Topics in AI (CPSC 532S):
Multimodal Learning with Vision, Language and Sound

Lecture 6: Convolutional Neural Networks (Part 3)

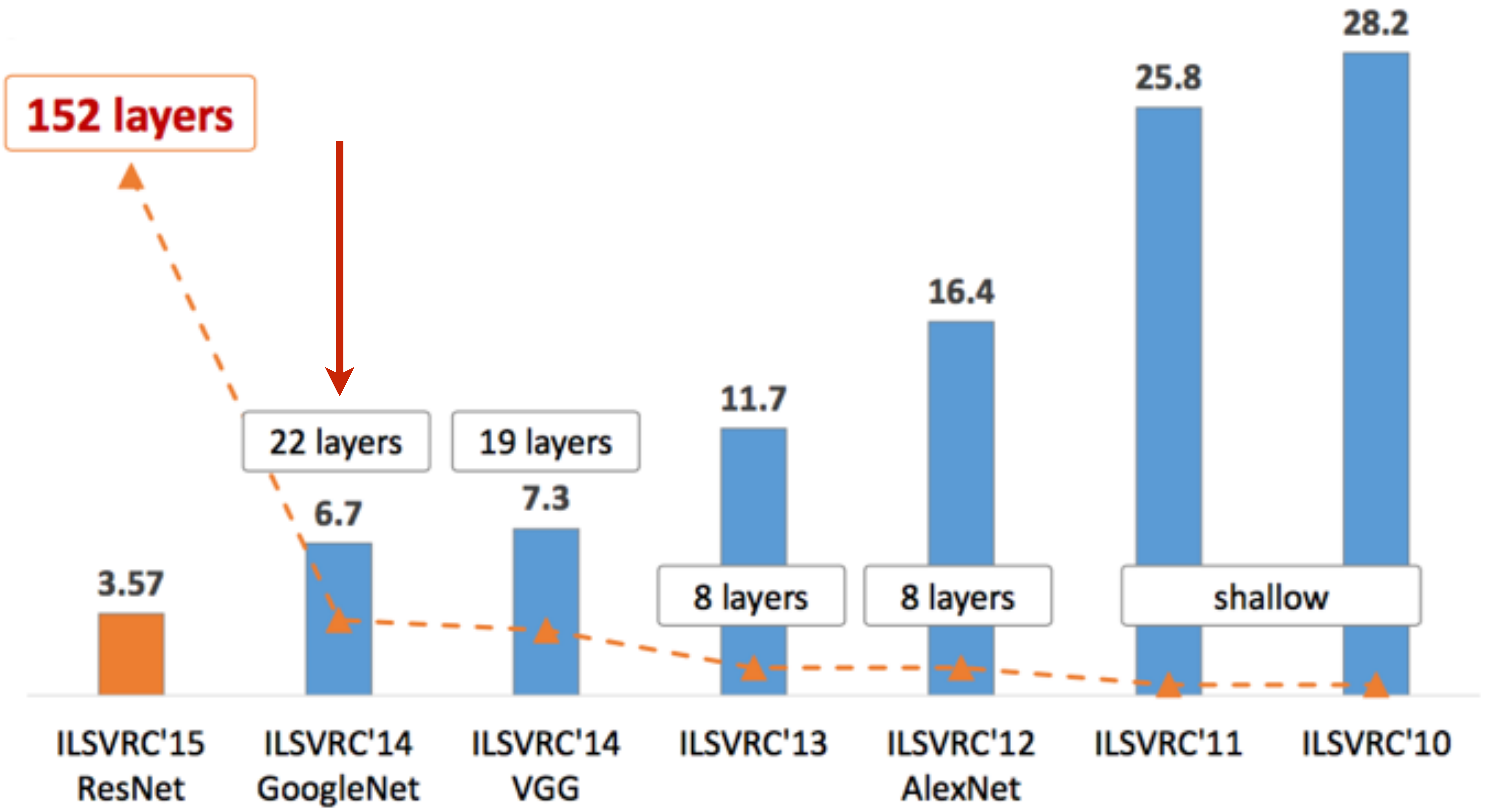
Logistics:

Assignment 2 is due on **Wednsday, 11:59pm**

Groups + idea for project by **Thursday, January 31st**

Paper list to be posted by **Monday, January 28th**

ILSVRC winner 2012



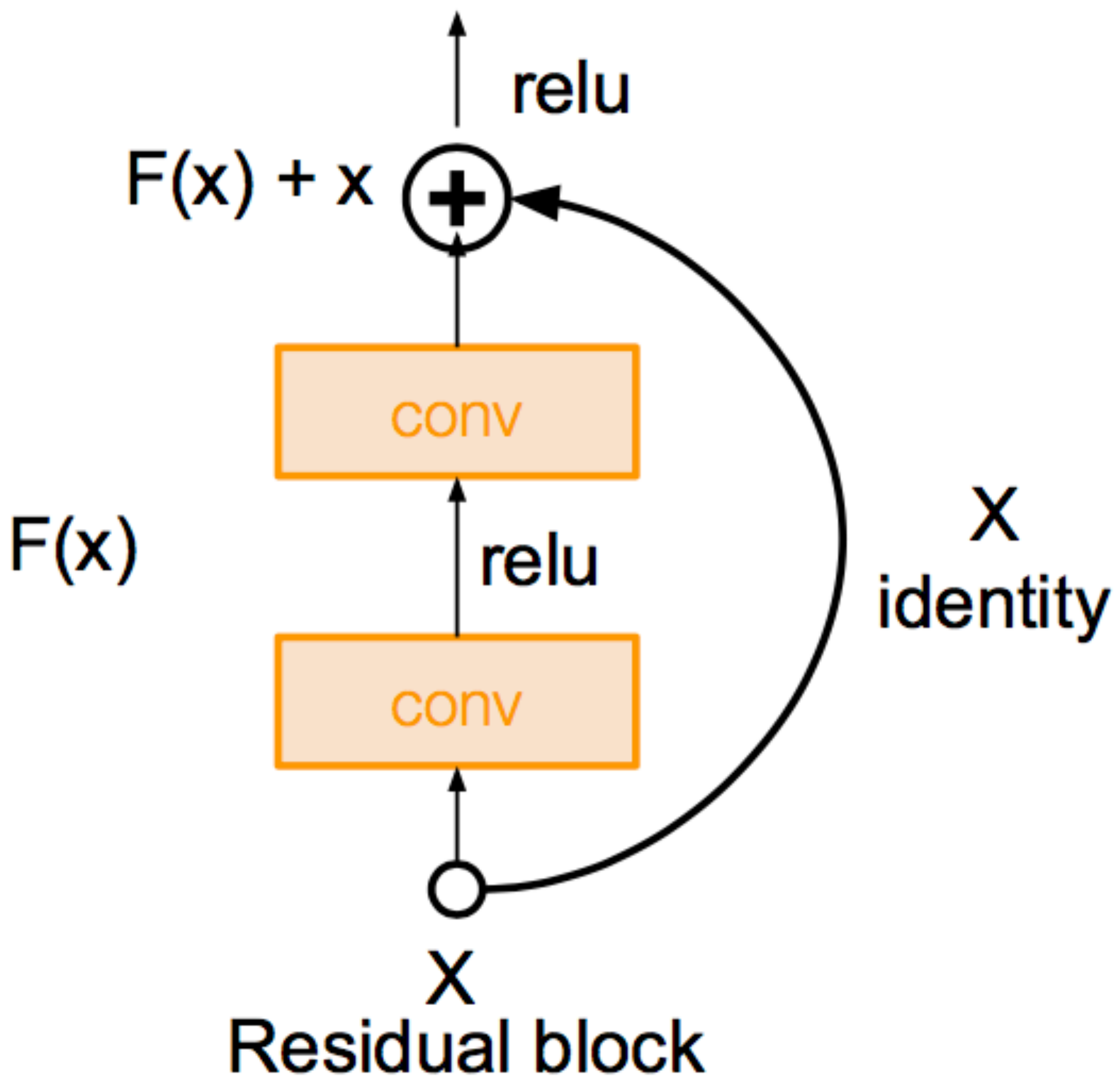
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

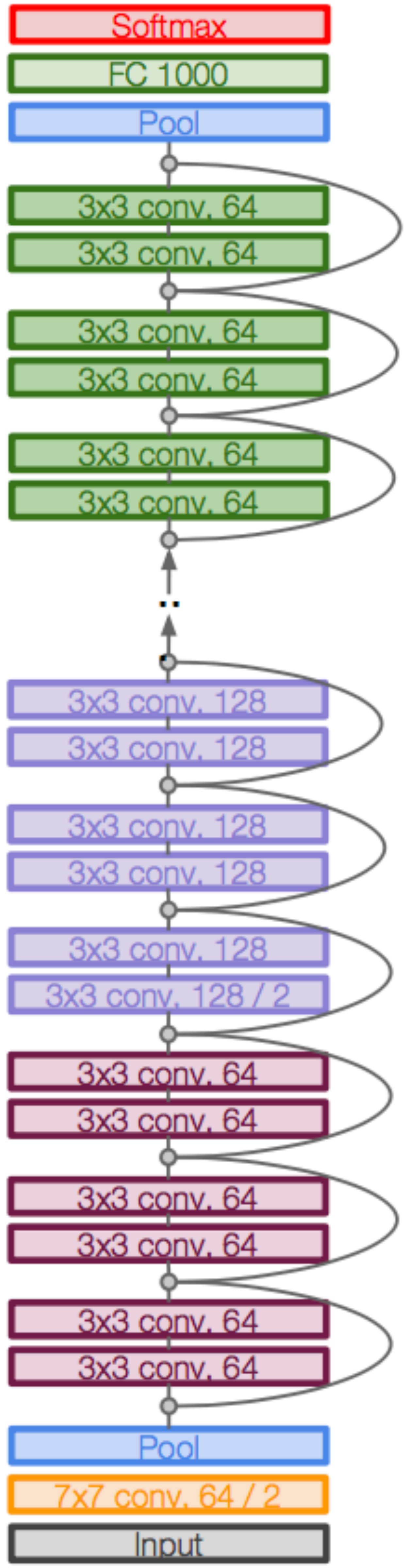
ResNet

[He et al., 2015]

even deeper — **152 layers**!

using residual connections



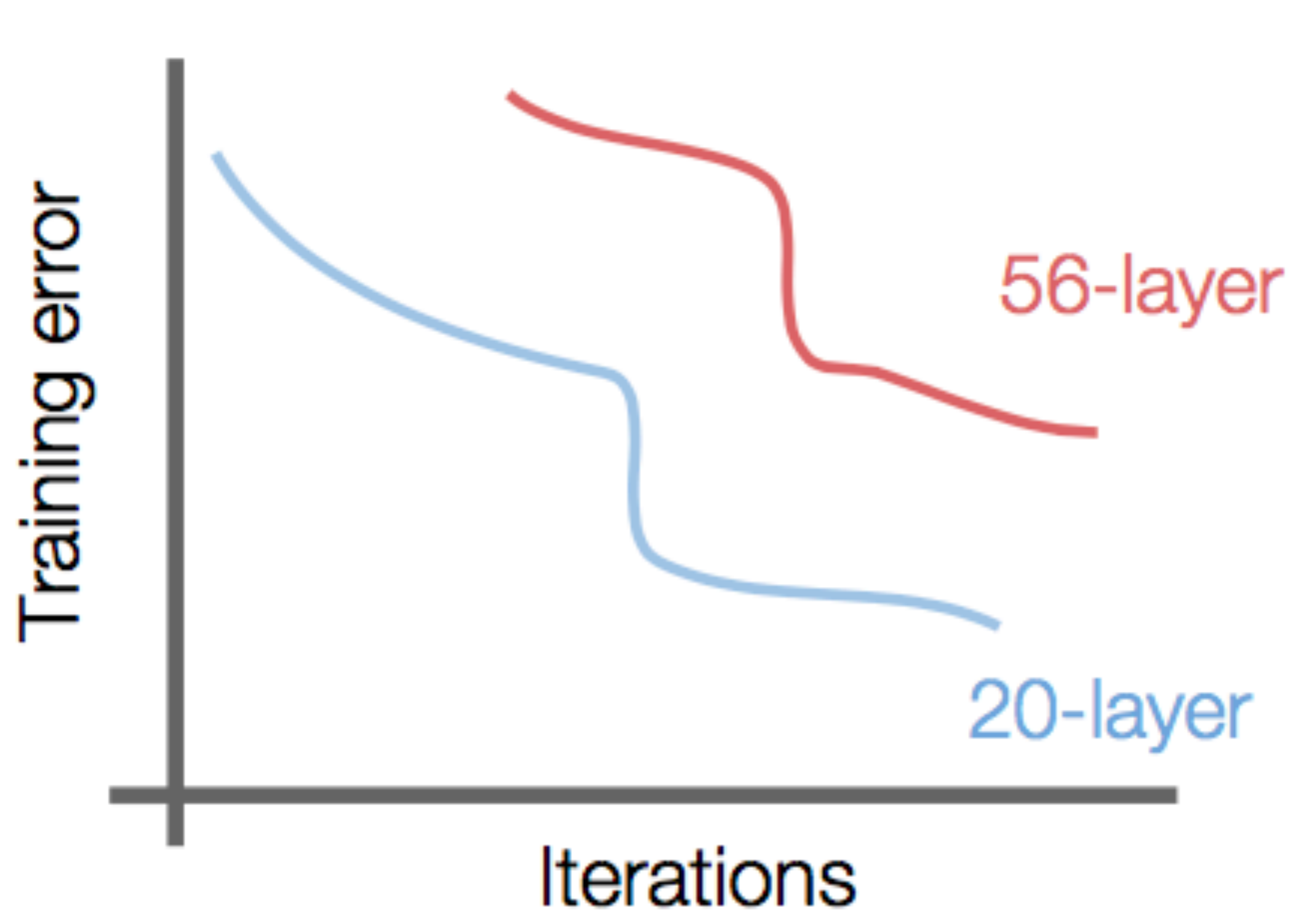


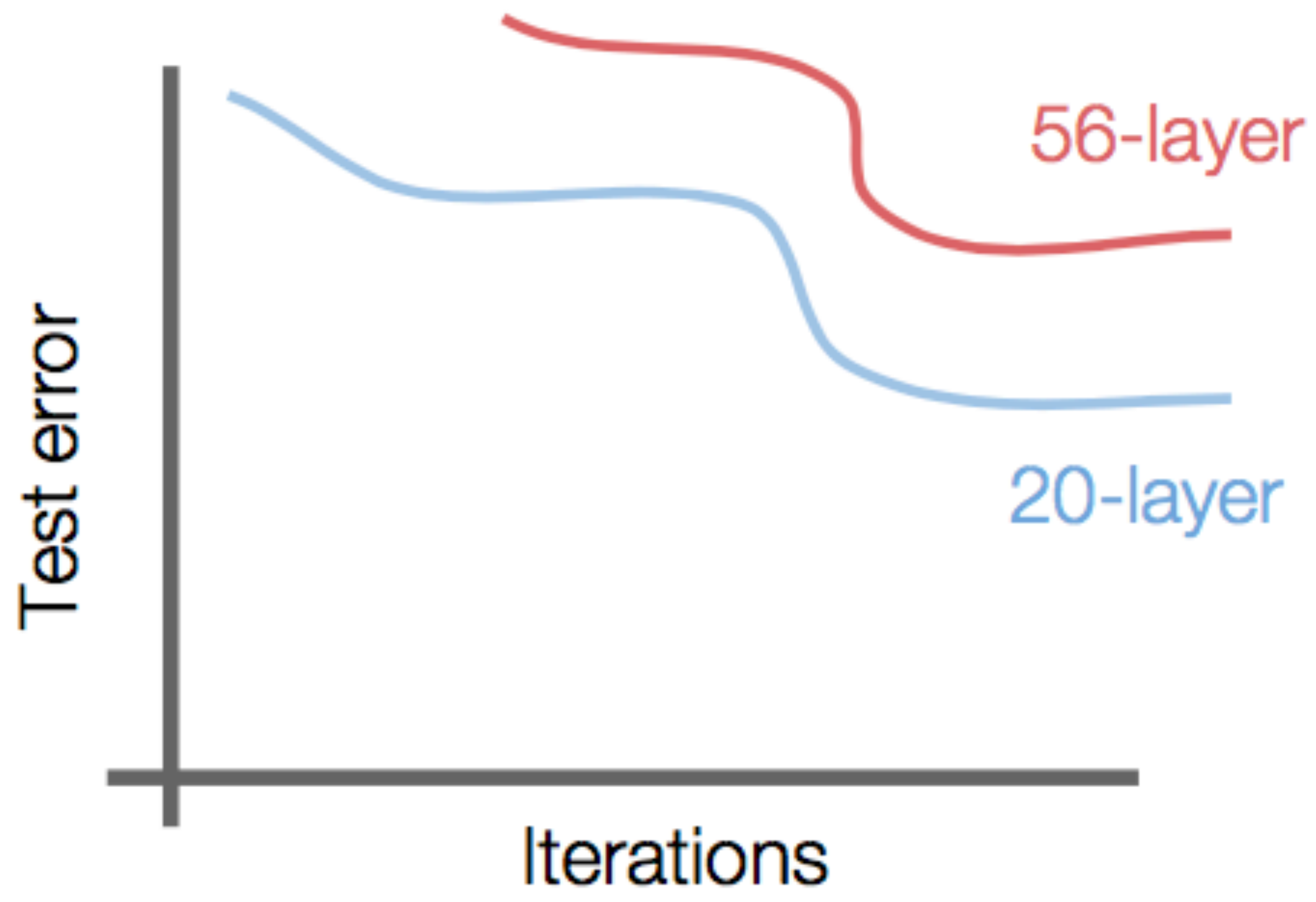
* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

ResNet: Motivation

[He et al., 2015]

What happens when we continue to stacking deeper layers on a “plain” CNN





Whats the **problem**?

