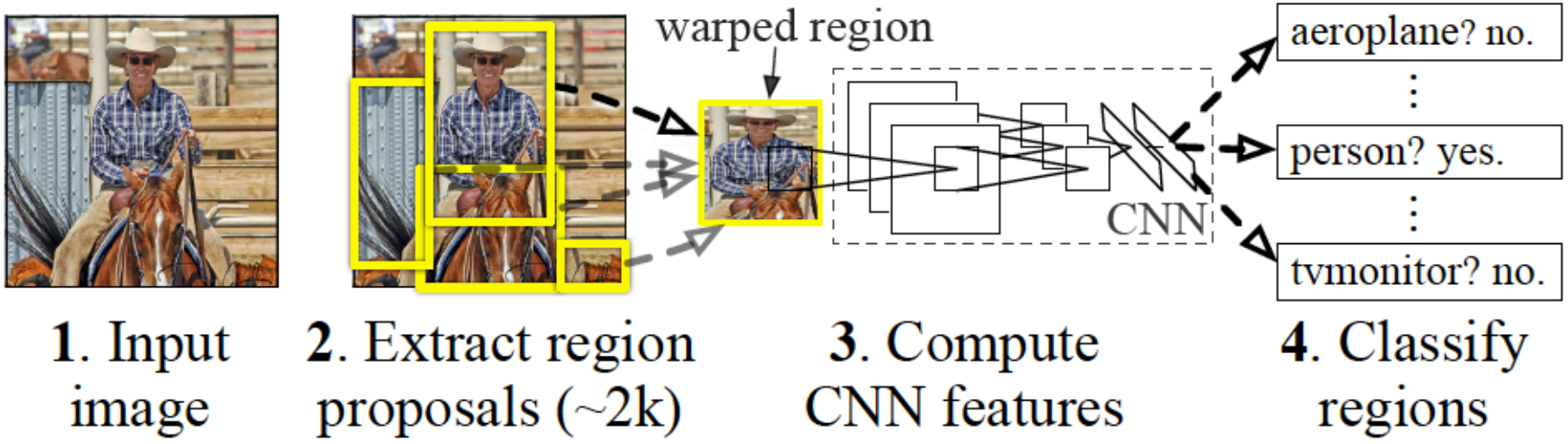




- Ordering determined by detector confidence (say, softmax score)
- Choose the IoU threshold
- Plot the precision-recall curve
- Average precision: area under the curve
- Mean AP: average AP across all categories

- AP-k: AP value with k% IoU
- Another metric: report the average of AP-k, for various k = [50, 55, …., 95].

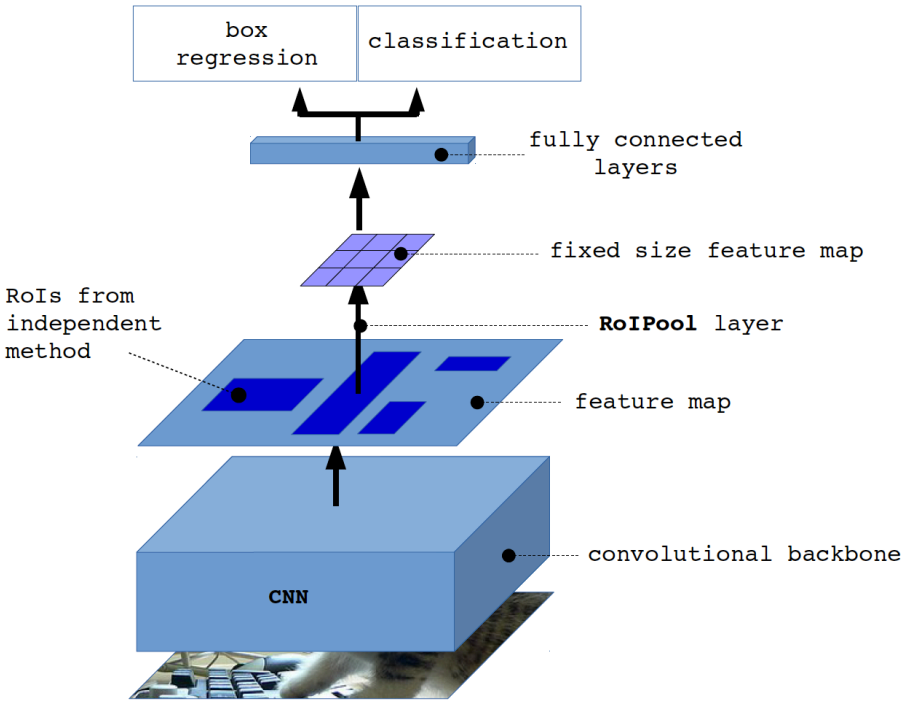
R-CNN

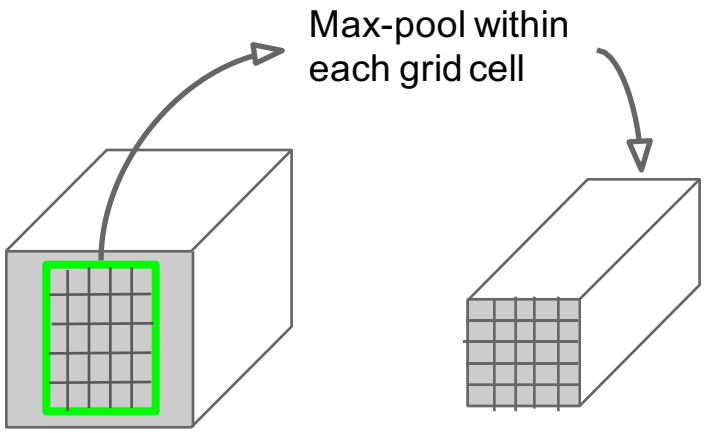


- Expensive
  - No feature reuse
  - Full forward pass of CNN for each region proposal
- Classification and regression are disjoint from feature extraction
  - CNN features not updated together with detection
- Complex training pipeline

[Girshick et al., Rich feature hierarchies]

Fast R-CNN





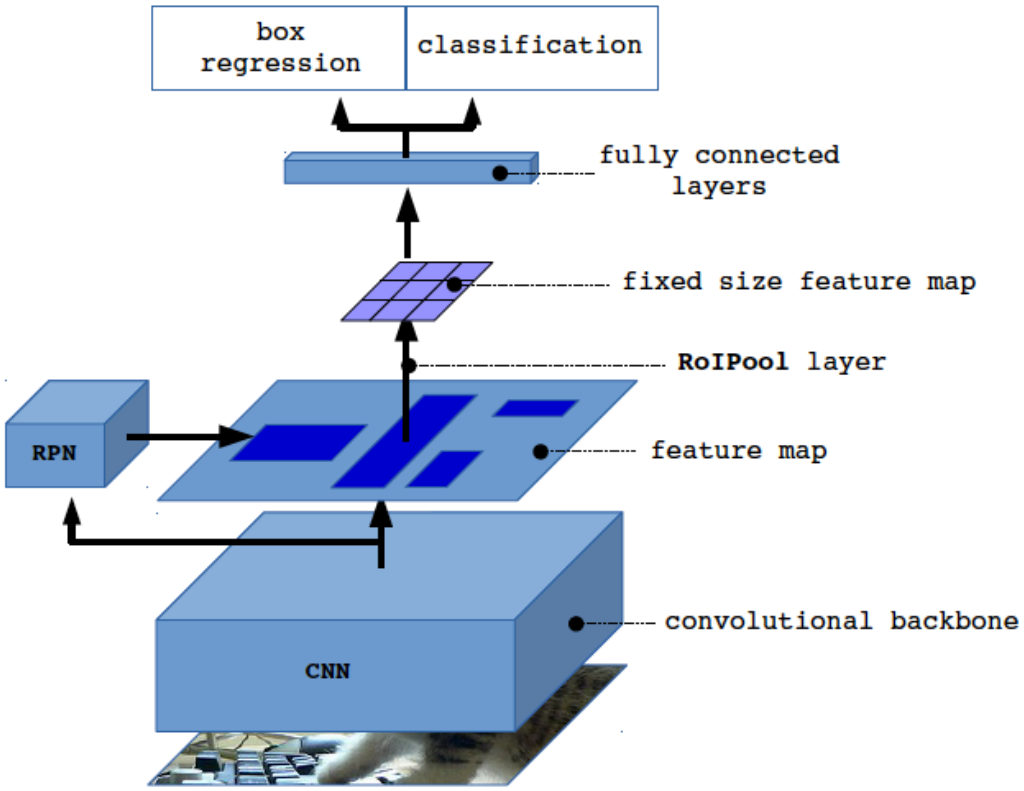
Hi-res conv features:
C x H x W
with region proposal

