The early history of ANNs for visual recognition

Jitendra Malik

Handwritten digit recognition (MNIST,USPS)

- LeCun's Convolutional Neural Networks variations (0.8%, 0.6% and 0.4% on MNIST)
- Tangent Distance(Simard, LeCun & Denker: 2.5% on USPS)
- Randomized Decision Trees (Amit, Geman & Wilder, 0.8%)
- SVM on orientation histograms(Maji & Malik, 0.8%)

Fig. 4. Size-normalized examples from the MNIST database.

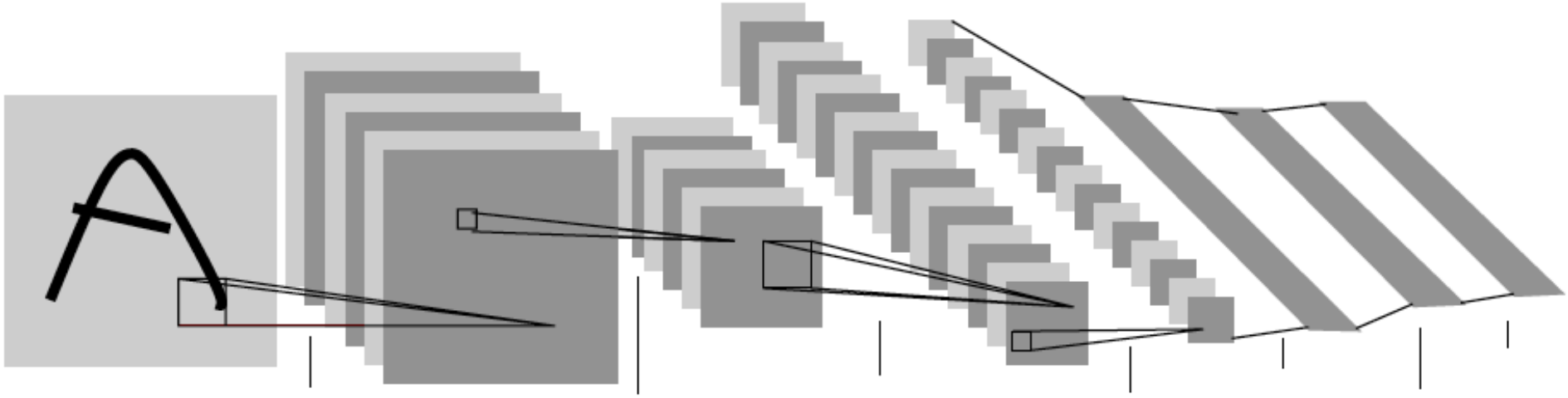
The MNIST DATABASE of handwritten digits

yann.lecun.com/exdb/mnist/

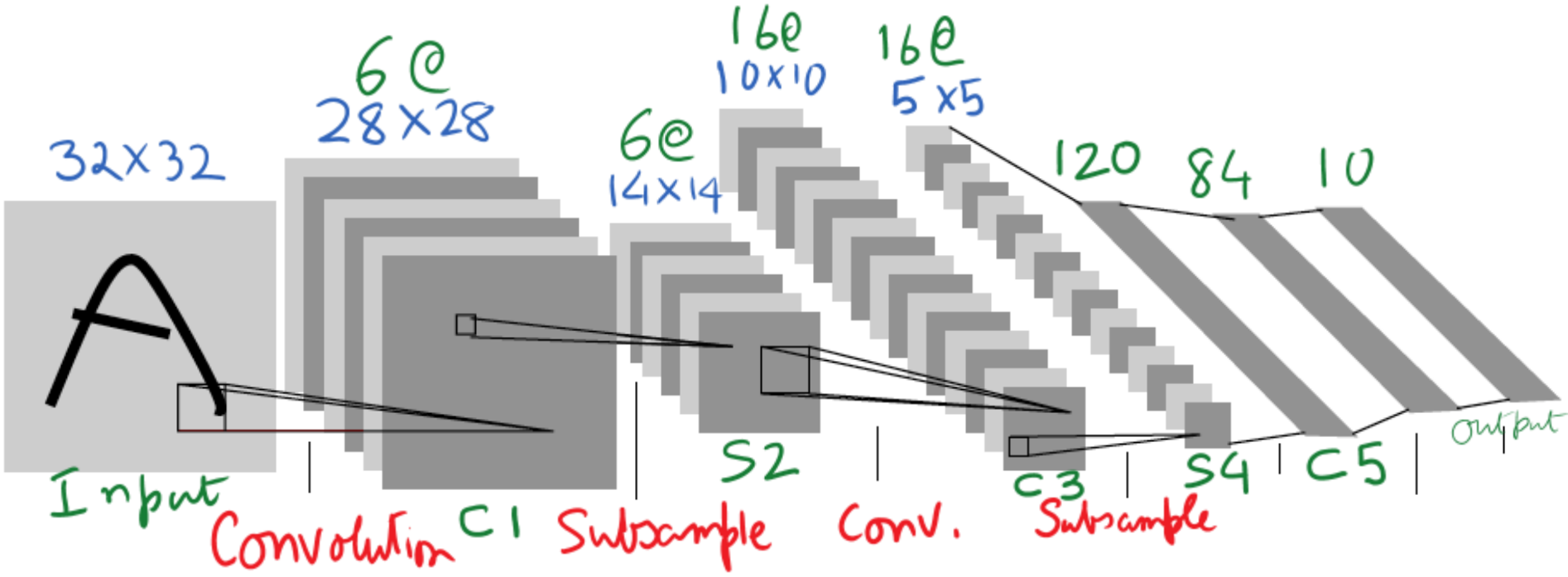
Yann LeCun & Corinna Cortes

- Has a training set of 60 K examples (6K examples for each digit), and a test set of 10K examples.
- Each digit is a 28 x 28 pixel grey level image. The digit itself occupies the central 20 x 20 pixels, and the center of mass lies at the center of the box.
- *"It is a good database for people who want to try learning techniques and pattern recognition methods on real-world data while spending minimal efforts on preprocessing and formatting."*

Convolutional Neural Networks
LeCun et al (1989)



Convolutional Neural Networks (LeCun et al)



This section describes in more detail the architecture of LeNet-5, the Convolutional Neural Network used in the experiments. LeNet-5 comprises 7 layers, not counting the input, all of which contain trainable parameters (weights). The input is a 32x32 pixel image. This is significantly larger than the largest character in the database (at most 20x20 pixels centered in a 28x28 field). The reason is that it is desirable that potential distinctive features such as stroke end-points or corner can appear *in the center* of the receptive field of the highest-level feature detectors. In LeNet-5 the set of centers of the receptive fields of the last convolutional layer (C3, see below) form a 20x20 area in the center of the 32x32 input. The values of the input pixels are normalized so that the background level (white) corresponds to a value of -0.1 and the foreground (black) corresponds to 1.175. This makes the mean input roughly 0, and the variance roughly 1 which accelerates learning [46].

In the following, convolutional layers are labeled Cx, subsampling layers are labeled Sx, and fully-connected layers are labeled Fx, where x is the layer index.

Layer C1 is a convolutional layer with 6 feature maps. Each unit in each feature map is connected to a 5x5 neighborhood in the input. The size of the feature maps is 28x28 which prevents connection from the input from falling off the boundary. C1 contains 156 trainable parameters, and 122,304 connections.

Layer S2 is a sub-sampling layer with 6 feature maps of size 14x14. Each unit in each feature map is connected to a 2x2 neighborhood in the corresponding feature map in C1. The four inputs to a unit in S2 are added, then multiplied by a trainable coefficient, and added to a trainable bias. The result is passed through a sigmoidal function. The 2x2 receptive fields are non-overlapping, therefore feature maps in S2 have half the number of rows and column as feature maps in C1. Layer S2 has 12 trainable parameters and 5,880 connections.

Layer C3 is a convolutional layer with 16 feature maps. Each unit in each feature map is connected to several 5x5 neighborhoods at identical locations in a subset of S2's feature maps. Table I shows the set of S2 feature maps

Layer S2 is a sub-sampling layer with 6 feature maps of size 14x14. Each unit in each feature map is connected to a 2x2 neighborhood in the corresponding feature map in C1. The four inputs to a unit in S2 are added, then multiplied by a trainable coefficient, and added to a trainable bias. The result is passed through a sigmoidal function. The 2x2 receptive fields are non-overlapping, therefore feature maps in S2 have half the number of rows and column as feature maps in C1. Layer S2 has 12 trainable parameters and 5,880 connections.

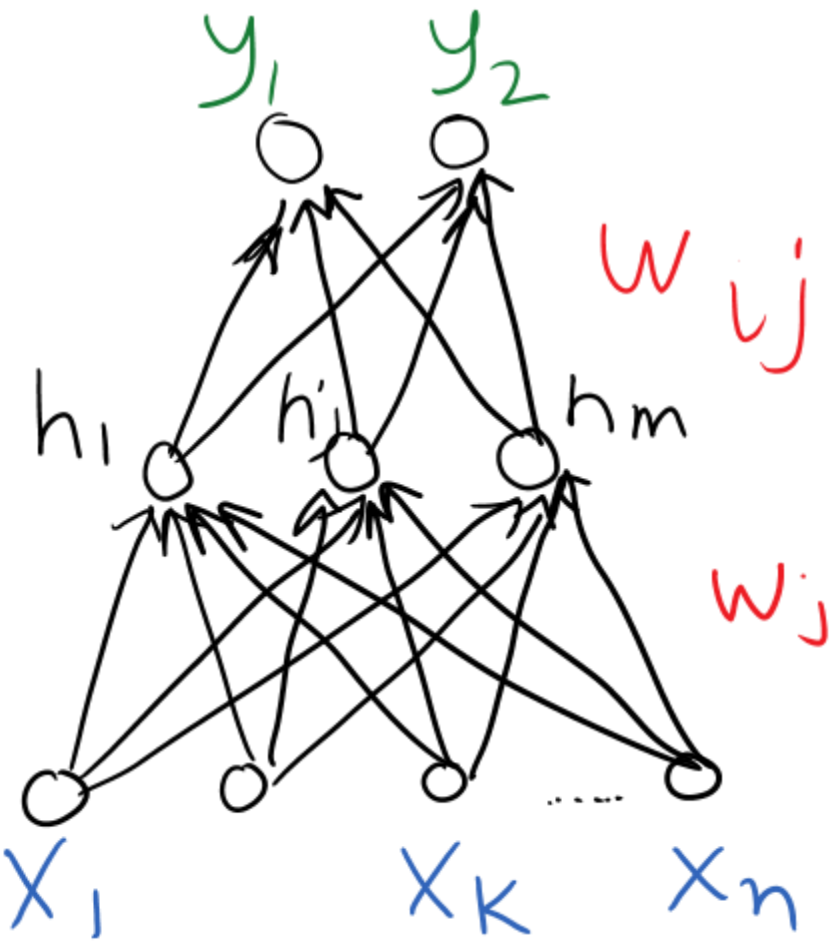
Layer C3 is a convolutional layer with 16 feature maps. Each unit in each feature map is connected to several 5x5 neighborhoods at identical locations in a subset of S2's feature maps. Table I shows the set of S2 feature maps combined by each C3 feature map. Why not connect every S2 feature map to every C3 feature map? The reason is twofold. First, a non-complete connection scheme keeps the number of connections within reasonable bounds. More importantly, it forces a break of symmetry in the network. Different feature maps are forced to extract different (hopefully complementary) features because they get different sets of inputs. The rationale behind the connection scheme in table I is the following. The first six C3 feature maps take inputs from every contiguous subsets of three feature maps in S2. The next six take input from every contiguous subset of four. The next three take input from some discontinuous subsets of four. Finally the last one takes input from all S2 feature maps. Layer C3 has 1,516 trainable parameters and 151,600 connections.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

TABLE I

EACH COLUMN INDICATES WHICH FEATURE MAP IN S2 ARE COMBINED

Training multi-layer networks



$$y_i = g\left(\sum_j w_{ij} h_j\right)$$

$$h_j = g\left(\sum_k w_{jk} x_k\right)$$

Minimize $(y_i - y_i^{desired})^2$ by suitable choice of w's

How do we find objects in real images?

The PASCAL VOC object detection challenge 2006-2015

The AlexNet paper from 2012

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

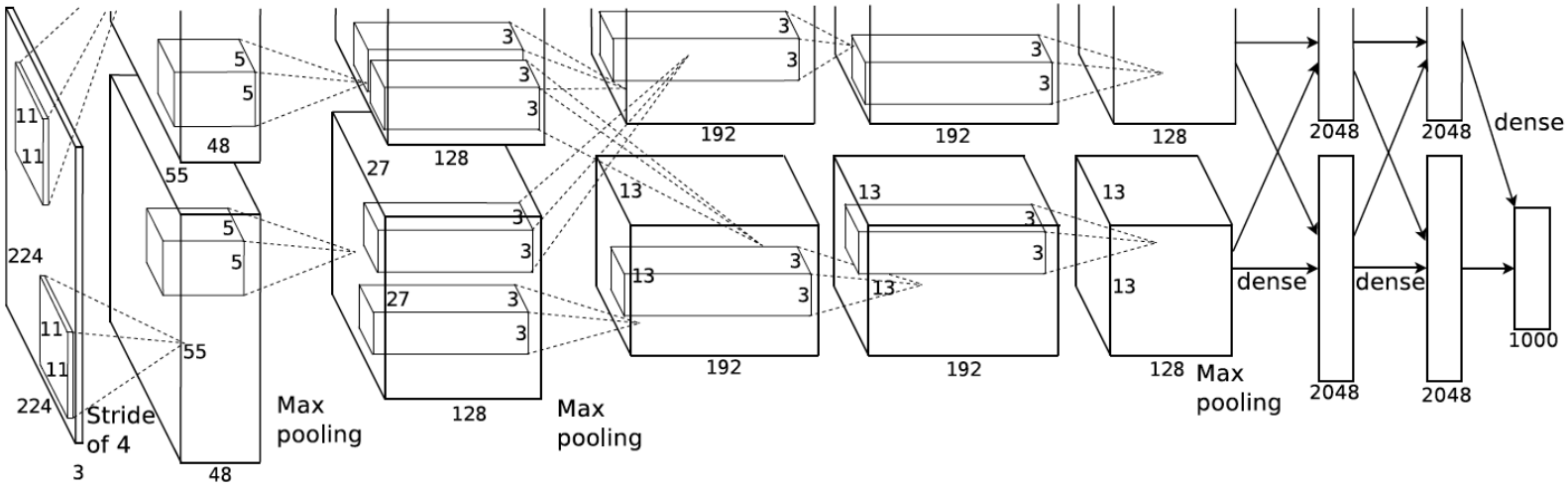


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

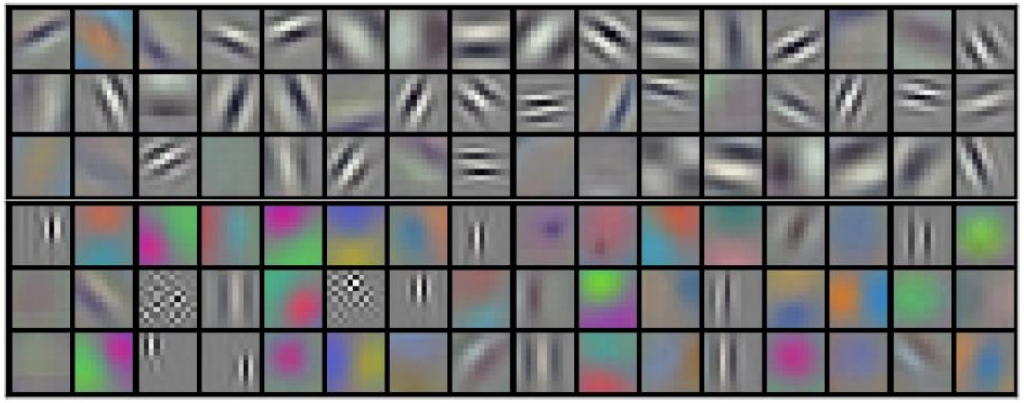


Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

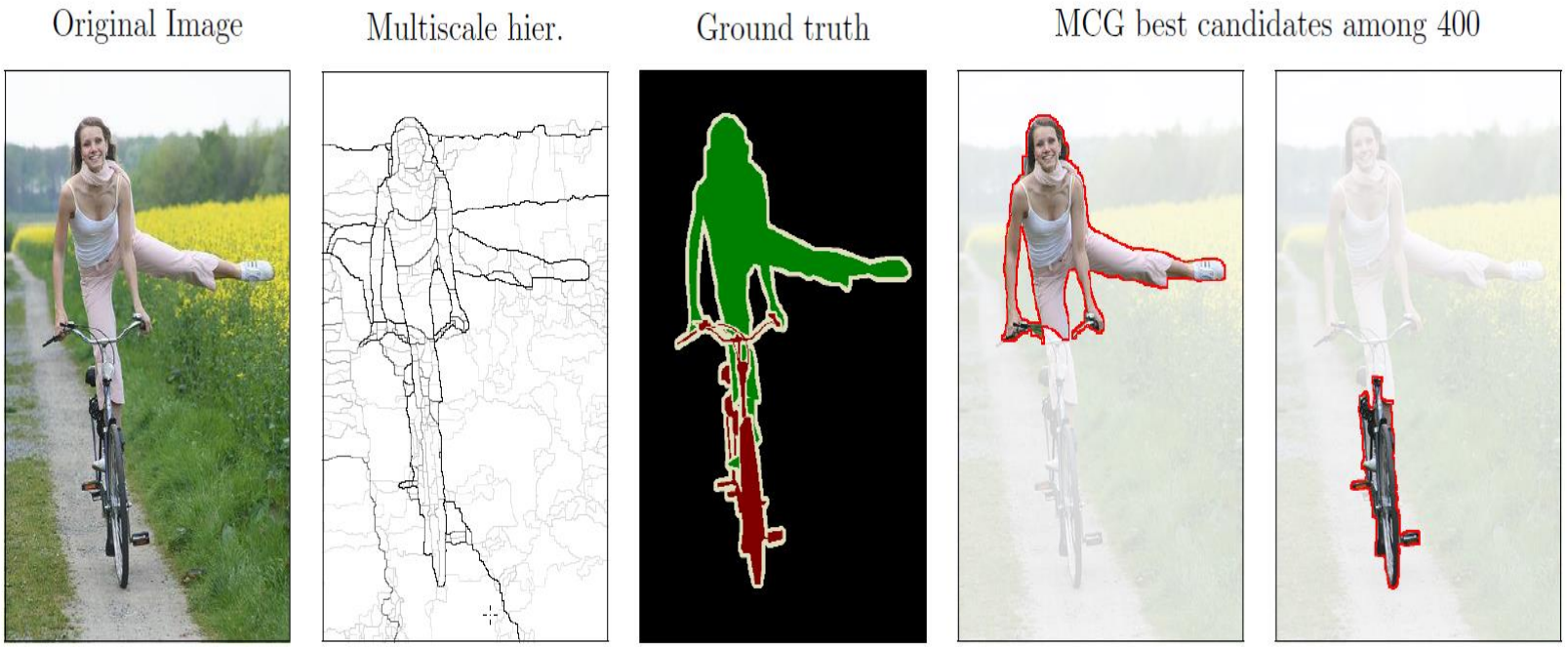
How do we find objects in real images?

The PASCAL VOC object detection challenge 2006-2015

Two main approaches

- Sliding windows at multiple scales
- Bottom-up region/bounding box proposals

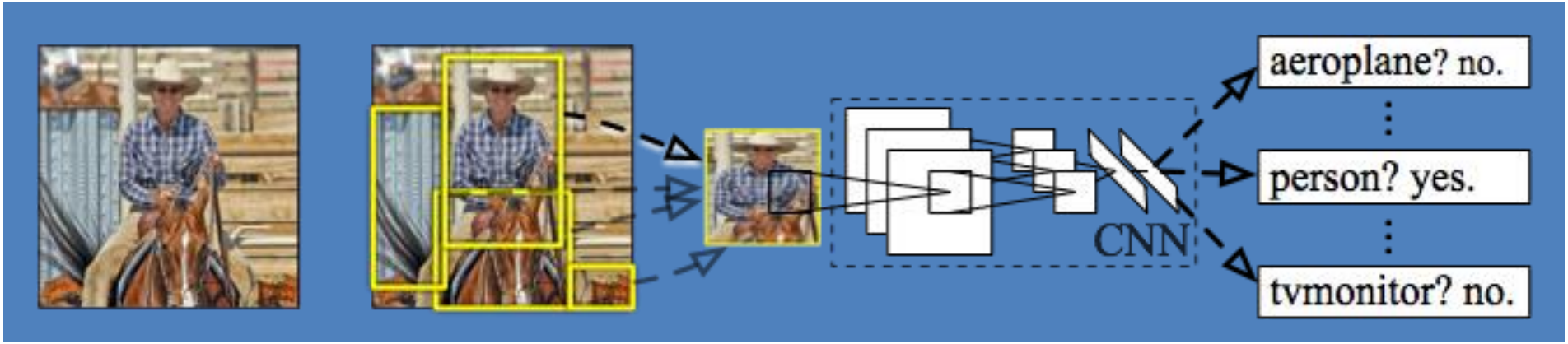
Bottom-up grouping as input to recognition



We produce superpixels of coherent color and texture first, then combine neighboring ones to generate object candidates

R-CNN: Regions with CNN features

Girshick, Donahue, Darrell & Malik (CVPR 2014)



Input image

Extract region proposals (~2k / image)

Compute CNN features

Classify regions (linear SVM)

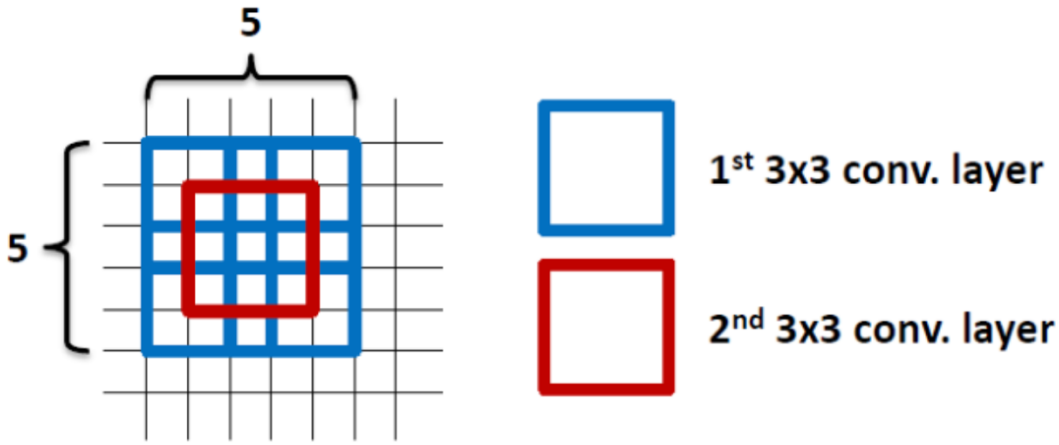
VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Karen Simonyan* **& Andrew Zisserman+**
Visual Geometry Group, Department of Engineering Science, University of Oxford
{karen,az}@robots.ox.ac.uk

ABSTRACT

In this work we investigate the effect of the convolutional network depth on its accuracy in the large-scale image recognition setting. Our main contribution is a thorough evaluation of networks of increasing depth using an architecture with very small (3×3) convolution filters, which shows that a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16–19 weight layers. These findings were the basis of our ImageNet Challenge 2014 submission, where our team secured the first and the second places in the localisation and classification tracks respectively. We also show that our representations generalise well to other datasets, where they achieve state-of-the-art results. We have made our two best-performing ConvNet models publicly available to facilitate further research on the use of deep visual representations in computer vision.