ResNet: Motivation

[He et al., 2015]

Hypothesis: deeper models are harder to optimize (optimization problem)

Observation: the deeper model should (conceptually) perform just as well (e.g., take shallower model and use identity for all remaining layers)

How do we implement this idea in practice

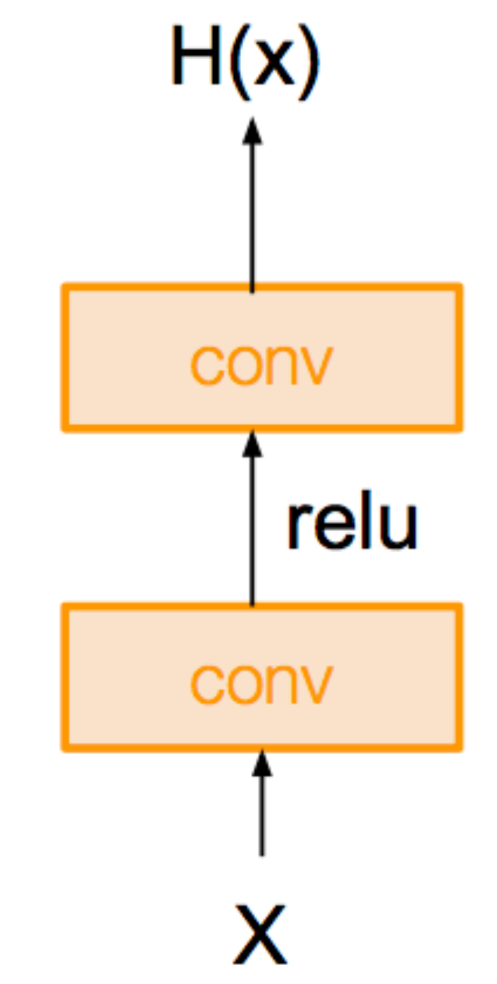
ResNet

[He et al., 2015]

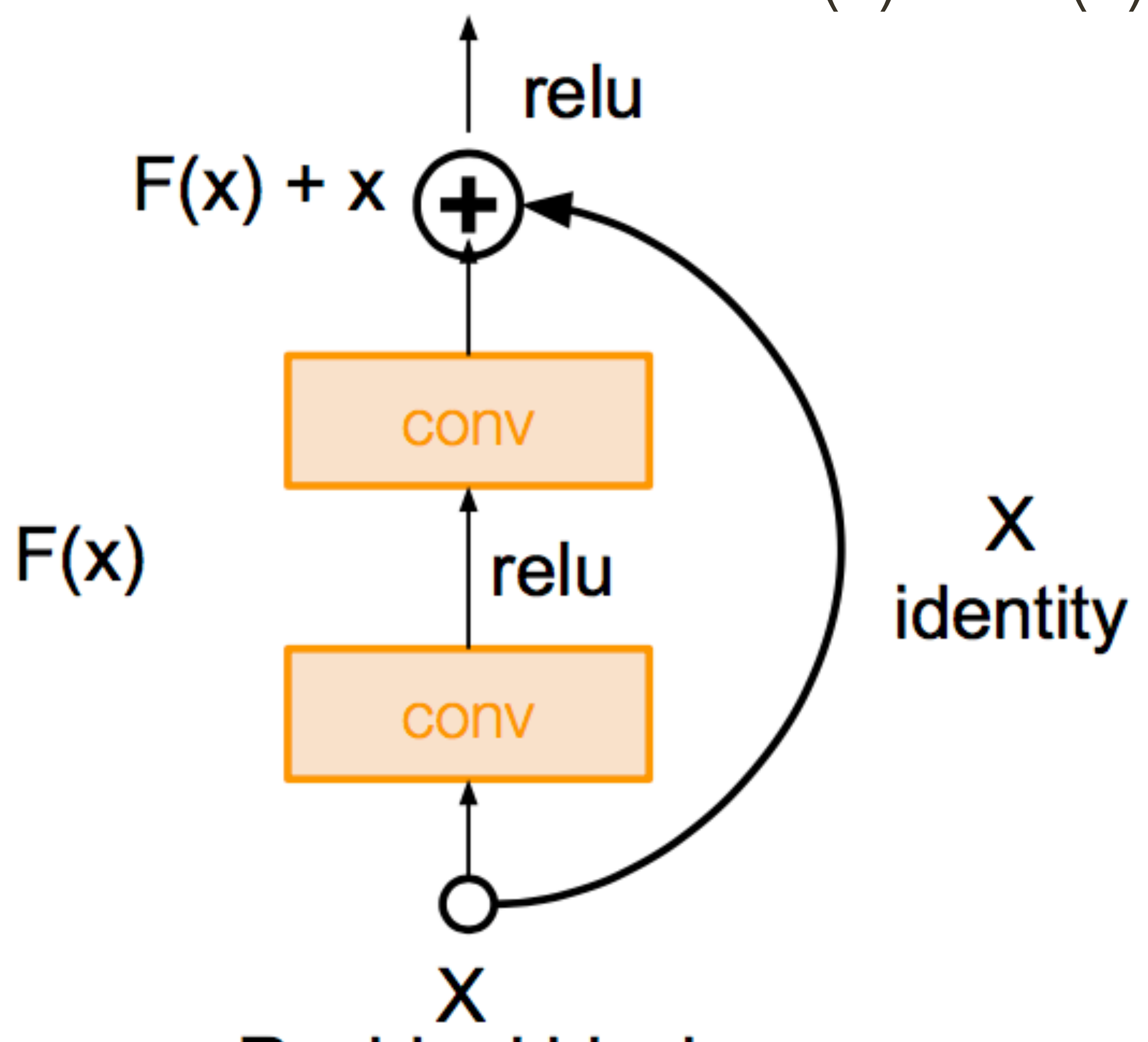
Solution: use network to fit residual mapping instead of directly trying to fit a desired underlying mapping

$$H(x) = F(x) + X$$

Use layers to fit **residual** $F(x) = H(x) - X$ instead of $H(x)$ directly



"Plain" layers



Residual block

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

ResNet

[He et al., 2015]

Full details

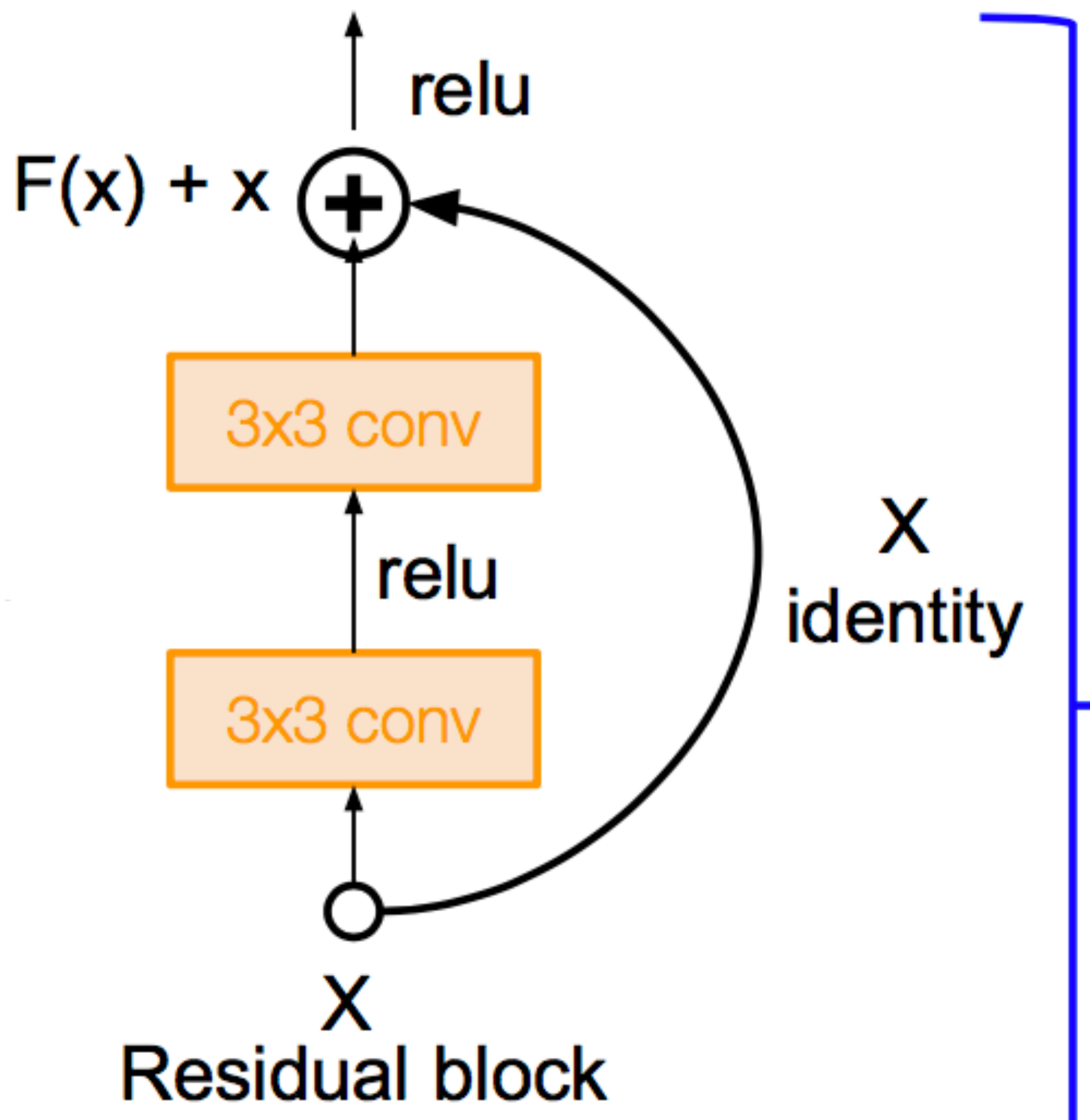
— Stacked **residual blocks**

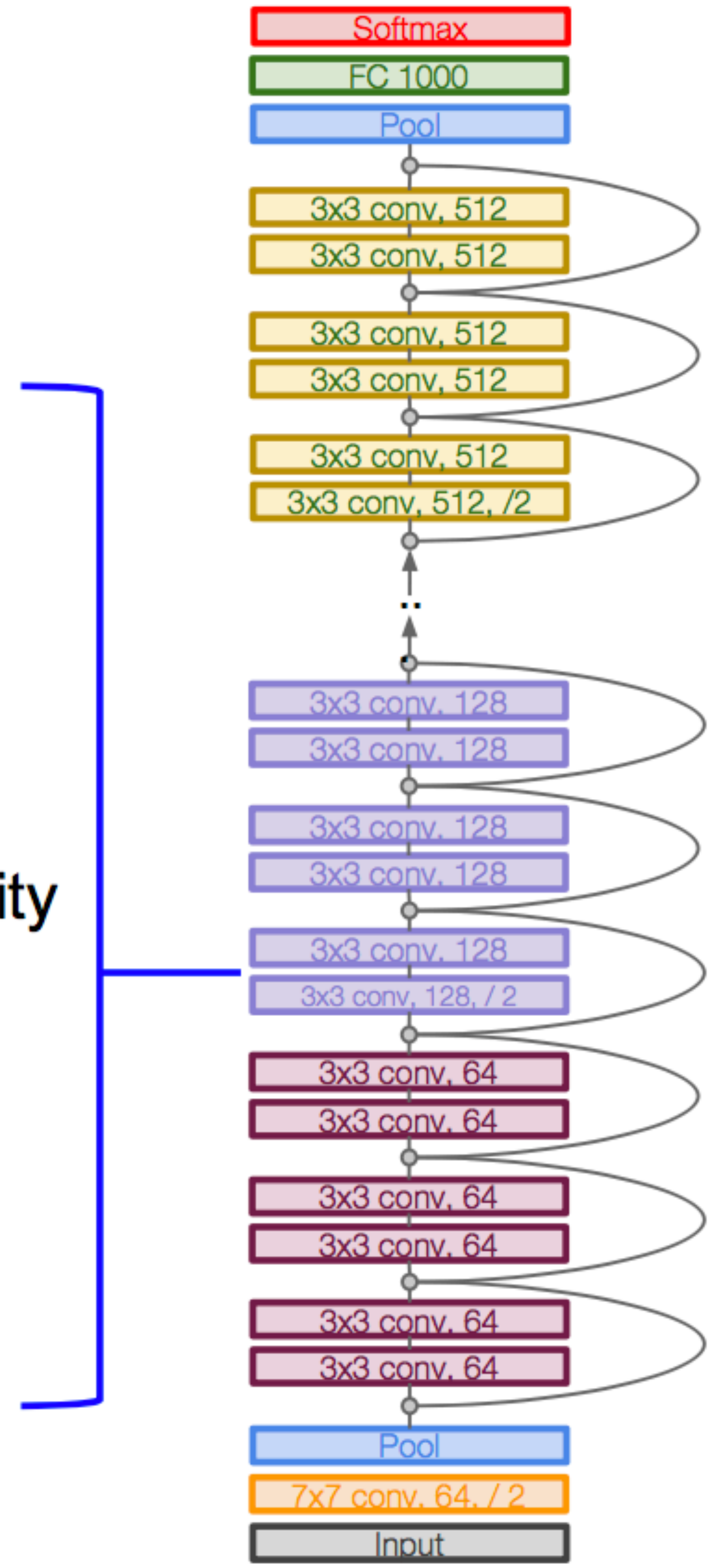
— Every residual block consists of **two 3x3 filters**

— Periodically double # of filters and downsample spatially using stride of 2

— Additional convolutional layer in the beginning

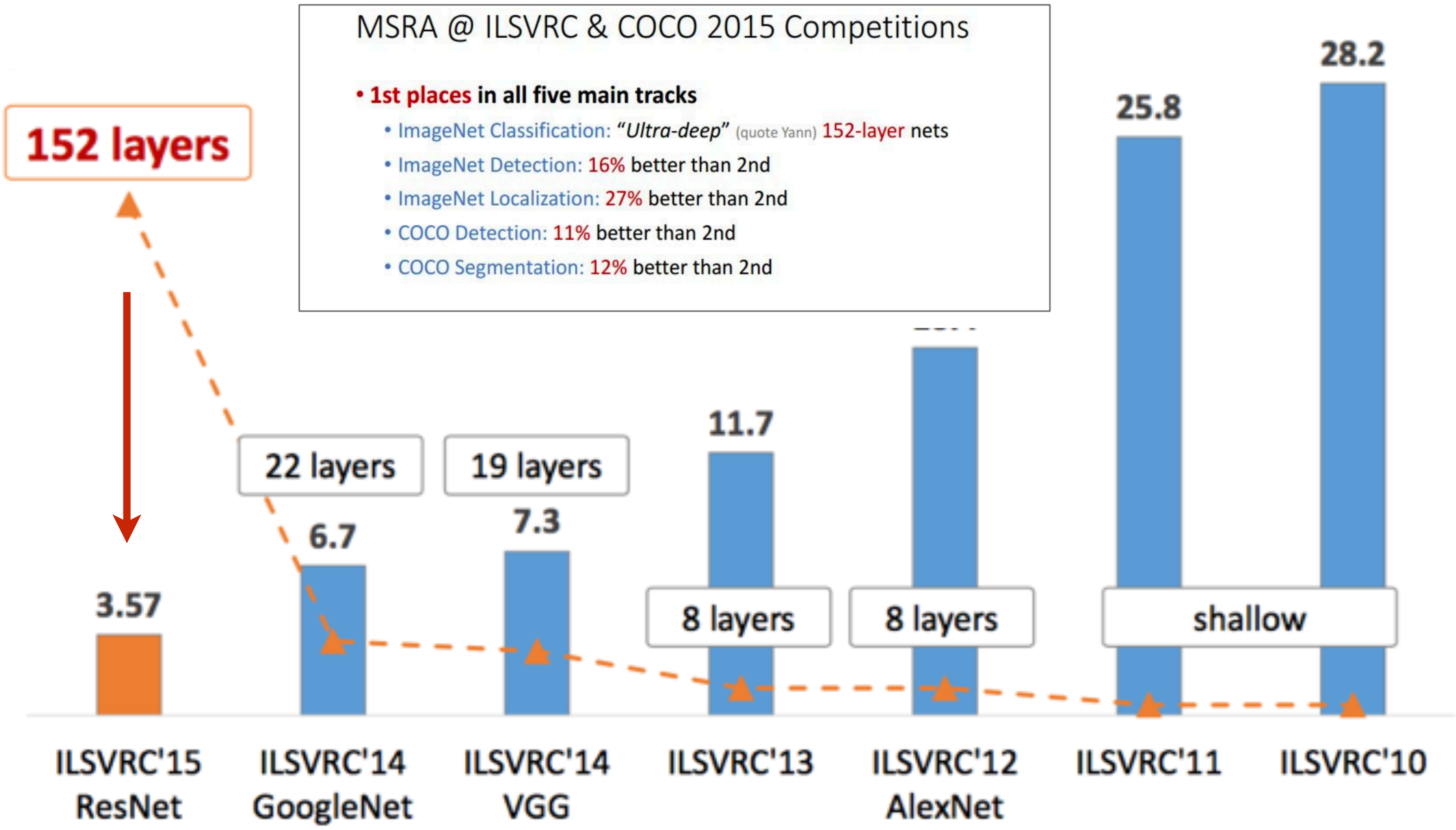
— **No FC layers** at the end (only FC to output 1000 classes)





* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

ILSVRC winner 2012



* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

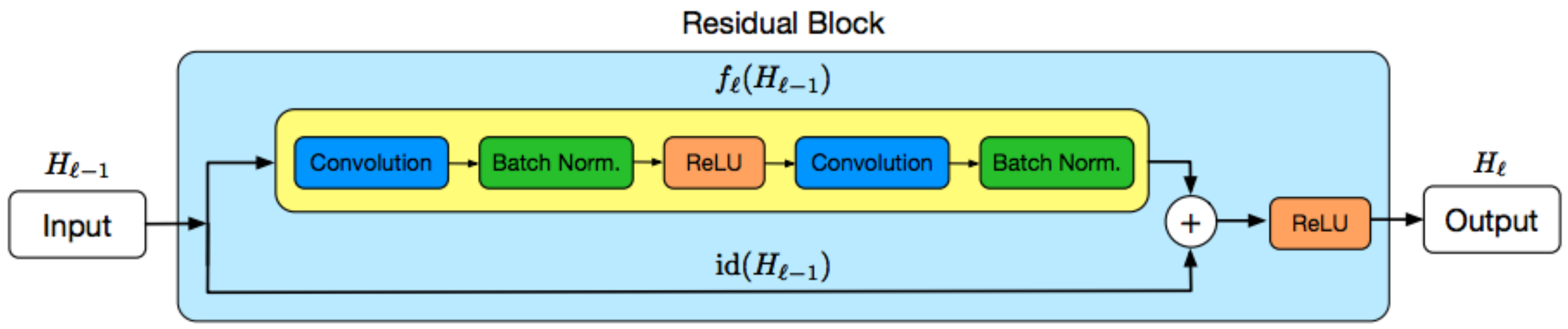
Regularization: Stochastic Depth

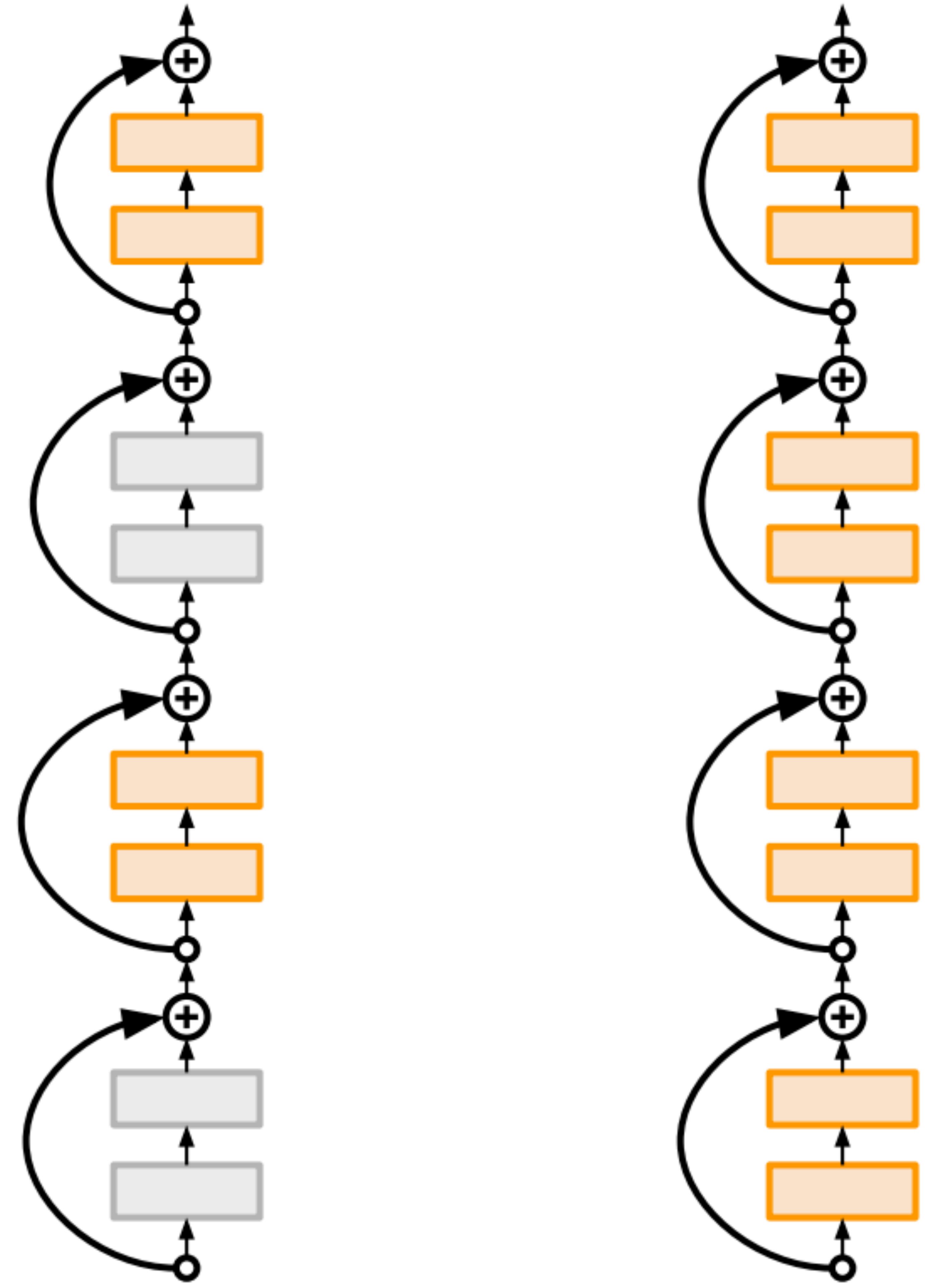
[Huang et al., ECCV 2016]

Effectively "dropout" but for layers

Stochastically with some probability **turn off some layer** (for each batch)

Effectively trains a collection of neural networks

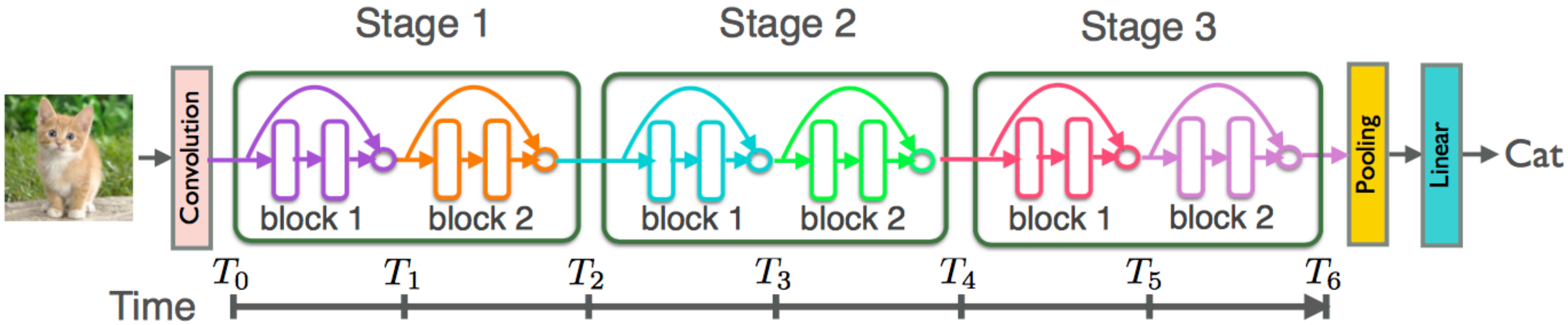




* adopted from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

ResNet: A little theory

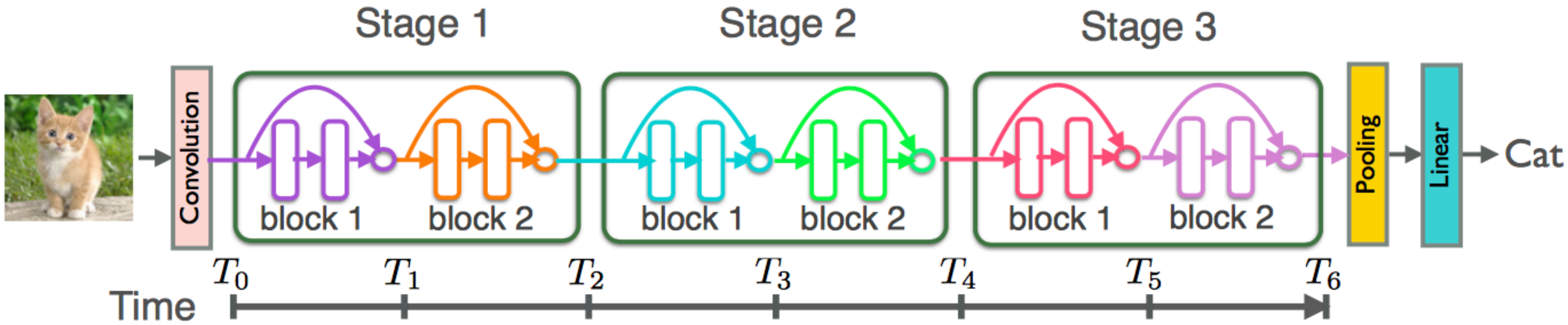
One can view a sequence of outputs from residual layers as a **Dynamical System**



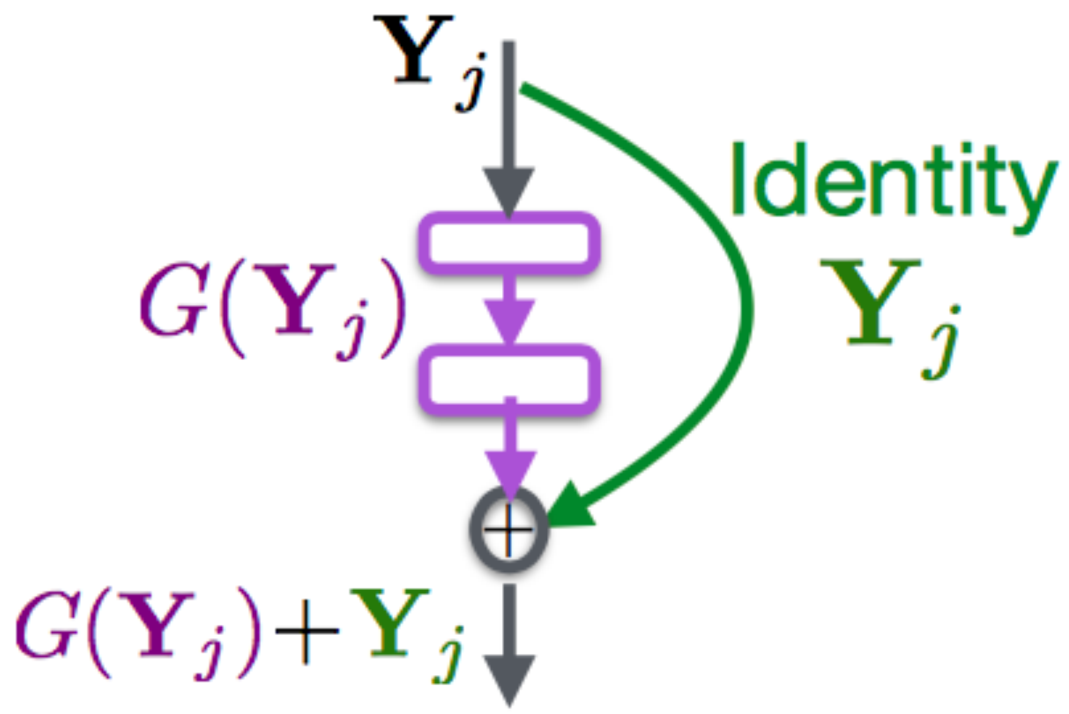
[Cheng et al., ICLR 2018]

ResNet: A little theory

One can view a sequence of outputs from residual layers as a **Dynamical System**



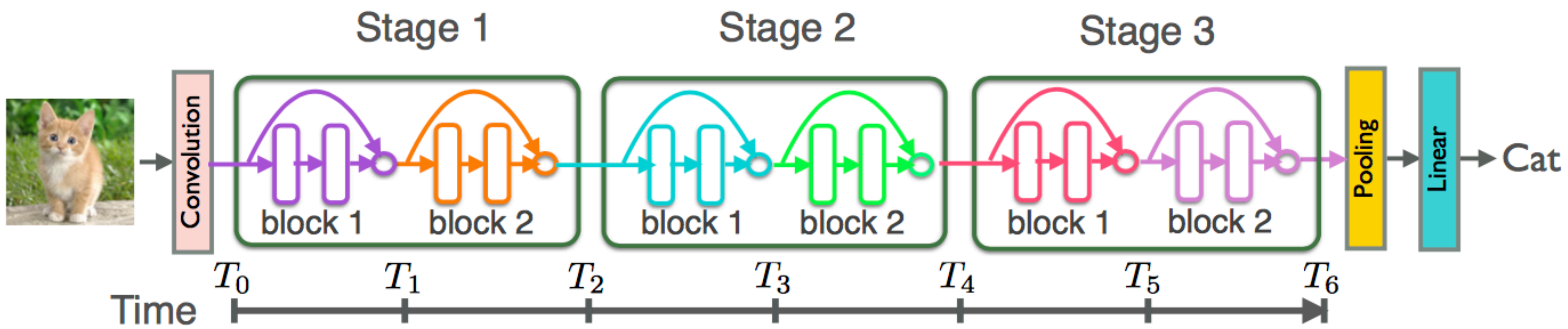
$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + G(\mathbf{Y}_j, \boldsymbol{\theta}_j)$$



[Cheng et al., ICLR 2018]

ResNet: A little theory

One can view a sequence of outputs from residual layers as a **Dynamical System**

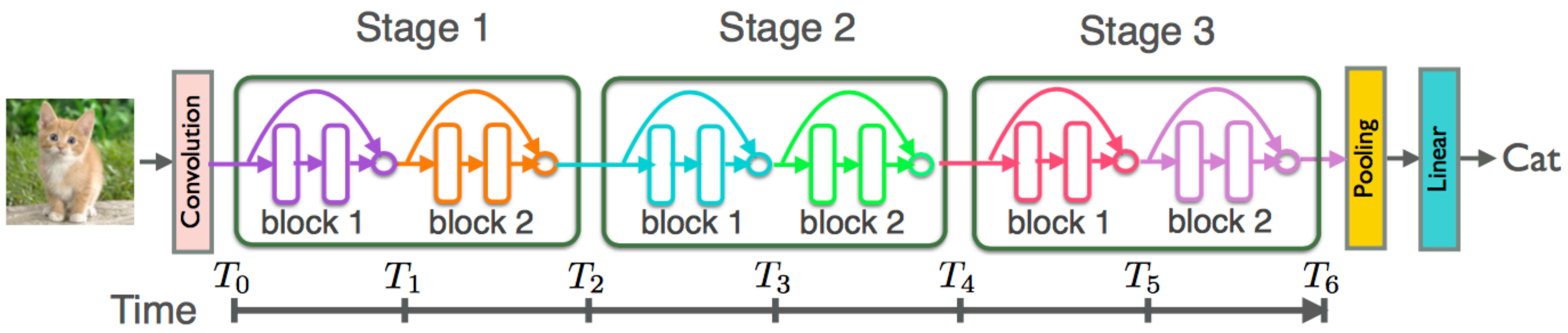


What happens if you take more layers and take smaller steps?

[Chen et al., NIPS 2018 **best paper**]

ResNet: A little theory

One can view a sequence of outputs from residual layers as a **Dynamical System**