RoI conv features:
Fixed size C x h x w
to classifier

- RoI pooling: connects convolutional layers to classifier
  - Features and classifier trained together
- Feature extraction just once for the whole image
  - Faster inference and training
- Proposals still from external mechanism

[Girschick, Fast R-CNN]

Faster R-CNN

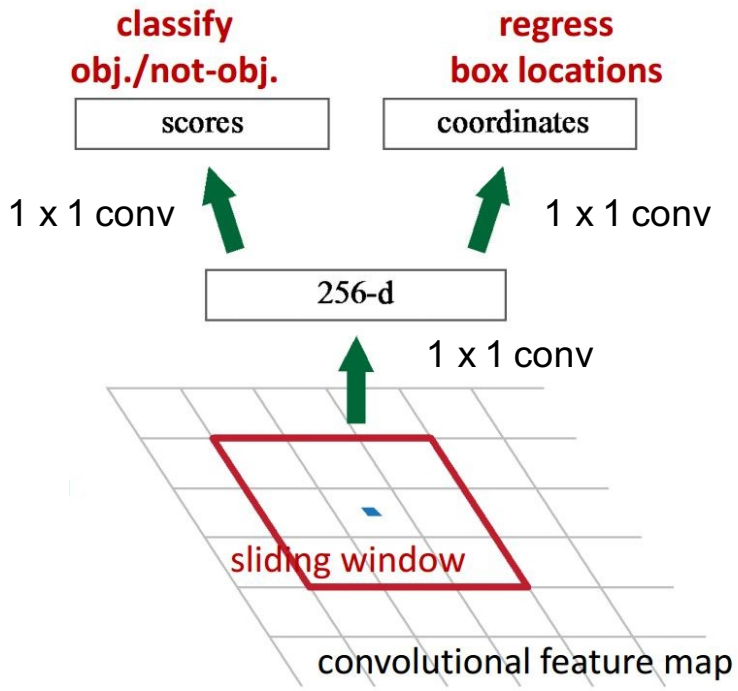


- Proposals still from separate mechanism in Fast R-CNN
- Insert a **Region Proposal Network (RPN)** after last convolutional layer
- RPN trained to produce region proposals directly, no need for external region proposals!
- After RPN, use RoI Pooling and an upstream classifier and regressor just like Fast R-CNN

[Ren et al., Faster R-CNN]

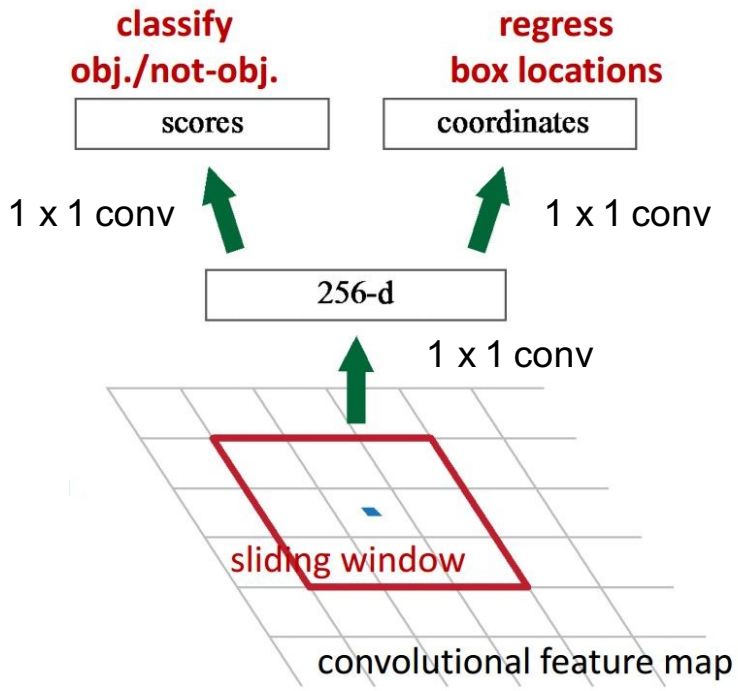
Faster R-CNN: Region Proposal Network

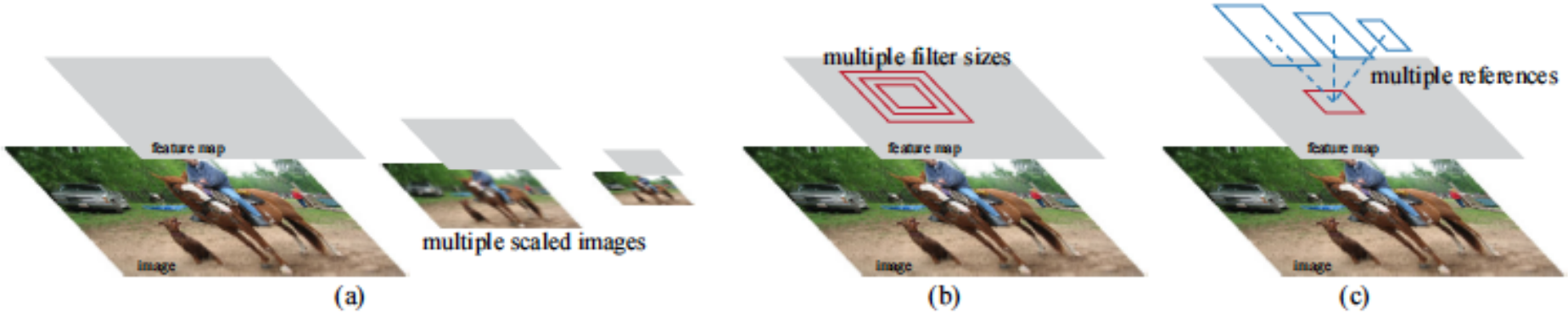
- Slide a small window on the feature map
- Build a small network for:
  - classifying object or not-object
  - regressing bounding box locations
- Position of sliding window gives location information with respect to the image
- Box regression gives finer localization with respect to this sliding window



Faster R-CNN: Need for Multiscale Proposals

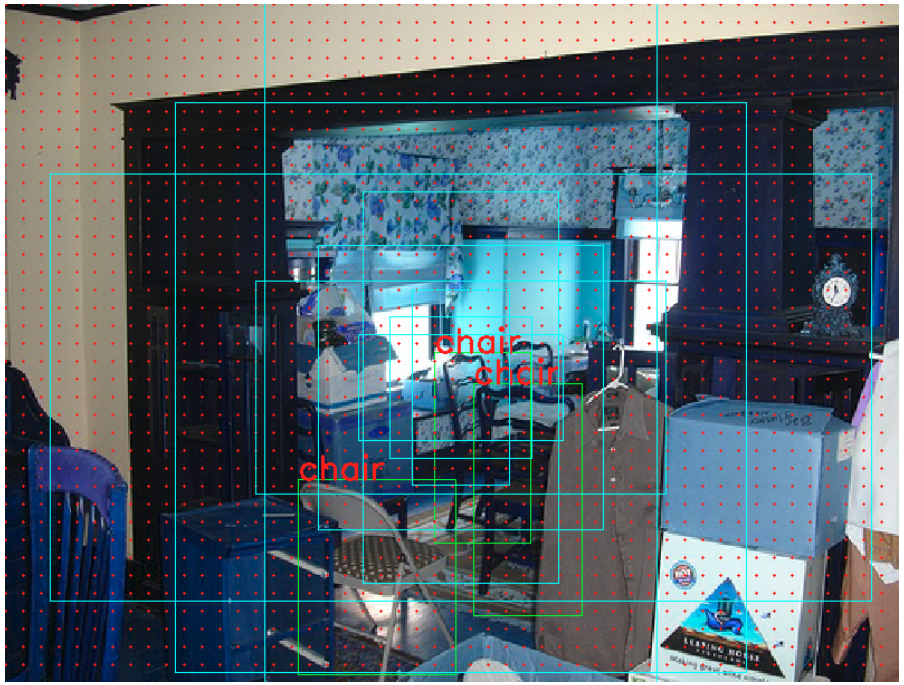
- Problem with region proposal using a convolutional network
  - Multiple scales of objects possible
- Possible solutions
  - Multiple scaled images
  - Filters of different sizes (usually together with above)
- But such approaches are time-consuming.