1. The Use of 3×3 Filters



2 layers of 3×3 filters already covered the 5×5 area

By using 2 layers of 3×3 filters, it actually have already covered 5×5 area as in the above figure. **By using 3 layers of 3×3 filters, it actually have already covered 7×7 effective area.** Thus, large-size filters such as 11×11 in AlexNet [3] and 7×7 in ZFNet [2] indeed are not needed. (If

Another reason is that **the number of parameters are fewer.** Suppose there is only 1 filter per layer, 1 layer at input, and exclude the bias:

1 layer of 11×11 filter, number of parameters = **11×11=121**
5 layer of 3×3 filter, number of parameters = **3×3×5=45**
Number of parameters is reduced by 63%

1 layer of 7×7 filter, number of parameters = **7×7=49**
3 layers of 3×3 filters, number of parameters = **3×3×3=27**
Number of parameters is reduced by 45%

By using **1 layer of 5×5 filter**, number of parameters = 5**×5=25**
By using **2 layers of 3×3 filters**, number of parameters = **3×3+3×3=18**
Number of parameters is reduced by 28%

Deep Residual Learning for Image Recognition

Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun
Microsoft Research
{kahe, v-xiangz, v-shren, jiansun}@microsoft.com

Abstract

Deeper neural networks are more difficult to train. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. We provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset we evaluate residual nets with a depth of up to 152 layers—8× deeper than VGG nets [41] but still having lower complexity. An ensemble of these residual nets achieves 3.57% error on the ImageNet test *set. This result won the 1st place on the ILSVRC 2015 classification task. We also present analysis on CIFAR-10 with 100 and 1000 layers.*

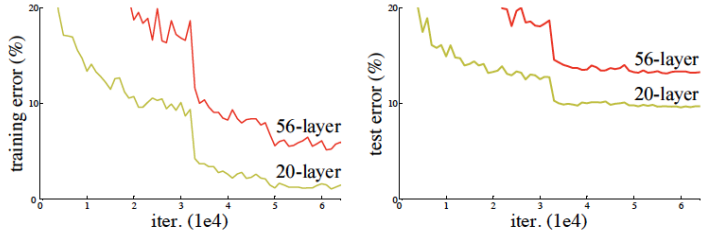


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

greatly benefited from very deep models.

Driven by the significance of depth, a question arises: *Is learning better networks as easy as stacking more layers?* An obstacle to answering this question was the notorious problem of vanishing/exploding gradients [1, 9], which

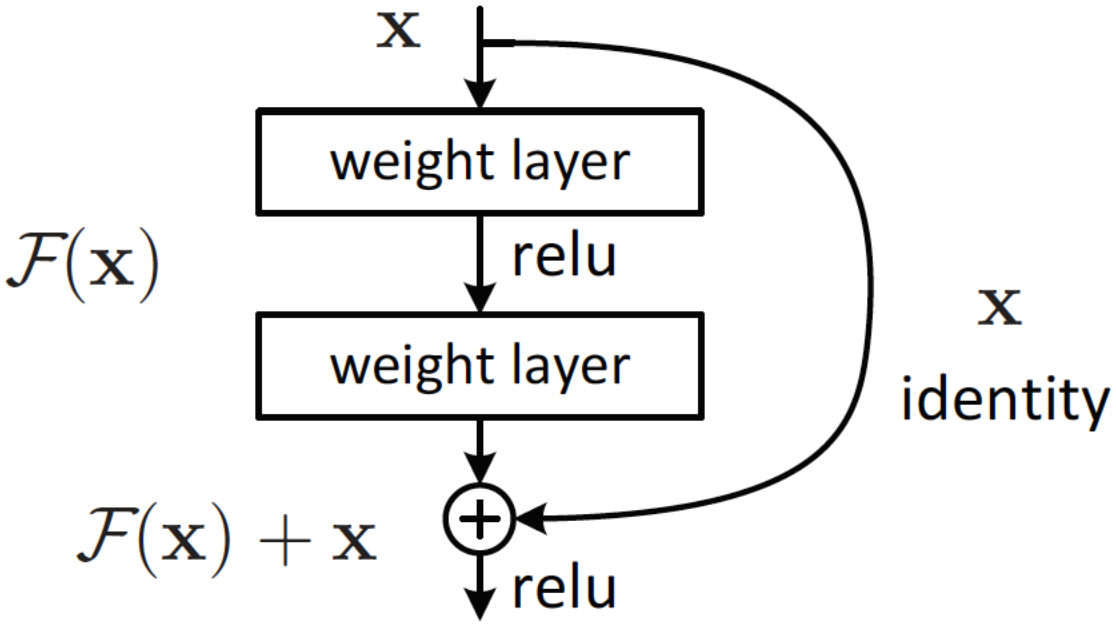


Figure 2. Residual learning: a building block.

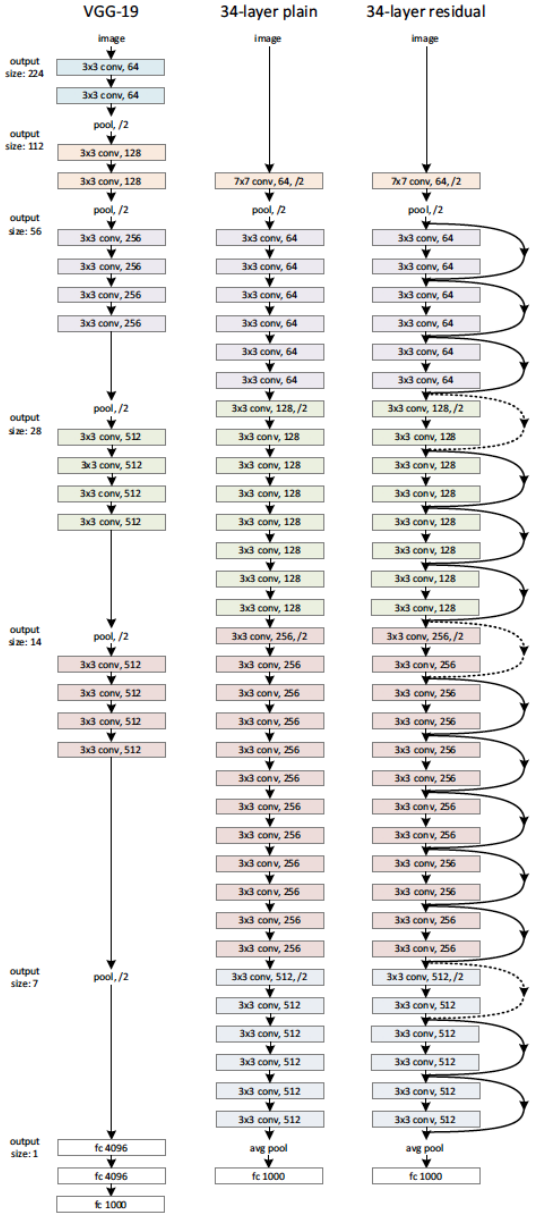


Figure 3. Example network architectures for ImageNet. **Left**: the VGG-19 model [41] (19.6 billion FLOPs) as a reference. **Middle**: a plain network with 34 parameter layers (3.6 billion FLOPs). **Right**: a residual network with 34 parameter layers (3.6 billion FLOPs). The dotted shortcuts increase dimensions. **Table 1** shows more details and other variants.

Now we can train deeper networks!

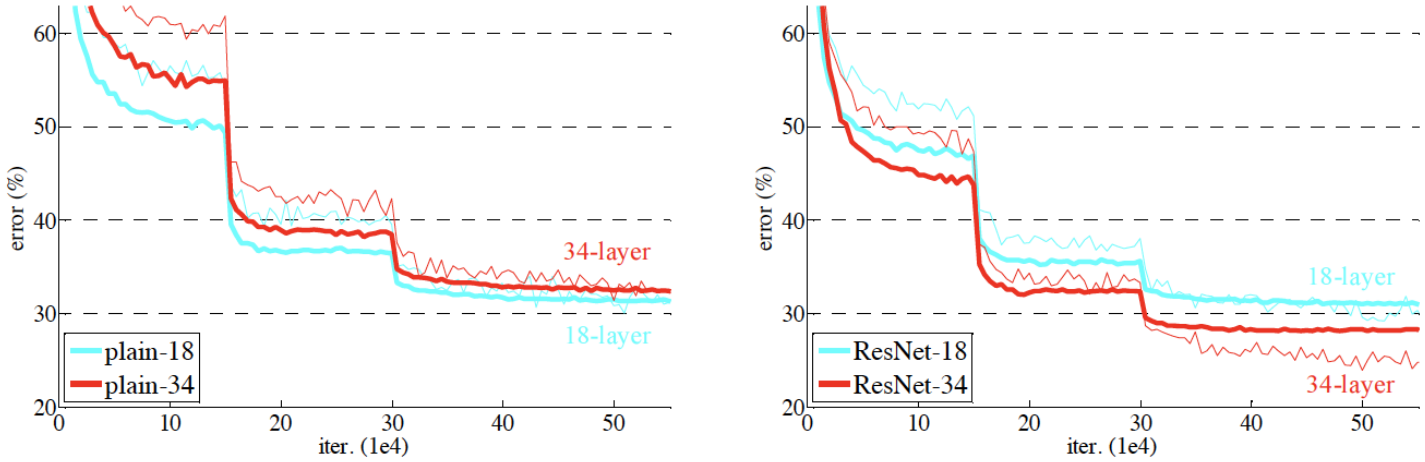


Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

	plain	ResNet
18 layers	27.94	27.88
34 layers	28.54	**25.03**

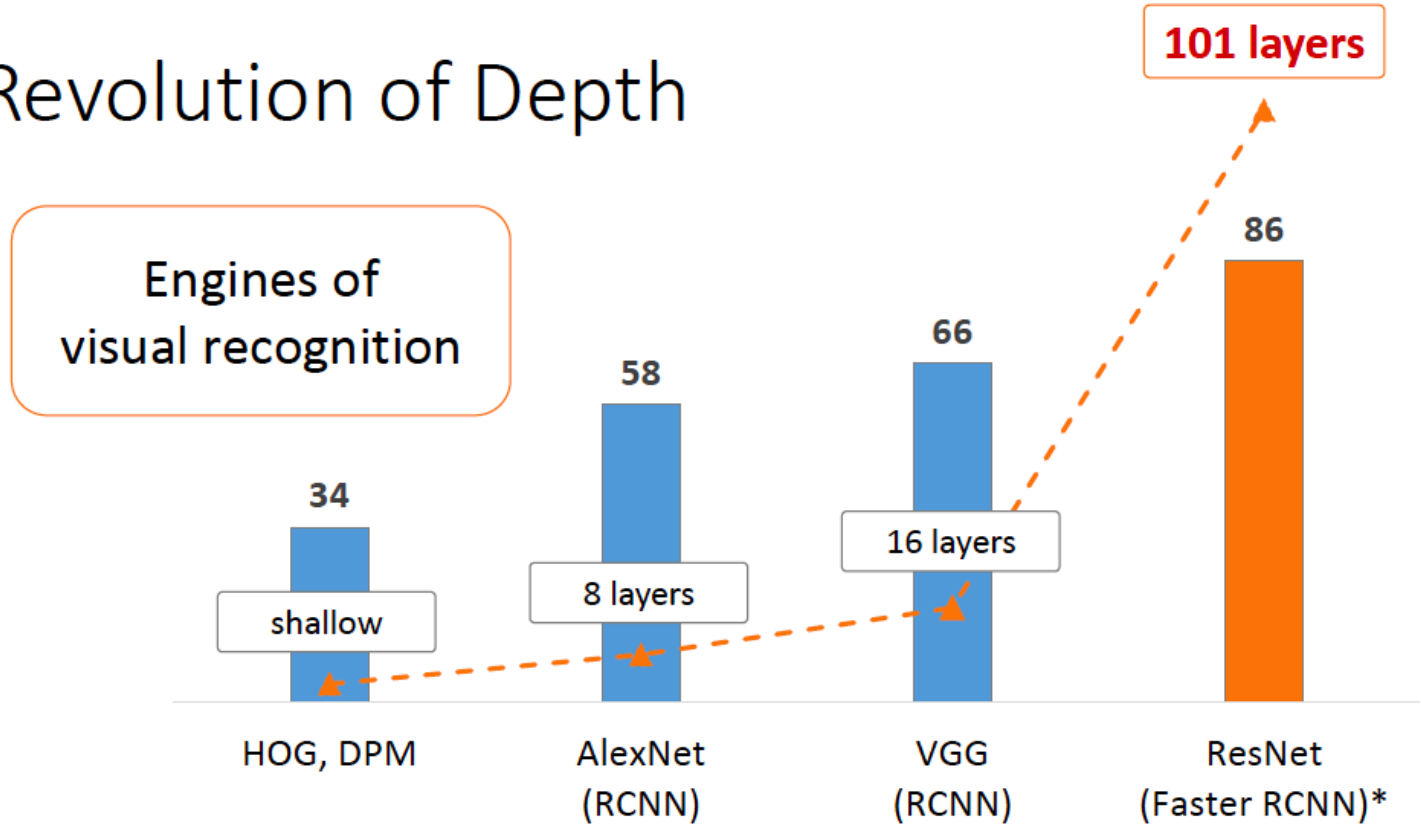
Table 2. Top-1 error (%, 10-crop testing) on ImageNet validation. Here the ResNets have no extra parameter compared to their plain counterparts. Fig. 4 shows the training procedures.

reducing of the training error[3]. The reason for such optimization difficulties will be studied in the future.

Residual Networks. Next we evaluate 18-layer and 34-layer residual nets (*ResNets*). The baseline architectures are the same as the above plain nets, expect that a shortcut connection is added to each pair of 3×3 filters as in Fig. 3

Microsoft Research

Revolution of Depth



PASCAL VOC 2007 **Object Detection** mAP (%)

ICCV15 International Conference on Computer Vision

*w/ other improvements & more data

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". arXiv 2015.

Training R-CNN

Bounding-box labeled detection data is scarce

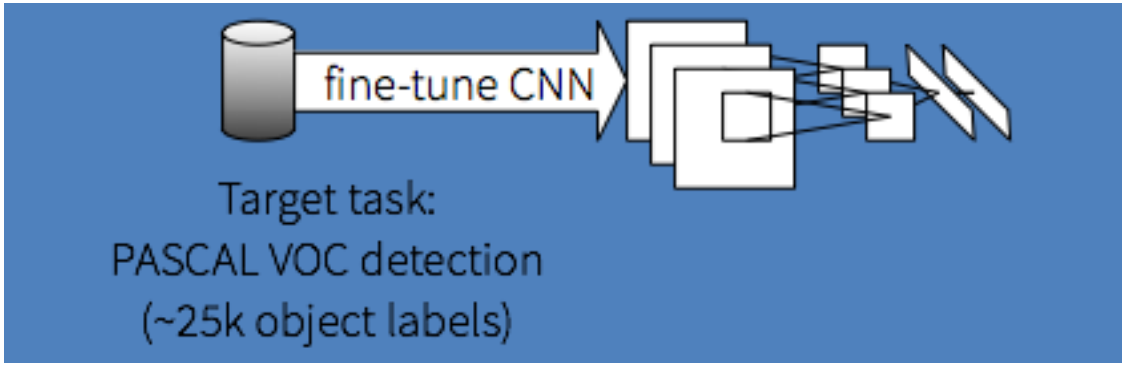
Key insight:

Use *supervised* pre-training on a data-rich auxiliary task and *transfer* to detection

R-CNN training: Step 2

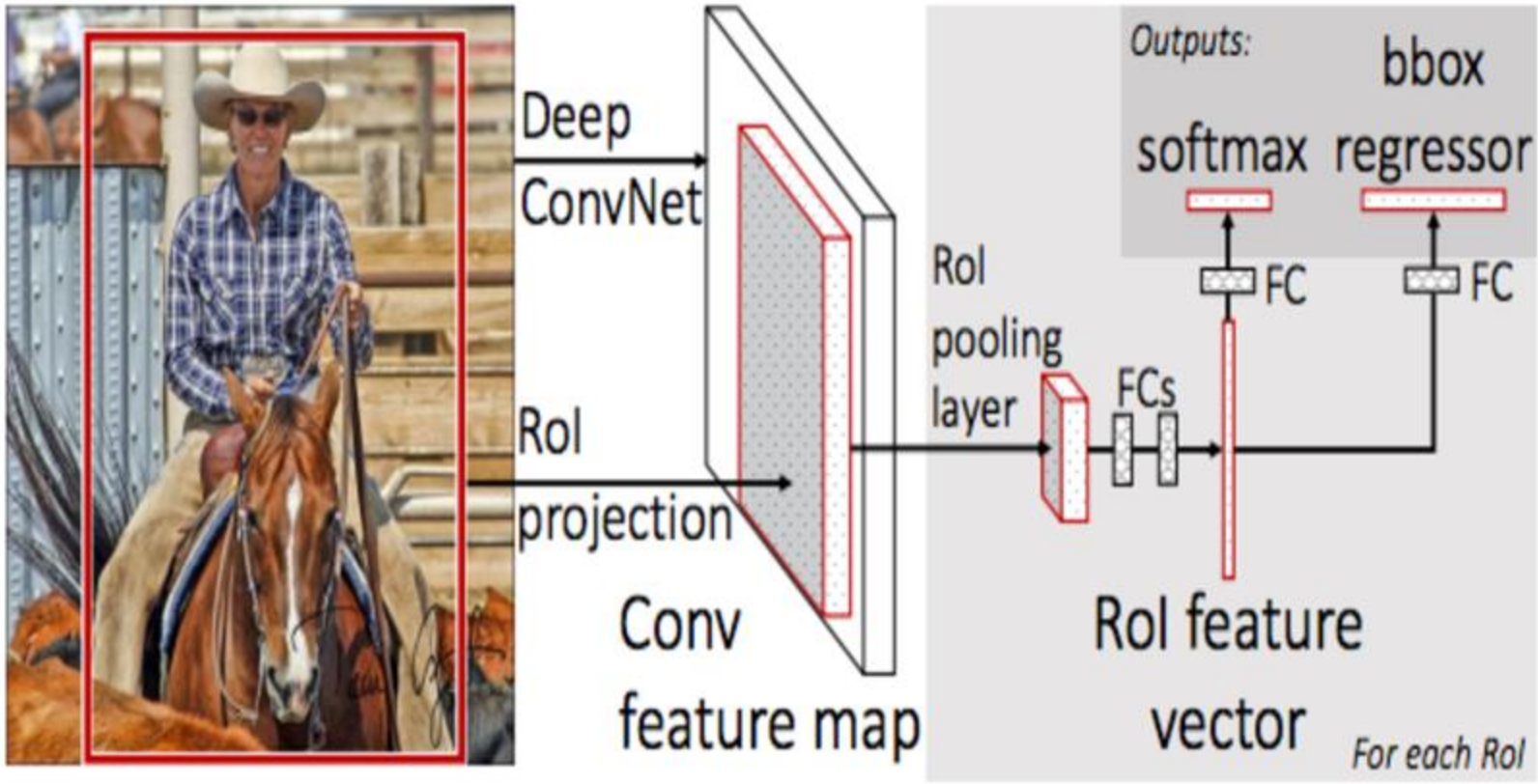
Fine-tune the CNN for detection

Transfer the representation learned for ILSVRC classification to PASCAL (or ImageNet detection)



Fast R-CNN (Girshick, 2015)

R-CNN with SPP features, no need to warp individual windows



There is also Faster R-CNN
which doesn't require external proposals

Learning Transferable Visual Models From Natural Language Supervision

Alec Radford [*1] Jong Wook Kim [*1] Chris Hallacy [1] Aditya Ramesh [1] Gabriel Goh [1] Sandhini Agarwal [1]
Girish Sastry [1] Amanda Askell [1] Pamela Mishkin [1] Jack Clark [1] Gretchen Krueger [1] Ilya Sutskever [1]

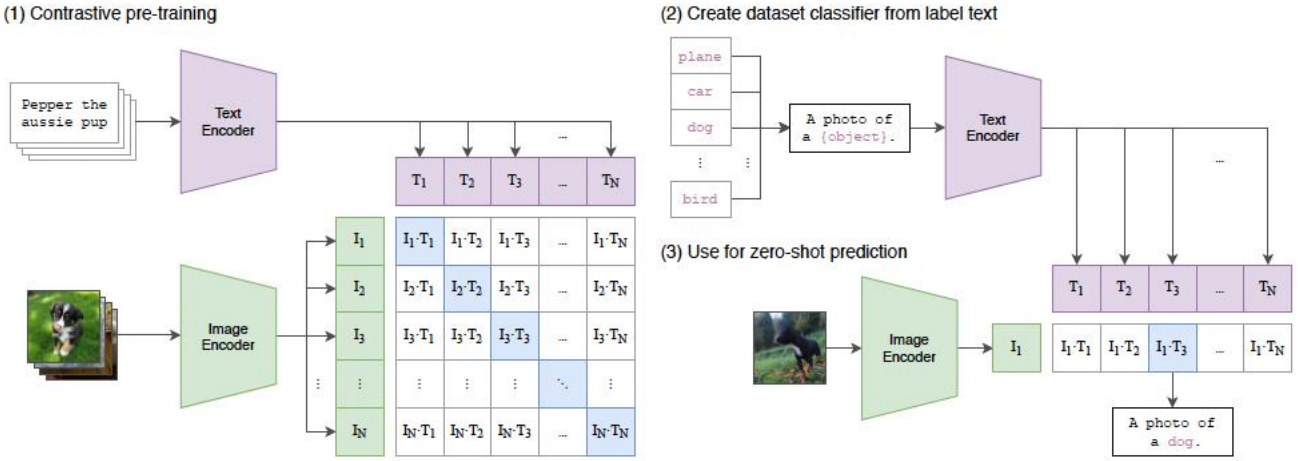


Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

CLIP : Contrastive Language Image Pretraining

Sigmoid Loss for Language Image Pre-Training

Xiaohua Zhai* Basil Mustafa Alexander Kolesnikov Lucas Beyer*
Google DeepMind, Zürich, Switzerland
{xzhai, basilm, akolesnikov, lbeyer}@google.com

Instead of the softmax-based contrastive loss, we propose a simpler alternative that does not require computing global normalization factors. The sigmoid-based loss processes every image-text pair independently, effectively turning the learning problem into the standard binary classification on the dataset of all pair combinations, with a positive labels for the matching pairs (I_i, T_i) and negative labels for all other pairs $(I_i, T_{j\neq i})$. It is defined as follows:

$$-\frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\sum_{j=1}^{|\mathcal{B}|}\underbrace{\log\frac{1}{1+e^{z_{ij}(-t\mathbf{x}_i\cdot\mathbf{y}_j+b)}}}_{\mathcal{L}_{ij}}$$

where z_{ij} is the label for a given image and text input, which equals 1 if they are paired and -1 otherwise. Note that at

The Past, Present, and Future of Object Detection

Ross Girshick (FAIR)

in collaboration with

Kaiming He (FAIR), Georgia Gkioxari (FAIR), Tsung-Yi Lin (Google Brain),
Bharath Hariharan (Cornell), and Piotr Dollár (FAIR)