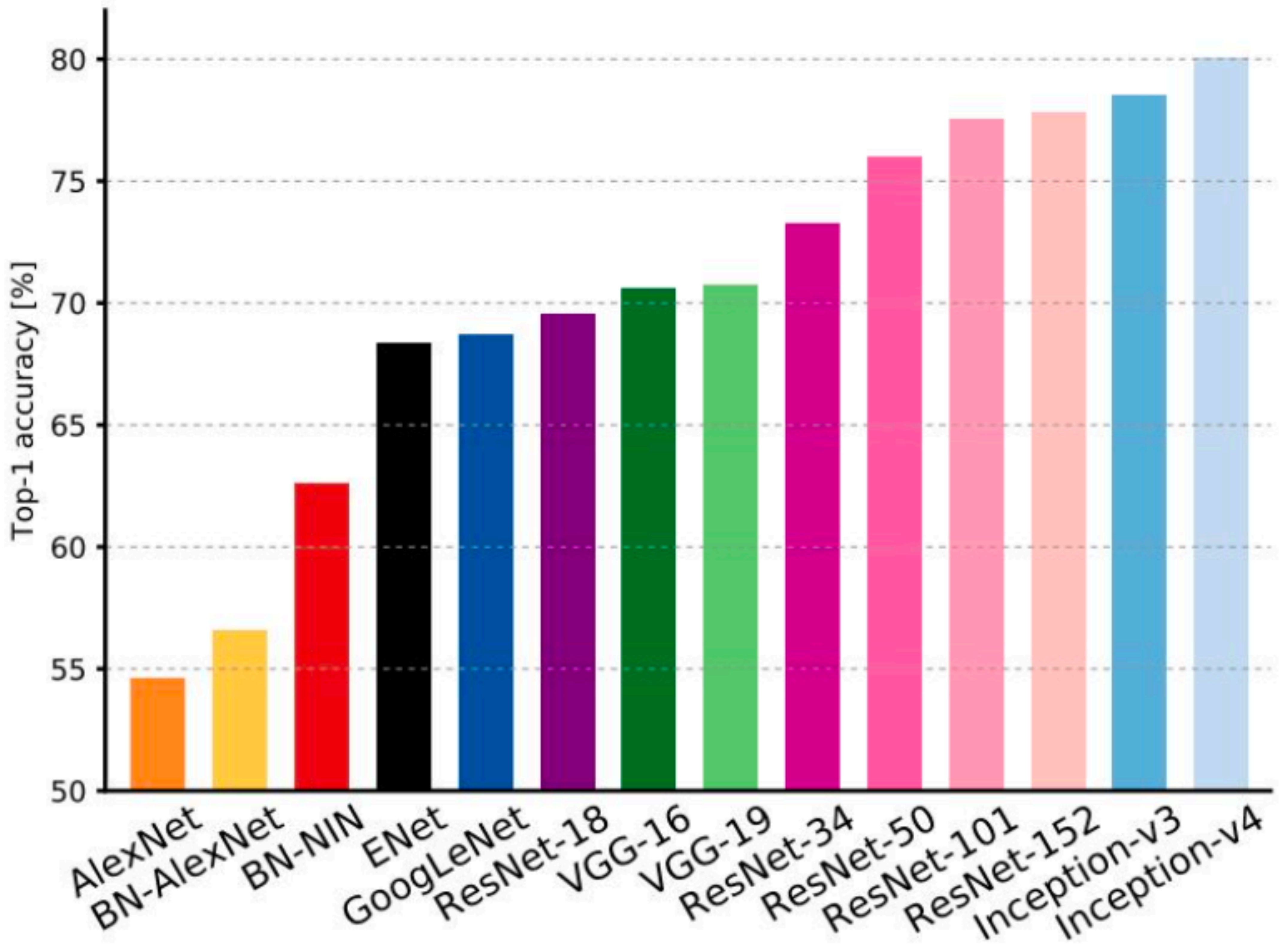


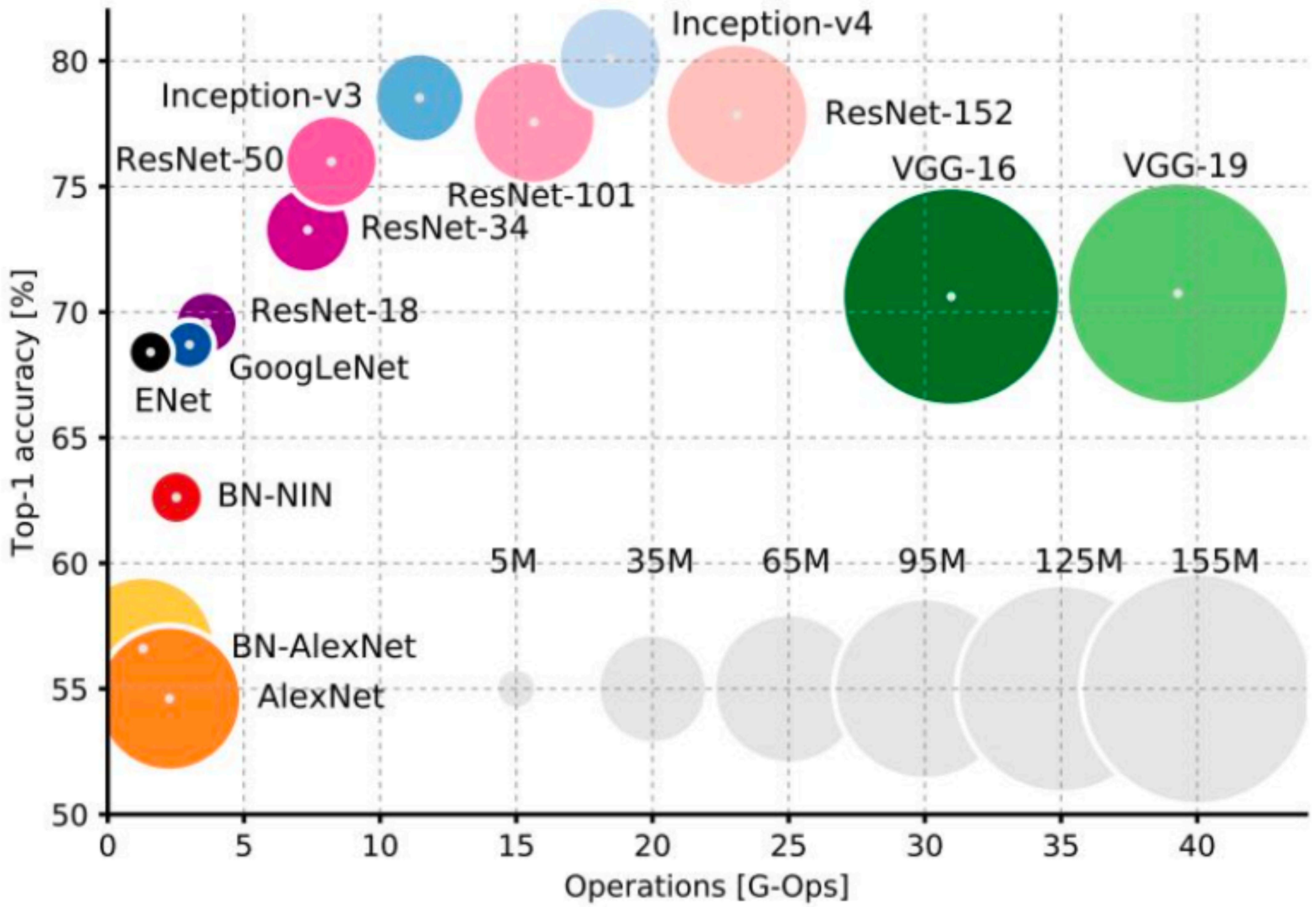
What happens if you take more layers and take smaller steps?

You can actually treat a neural network as an **ODE**: $$\frac{d\mathbf{h}(t)}{dt} = f(\mathbf{h}(t), t, \theta)$$

[Chen et al., NIPS 2018 **best paper**]

Comparing **Complexity**





An Analysis of Deep Neural Network Models for Practical Applications, 2017.

Computer **Vision Problems** (no language for now)

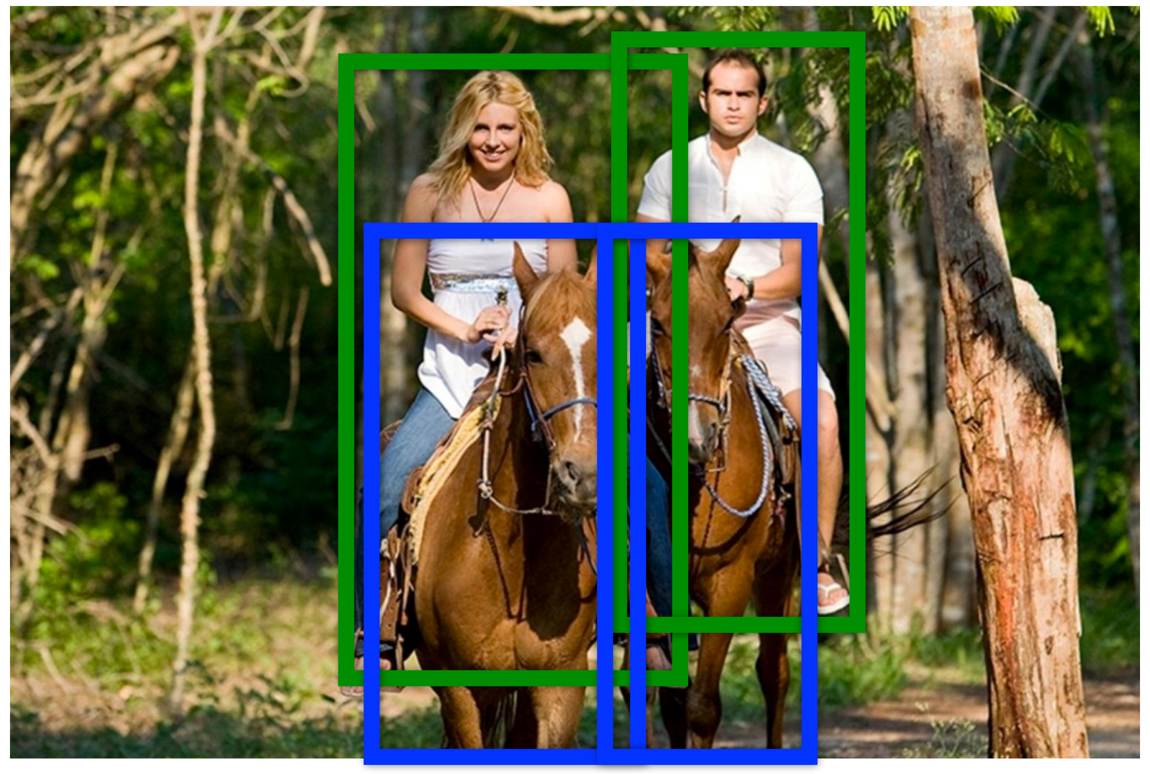
Categorization

Multi-**class:** Horse
Church
Toothbrush
Person

IMAGENET

Multi-**label**: **Horse**
Church
Toothbrush
Person

Detection



Horse (x, y, w, h)
Horse (x, y, w, h)
Person (x, y, w, h)
Person (x, y, w, h)

COCO
Common Objects in Context

Segmentation

Horse
Person

COCO
Common Objects in Context

Instance Segmentation

Horse1
Horse2
Person1
Person2

Computer **Vision Problems** (no language for now)

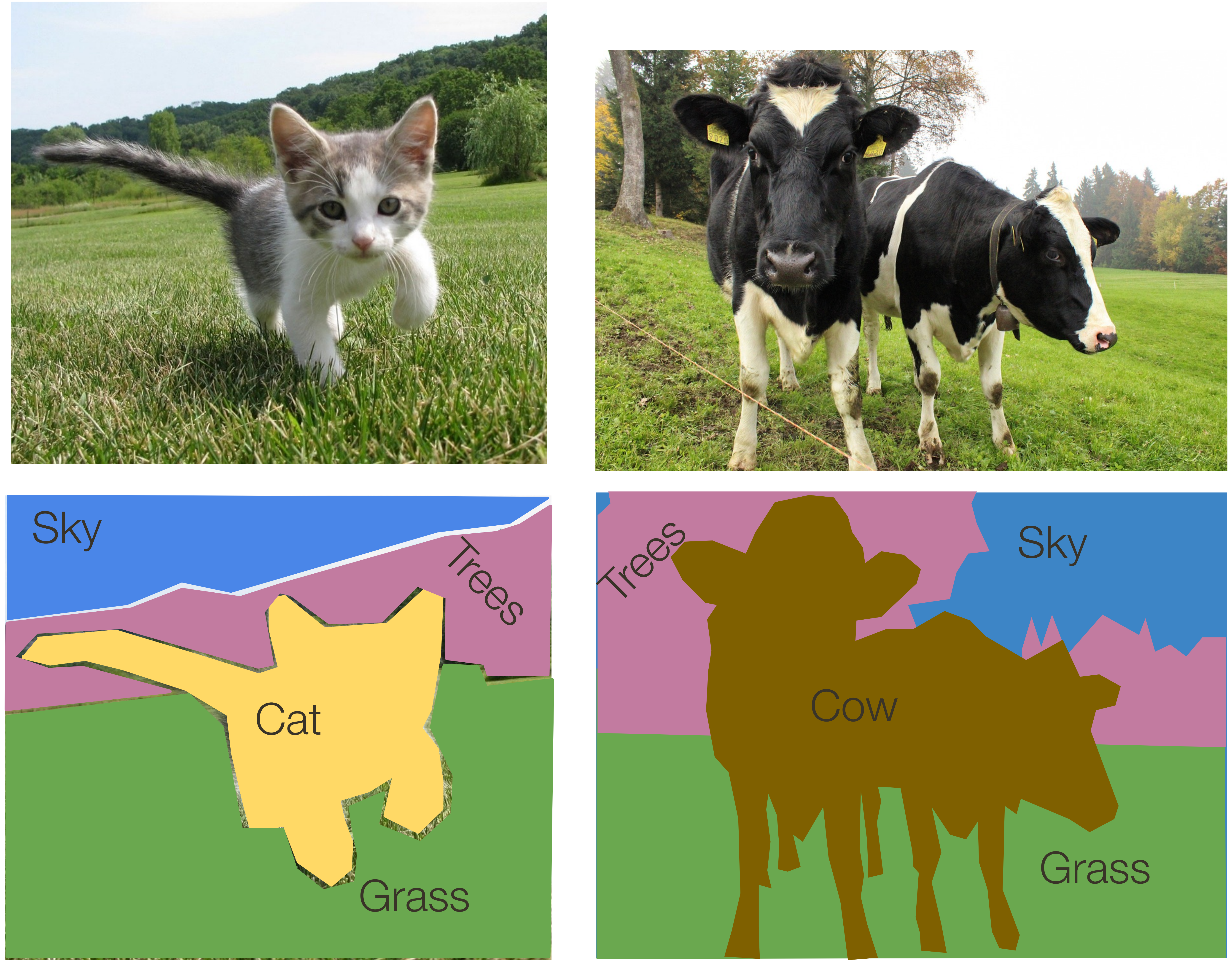
Segmentation

Horse
Person

COCO
Common Objects in Context

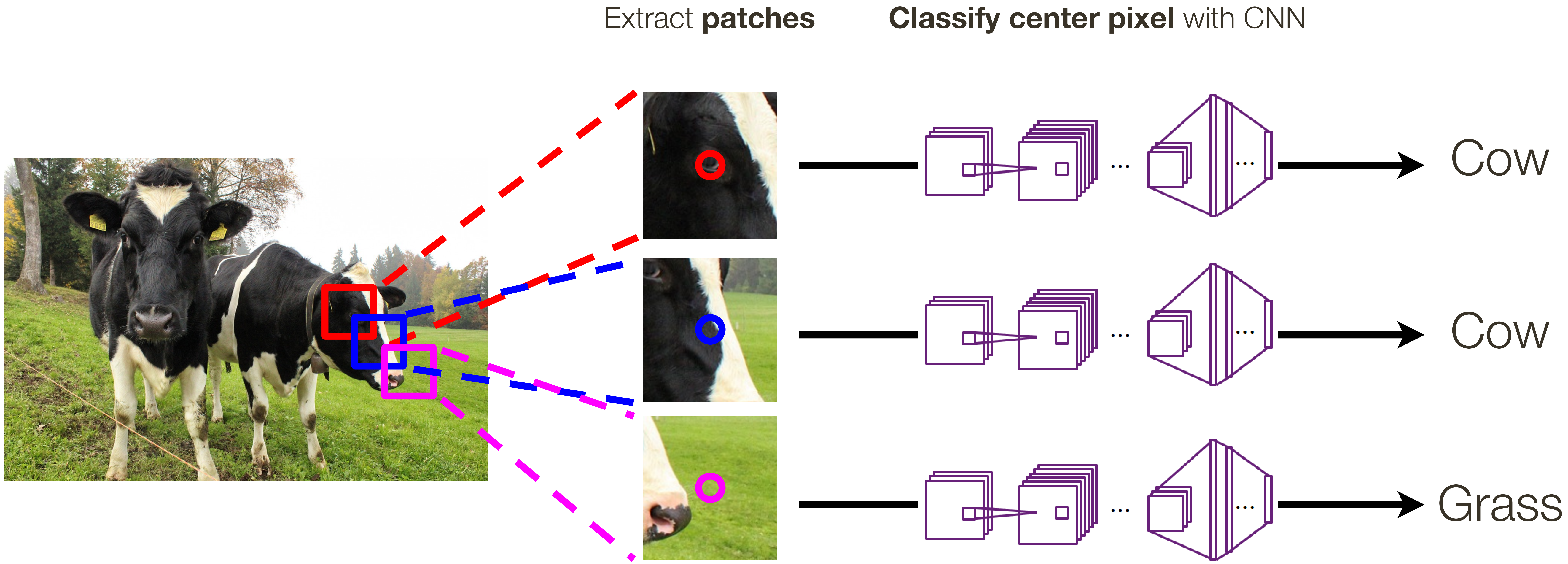
Semantic **Segmentation**

Label **every pixel** with a category label (without differentiating instances)



Semantic **Segmentation:** Sliding Window

[Farabet et al, TPAMI 2013]
[Pinheiro et al, ICML 2014]

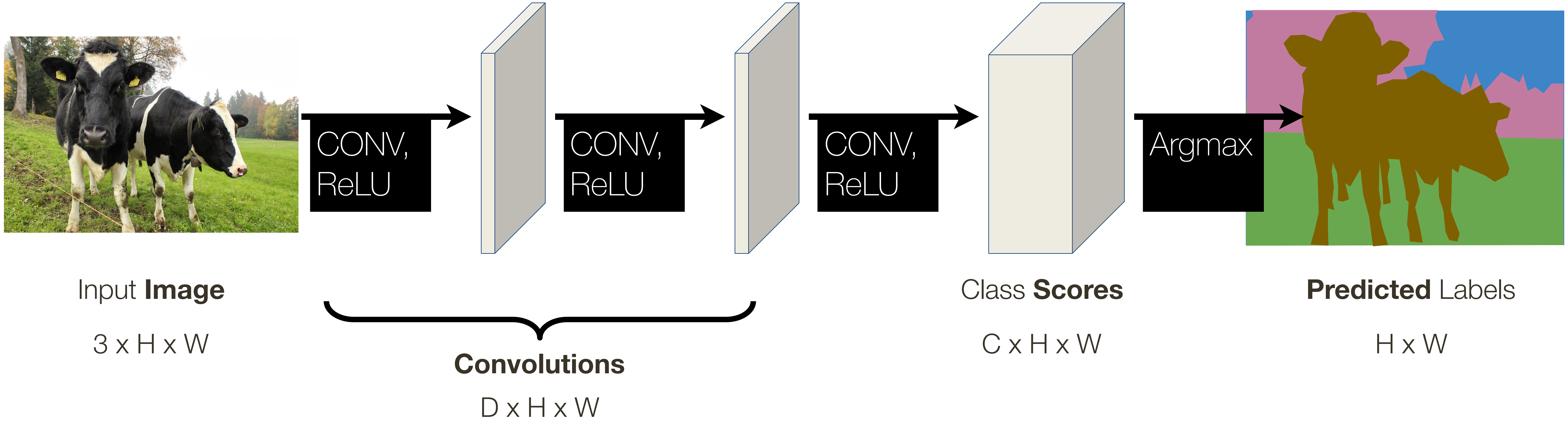


Problem: VERY inefficient, no reuse of computations for overlapping patches

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Semantic **Segmentation:** Fully Convolutional CNNs