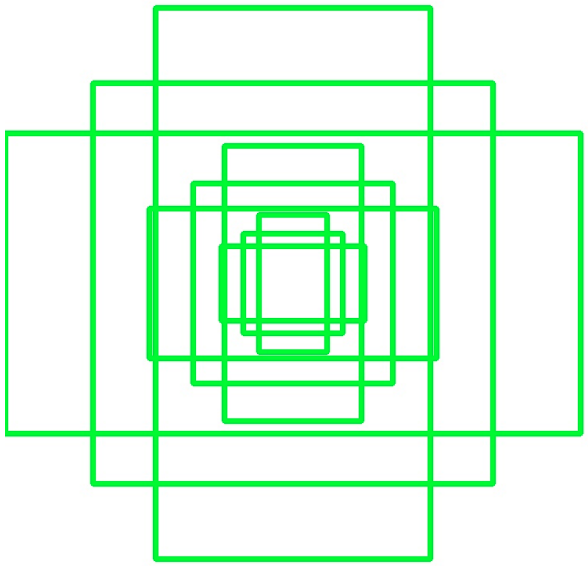




Faster R-CNN: Anchors

- Consider single object localization
  - Simply have a classification and regression head
- But now consider multiple objects, possibly overlapping
  - Regression for objects will interfere with each other
- Anchors: a set of reference positions on the feature map
  - Anchor box with high ground truth overlap responsible for regressing position
  - Determines reference and spatial extent for predicting object



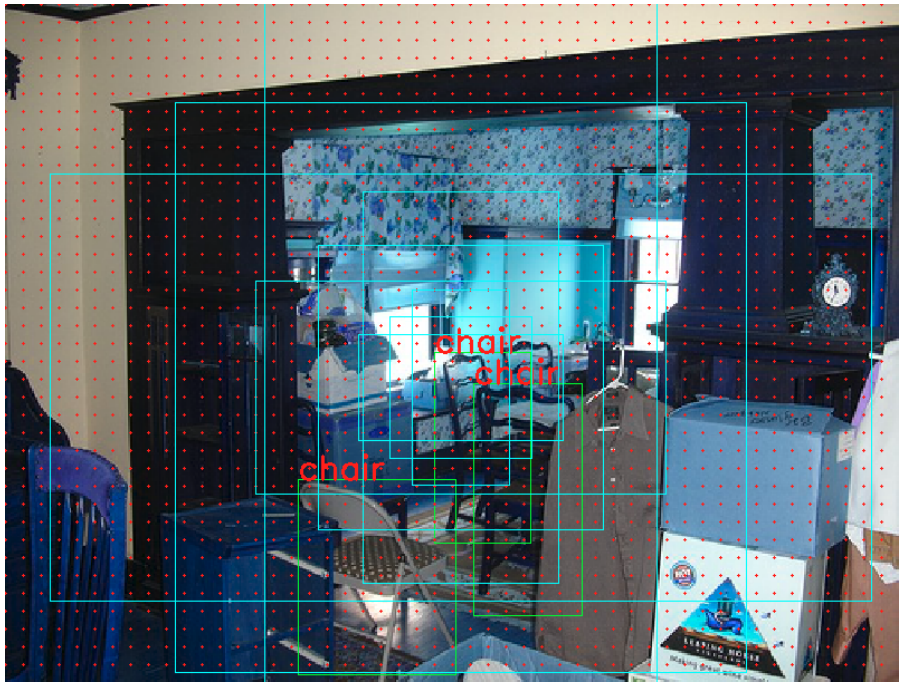


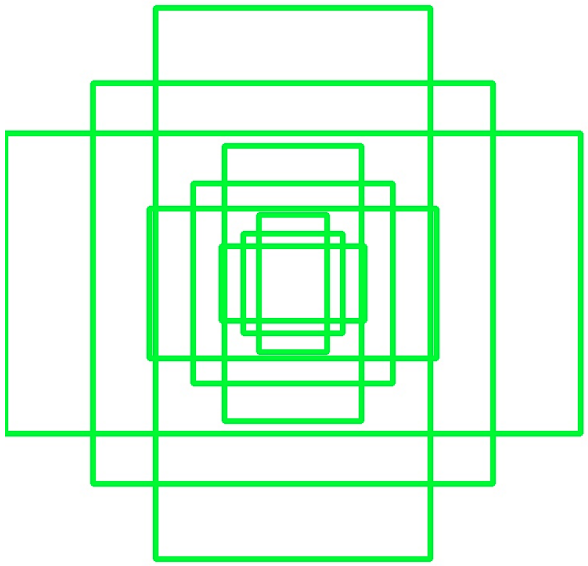
Typically: 3 scales and 3 aspect ratios

Faster R-CNN: Anchors

- A cost-efficient way to achieve multi-scale outputs
  - Relies on image and feature maps at single scale
  - Feature computation is shared across anchor boxes at different scales

- Translation invariance
  - Use same convolutional RPN, anchor boxes for spatial localization
  - A translated object will lead to accordingly shifted proposals





Typically: 3 scales and 3 aspect ratios

Training RPN

- Place anchors uniformly across image, n boxes at every position (typically n = 9)
  - For 40 x 60 feature map, about 21k anchor boxes
- RPN output: 4n regression (x, y, w, h) and 2n classification (object, background)
- Non-maximum suppression to pick about 2k boxes
  - Remove boxes that overlap with others of higher score
- Labels for RPN classification: compute IoU of anchor boxes with ground truth
  - Assign IoU > 0.7 as object and IoU < 0.3 as background
- Bounding box regression
  - Displacement target: distance between centers of ground truth and anchor boxes
  - Size target: log ratio of anchor and ground truth dimensions
- Sample 256 anchors to form mini-batch for training

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*).$$

Cross-entropy

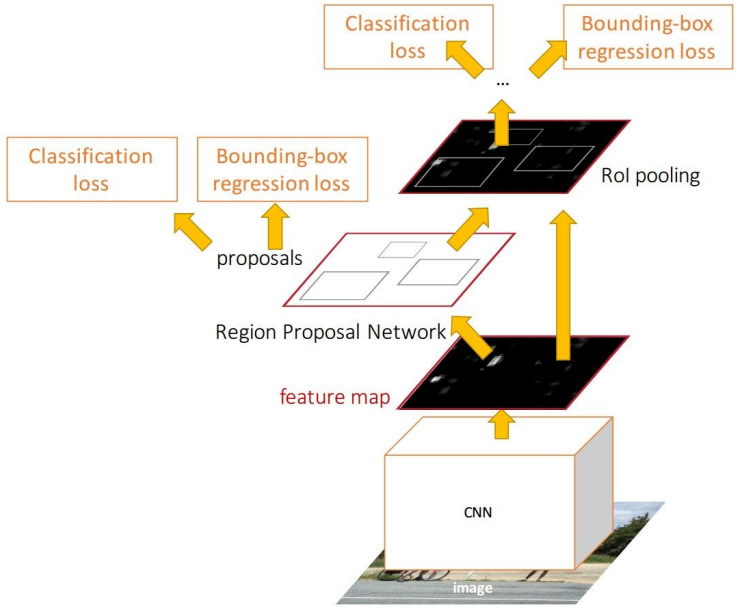
Bounding box label

Smooth L1

Regression target

Training Overall Detector

- Train CNN + RPN
  - Initialize with ImageNet pretrained weights
  - Train end-to-end for region proposal task
- Train CNN + Detector
  - Use fixed proposals from above RPN
  - Train Fast R-CNN for detection task
- Fine-tune RPN
  - Use CNN from above step
  - Fine-tune RPN for proposal task
- Fine-tune detector
  - Keep CNN fixed
  - Fine-tune Fast R-CNN layers for detection task