Facebook AI Research

@ UC Berkeley Oct 2, 2017

COCO

Dataset

Images

COCO

Dataset

Images

COCO

Annotations

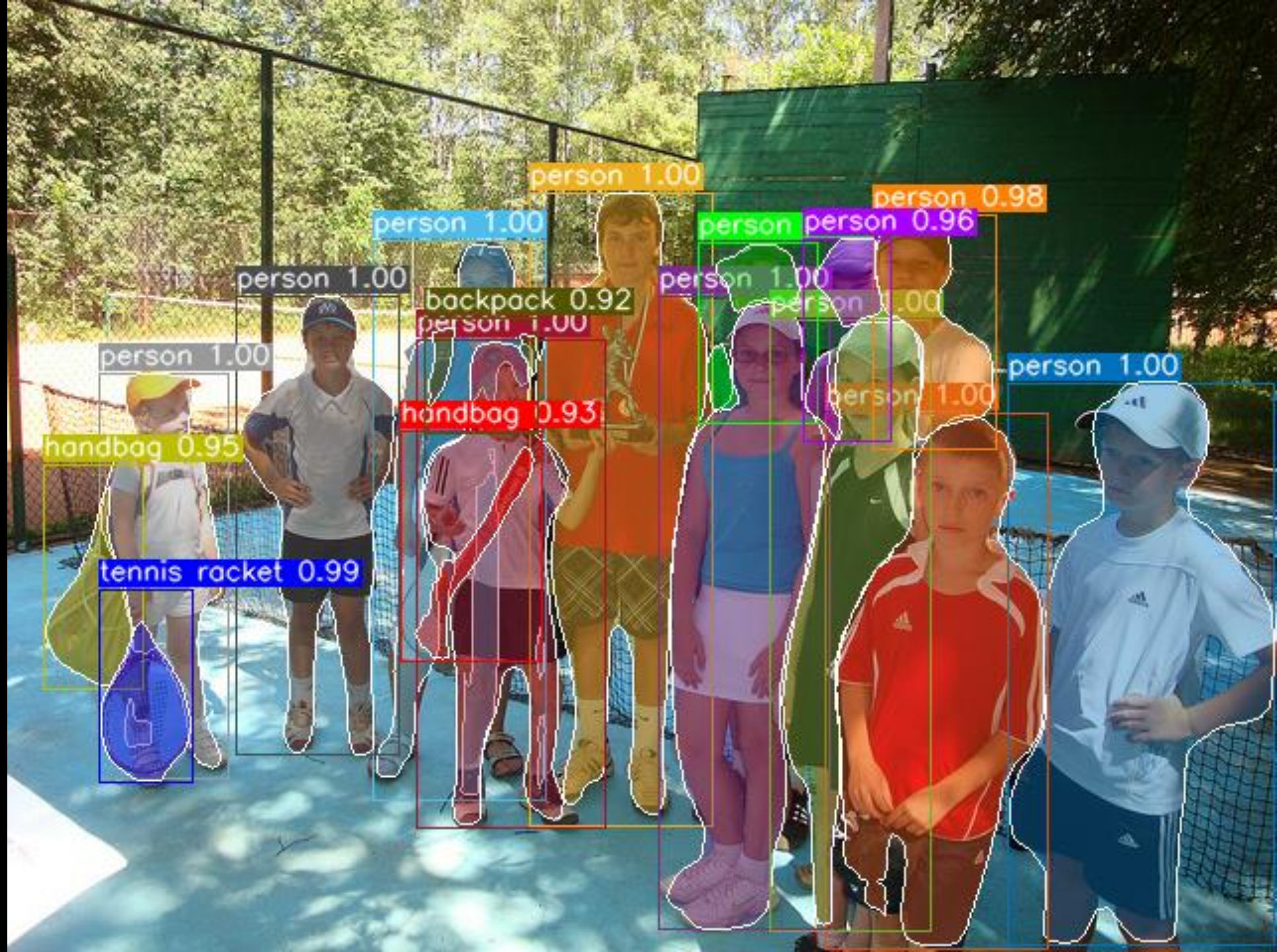
COCO

Annotations

Example

Mask R-CNN

Output on Unseen Images



Example

Mask R-CNN

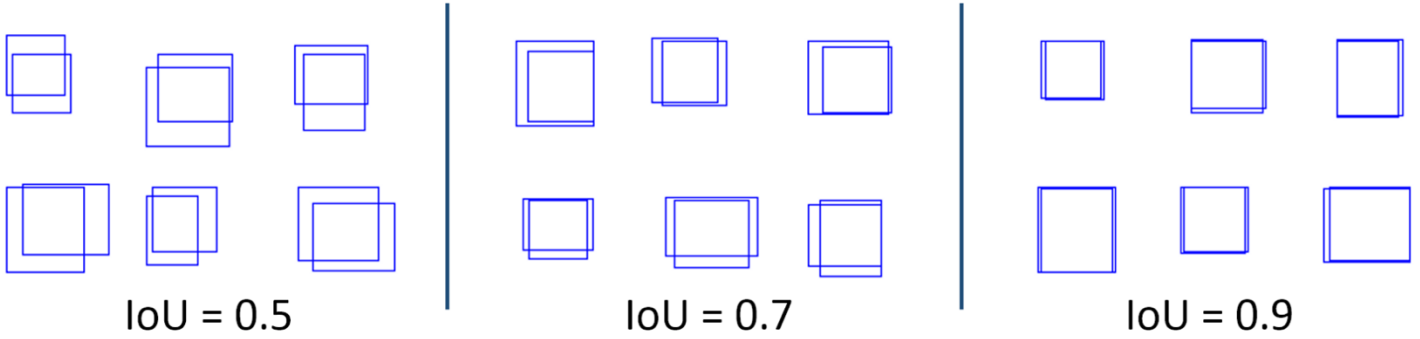
Output on Unseen Images

COCO Object Detection Average Precision (%)

Area under a detector's precision-recall curve, averaged over…

- Object categories
- True positive overlap requirement (IoU from 0.5 to 0.95; see below)

boxes



masks

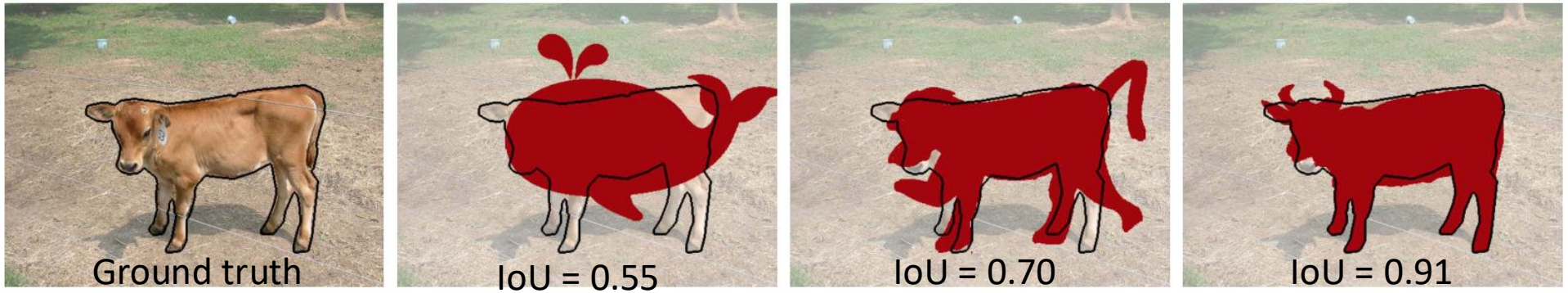
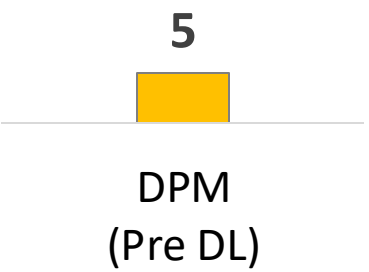


Figure credits: Dollár and Zitnick (top), Krähenbühl and Kulton (bottom)

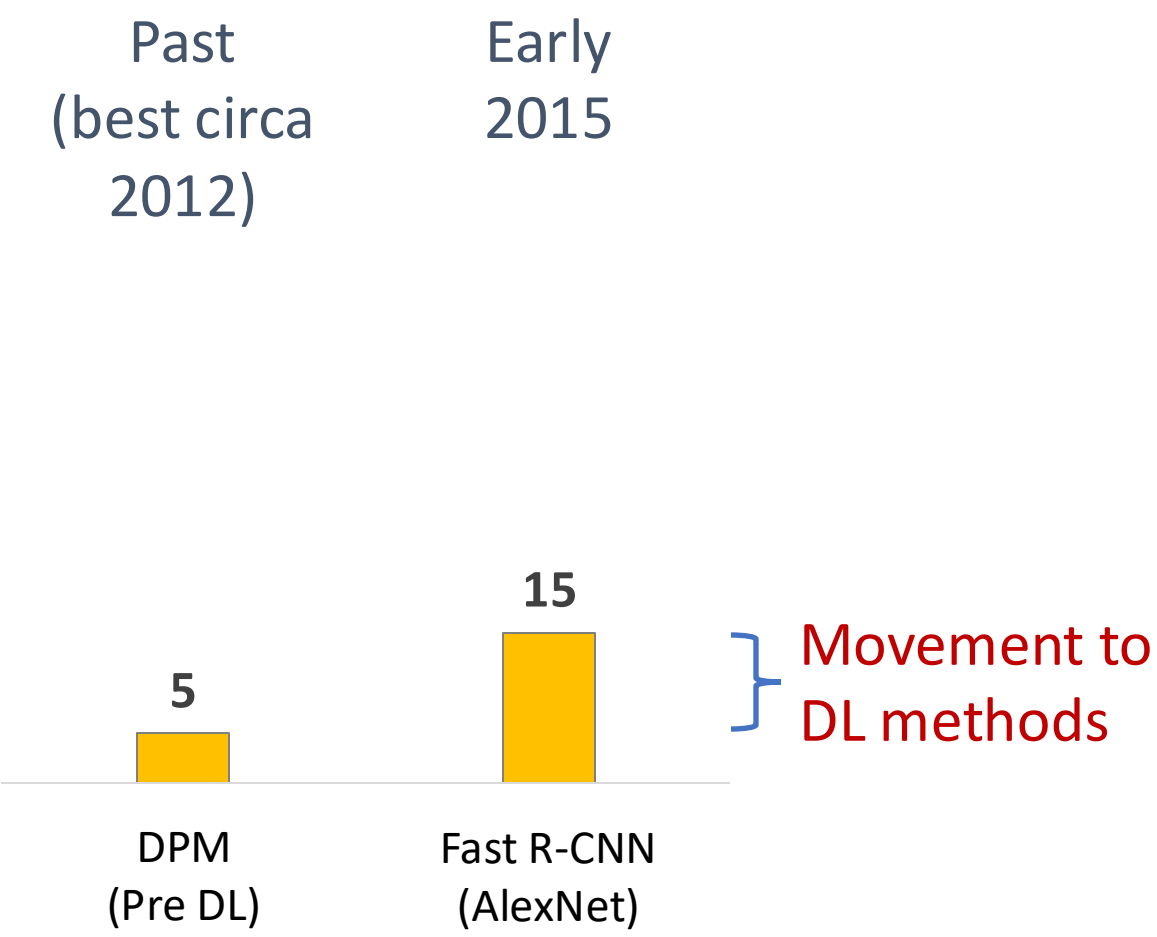
COCO Object Detection Average Precision (%)

Past
(best circa
2012)



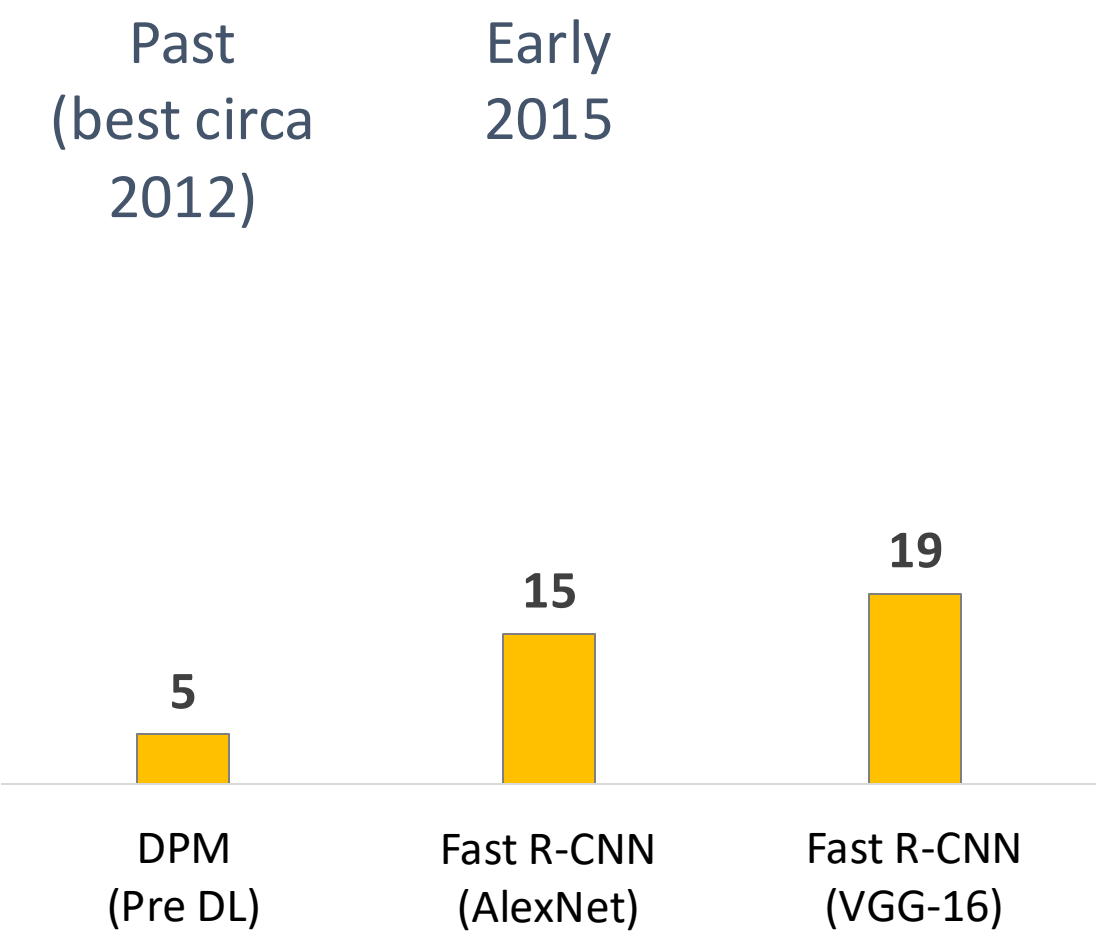
Felzenszwalb, Girshick, McAllester, Ramanan. Object Detection with Discriminatively Trained Part Based Models. PAMI 2010.

COCO Object Detection Average Precision (%)



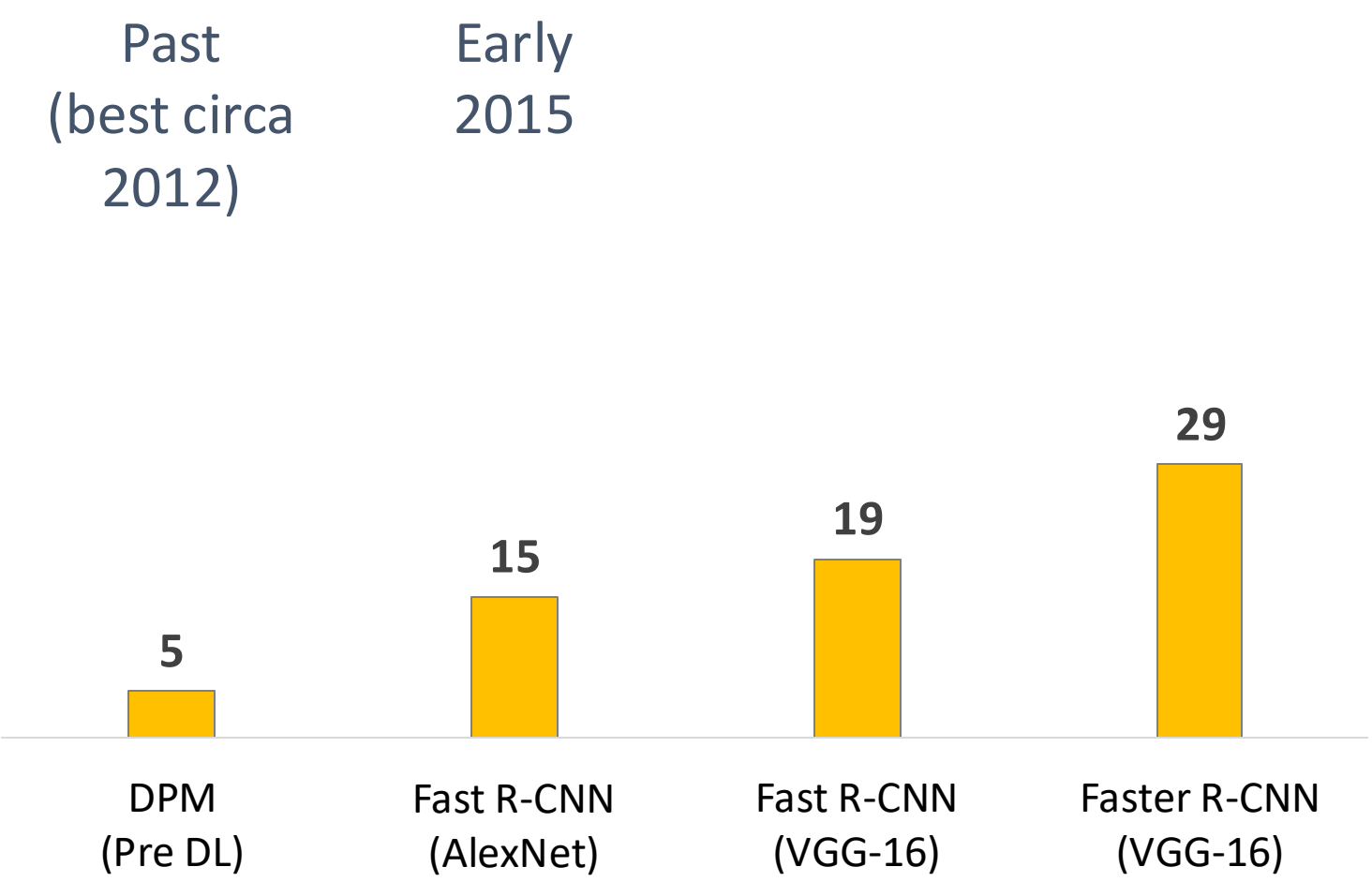
Girshick. Fast R-CNN. ICCV 2015.

COCO Object Detection Average Precision (%)



Girshick. Fast R-CNN. ICCV 2015.

COCO Object Detection Average Precision (%)

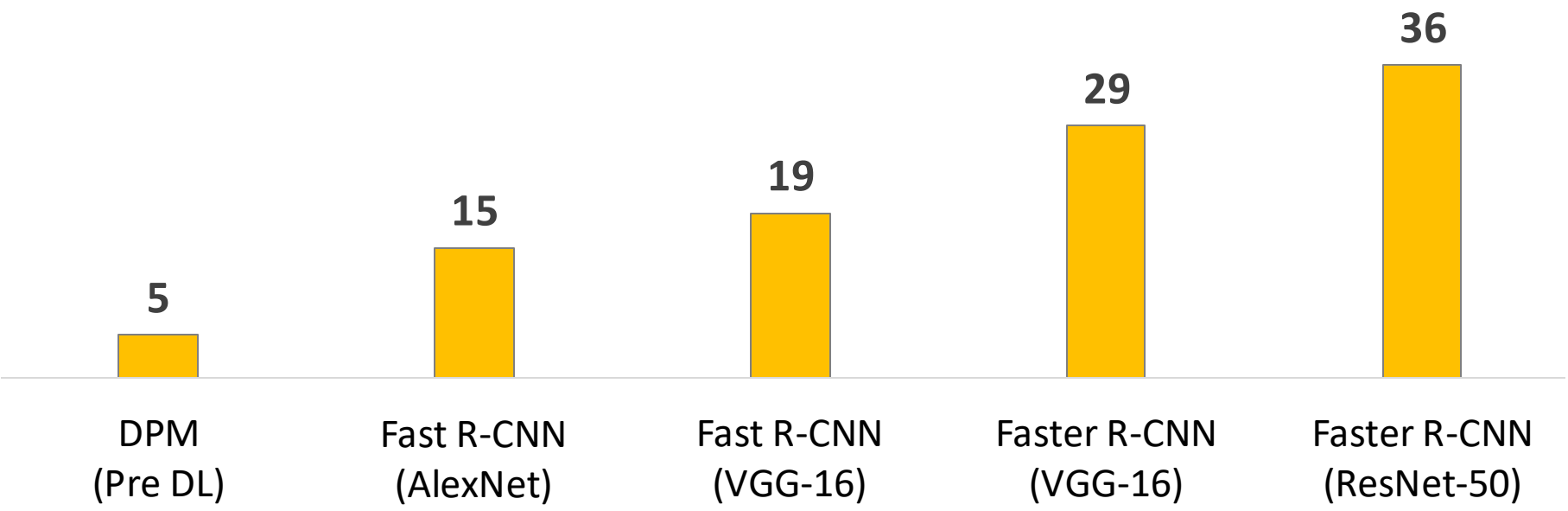


Ren, He, Girshick, Sun. Faster R-CNN: Towards Real-Time Object Detection. NIPS 2015.

COCO Object Detection Average Precision (%)

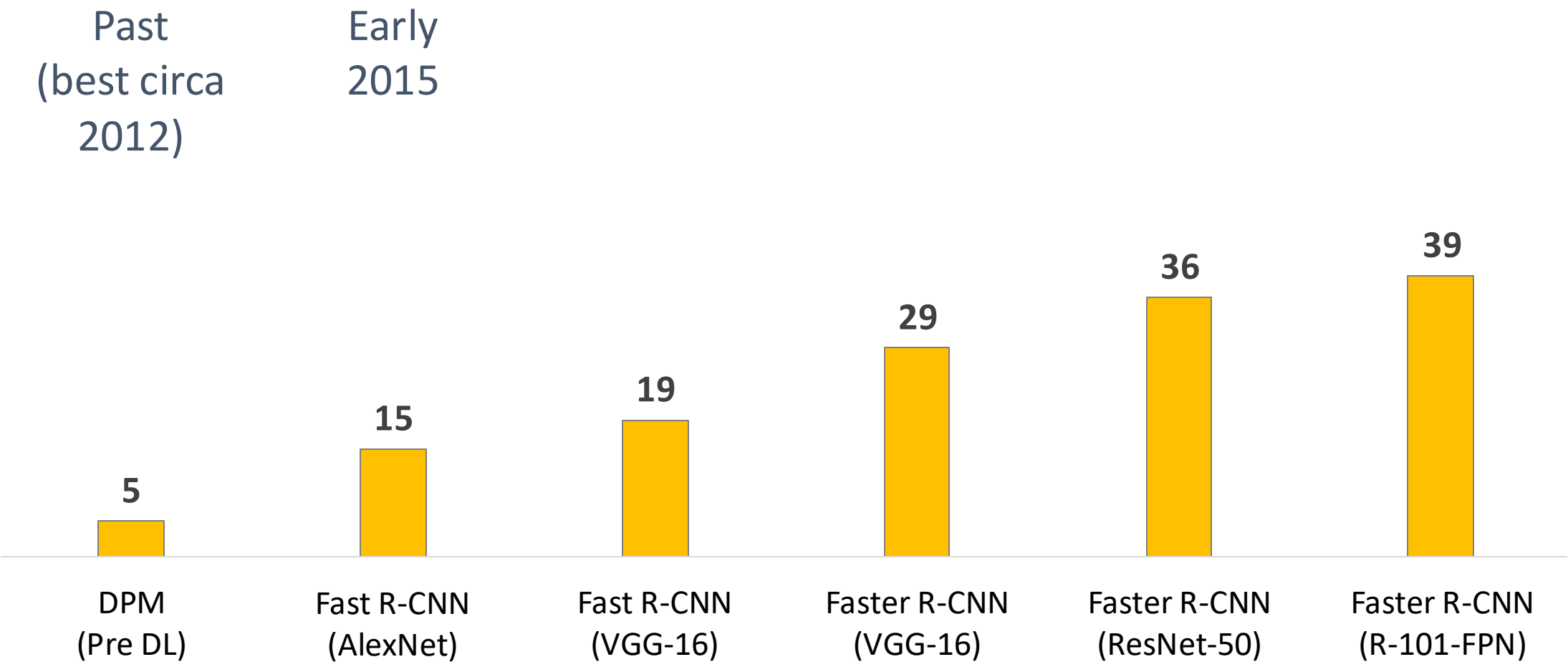
Past (best circa 2012)

Early 2015



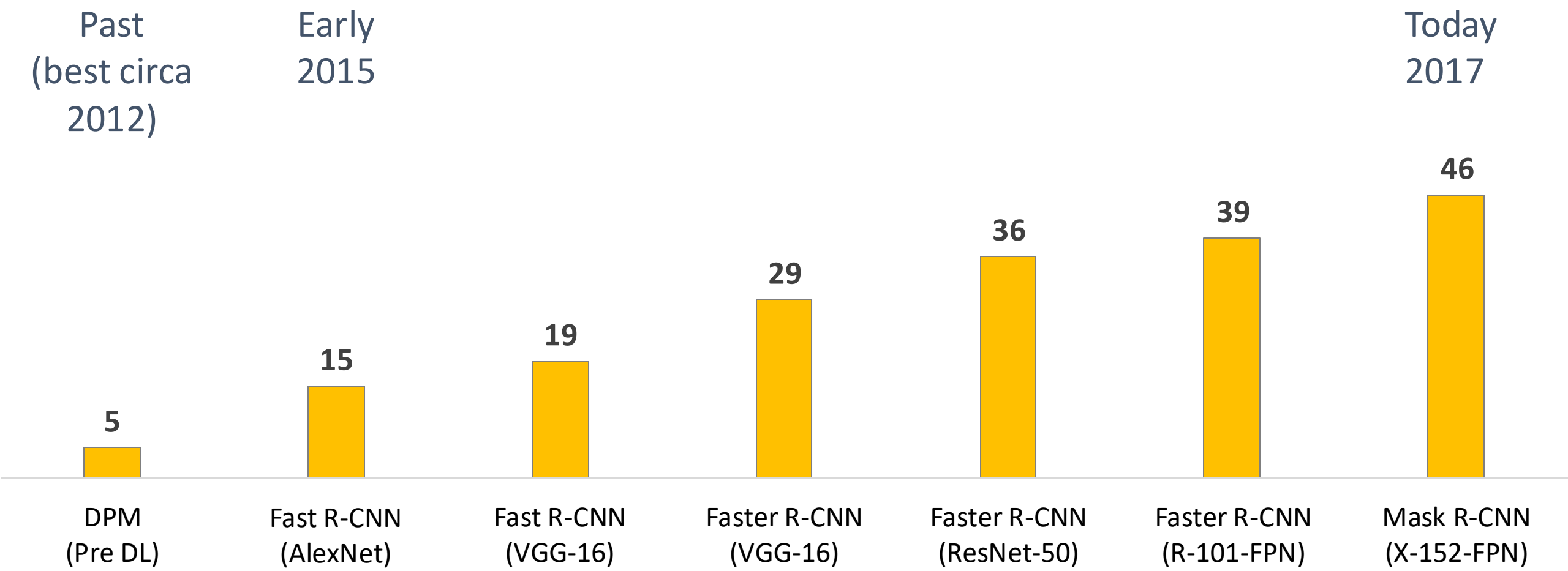
Ren, He, Girshick, Sun. Faster R-CNN: Towards Real-Time Object Detection. NIPS 2015.