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



Problem: Convolutions at the original image scale will be very expensive

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Semantic **Segmentation:** Fully Convolutional CNNs

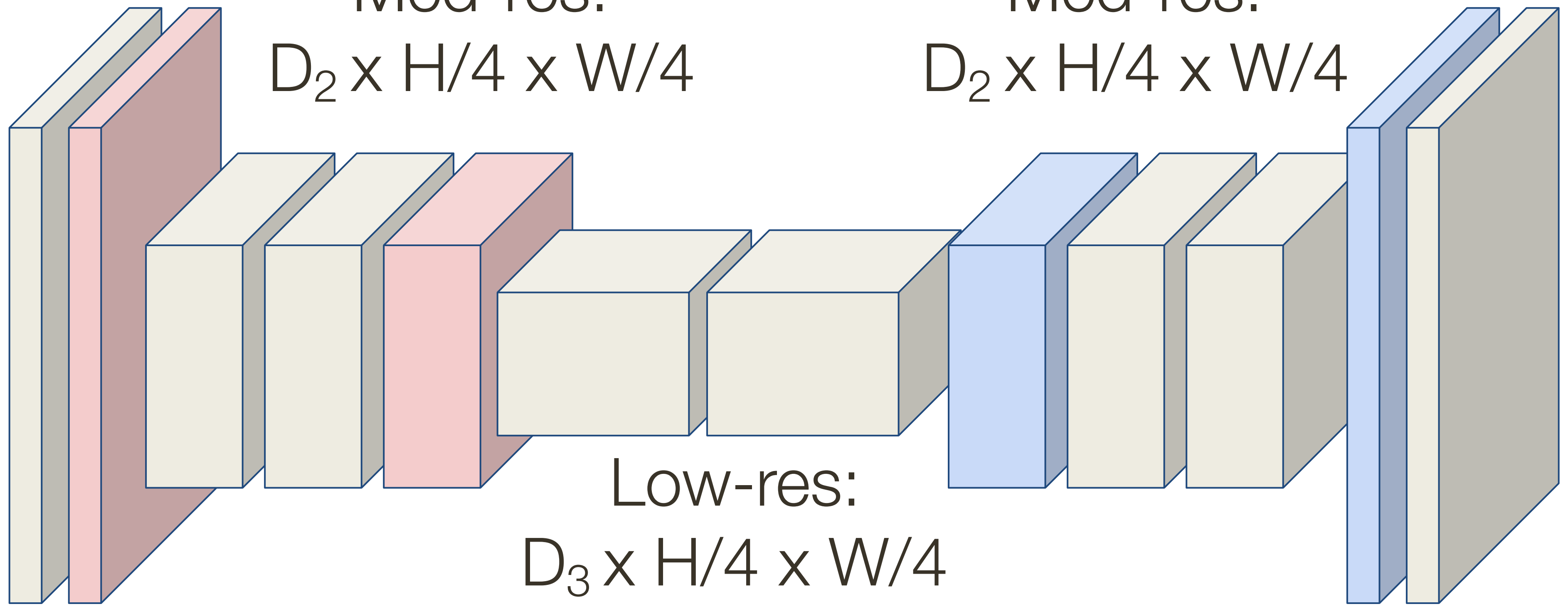
Design a network as a number of convolutional layers with **downsampling** and **upsampling** inside the network!

Input **Image**

3 x H x W

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

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



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

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

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

Predicted Labels

H x W

Downsampling = Pooling

Upsampling = ???

[Long et al, CVPR 2015]
[Noh et al, ICCV 2015]

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

In-network **Up Sampling** (a.k.a "Unpooling")

Nearest Neighbor

$$\begin{matrix}1 & 2\\ 3 & 4\end{matrix} \quad \longrightarrow \quad \begin{array}{cc|cc}1 & 1 & 2 & 2\\ 1 & 1 & 2 & 2\\ \hline 3 & 3 & 4 & 4\\ 3 & 3 & 4 & 4\end{array}$$

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

"Bed of Nails"

$$\begin{matrix}1 & 2\\ 3 & 4\end{matrix} \quad \longrightarrow \quad \begin{array}{cc|cc}1 & 0 & 2 & 0\\ 0 & 0 & 0 & 0\\ \hline 3 & 0 & 4 & 0\\ 0 & 0 & 0 & 0\end{array}$$

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

In-network **Up Sampling:** Max Unpooling

Max Pooling
Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4

→

5	6
7	8

Output: 2 x 2

→ … →

Rest of the network

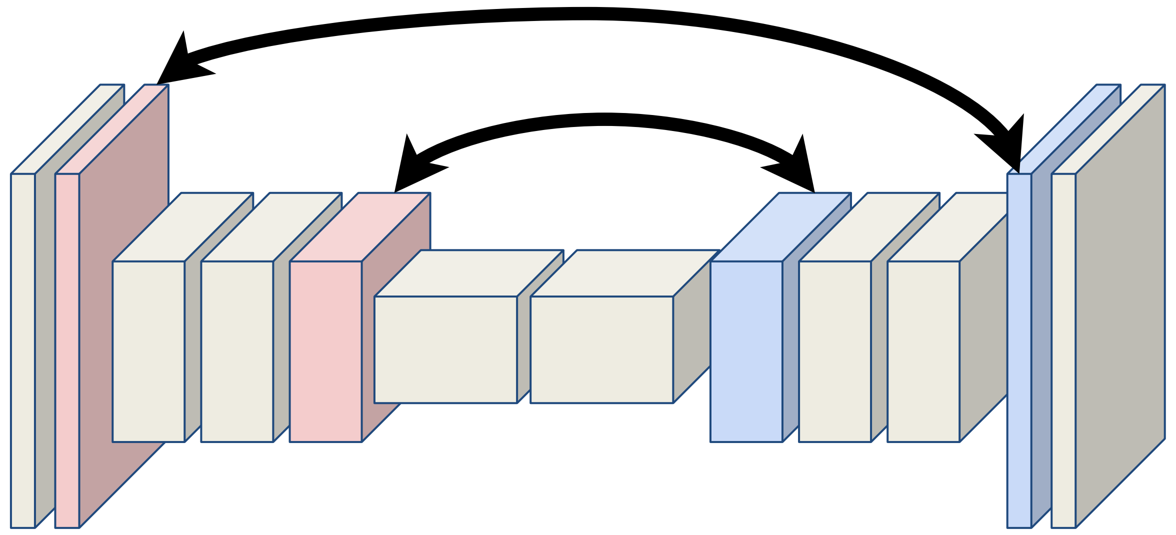
Max Unpooling
Use positions from pooling layer