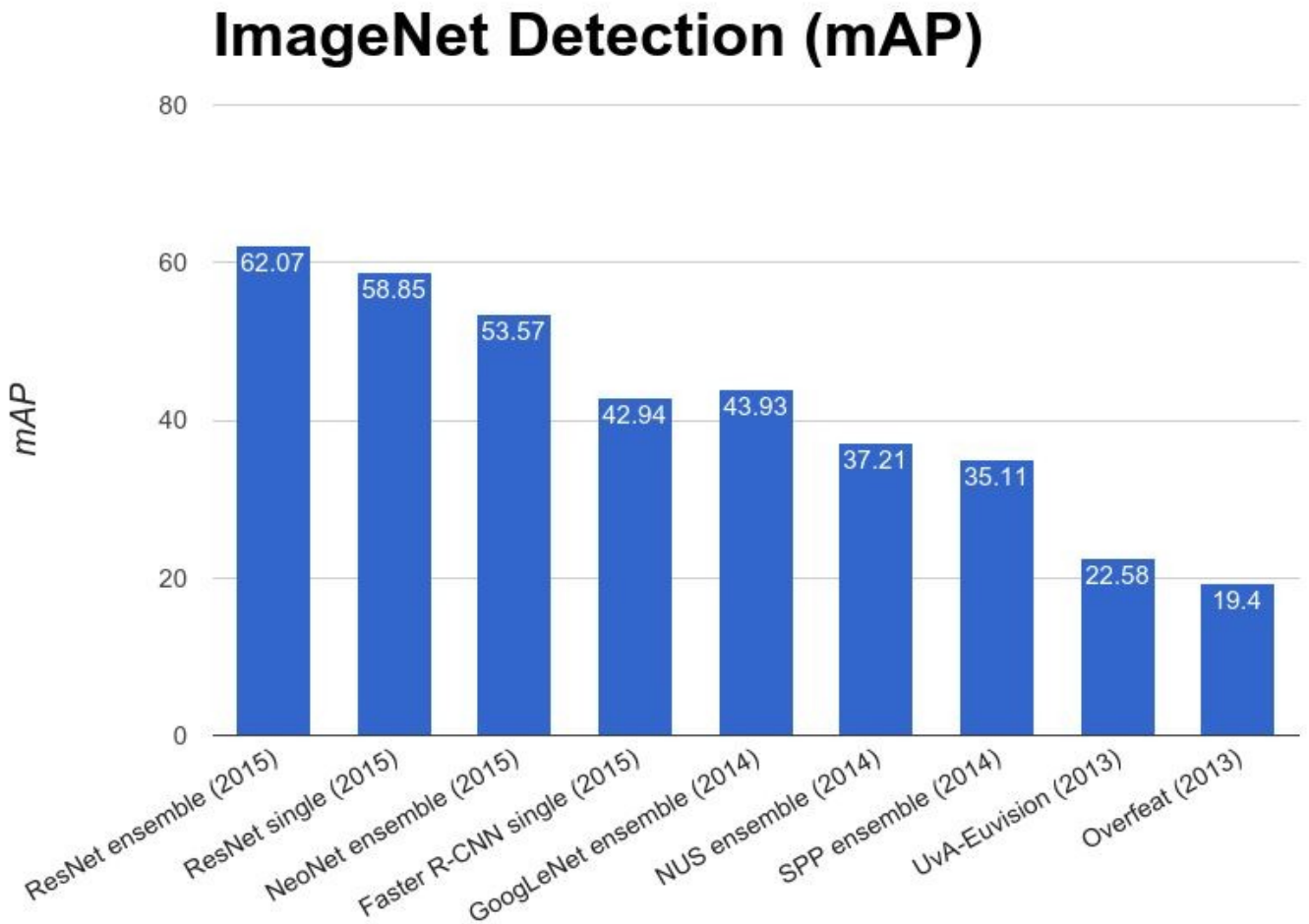


- Subsequently, a joint training is also available with all four losses
  - RPN: classification (object or background), regression (anchor to proposal)
  - Fast R-CNN: classification (object category), regression (proposal to bounding box)

Faster R-CNN Improvements

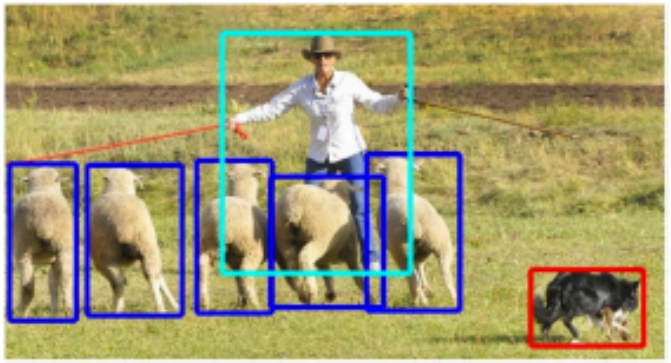
	R-CNN	**Fast R-CNN**	**Faster R-CNN**
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
mAP (VOC 2007)	66.0	**66.9**	**66.9**

Faster R-CNN Improvements



Instance Segmentation

Image classification



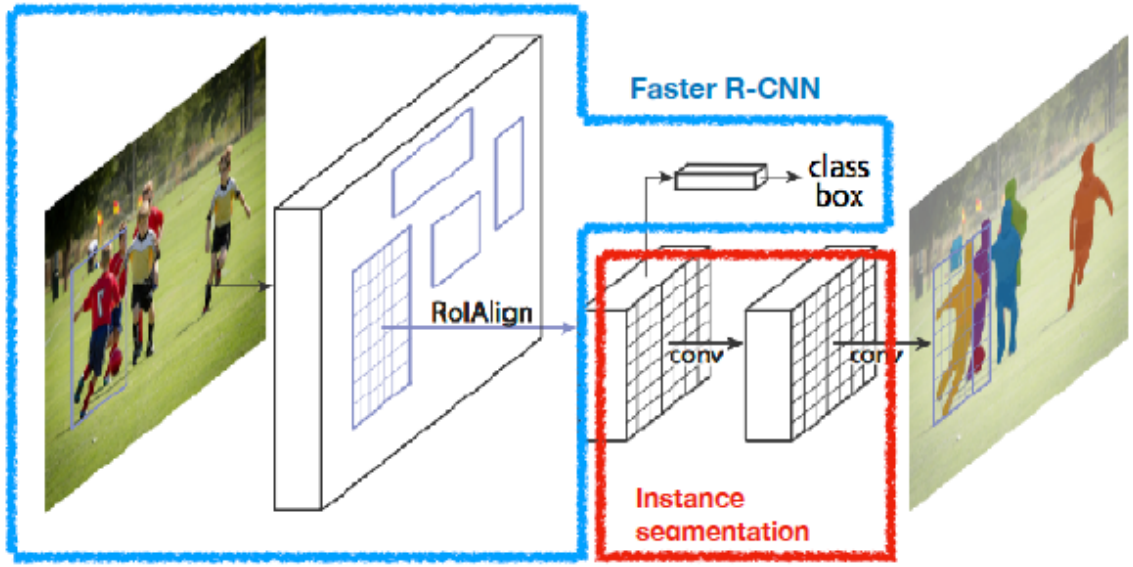
Object detection

Semantic segmentation

Instance segmentation

[Lin et al., MS COCO]