COCO Object Detection Average Precision (%)



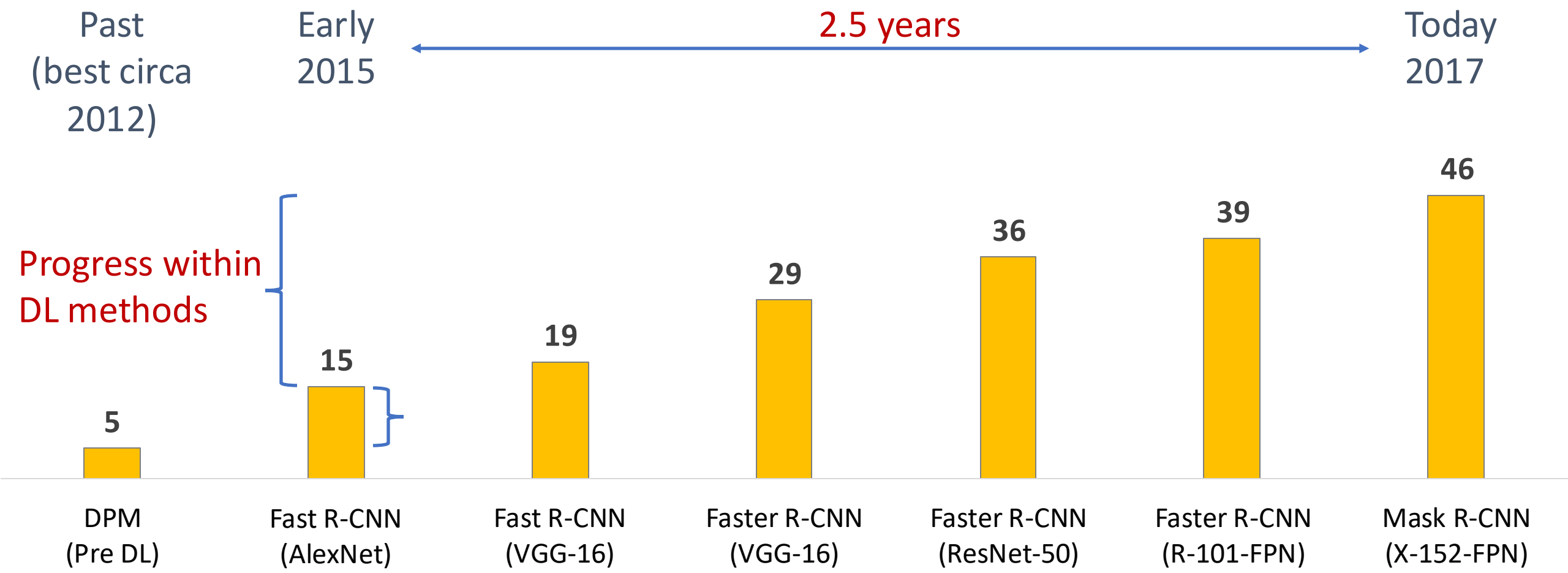
Lin et al. Feature Pyramid Networks. CVPR 2017.

COCO Object Detection Average Precision (%)

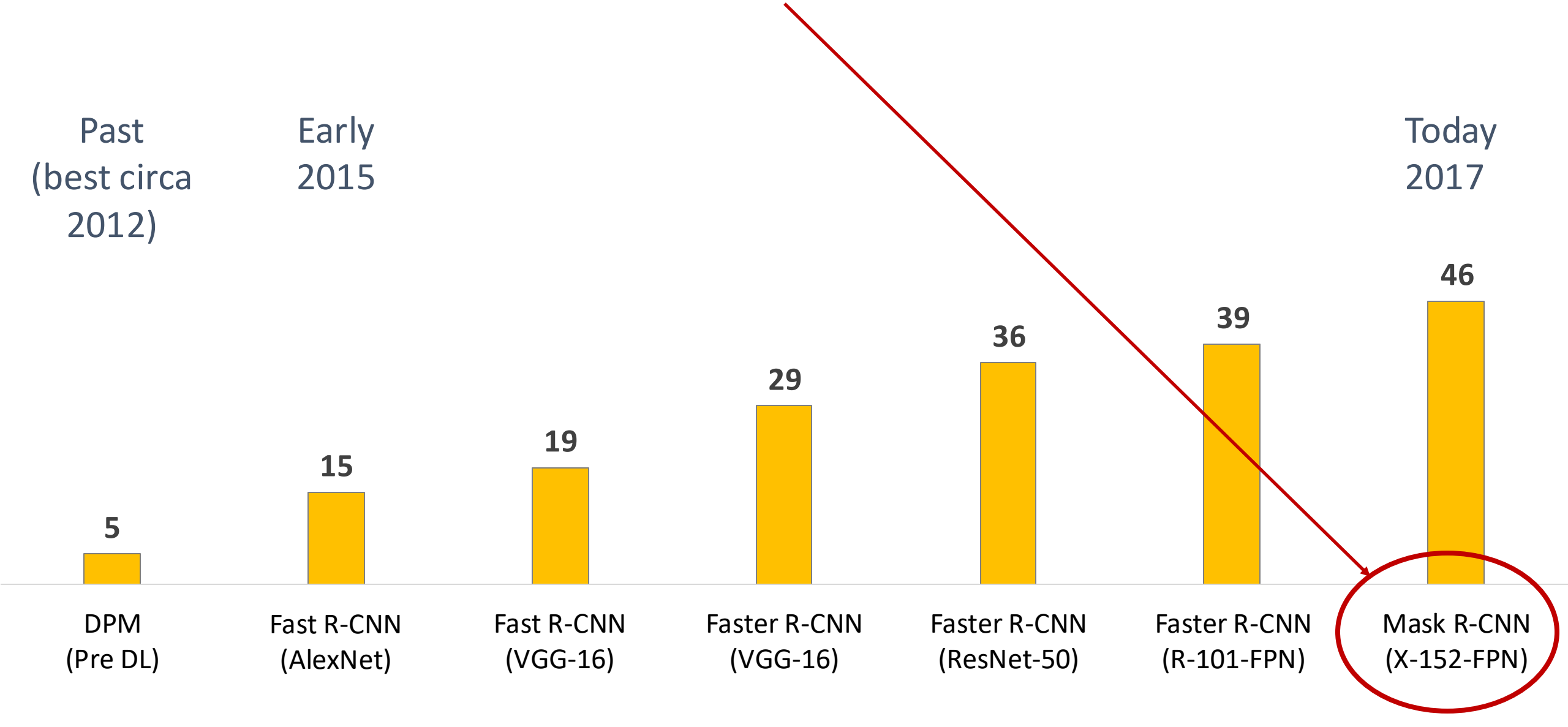


He, Gkioxari, Dollár, Girshick. Mask R-CNN. ICCV 2017.

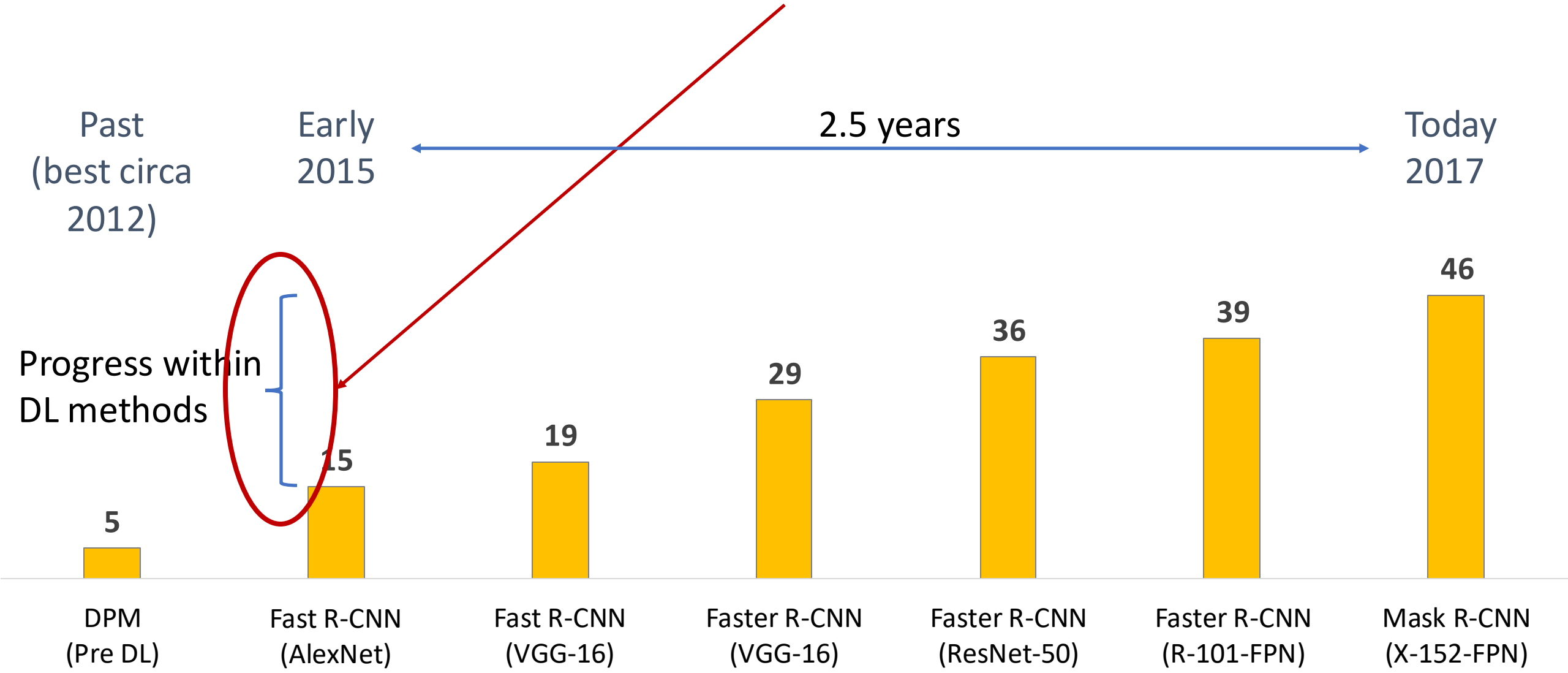
COCO Object Detection Average Precision (%)



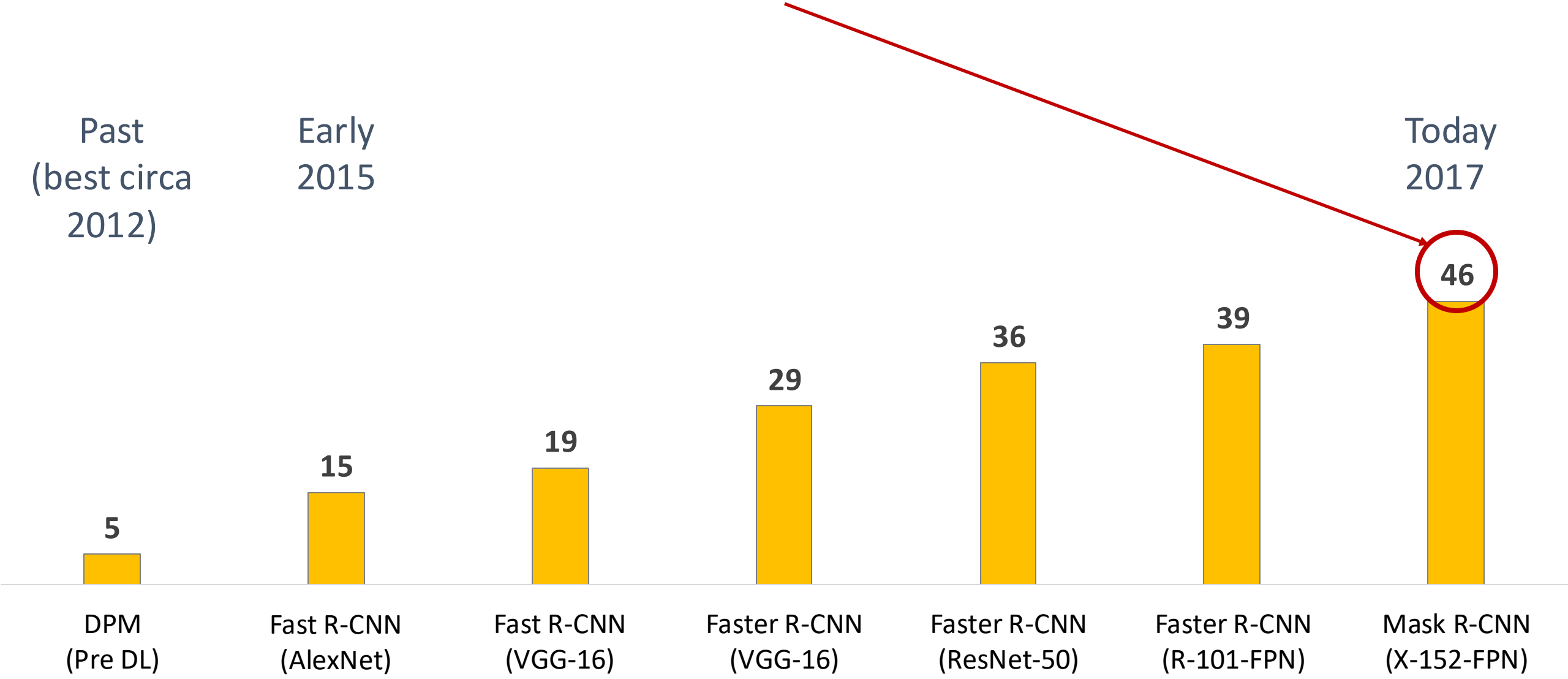
1. What is this?



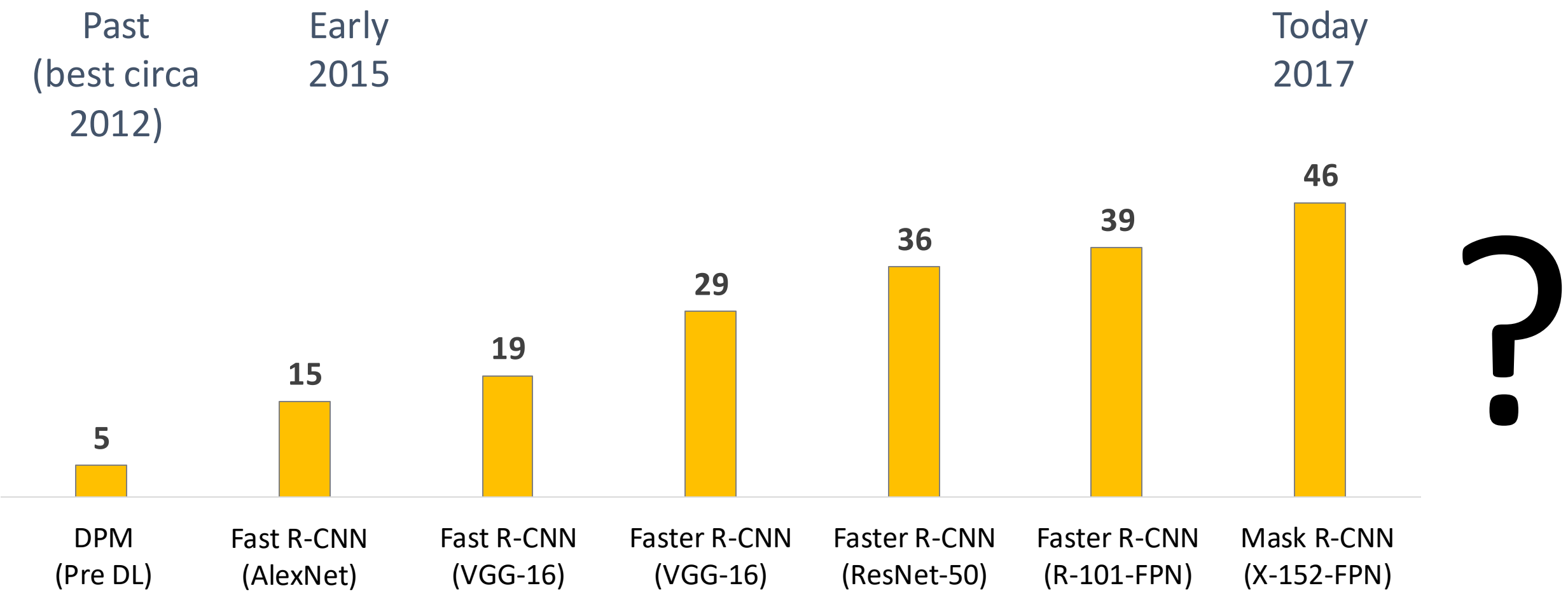
2. What made this possible?



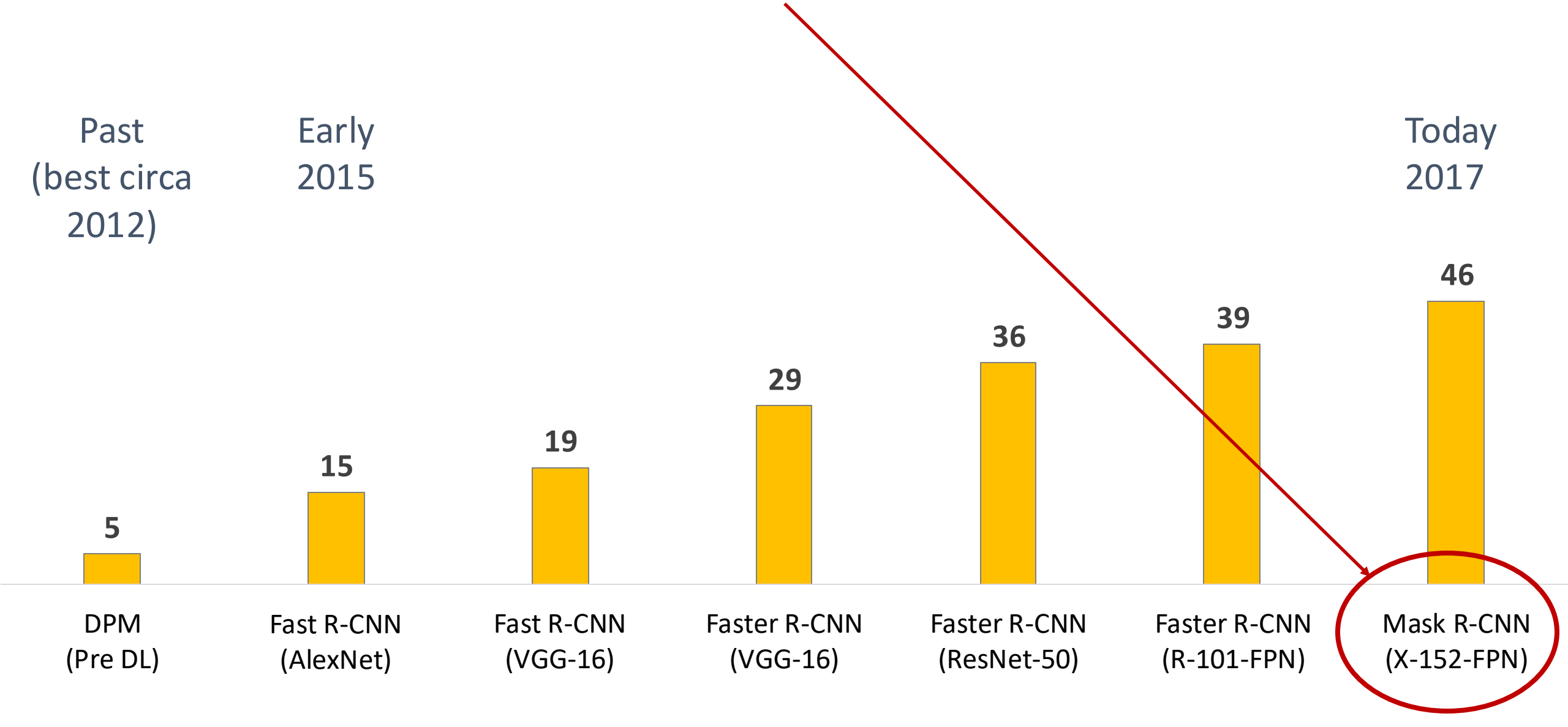
3a. How good is this?



3b. What's next?



Let's start: What is this?