1	2
3	4

Input: 2 x 2

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

Output: 4 x 4

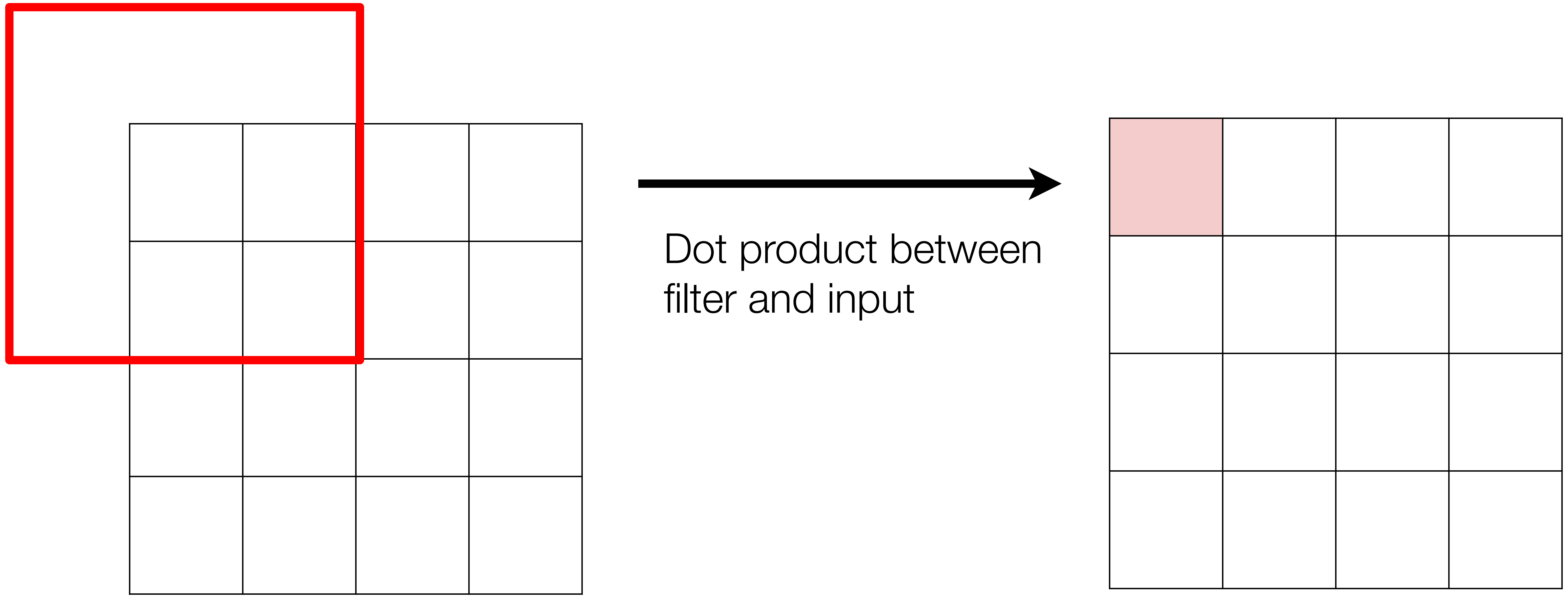


Corresponding pairs of downsampling and upsampling layers

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

In-network **Up Sampling:** Transpose Convolution

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



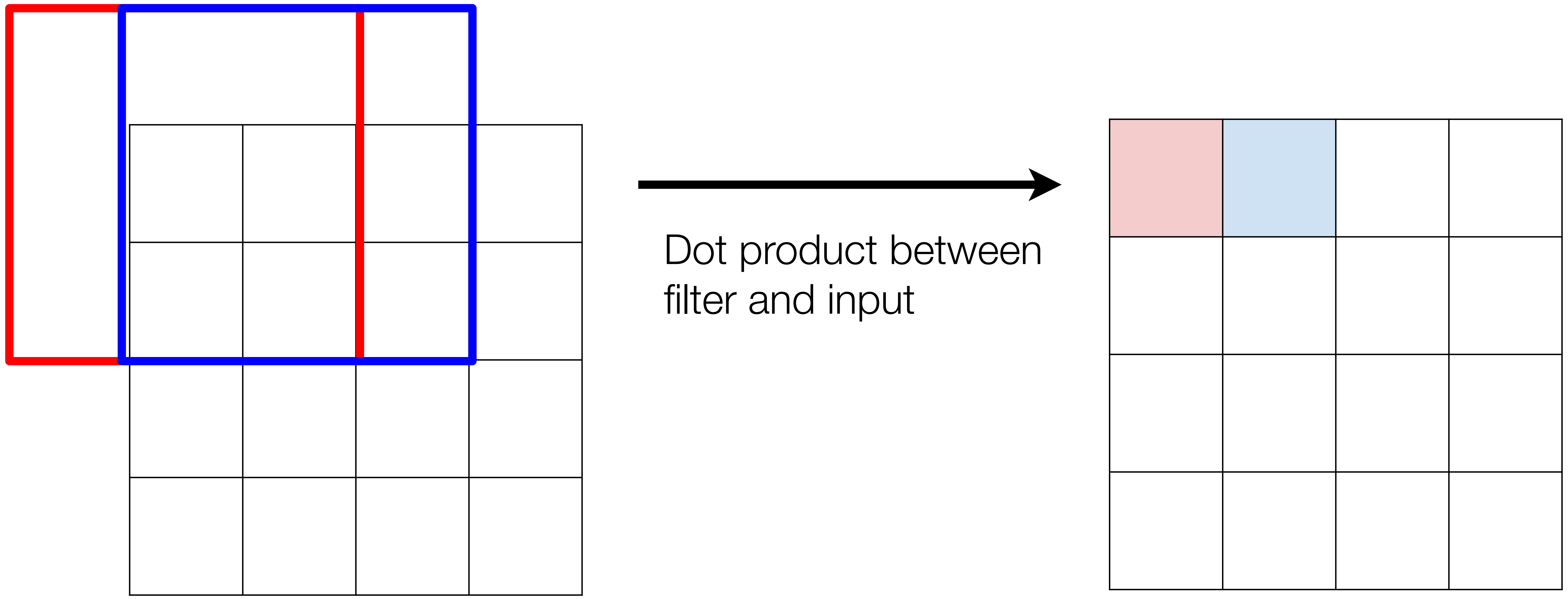
Input: 4 x 4

Output: 4 x 4

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

In-network **Up Sampling:** Transpose Convolution

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

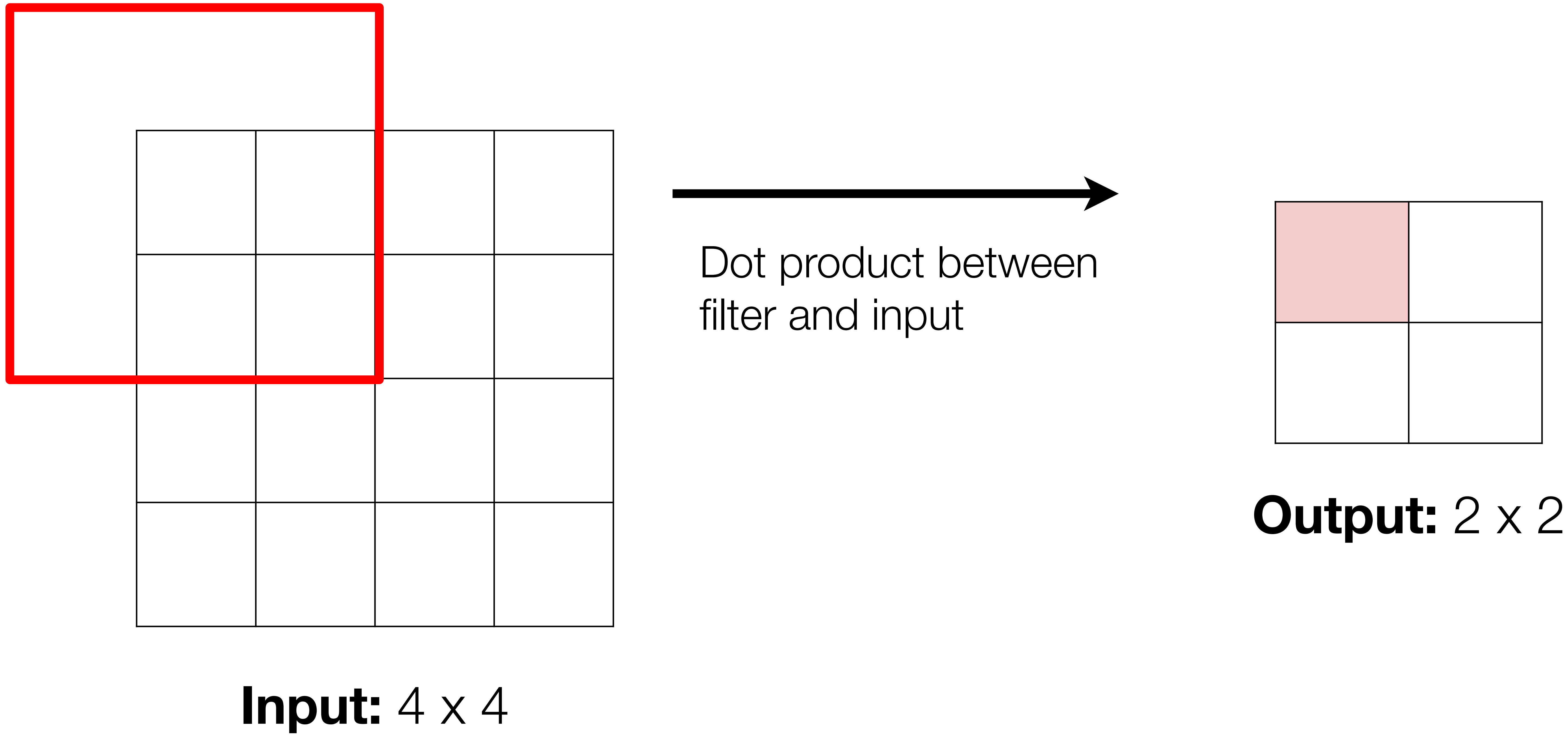


Input: 4 x 4

Output: 4 x 4

In-network **Up Sampling:** Transpose Convolution

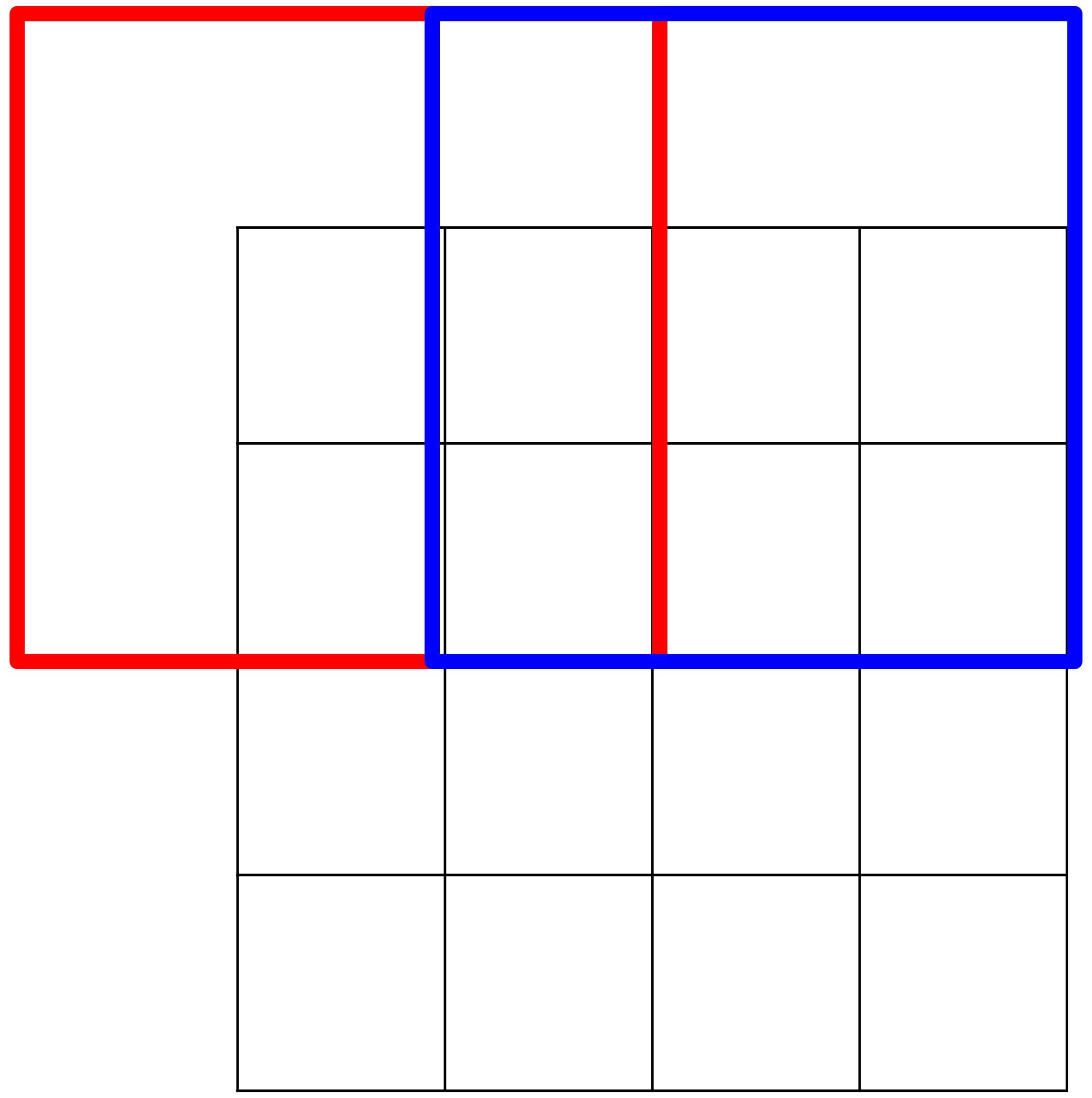
Recall: Normal 3 x 3 convolution, stride 2 pad 1



* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

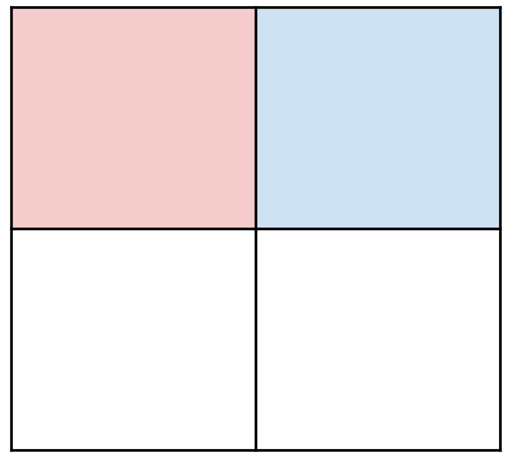
In-network **Up Sampling:** Transpose Convolution

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



Input: 4 x 4

Dot product between filter and input



Output: 2 x 2

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

Stride gives ratio in movement in input vs output

In-network **Up Sampling:** Transpose Convolution

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

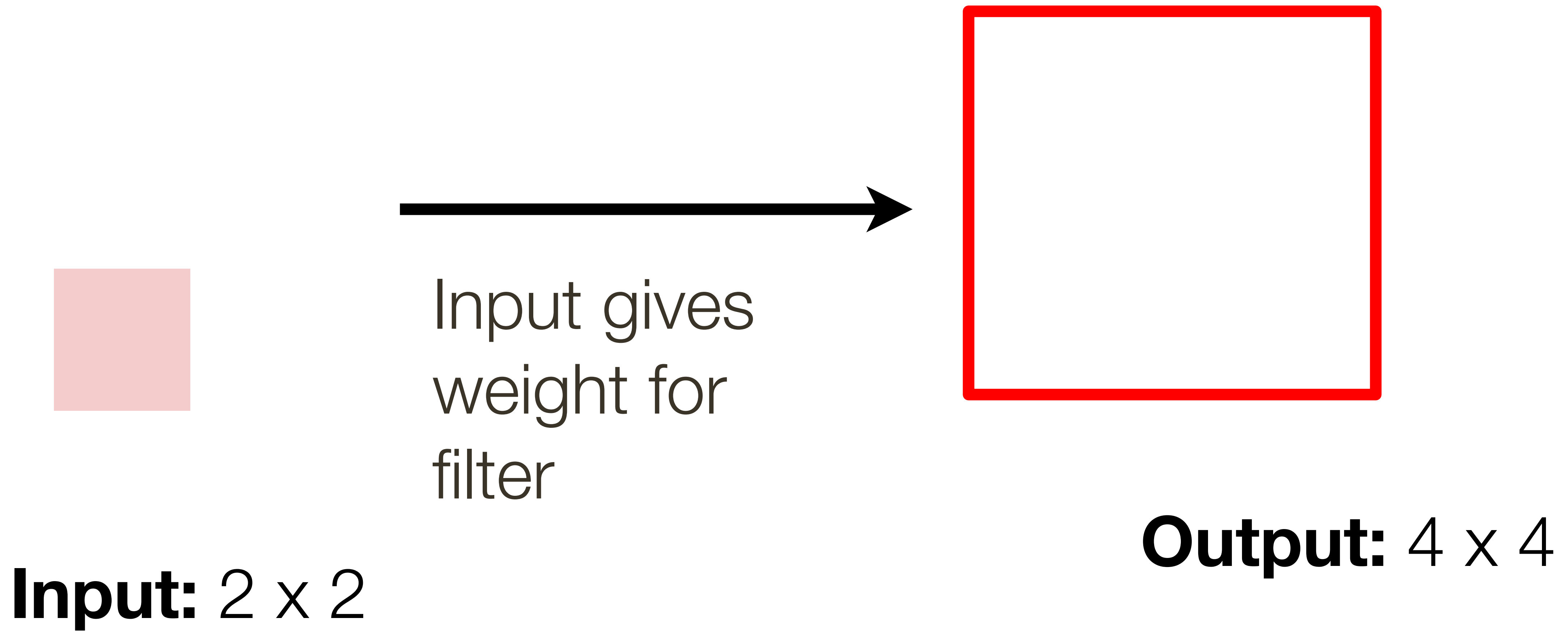
Input: 2 x 2

Output: 4 x 4

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

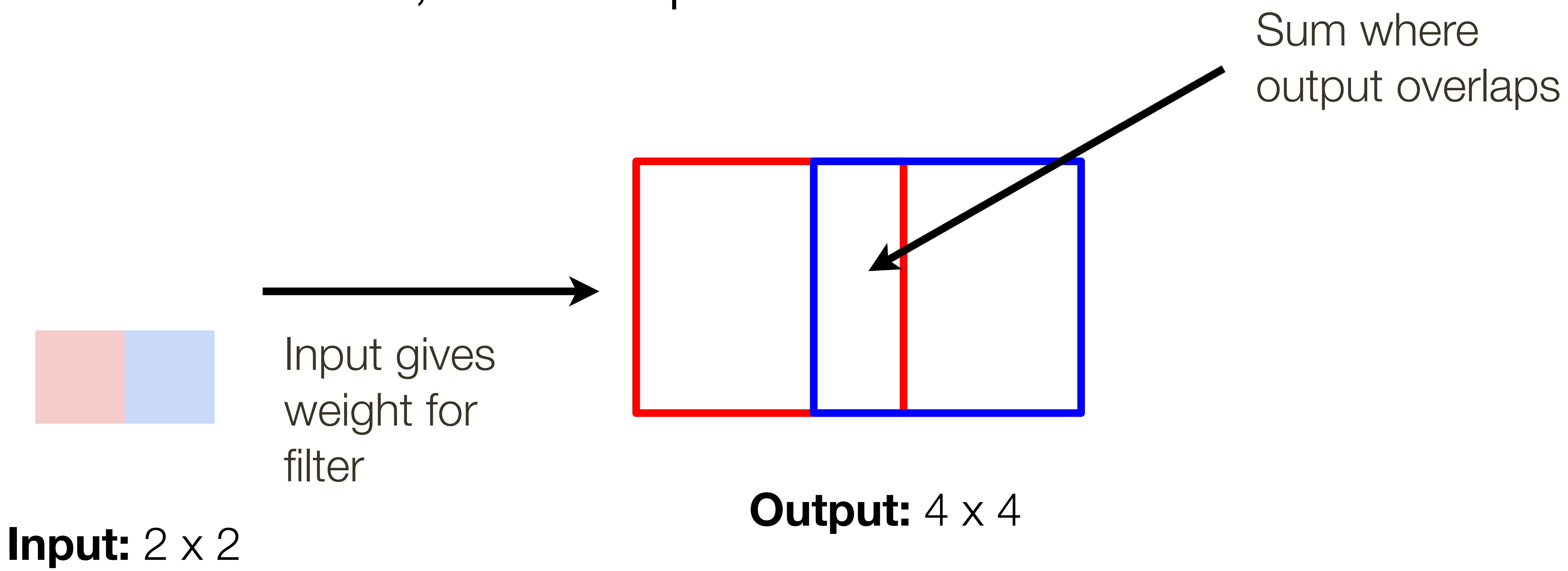
In-network **Up Sampling:** Transpose Convolution

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



In-network **Up Sampling:** Transpose Convolution

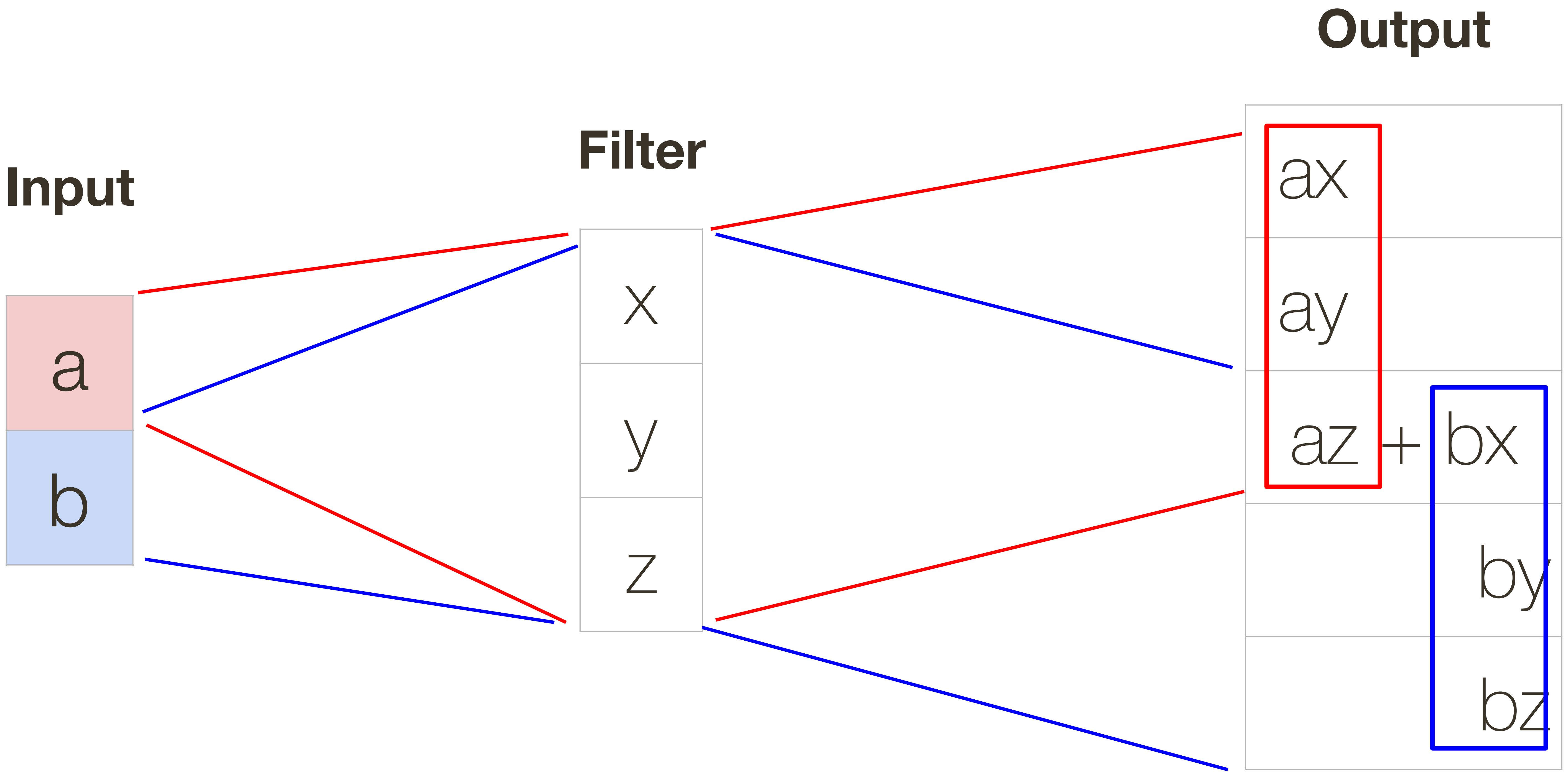
3 x 3 **transpose** convolution, stride 2 pad 1



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

Stride gives ratio in movement in output vs input

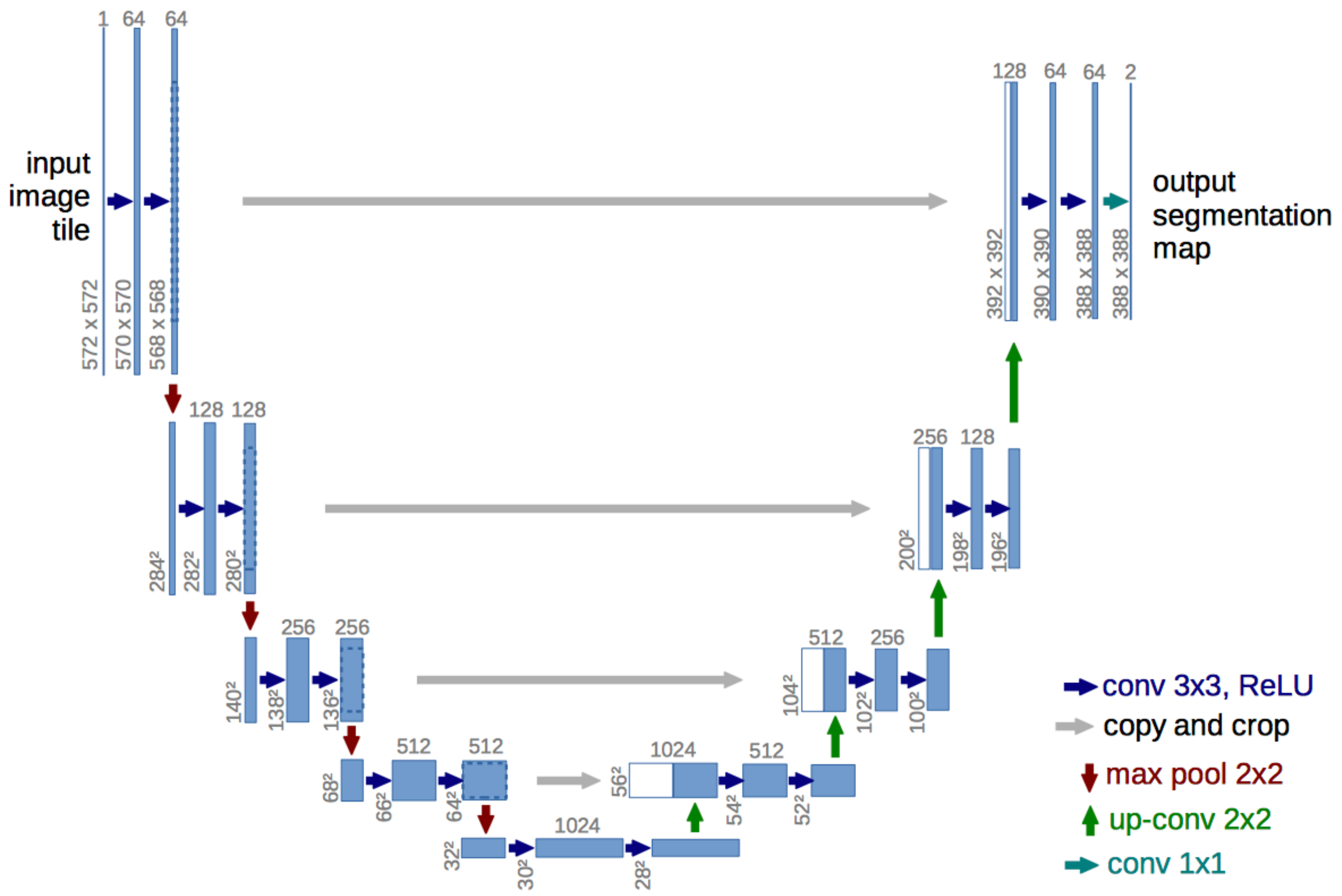
Transpose Convolution: 1-D Example



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

U-Net Architecture

ResNet-like Fully convolutional CNN



[Ronneberger et al, CVPR 2015]