Instance Segmentation



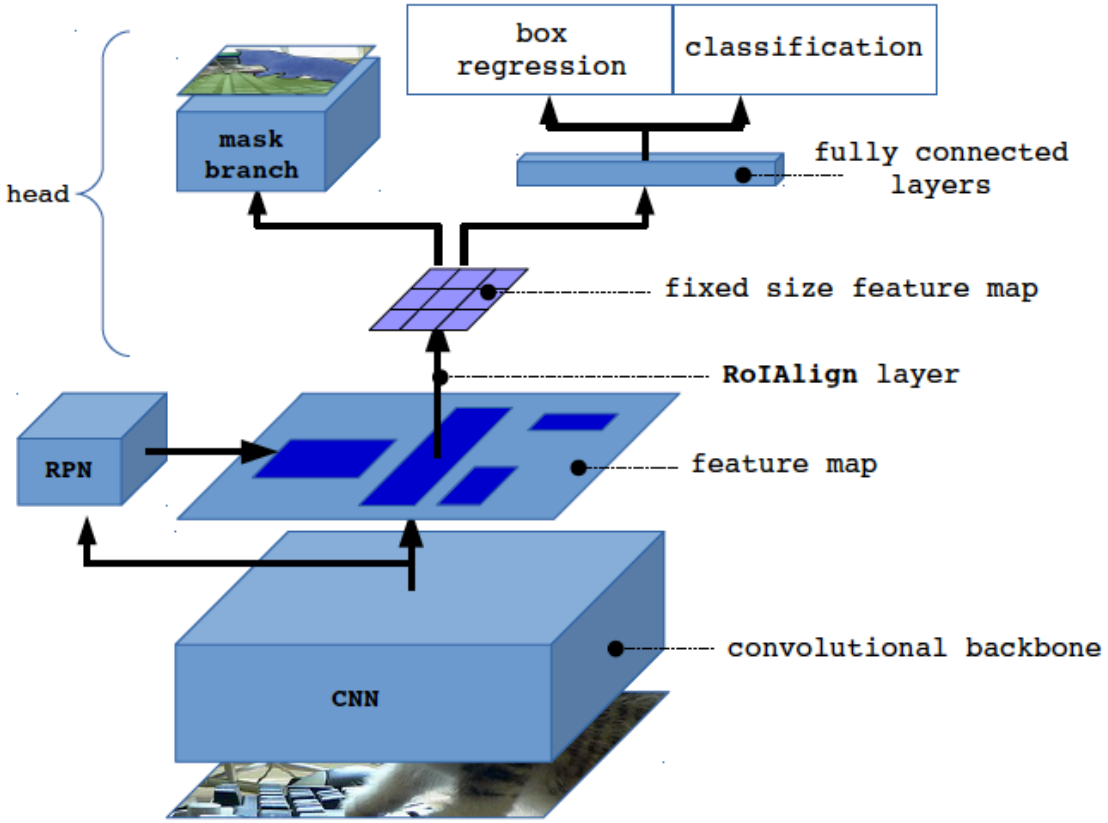
- Two choices:
  - Predict instance first, then classify
  - Segment first, then break into instances
  - Mask R-CNN is instance-first

- Also applicable to human pose estimation with keypoints as one-hot masks

Instance segmentation

[He et al., Mask R-CNN]

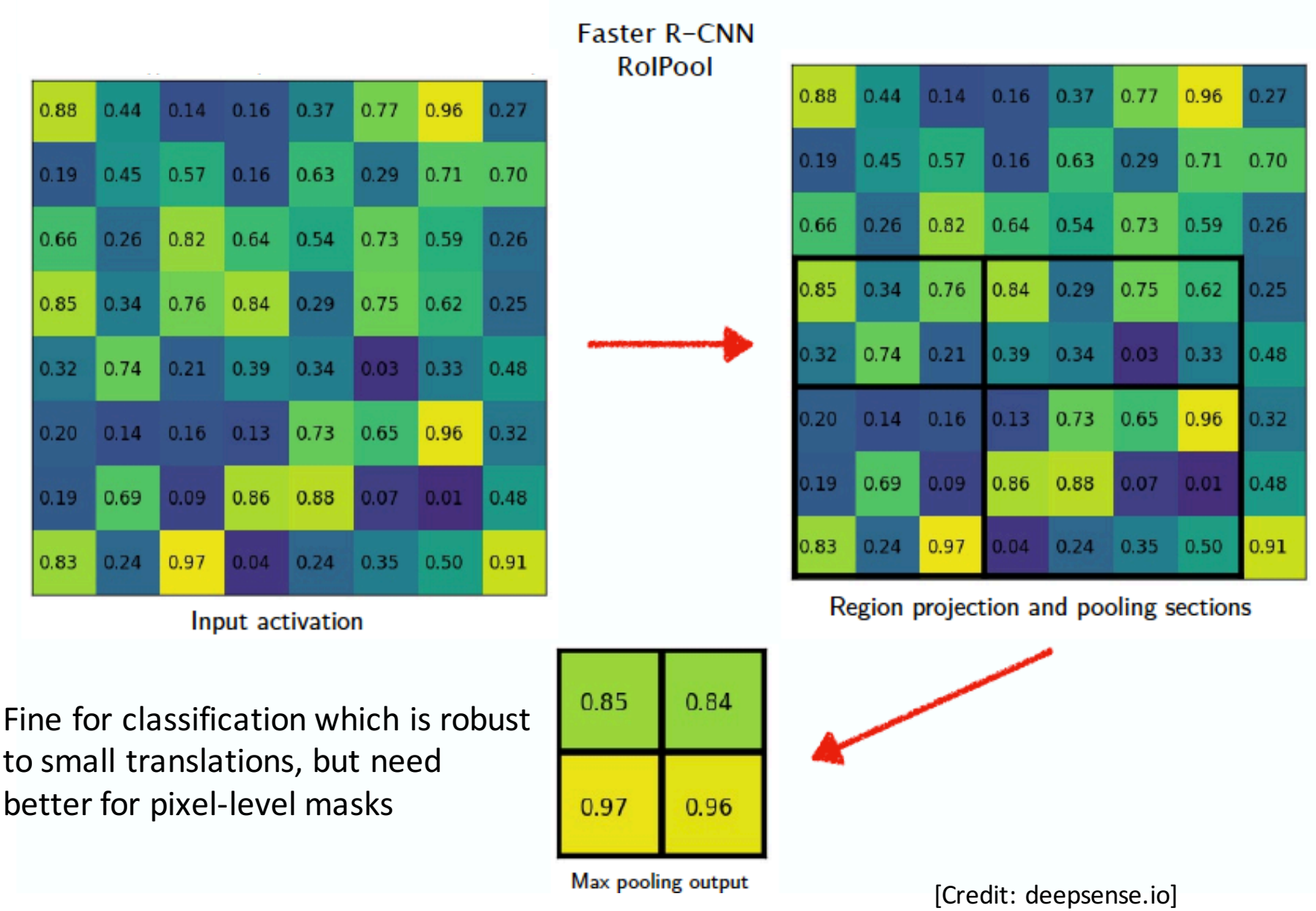
Mask R-CNN



- Predict mask in parallel to class (rather than before class)
- Decouple mask (FCN) and class prediction (Faster R-CNN)
- RoIAlign: quantization-free for better localization
- Training (2 days on 8 GPUs) and inference (200ms) speed

[He et al., Mask R-CNN]