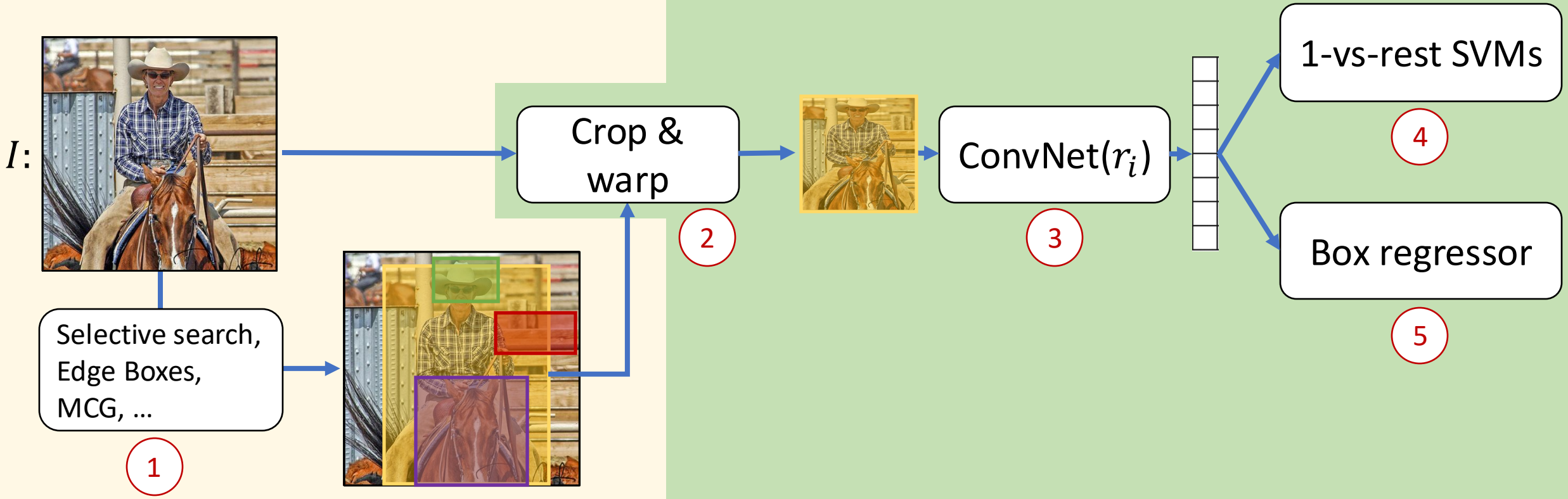


Girshick, Donahue, Darrell, Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. CVPR 2014.

"Slow" R-CNN

Per-image computation

Per-region computation for each $r_i \in r(I)$

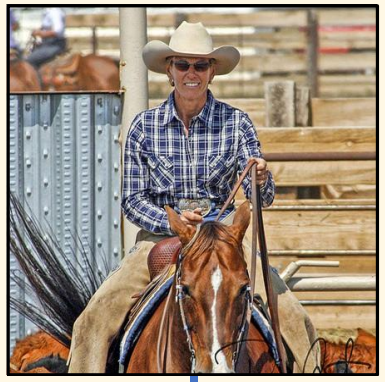


"Slow" R-CNN

Per-image computation

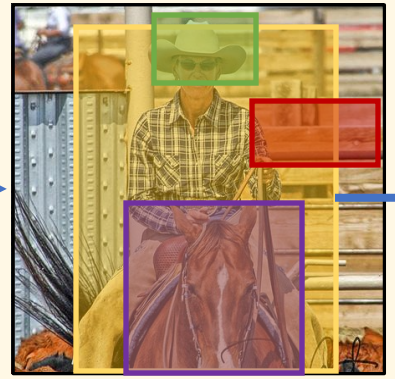
Per-region computation for each $r_i \in r(I)$

I:



Selective search, Edge Boxes, MCG, …

1



Crop & warp

2

ConvNet(r_i)

3

1-vs-rest SVMs

4

Box regressor

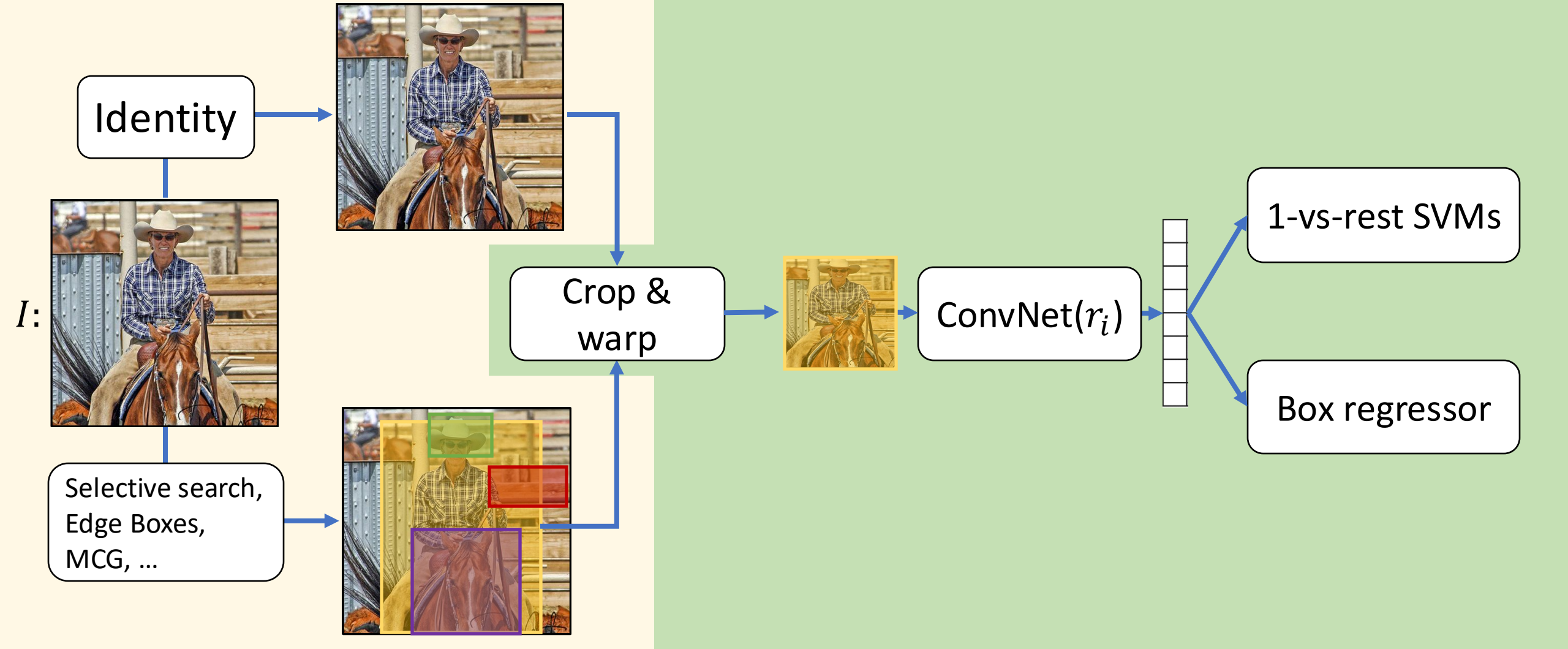
5

Very heavy per-region computation
E.g., 2000 full network evaluations

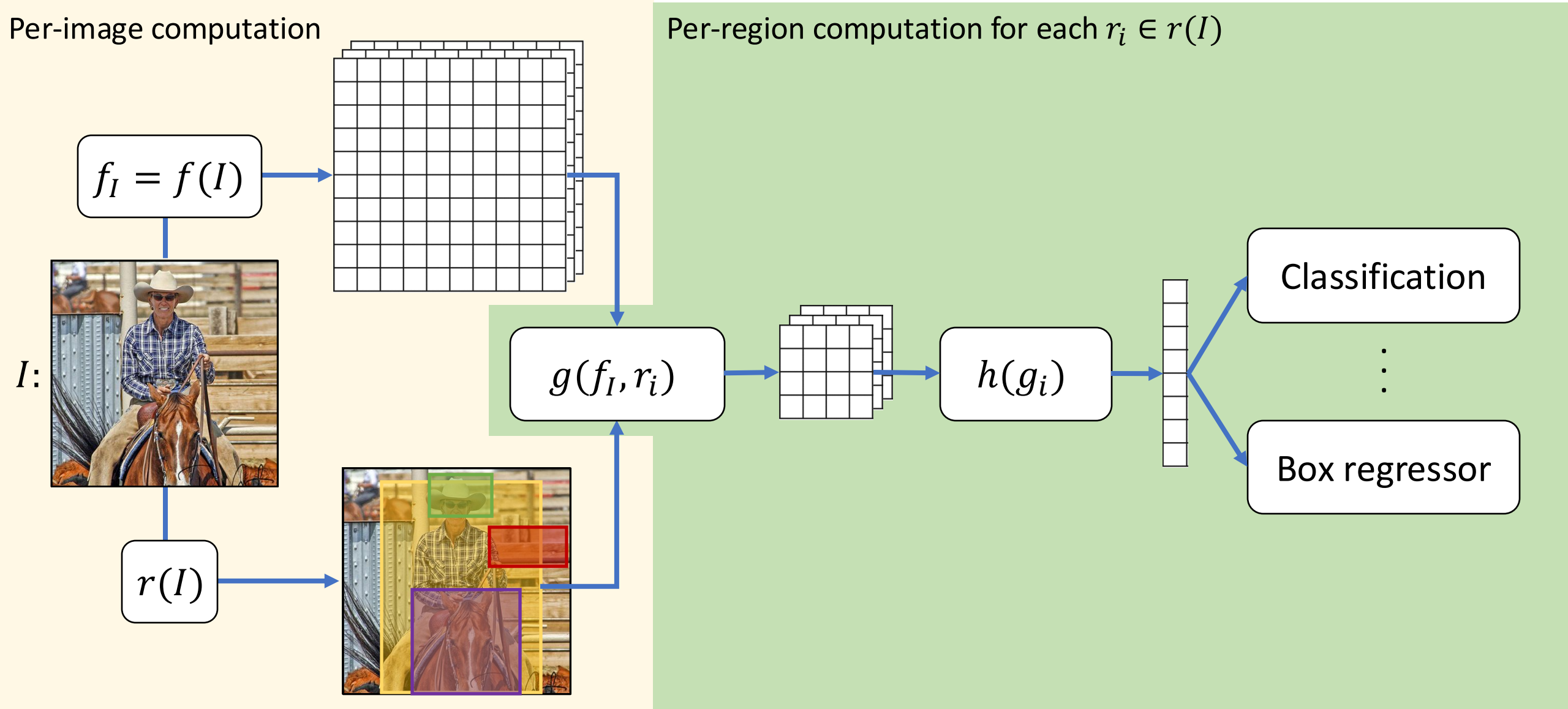
"Slow" R-CNN

Per-image computation

Per-region computation for each $r_i \in r(I)$



Generalized R-CNN Approach to Detection



Girshick. Fast R-CNN. ICCV 2015.

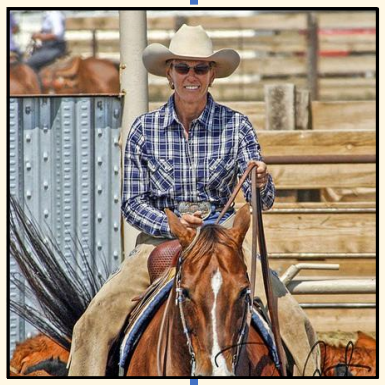
Fast R-CNN

Per-image computation

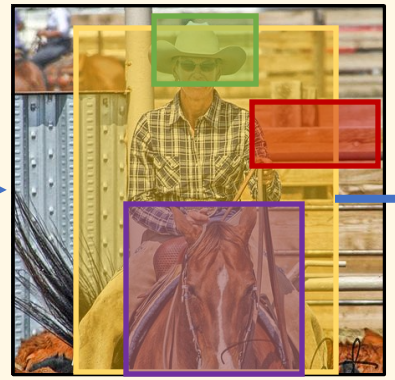
Per-region computation for each $r_i \in r(I)$

FCN(I)

I:



Selective search,
Edge Boxes,
MCG, …



RoIPool

MLP

Softmax clf.

Box regressor

Lightweight per-region computation

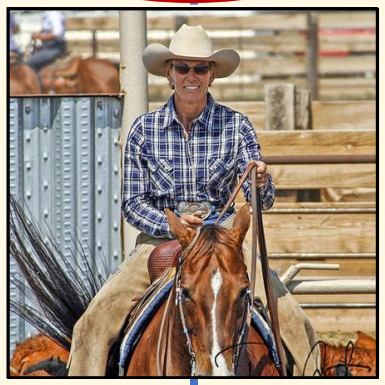
Fast R-CNN

Per-image computation

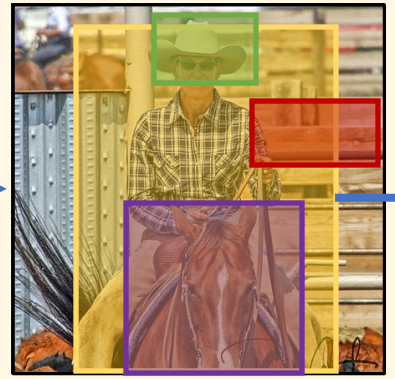
Per-region computation for each $r_i \in r(I)$

FCN(I)

I:



Selective search,
Edge Boxes,
MCG, …



RoIPool

MLP

Softmax clf.

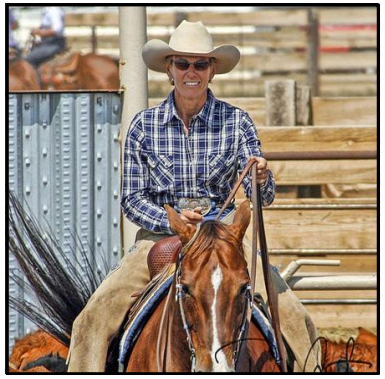
Box regressor

Whole-image FCN

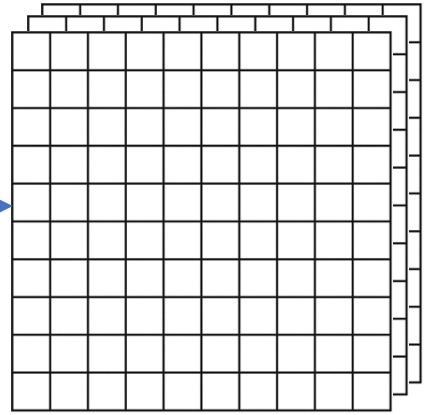
Use any standard ConvNet as the "backbone architecture"

- AlexNet, VGG, ResNet, Inception, Inception-ResNet, ResNeXt, DenseNet, …
- Use the first *N* layers with spatial extent (e.g., up to "conv5")

I:

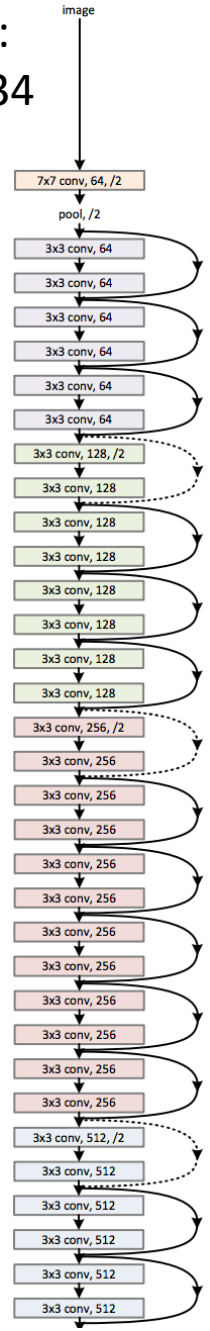


FCN(I)

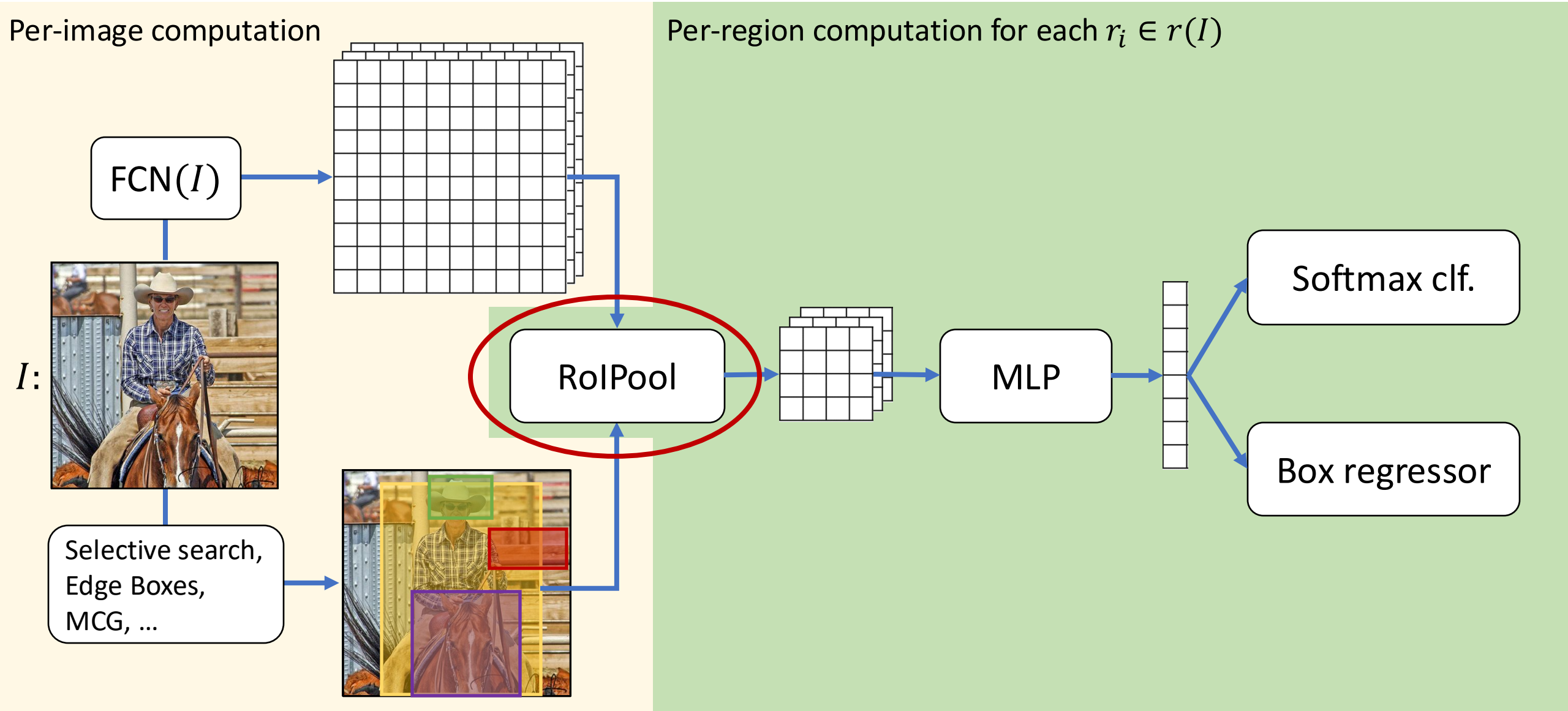


Example feature map dimensions:
(512, H/16, W/16)

Example:
ResNet-34

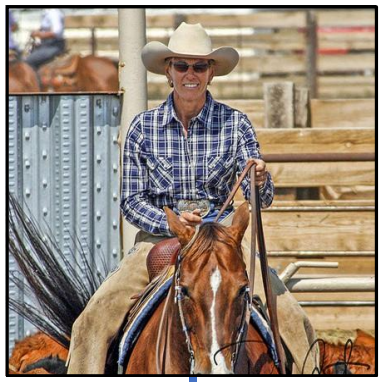


Fast R-CNN

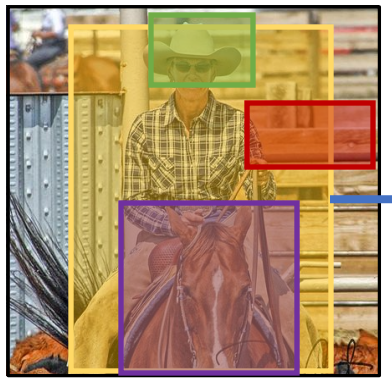

Per-image computation
Per-region computation for each $r_i \in r(I)$
FCN(I)
I:
Selective search,
Edge Boxes,
MCG, …
RoIPool
MLP
Softmax clf.
Box regressor

RoIPool (on each Proposal)

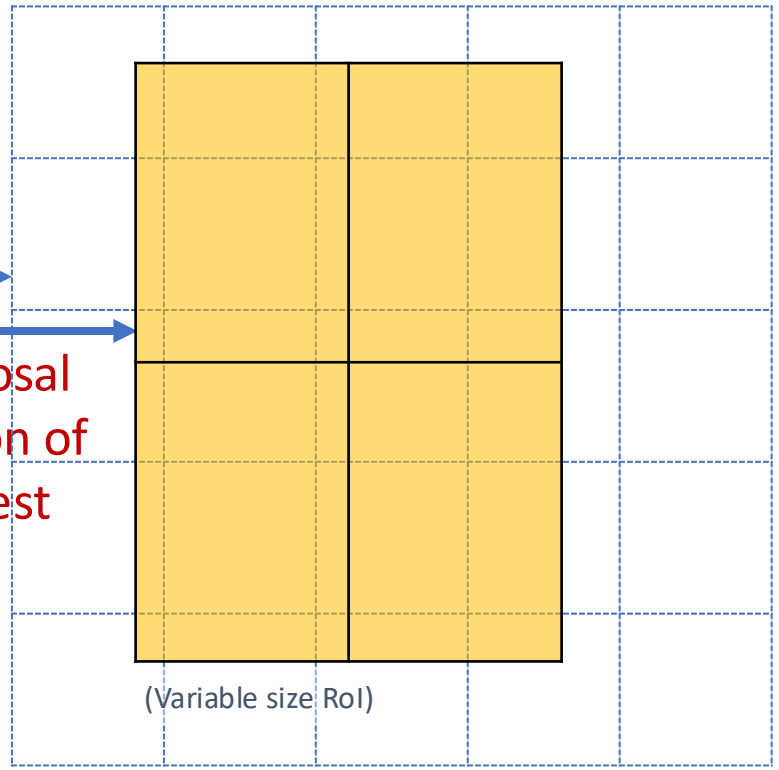
Transform arbitrary size proposal into a fixed-dimensional representation (e.g., 2x2)



f_I = FCN(I)

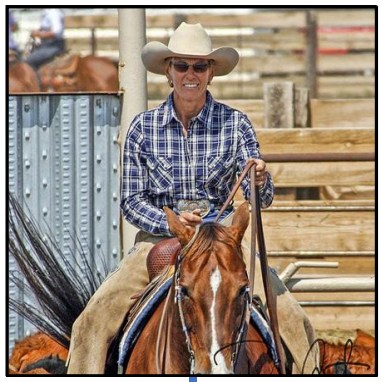


Proposal Region of Interest (RoI)

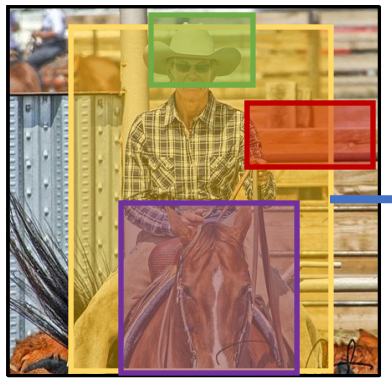


RoIPool (on each Proposal)

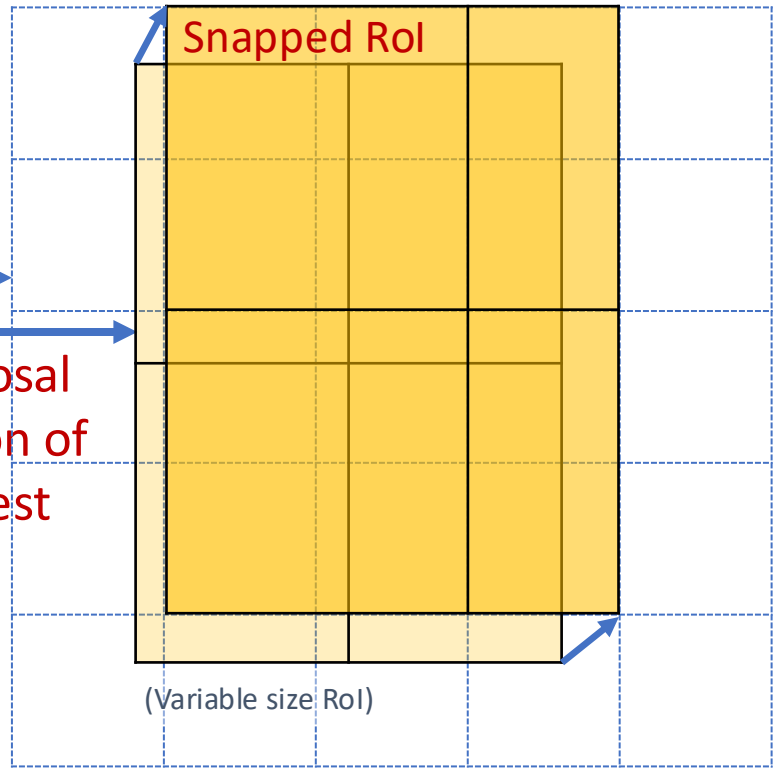
Transform arbitrary size proposal into a fixed-dimensional representation (e.g., 2x2)



f_I = FCN(I)

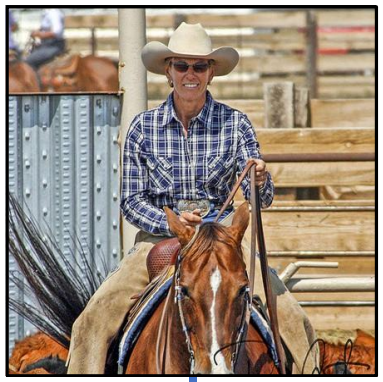


Proposal Region of Interest (RoI)

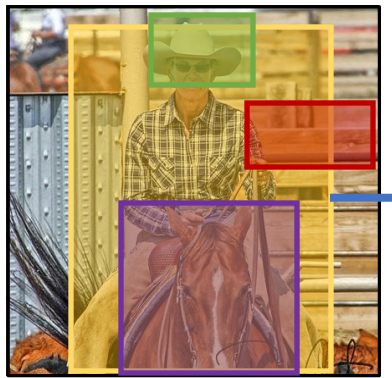


RoIPool (on each Proposal)

Transform arbitrary size proposal into a fixed-dimensional representation (e.g., 2x2)



f_I = FCN(I)



Proposal Region of Interest (RoI)

Snapped RoI

(Variable size RoI)

RoIPool transform

(Fixed dimensional representation)

Feature value is max over input cells

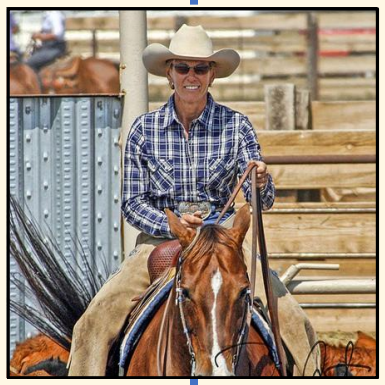
Fast R-CNN

Per-image computation

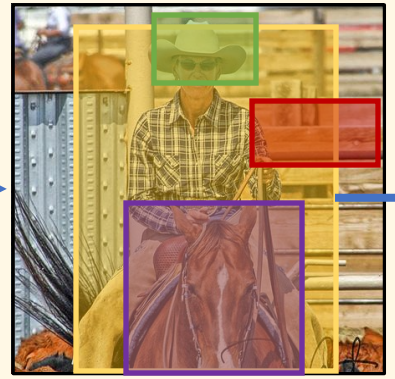
Per-region computation for each $r_i \in r(I)$

FCN(I)

I:



Selective search,
Edge Boxes,
MCG, …



RoIPool

MLP

Softmax clf.