Computer **Vision Problems** (no language for now)

Categorization

Multi-**class**:

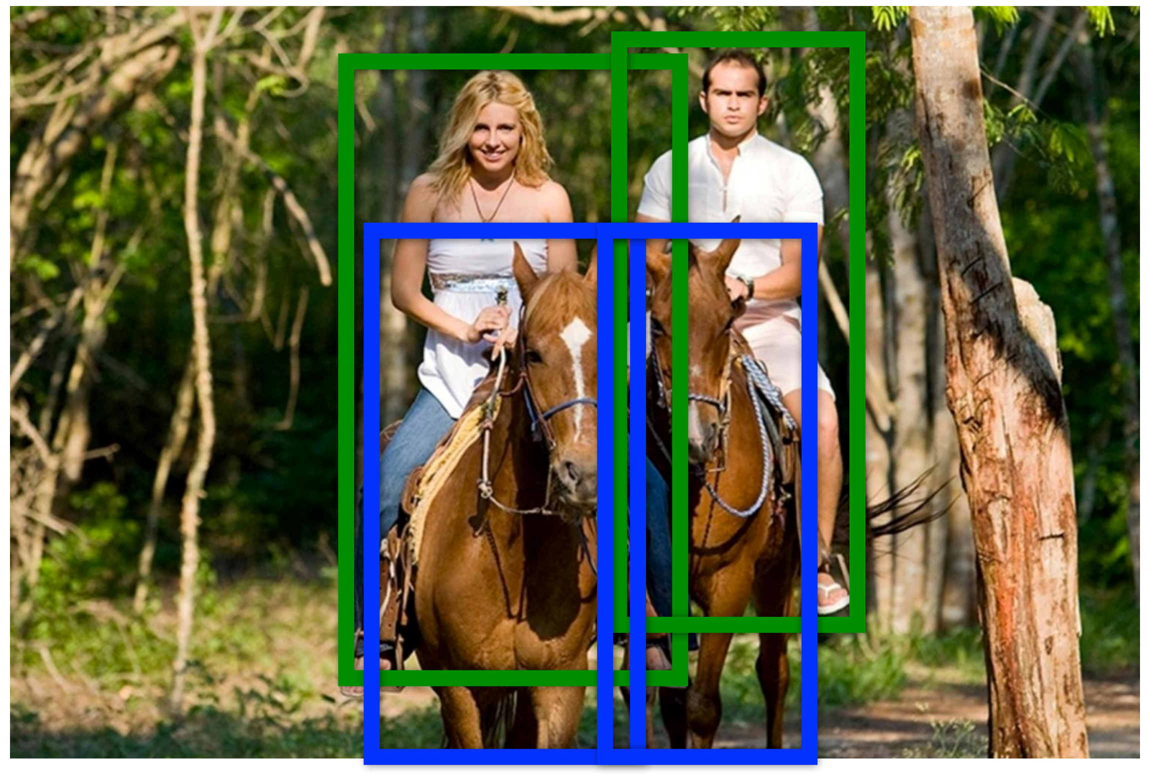
Horse
Church
Toothbrush
Person

IMAGENET

Multi-**label**:

Horse
Church
Toothbrush
Person

Detection



Horse (x, y, w, h)
Horse (x, y, w, h)
Person (x, y, w, h)
Person (x, y, w, h)

COCO Common Objects in Context

Segmentation

Horse
Person

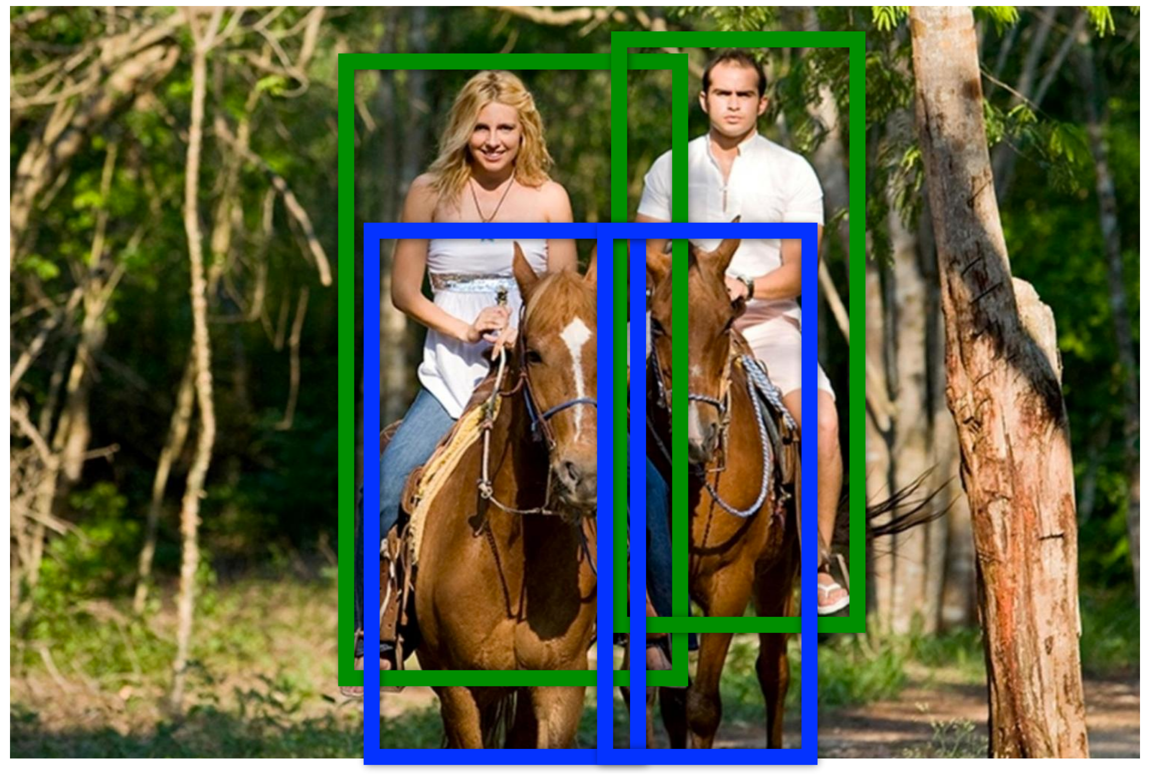
COCO Common Objects in Context

Instance Segmentation

Horse1
Horse2
Person1
Person2

Computer **Vision Problems** (no language for now)

Detection



Horse (x, y, w, h)
Horse (x, y, w, h)
Person (x, y, w, h)
Person (x, y, w, h)

COCO
Common Objects in Context

Datasets: Pascal VOC

20 classes: aeroplane, bicycle, boat, bottle, bus, car, cat, chair, cow, dining table, dog, horse, motorbike, person, potted plant, sheep, train, TV

	Training	Testing
Images	10,103	9,637
Objects	23,374	22,992

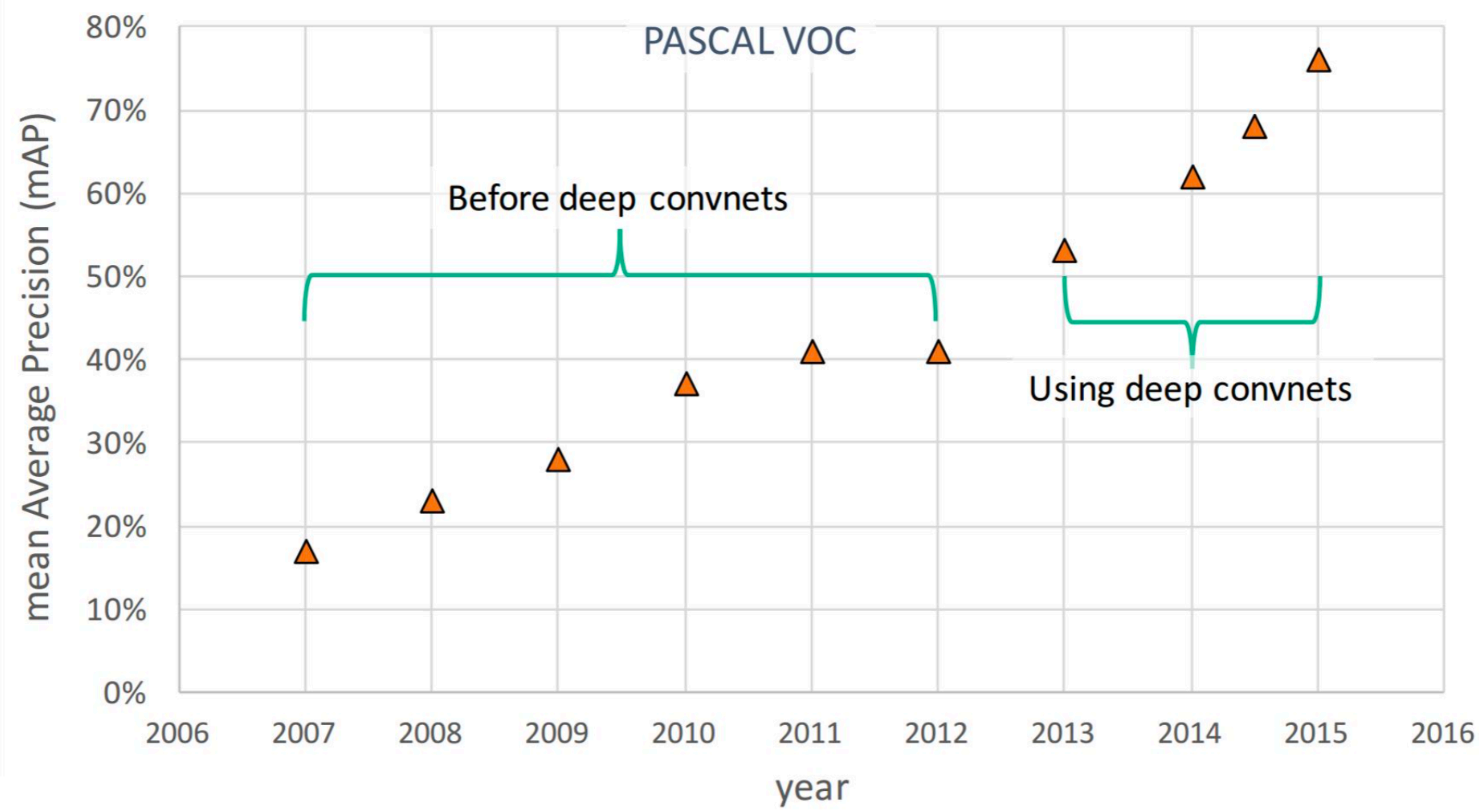
Real images downloaded from flickr, not filtered for “quality”

* slide from Andrew Zisserman

Datasets: COCO

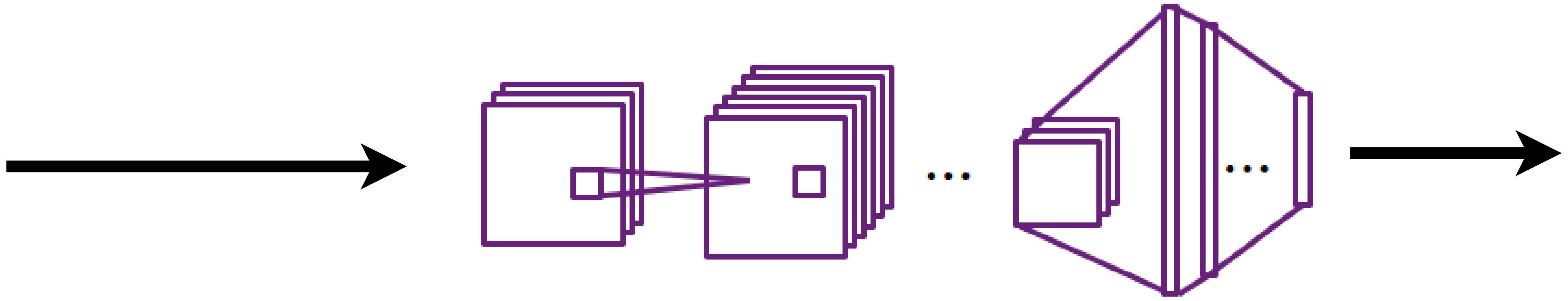
- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints

Object **Detection**



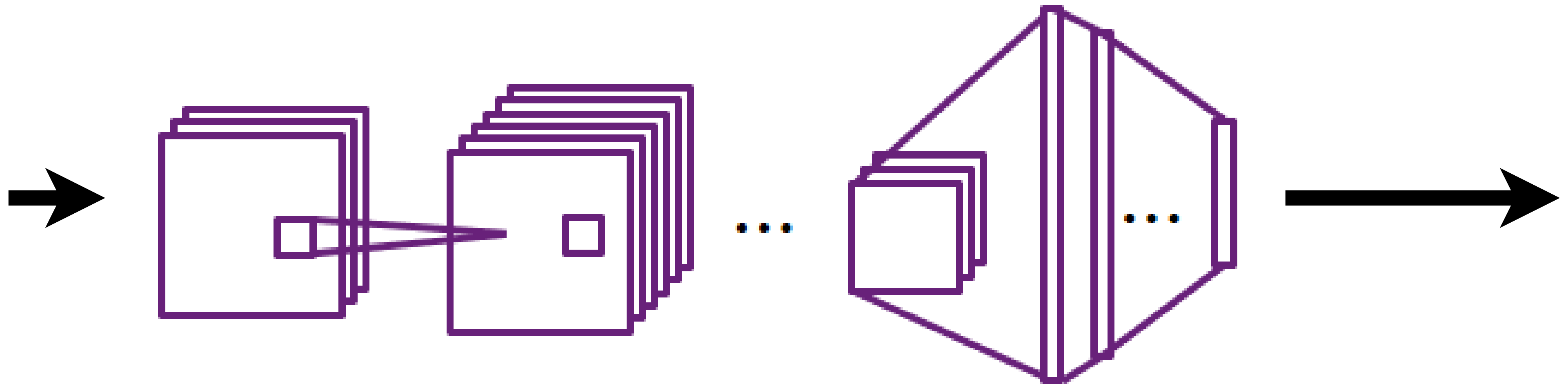
* plot from Ross Girshick, 2015

Object **Detection** as Regression Problem



CAT (x, y, w ,h)

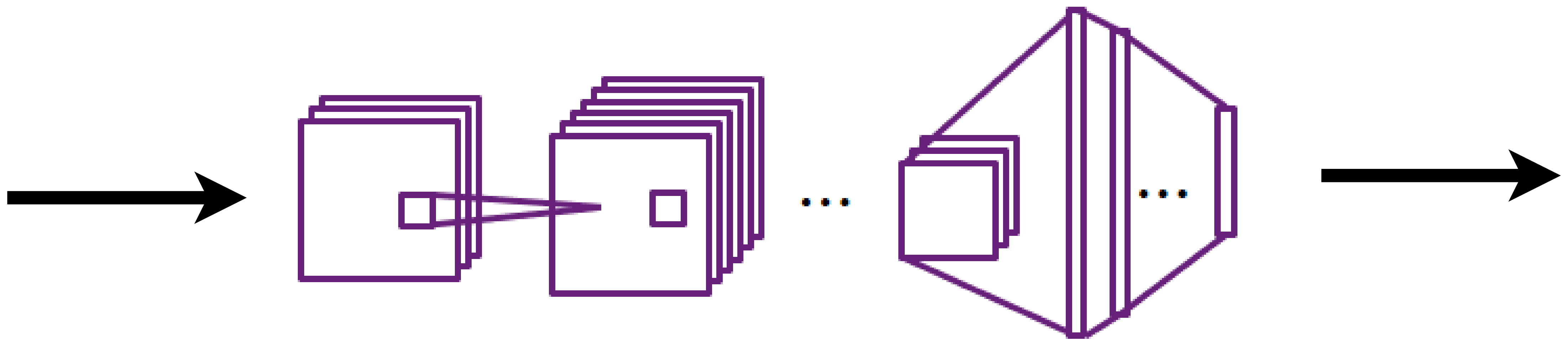
Problem: each image needs a different number of outputs



DUCK (x, y, w ,h)
DUCK (x, y, w ,h)
DUCK (x, y, w ,h)
DUCK (x, y, w ,h)
DUCK (x, y, w ,h)
DUCK (x, y, w ,h)
DUCK (x, y, w ,h)
…

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Object **Detection** as Classification Problem