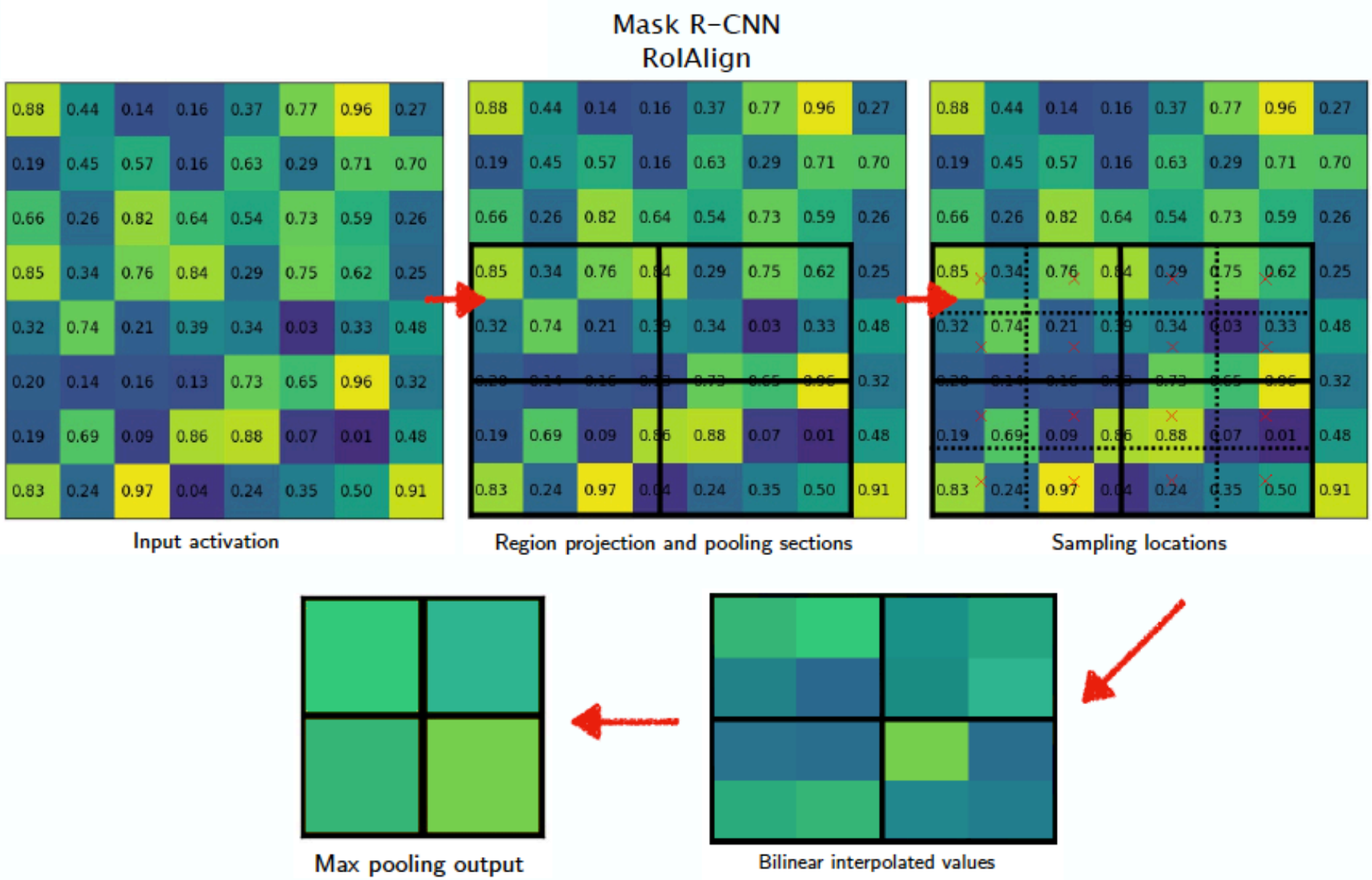
RoI-Pooling in Faster R-CNN



Fine for classification which is robust to small translations, but need better for pixel-level masks

[Credit: deepsense.io]

RoI-Align in Mask R-CNN



[Credit: Silvio Galesso]

Decoupled segmentation and classification

- Decoupling achieved with per-class binary masks (sigmoid)
- Alternative in FCN: multinomial masks (softmax)

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

A. Fast R-CNN B.

$K \cdot (m \times m)$ sigmoid outputs:
→ pixel-wise binary classification
→ one mask for each class, no competition

L_{mask}: mean binary cross-entropy

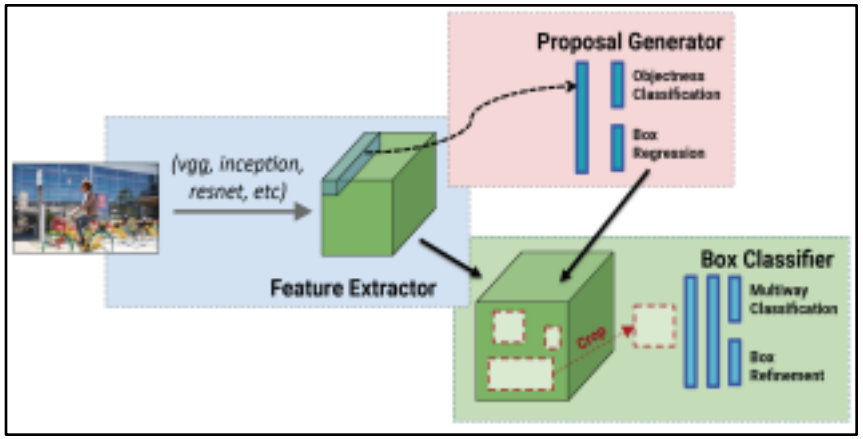
- Generate mask for every class without competition between classes
- Dedicated classification branch to predict class label to select output mask

	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*

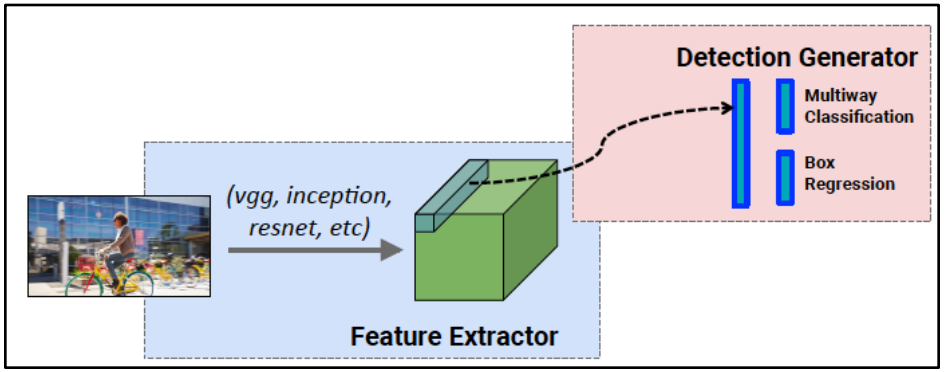
Single-Shot Detector

- Two step process of Faster R-CNN can be expensive
  - Box classification and regression are done twice
  - RPN produces boxes, but they are used to pool features
  - Separate classifier then used for evaluation

- Key motivation for SSD: achieve good accuracy-speed trade-off
  - Use something similar to RPN for directly scoring anchors
  - Extract anchors at multiple scales to get accuracy with minimal overhead