Box regressor

Region proposals have very poor recall
(ok for PASCAL VOC, major bottleneck for COCO)
Also, they can be slow

Ren, He, Girshick, Sun. Faster R-CNN:
Towards Real-Time Object Detection. NIPS 2015.

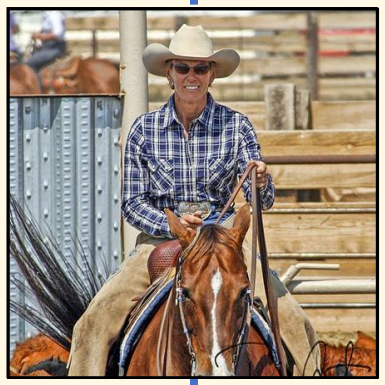
Faster R-CNN

Per-image computation

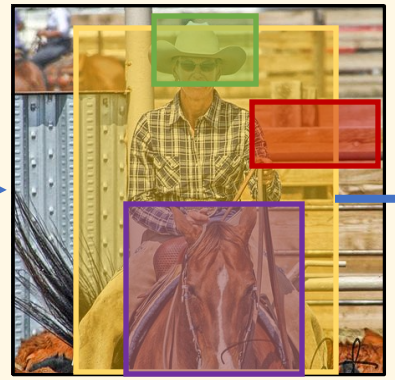
Per-region computation for each $r_i \in r(I)$

f_I = FCN(I)

I:



RPN(f_I)



RoIPool

MLP

Softmax clf.

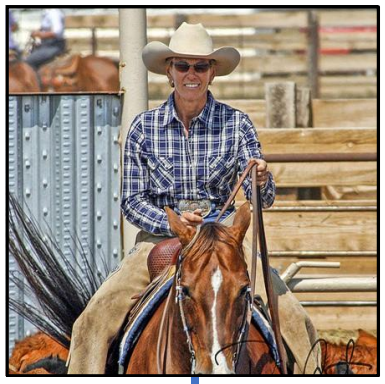
Box regressor

Learned proposals

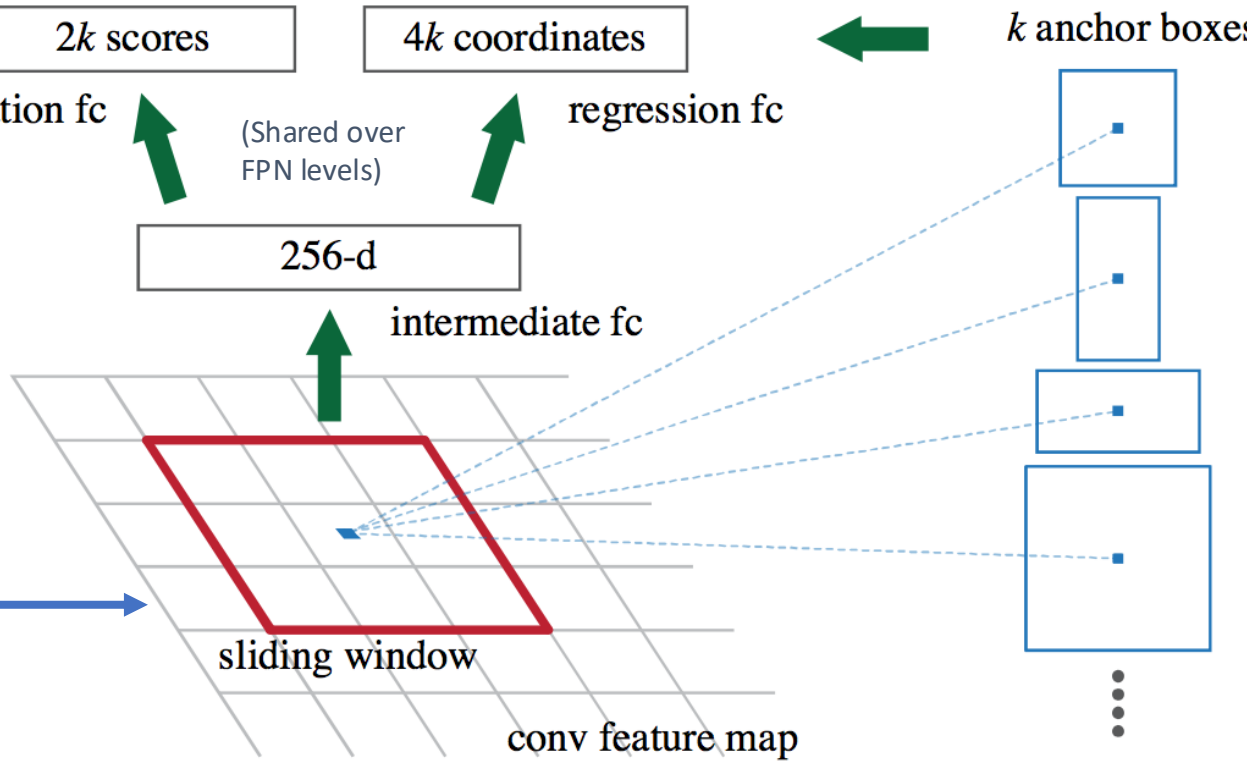
Sharing computation with whole-image network

Region Proposal Network (RPN)

Proposals = sliding window object/not-object classifier + box regression
inside the same network



f_I = FCN(I)



Anchors are prototypical object boxes

He, Gkioxari, Dollár, Girshick.
Mask R-CNN. ICCV 2017.

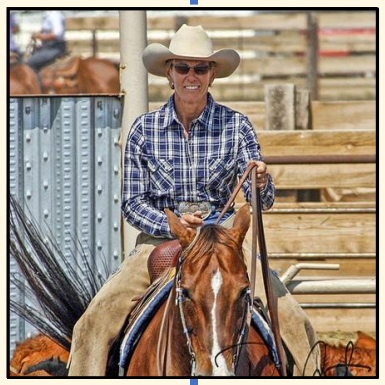
Mask R-CNN

Per-image computation

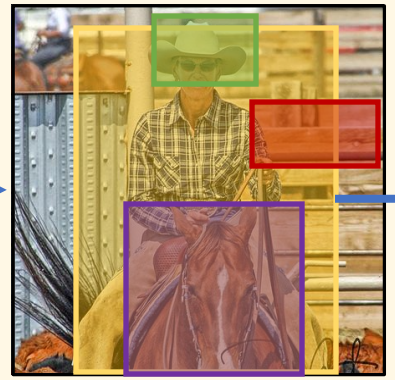
Per-region computation for each $r_i \in r(I)$

f_I = FCN(I)

I:



RPN(f_I)



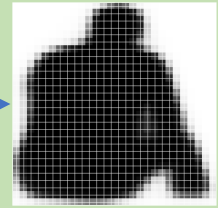
RoIAlign

MLP

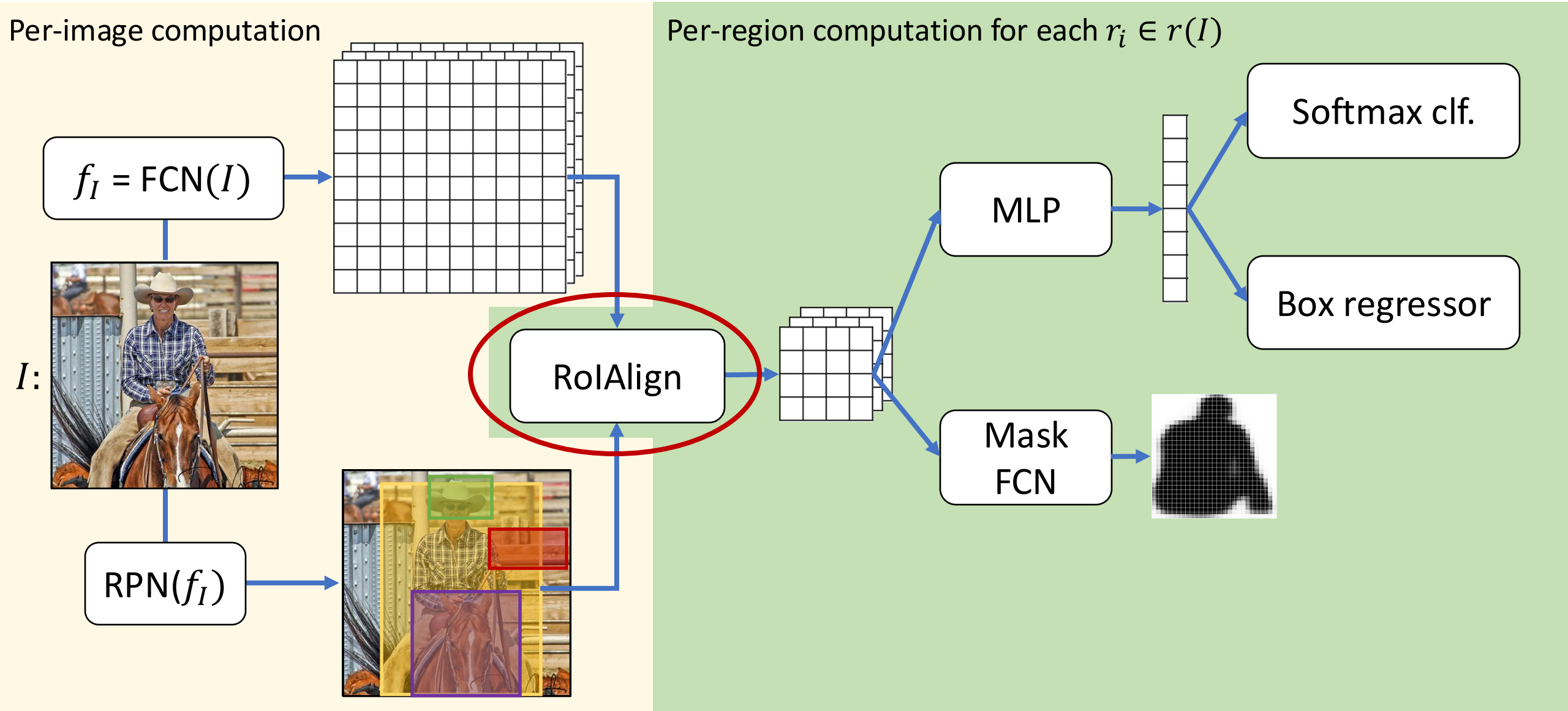
Softmax clf.

Box regressor

Mask FCN

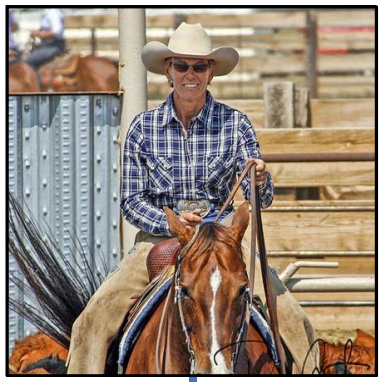


Mask R-CNN

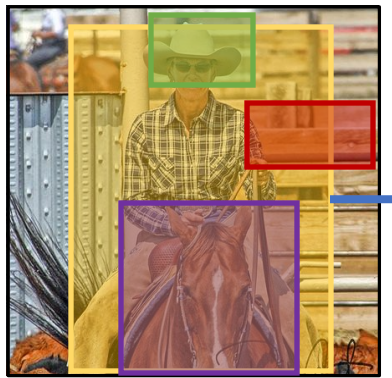

Per-image computation
Per-region computation for each $r_i \in r(I)$
f_I = FCN(I)
I:
RPN(f_I)
RoIAlign
MLP
Softmax clf.
Box regressor
Mask FCN

RoIAlign (on each Proposal)

Smoothly transform RoI features into
a fixed-dimensional representation (e.g., 2x2)

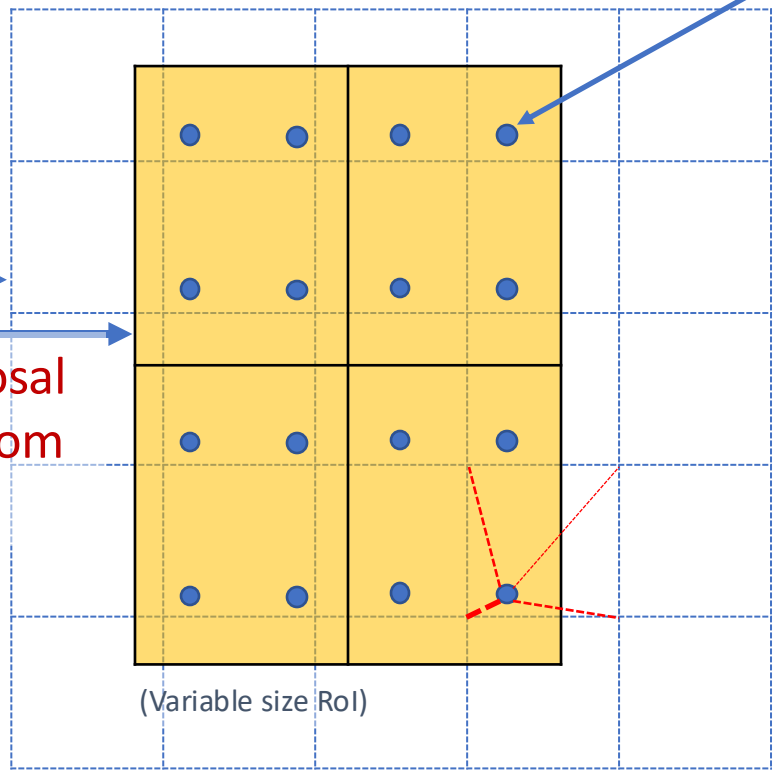


f_I = FCN(I)



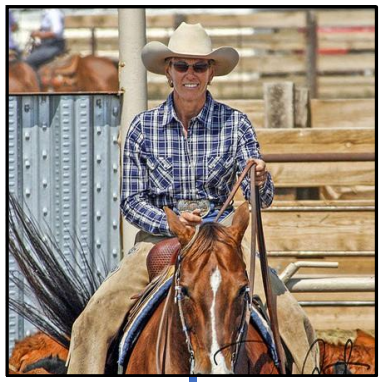
Proposal
RoI from
RPN

Grid of bilinear
interpolation points

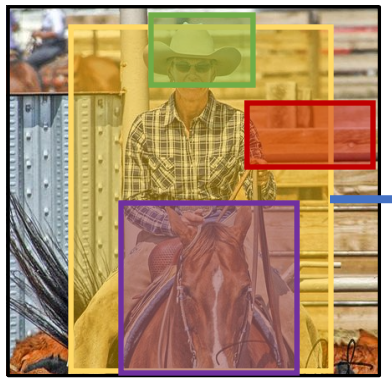


RoIAlign (on each Proposal)

Smoothly transform RoI features into
a fixed-dimensional representation (e.g., 2x2)



f_I = FCN(I)



Proposal RoI from RPN

Grid of bilinear interpolation points

(Variable size RoI)

RoIAlign transform

(Fixed dimensional representation)

Feature value is average of interpolated values on grid

Compare to RoIPool

Preserve alignment or not?

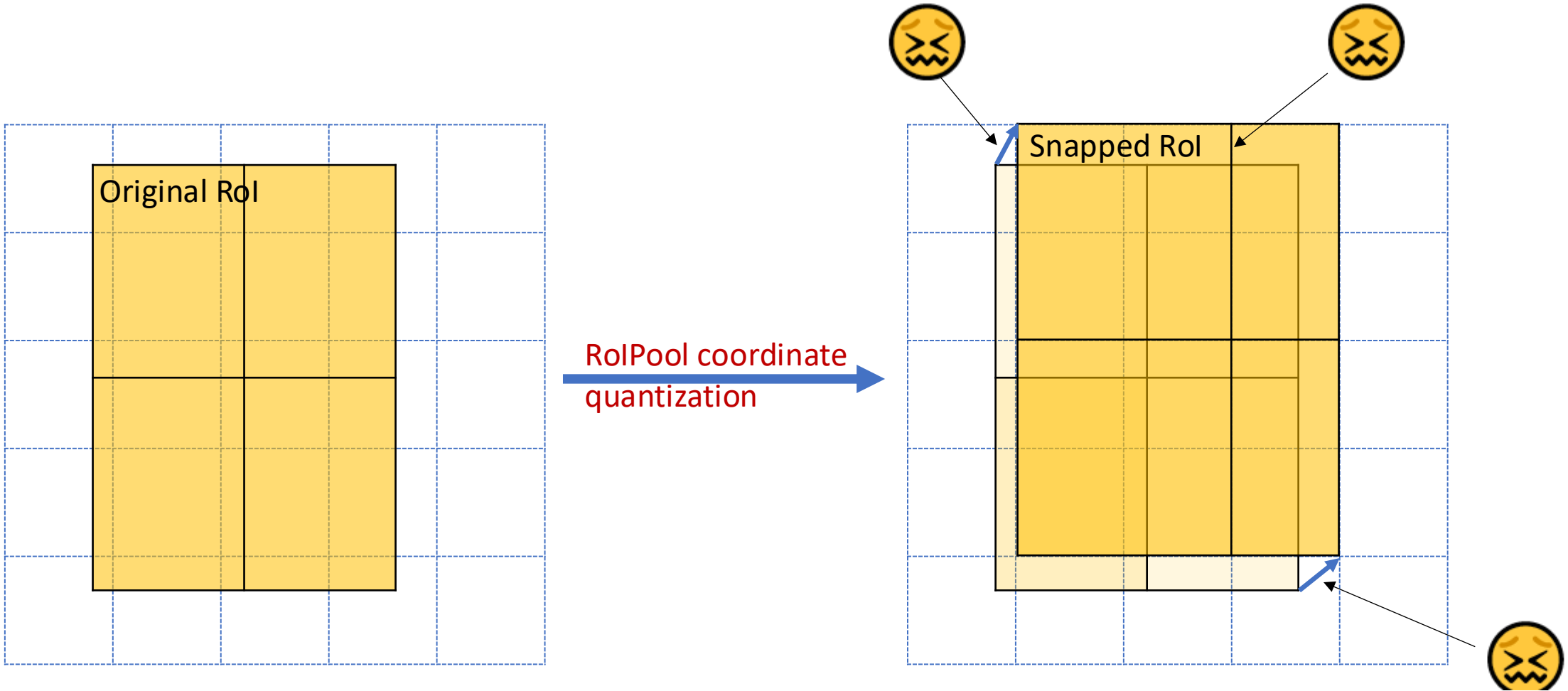
	align?	bilinear?	agg.	AP	AP_{50}	AP_{75}
RoIPool [12]			max	26.9	48.8	26.4
RoIWarp [10]		✓	max	27.2	49.2	27.1
		✓	ave	27.1	48.9	27.1
RoIAlign	✓	✓	max	**30.2**	**51.0**	**31.8**
	✓	✓	ave	**30.3**	**51.2**	**31.5**

+20% relative at high IoU

(c) **RoIAlign** (ResNet-50-C4): Mask results with various RoI layers. Our RoIAlign layer improves AP by ~3 points and AP_{75} by ~5 points. Using proper alignment is the only factor that contributes to the large gap between RoI layers.

Compare to RoIPool

Quantization breaks pixel-to-pixel alignment



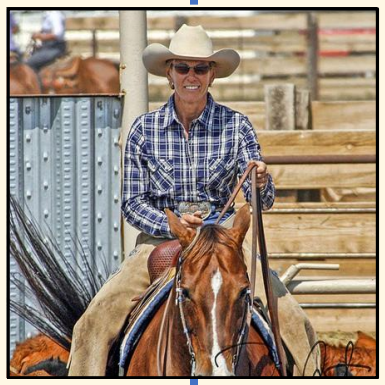
Mask R-CNN

Per-image computation

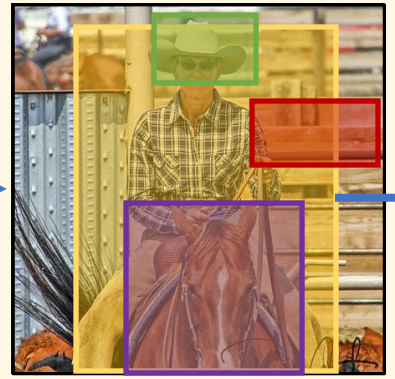
Per-region computation for each $r_i \in r(I)$

f_I = FCN(I)

I:



RPN(f_I)



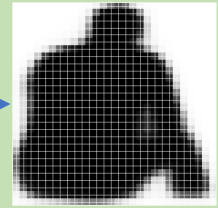
RoIAlign

MLP

Softmax clf.