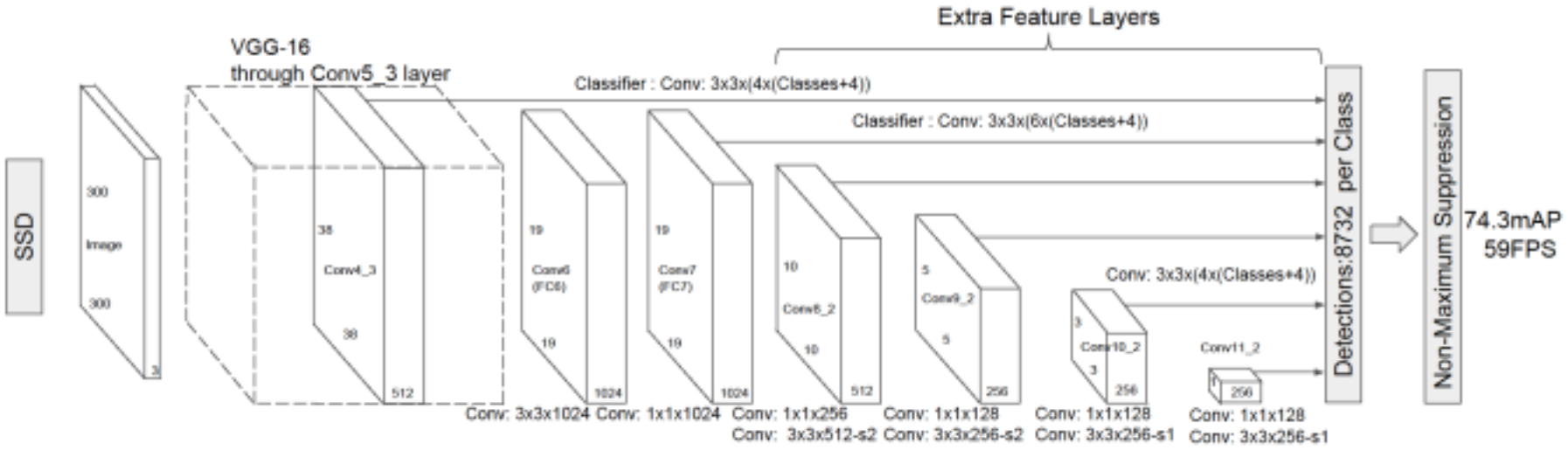


Faster R-CNN



SSD

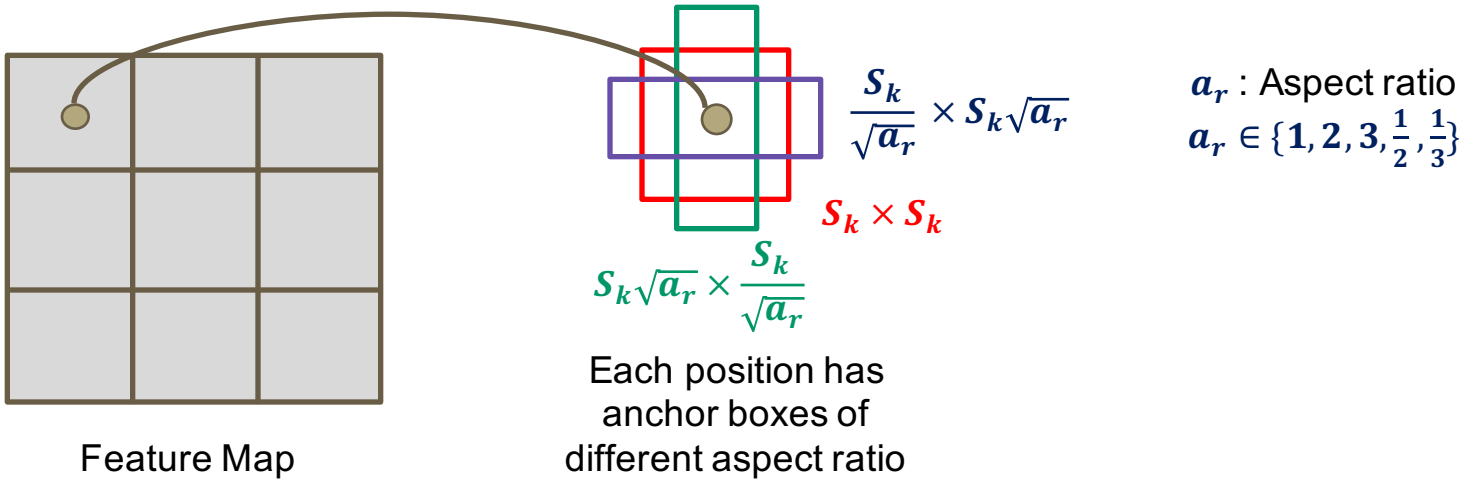
Single-Shot Detector



- Standard VGG (or ResNet) base network
- Add further layers with progressively decreasing size
  - Used for predicting detections at multiple scales
  - Layers with wider receptive fields expected to detect larger objects

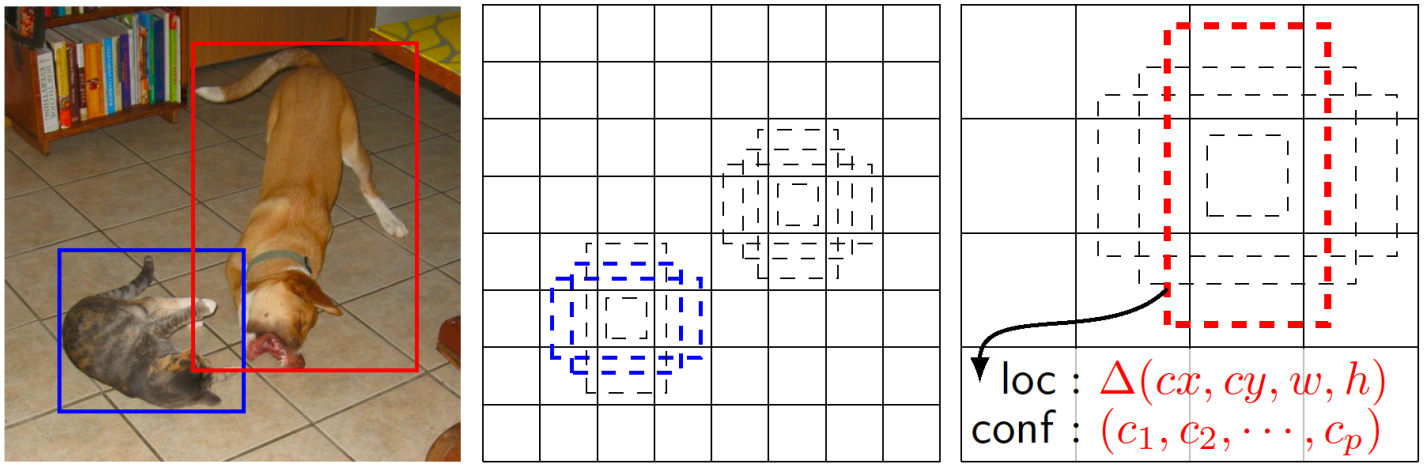
[Liu et al., SSD]

SSD: Default boxes (anchors)



- SSD uses default boxes, which are similar to anchors in Faster R-CNN
- Default boxes located at each cell on the feature map
  - Multiple boxes corresponding to different aspect ratios
- Default boxes across feature layers for better discretization of scale

SSD: Default boxes (anchors)

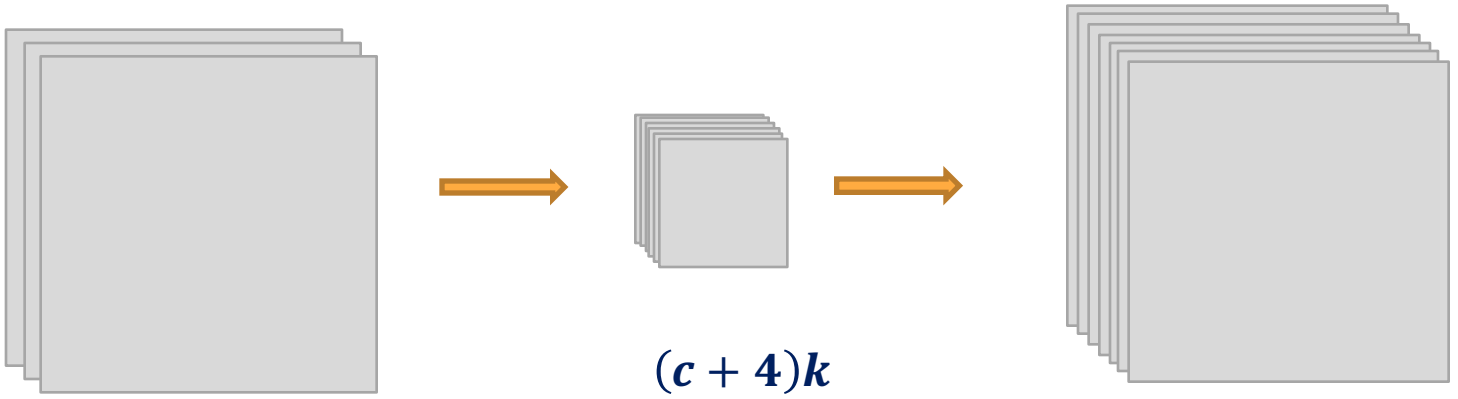


(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

- Anchor boxes of different feature maps have different scales
  - Various feature maps responsible for detecting objects of different sizes
- Let *m* be the number of feature maps used for prediction
- Let s_k denote the scale of feature map *k*
- Define $s_{\text{min}} = 0.2$ and $s_{\text{max}} = 0.9$

$$s_k = s_{\text{min}} + \frac{s_{\text{max}} - s_{\text{min}}}{m - 1}(k - 1), \quad k \in [1, m]$$

SSD: Convolution prediction at multiple scales



- On each feature map, two types of convolutional filters applied
  - c filters for class prediction, where c is the number of object categories
  - 4 filters for bounding box regression, for coordinates x, y, w, h
- With k default boxes at every cell, there are $(c+4)k$ filters for each feature map
- Output for $m \times n$ feature map is a $m \times n \times (c+4)k$ map
  - Represents category label and regression offsets for each default box

SSD: Training

- Need to match ground truth to default boxes to allow training
- Ground truth positives: match ground truth to each default box with IoU > 0.5
- Ground truth negatives: all other default boxes

SSD: Training

- Need to match ground truth to default boxes to allow training
- Ground truth positives: match ground truth to each default box with IoU > 0.5
- Ground truth negatives: all other default boxes
- Loss: Combination of classification confidence and localization accuracy
- Classification loss is softmax for multi-class prediction