Box regressor

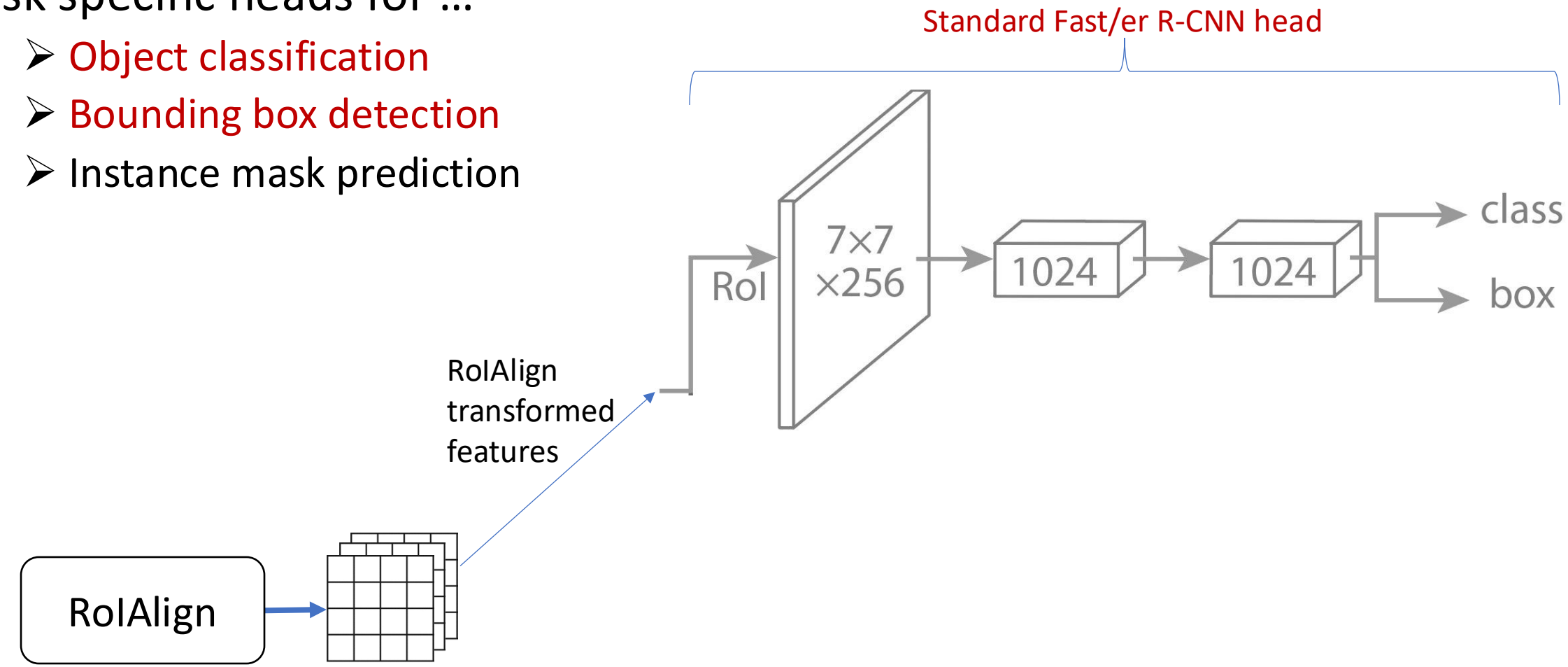
Mask FCN



Mask Head (on each Proposal)

Task specific heads for …

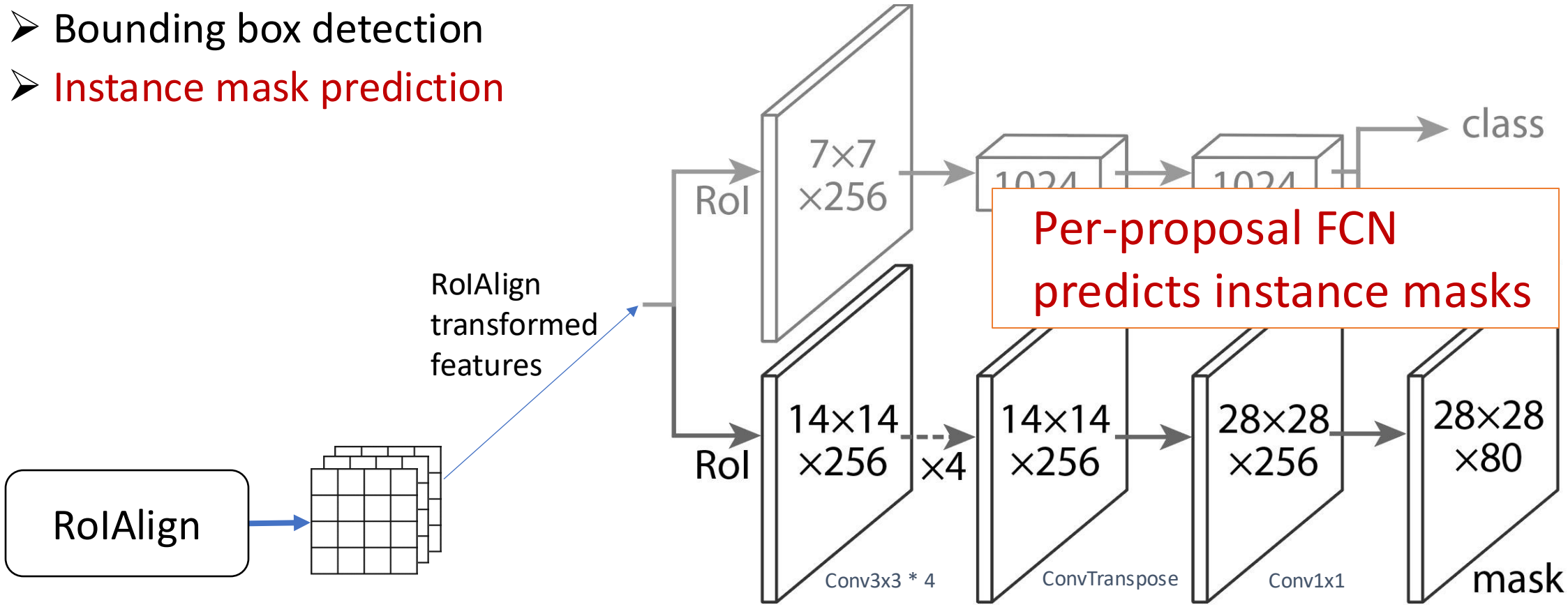
- Object classification
- Bounding box detection
- Instance mask prediction



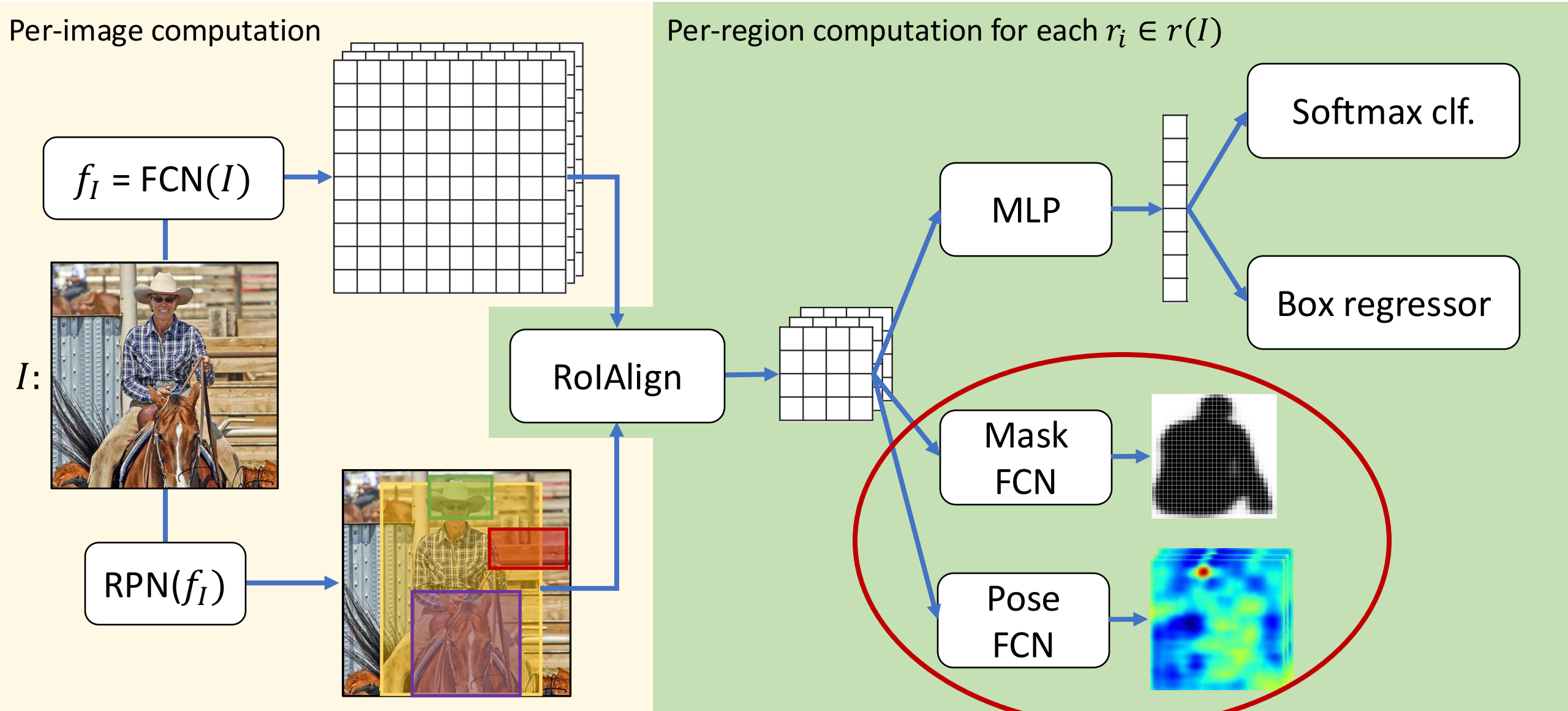
Mask Head (on each Proposal)

Task specific heads for …

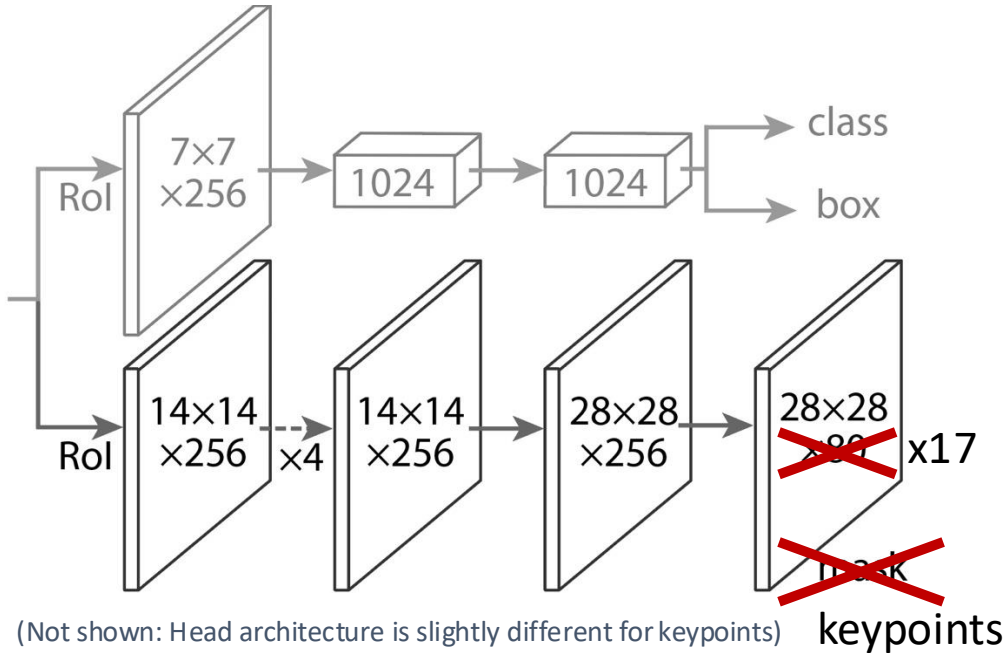
- Object classification
- Bounding box detection
- Instance mask prediction



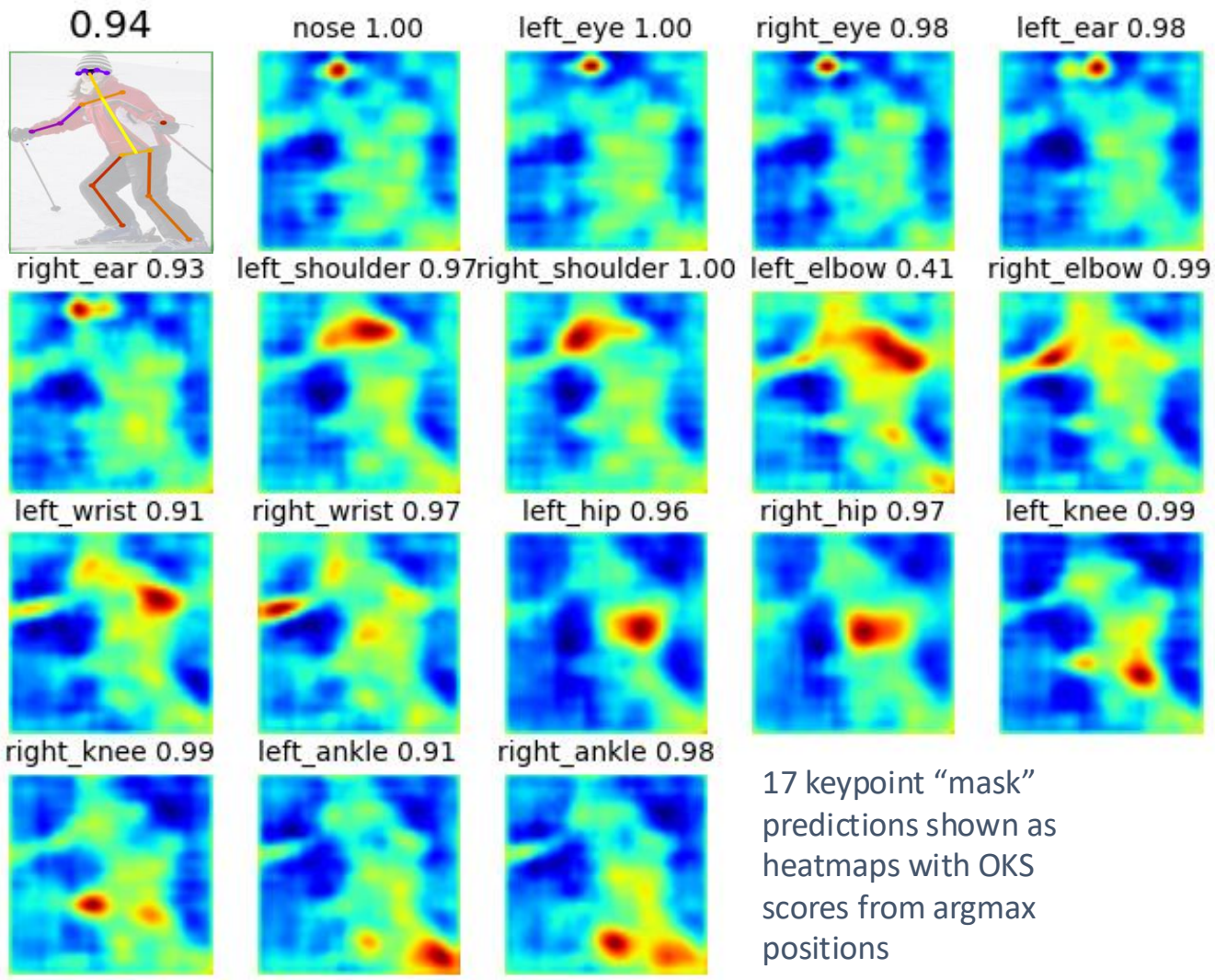
Mask R-CNN: Extension to 2D Human Pose



Pose Head



(Not shown: Head architecture is slightly different for keypoints)



17 keypoint "mask" predictions shown as heatmaps with OKS scores from argmax positions

- Add keypoint head (28x28x17)
- Predict one "mask" for each keypoint
- Softmax over spatial locations (encodes one keypoint per mask "prior")

Mask R-CNN: Training

Same as “image centric” Fast/er R-CNN training

But with training targets for masks

Example Mask Training Targets

Image with training proposal

28x28 mask target

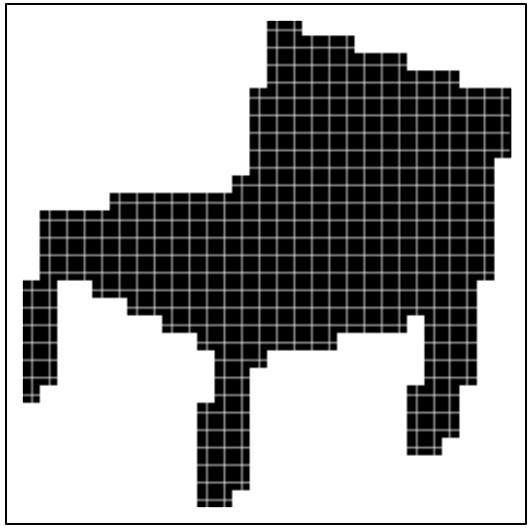
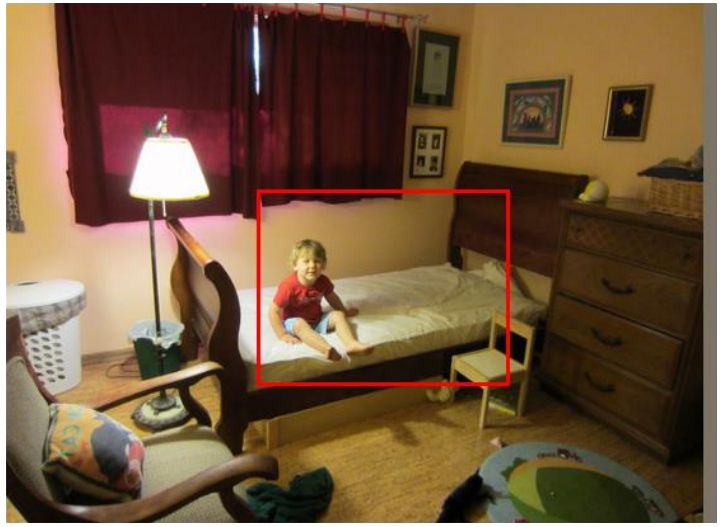
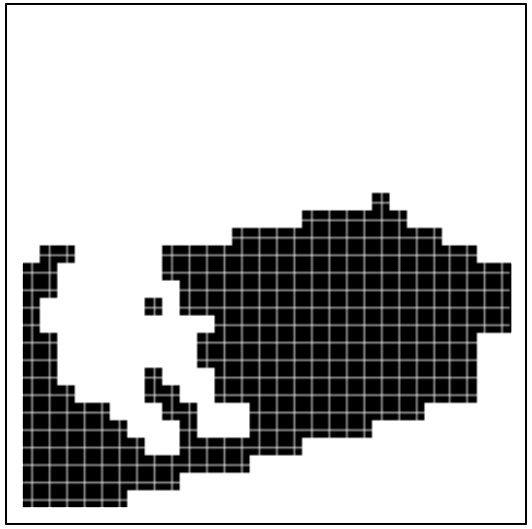
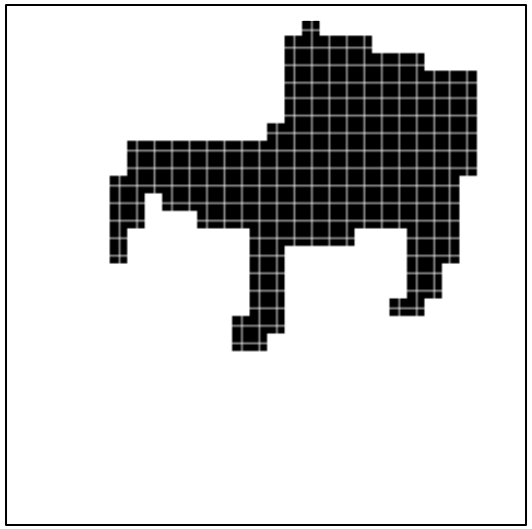


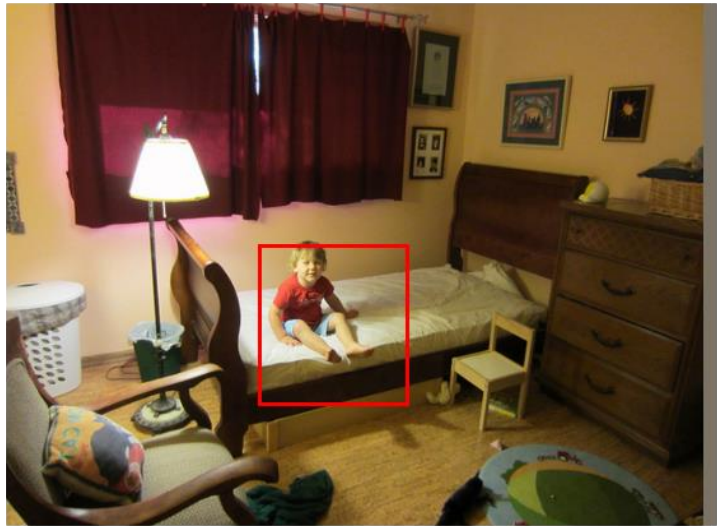
Image with training proposal

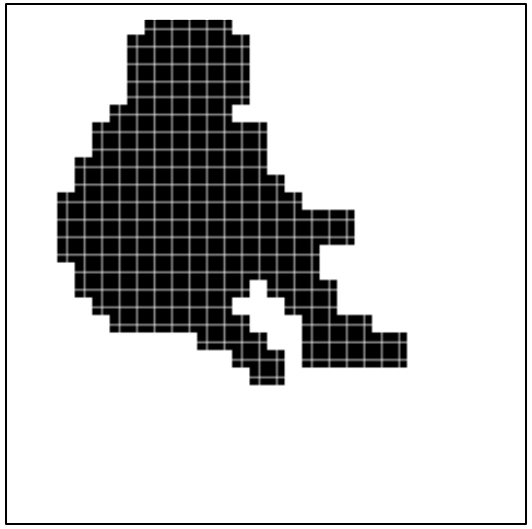


28x28 mask target









Mask R-CNN: Inference

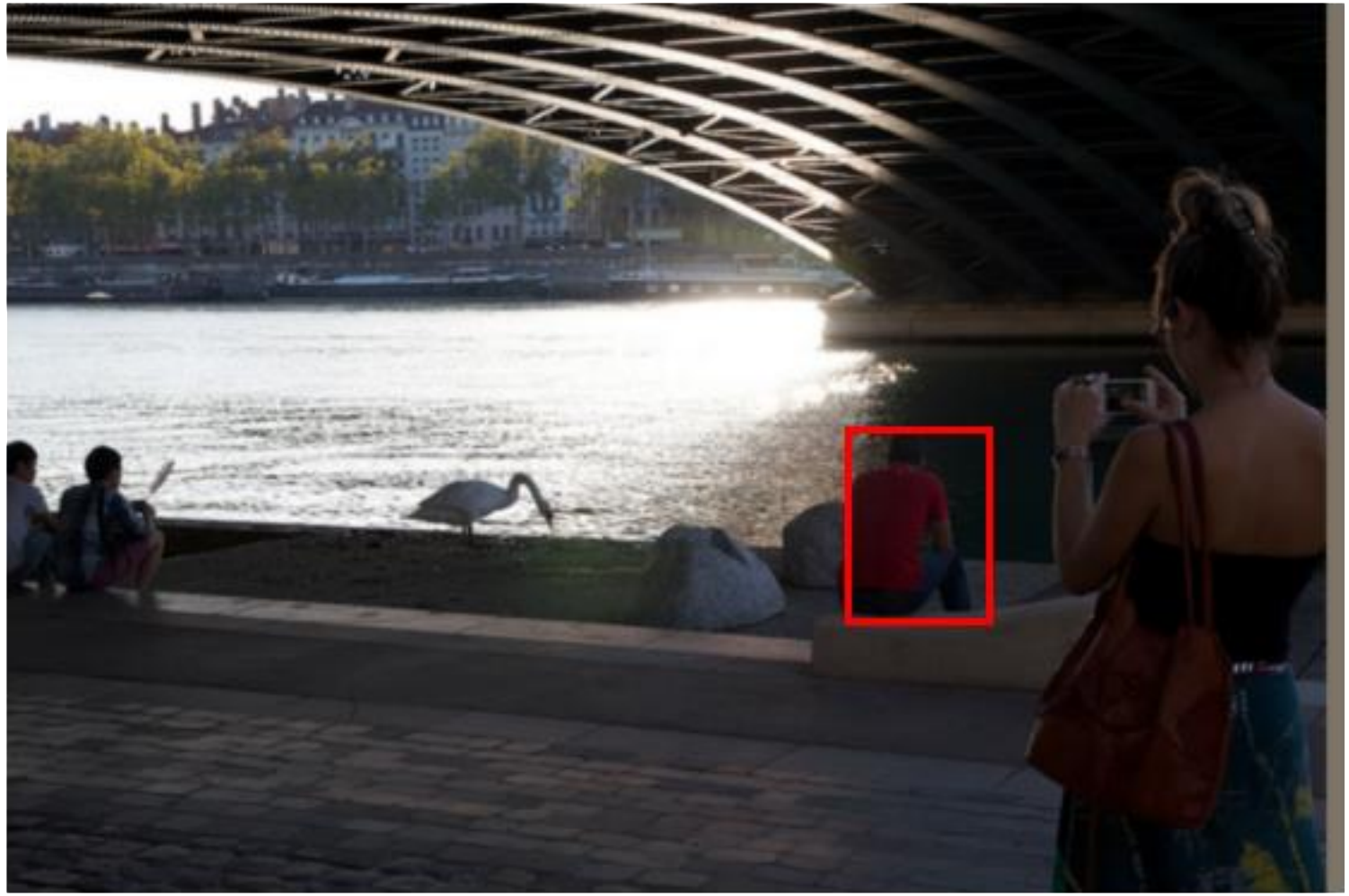
1. Perform Faster R-CNN inference

- Run backbone FCN
- Generate proposals with RPN
- Score the proposals with clf. head
- Refine proposals with box regressor
- Apply NMS and take the top *K* (= 100, e.g.)