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^{p} log(\hat{c}_i^p) - \sum_{i \in Neg} log(\hat{c}_i^0) \quad \text{where} \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$$

x_{ij}^p = 1 if default box *i* matches ground truth box *j* of class *p*, otherwise 0

- Smooth L1 loss for localization

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^{k} \text{smooth}_{\text{L1}}(l_i^m - \hat{g}_j^m)$$

Offsets — Predicted location — Ground truth for box *j*

SSD: Training

- Need to match ground truth to default boxes to allow training
- Ground truth positives: match ground truth to each default box with IoU > 0.5
- Ground truth negatives: all other default boxes
- Loss: Combination of classification confidence and localization accuracy
- Classification loss is softmax for multi-class prediction

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^{p} log(\hat{c}_i^p) - \sum_{i \in Neg} log(\hat{c}_i^0) \quad \text{where} \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$$

x_{ij}^p = 1 if default box *i* matches ground truth box *j* of class *p*, otherwise 0

- Smooth L1 loss for localization

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^{k} \text{smooth}_{\text{L1}}(l_i^m - \hat{g}_j^m)$$

Offsets — Predicted location — Ground truth for box *j*

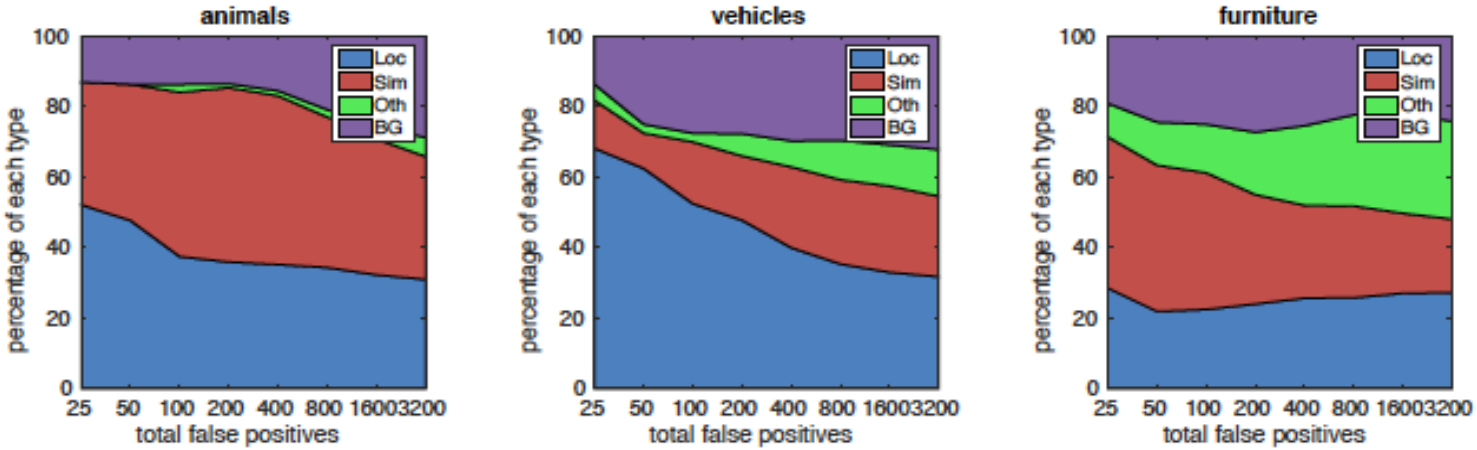
- Number of negatives far exceeds positives
  - Hard negative mining: choose default boxes with highest classification loss

SSD: Analysis

- Helps to use multiple layers for prediction

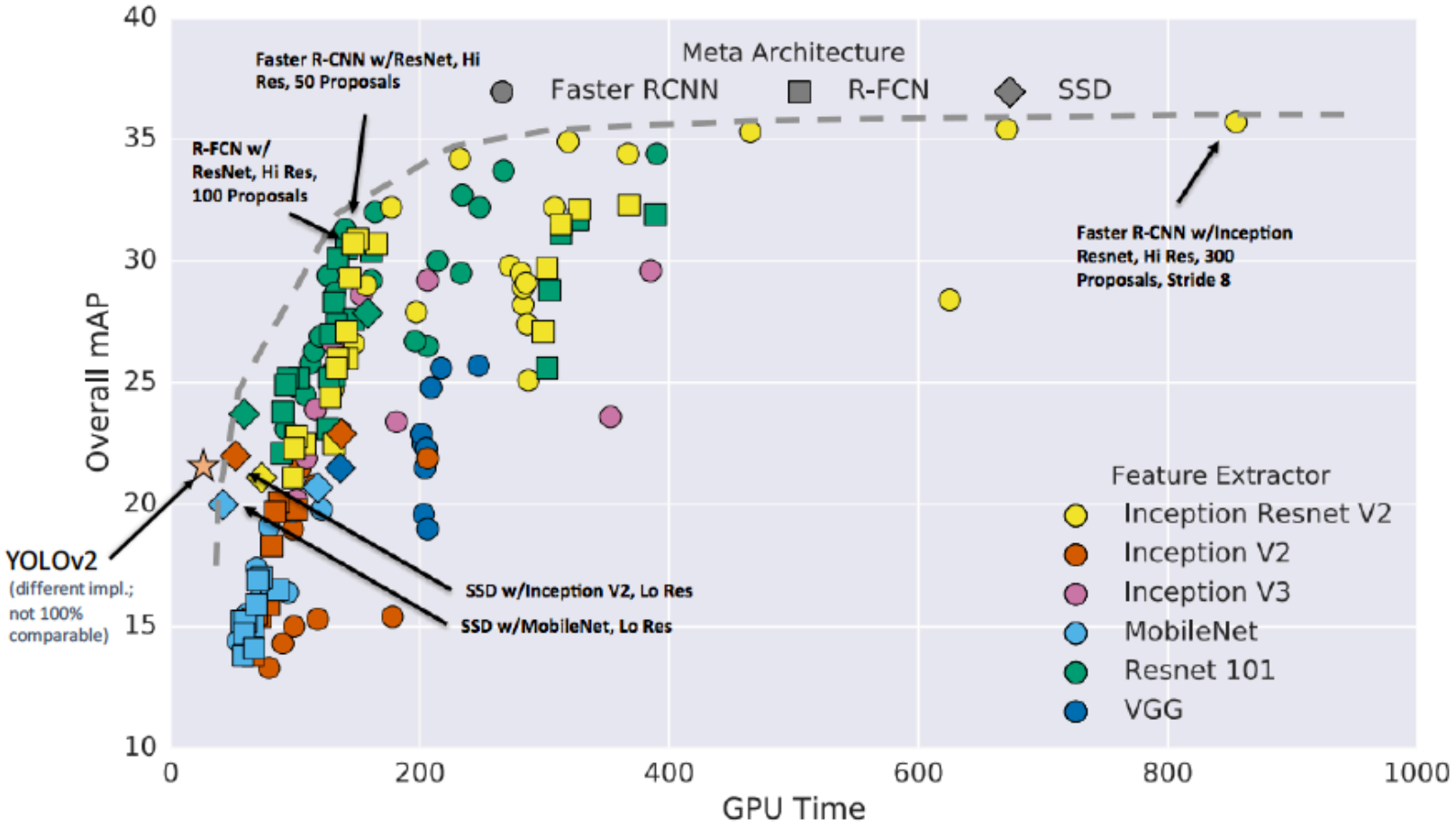
Prediction source layers from:						mAP use boundary boxes?		# Boxes
conv4_3	conv7	conv8_2	conv9_2	conv10_2	conv11_2	Yes	No	
✔	✔	✔	✔	✔	✔	74.3	63.4	8732
✔	✔	✔	✔	✔		**74.6**	63.1	8764
✔	✔	✔	✔			73.8	68.4	8942
✔	✔	✔				70.7	69.2	9864
✔	✔					64.2	64.4	9025
	✔					62.4	64.0	8664

- In general, most errors in SSD are due to poorer localization
  - Explicit pooling of region proposals in Faster R-CNN prevents this
  - Possibly also cause for more confusion between similar categories in SSD



- SSD also tends to do better for larger objects compared to smaller ones

Object Detection: Accuracy and Speed



Huang et al. Speed/Accuracy Tradeoffs for Modern Convolutional Object Detectors. CVPR 2017