2. Run RoIAlign and mask head on top-*K* refined, post-NMS boxes

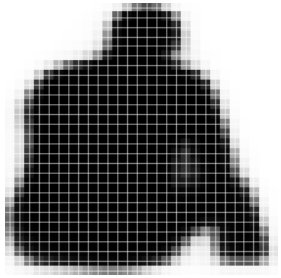
- Fast (only compute masks for top-*K* detections)
- Improves accuracy (uses *refined* detection boxes, not proposals)

Mask Prediction



Validation image with box detection shown in red

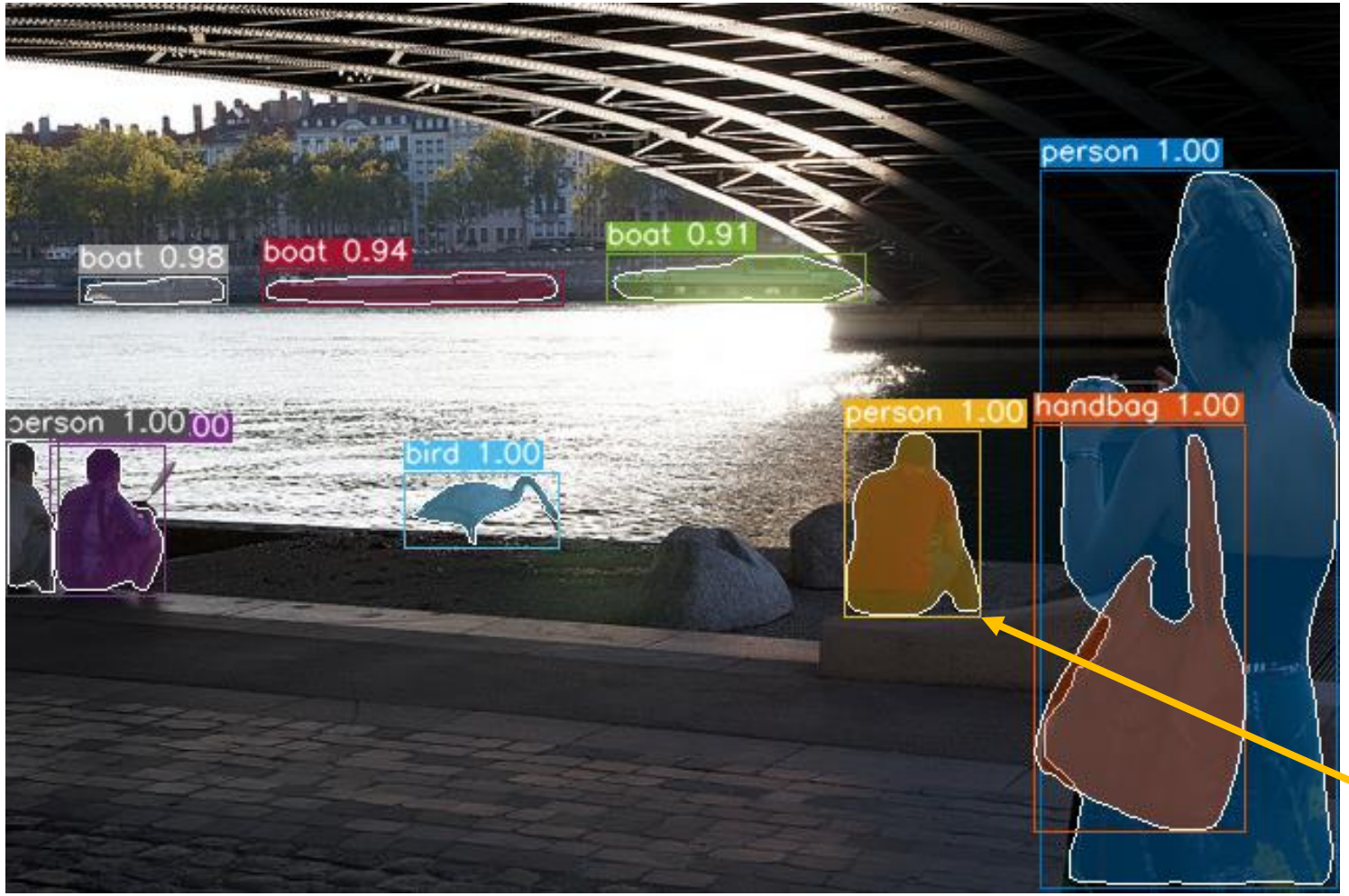
28x28 soft prediction from Mask R-CNN
(enlarged)



Soft prediction resampled to image coordinates
(bilinear and bicubic interpolation work equally well)

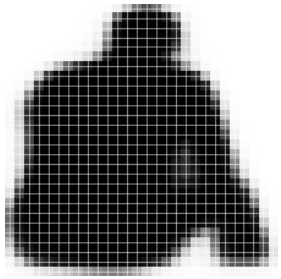
Final prediction (threshold at 0.5)

Mask Prediction



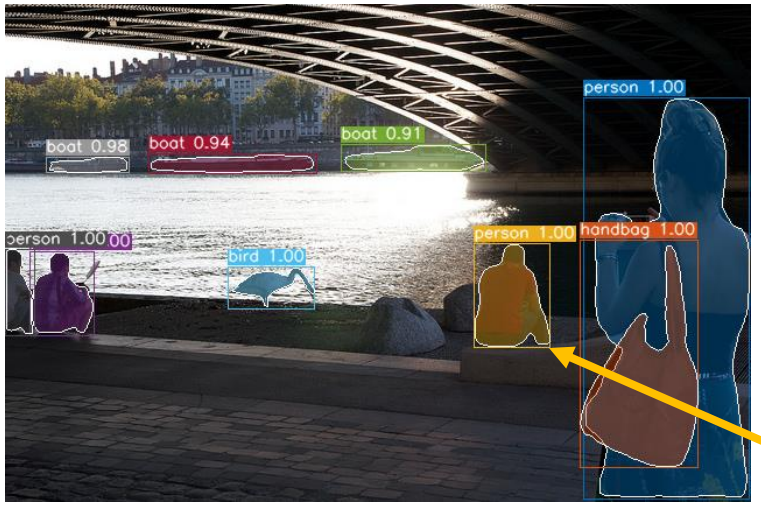
Validation image with box detection shown in red

28x28 soft prediction from Mask R-CNN
(enlarged)

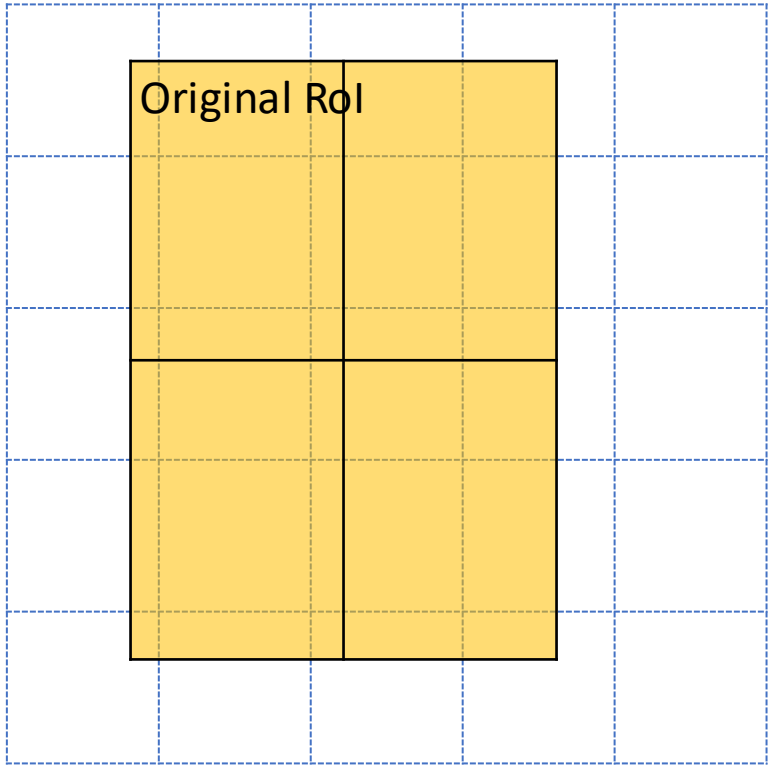


Soft prediction resampled to image coordinates
(bilinear and bicubic interpolation work equally well)

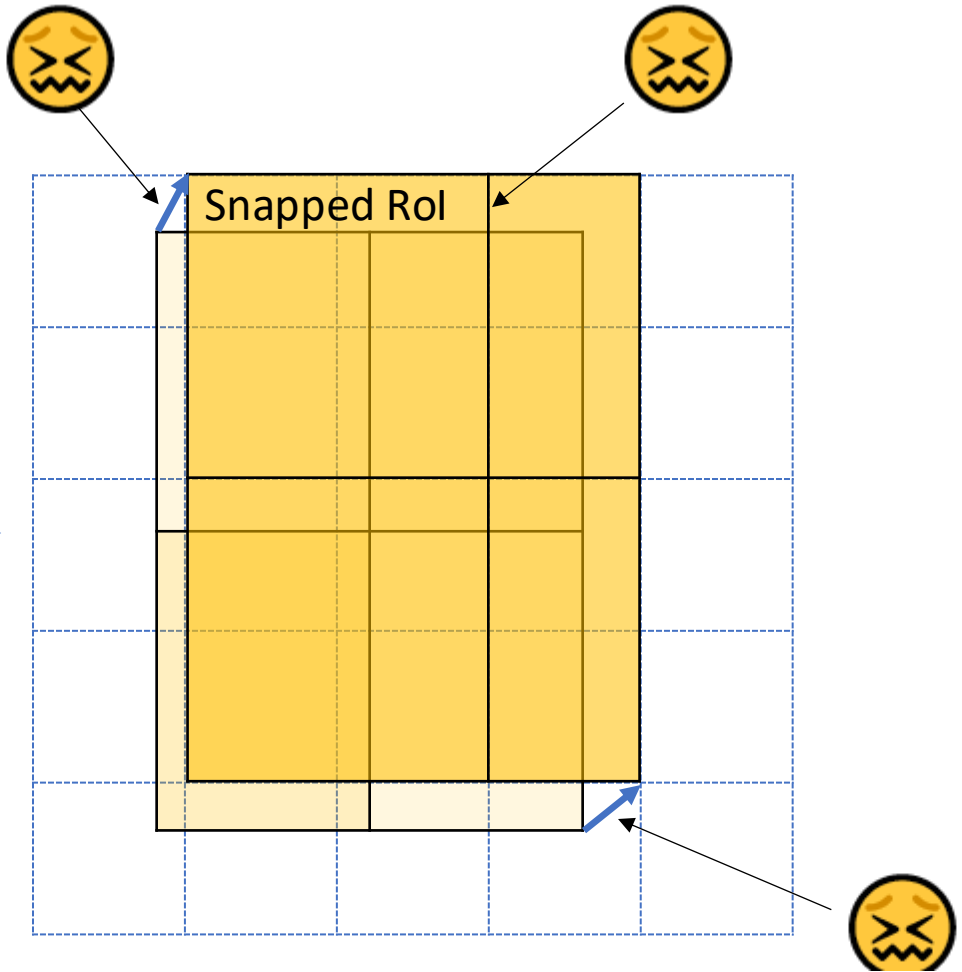
Final prediction (threshold at 0.5)



Quantization breaks pixel-to-pixel alignment



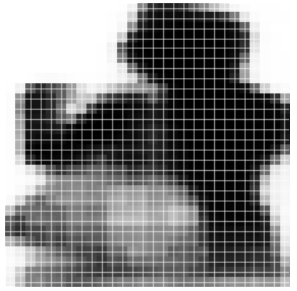
RoIPool coordinate quantization



Mask Prediction

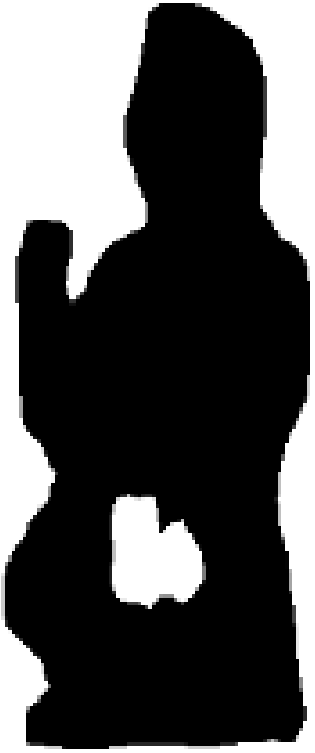
Validation image with box detection shown in red

28x28 soft prediction



Resized soft prediction

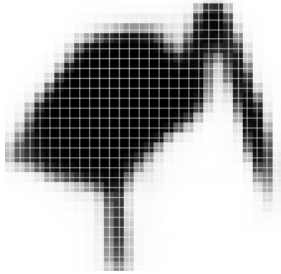
Final mask



Mask Prediction

Validation image with box detection shown in red

28x28 soft prediction



Resized Soft prediction

Final mask


person1.00
person1.00
person1.00
person.98
surfboard1.00
person1.00
person.91
surfboard1.00
surfboard.98
surfboard1.00
person.74
surfboard1.00


person.88
person1.00
tv.98
tv.84
person1.00
person1.00
bottle.97
wine glass.99
dining table.95
wine glass1.00
wine glass1.00

On Face Recognition

Jitendra Malik

Face Recognition by Humans: Nineteen Results All Computer Vision Researchers Should Know About

Increased knowledge about the ways people recognize each other may help to guide efforts to develop practical automatic face-recognition systems.

By Pawan Sinha, Benjamin Balas, Yuri Ostrovsky, and Richard Russell

Recognition as a function of available spatial resolution

Result 1: Humans can recognize familiar faces in very low-resolution images.

Result 2: The ability to tolerate degradations increases with familiarity.

Result 3: High-frequency information by itself is insufficient for good face recognition performance.

The nature of processing: Piecemeal versus holistic

Result 4: Facial features are processed holistically.

Result 5: Of the different facial features, eyebrows are among the most important for recognition.

Result 6: The important configural relationships appear to be independent across the width and height dimensions.

The nature of cues used: Pigmentation, shape and motion

Result 7: Face-shape appears to be encoded in a slightly caricatured manner.

Result 8: Prolonged face viewing can lead to high-level aftereffects, which suggest prototype-based encoding.

Result 9: Pigmentation cues are at least as important as shape cues.

Result 10: Color cues play a significant role, especially when shape cues are degraded.

Result 11: Contrast polarity inversion dramatically impairs recognition performance, possibly due to compromised ability to use pigmentation cues.

Result 12: Illumination changes influence generalization.

Result 13: View-generalization appears to be mediated by temporal association.

Result 14: Motion of faces appears to facilitate subsequent recognition.

Developmental progression

Result 15: The visual system starts with a rudimentary preference for face-like patterns.

Result 16: The visual system progresses from a piecemeal to a holistic strategy over the first several years of life.

Neural underpinnings

Result 17: The human visual system appears to devote specialized neural resources for face perception.

Result 18: Latency of responses to faces in inferotemporal (IT) cortex is about 120 ms, suggesting a largely feedforward computation.

Result 19: Facial identity and expression might be processed by separate systems.


1
2
3
4
5
6
7
8
9
10
11
12

Fig. 1. *Unlike current machine-based systems, human observers are able to handle significant degradations in face images. For instance, subjects are able to recognize more than half of all familiar faces shown to them at the resolution depicted here. Individuals shown in order are: Michael Jordan, Woody Allen, Goldie Hawn, Bill Clinton, Tom Hanks, Saddam Hussein, Elvis Presley, Jay Leno, Dustin Hoffman, Prince Charles, Cher, and Richard Nixon.*

Fig. 6. *Even drastic compressions of faces do not render them unrecognizable. Here, celebrity faces have been compressed to 25% of their original width. Yet, recognition performance with this set is the same as that obtained with the original faces.*

FaceNet: A Unified Embedding for Face Recognition and Clustering

Florian Schroff
fschroff@google.com
Google Inc.

Dmitry Kalenichenko
dkalenichenko@google.com
Google Inc.

James Philbin
jphilbin@google.com
Google Inc.

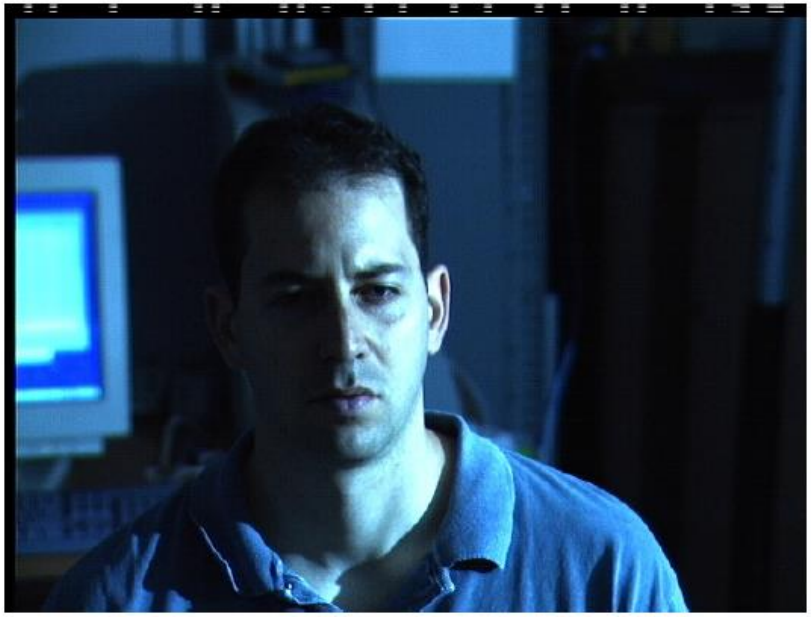
1.22

1.04

1.33

1.33

0.78



1.26

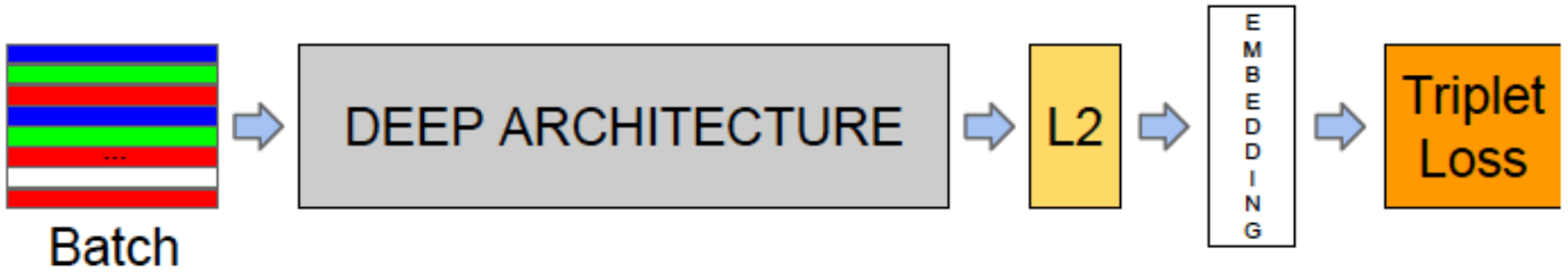


Figure 2. **Model structure.** Our network consists of a batch input layer and a deep CNN followed by L_2 normalization, which results in the face embedding. This is followed by the triplet loss during training.

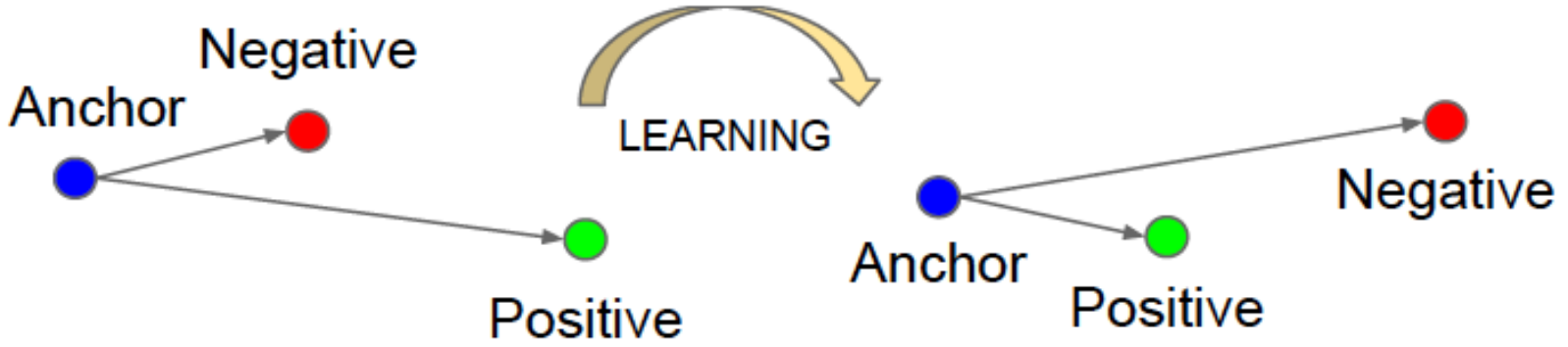


Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.

$$\sum_{i}^{N} \left[\| f(x_i^a) - f(x_i^p) \|_2^2 - \| f(x_i^a) - f(x_i^n) \|_2^2 + \alpha \right]_+$$