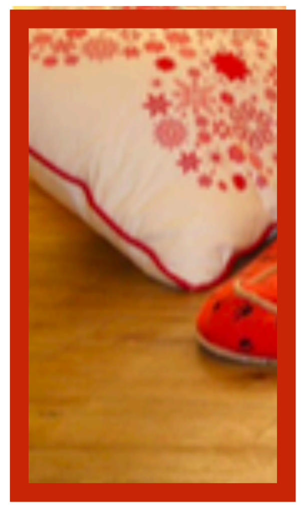


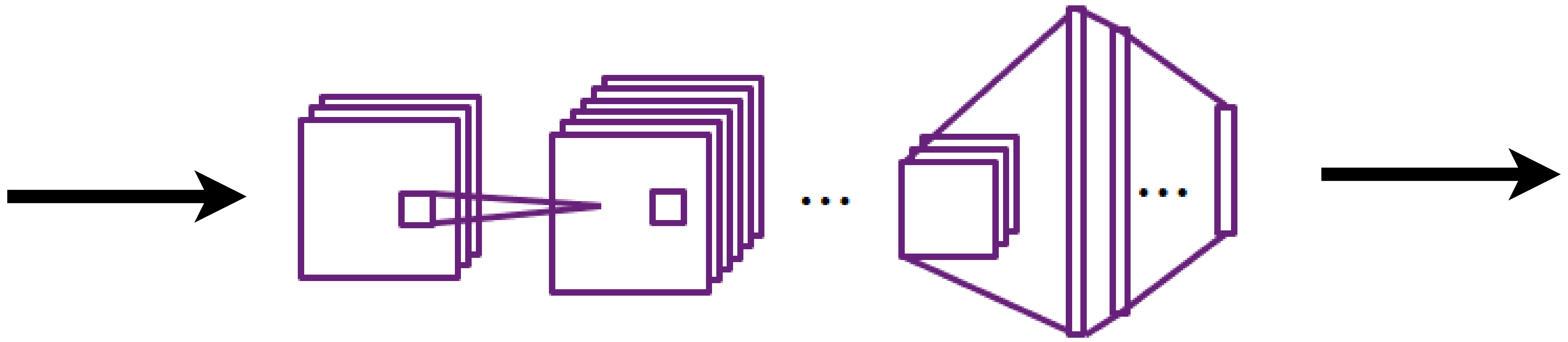
Category	**Prediction**
Dog	No
Cat	No
Couch	No
Flowers	No
Background	**Yes**
…	…

Apply CNN to many different crops in the image and (classification) CNN classifies each patch as object or background

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Object **Detection** as Classification Problem



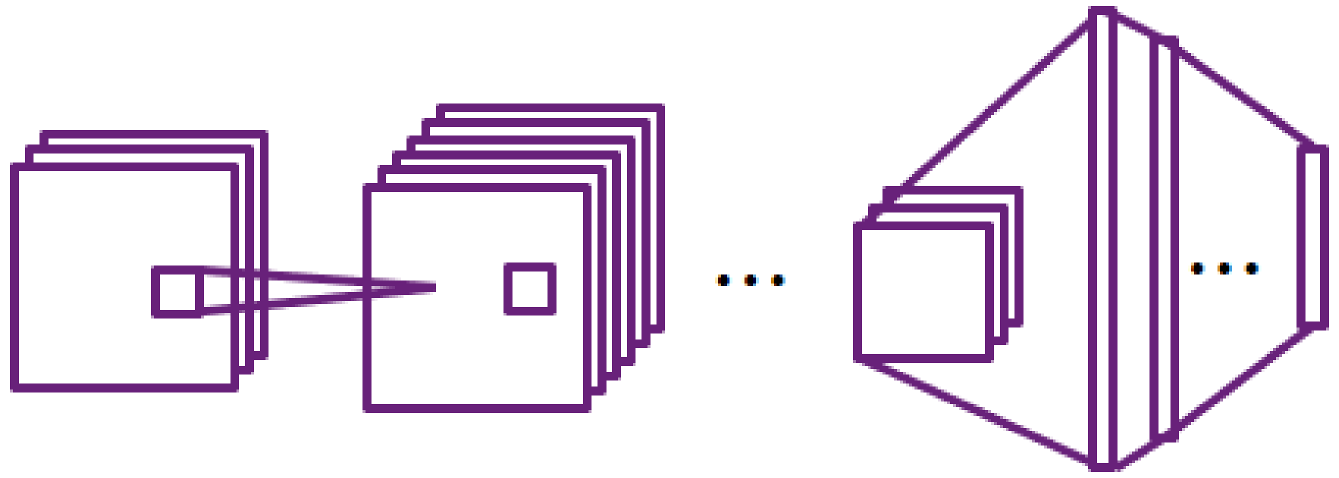


Category	**Prediction**
Dog	No
Cat	No
Couch	No
Flowers	No
Background	**Yes**
…	…

Apply CNN to many different crops in the image and (classification) CNN classifies each patch as object or background

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

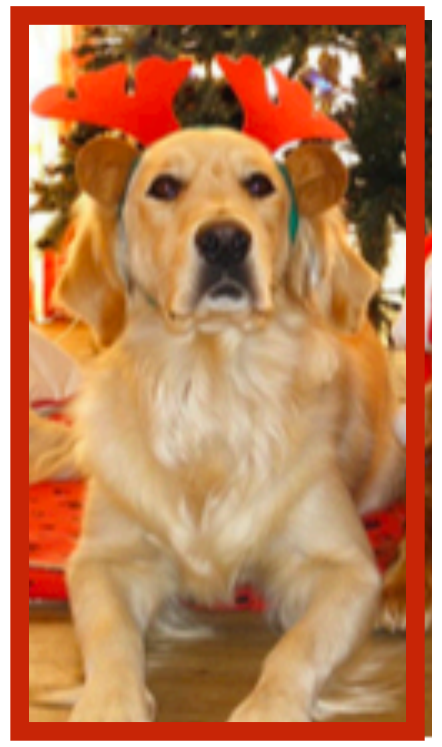
Object **Detection** as Classification Problem

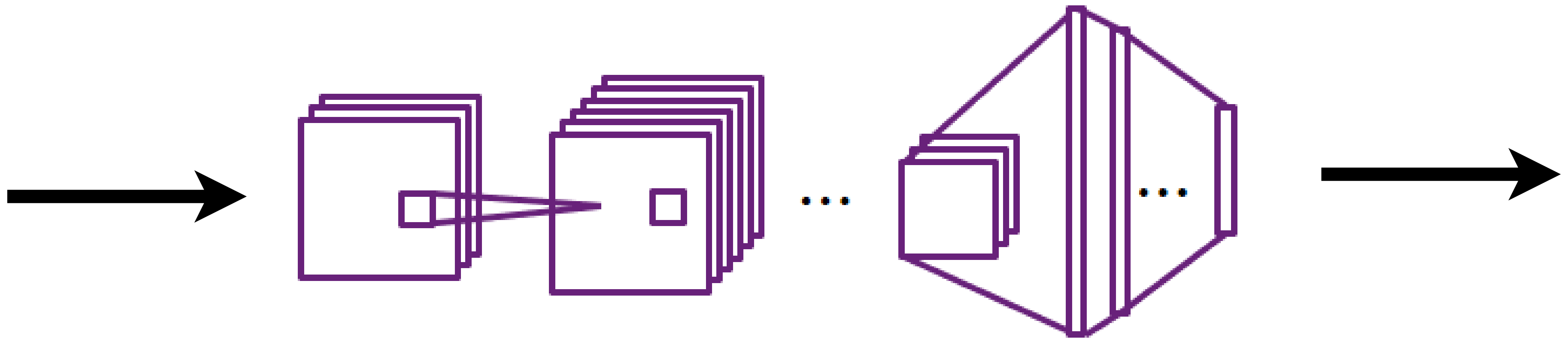


Category	**Prediction**
Dog	**Yes**
Cat	No
Couch	No
Flowers	No
Background	No
…	…

Apply CNN to many different crops in the image and (classification) CNN classifies each patch as object or background

Object **Detection** as Classification Problem





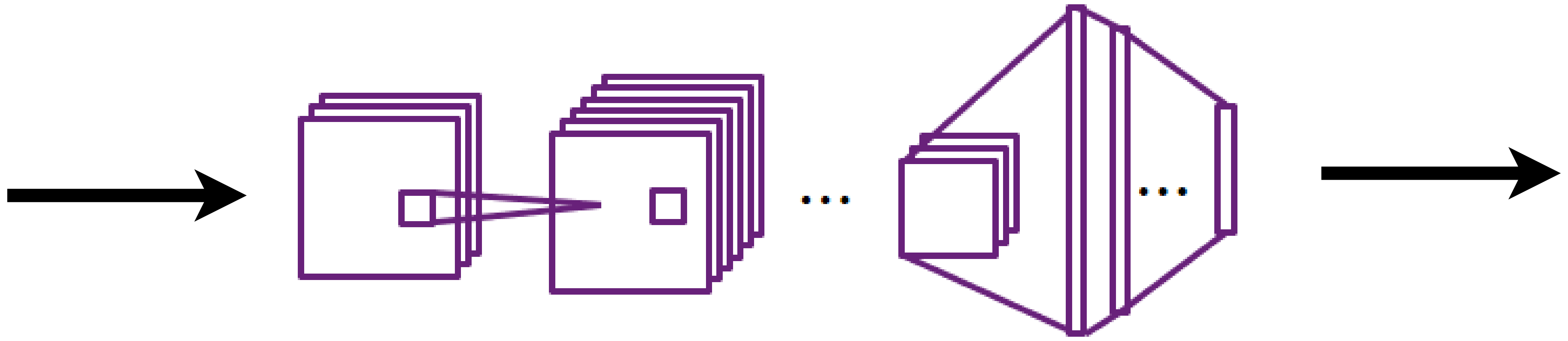
Category	**Prediction**
Dog	**Yes**
Cat	No
Couch	No
Flowers	No
Background	No
…	…

Apply CNN to many different crops in the image and (classification) CNN classifies each patch as object or background

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

Object **Detection** as Classification Problem

Problem: Need to apply CNN to **many** patches in each image



Category	**Prediction**
Dog	No
Cat	**Yes**
Couch	No
Flowers	No
Background	No
…	…

Apply CNN to many different crops in the image and (classification) CNN classifies each patch as object or background

* slide from Fei-Dei Li, Justin Johnson, Serena Yeung, **cs231n Stanford**

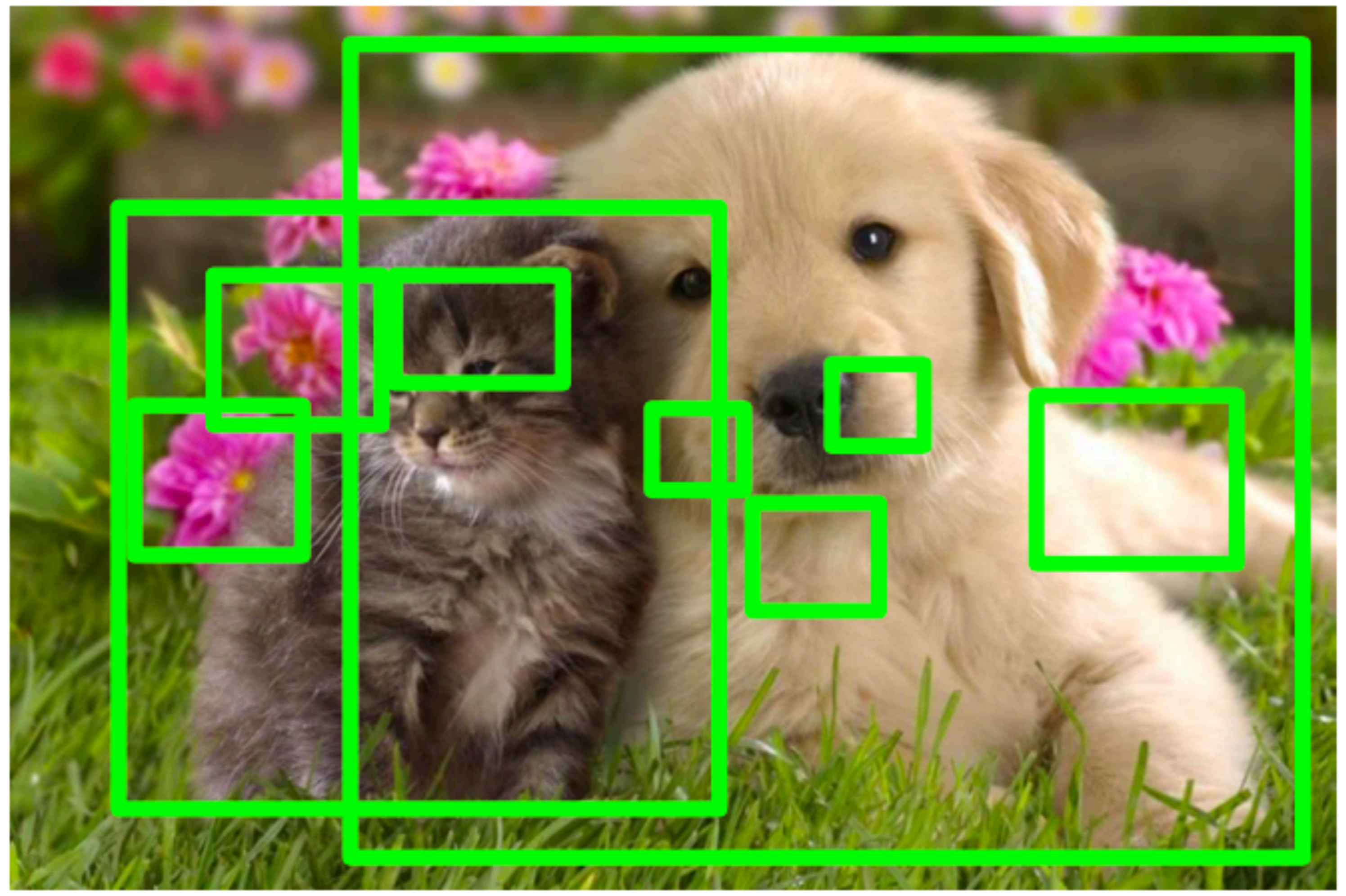
Region Proposals (older idea in vision)

[Alexe et al, TPAMI 2012]
[Uijkings et al, IJCV 2013]
[Cheng et al, CVPR 2014]
[Zitnick and Dollar, ECCV 2014]

Find image **regions that are likely contain objects** (any object at all)

- typically works by looking at histogram distributions, region aspect ratio, closed contours, coherent color

Relatively **fast to run** (Selective Search gives 1000 region proposals in a few seconds on a CPU)



Goal: Get "true" object regions to be in as few top K proposals as possible

R-CNN

[Girshick et al, CVPR 2014]

Input **Image**

* image from Ross Girshick

R-CNN

[Girshick et al, CVPR 2014]

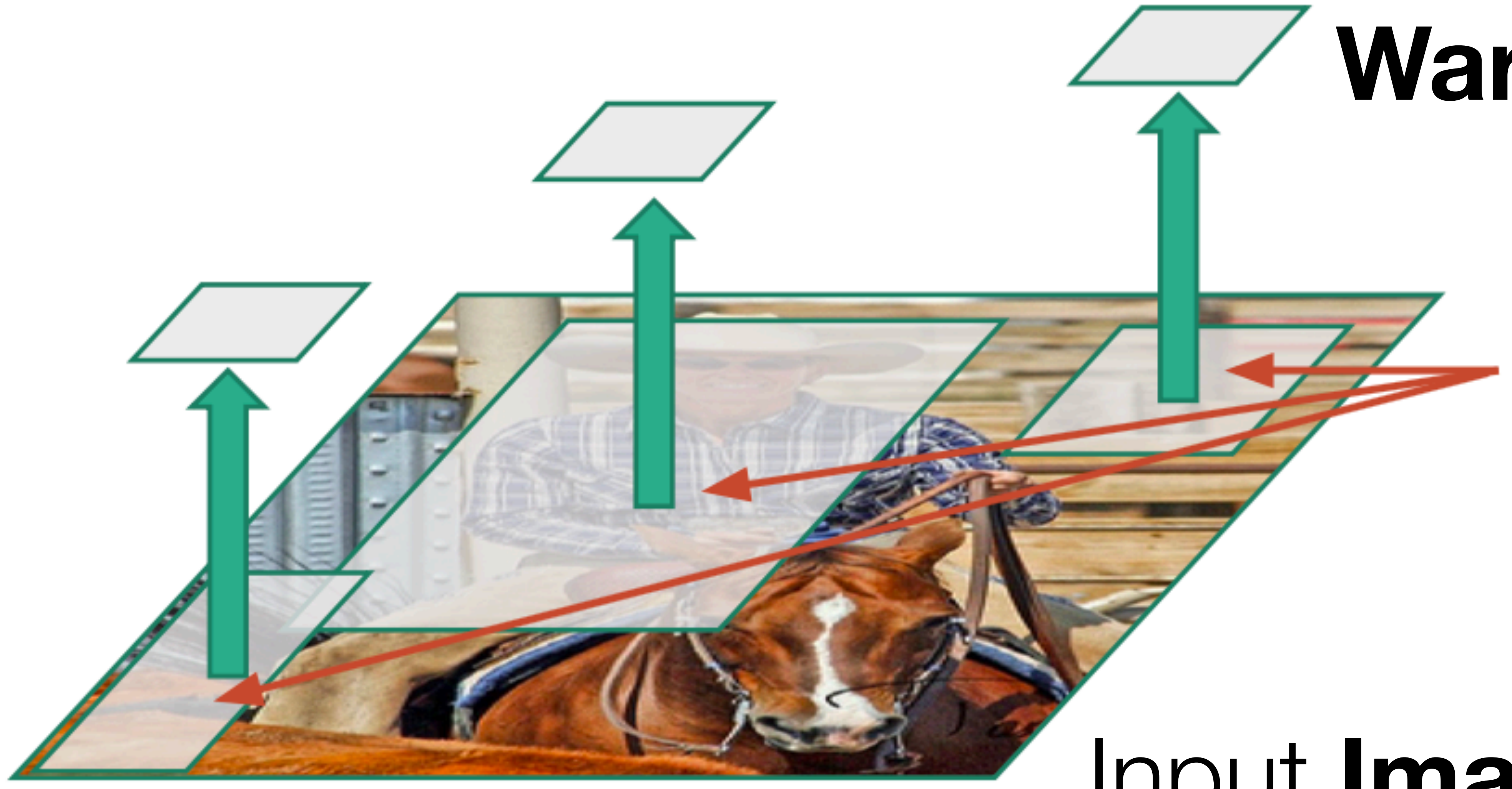
Regions of Interest from a proposal method (~2k)

Input **Image**

* image from Ross Girshick

R-CNN

[Girshick et al, CVPR 2014]



Warped image regions

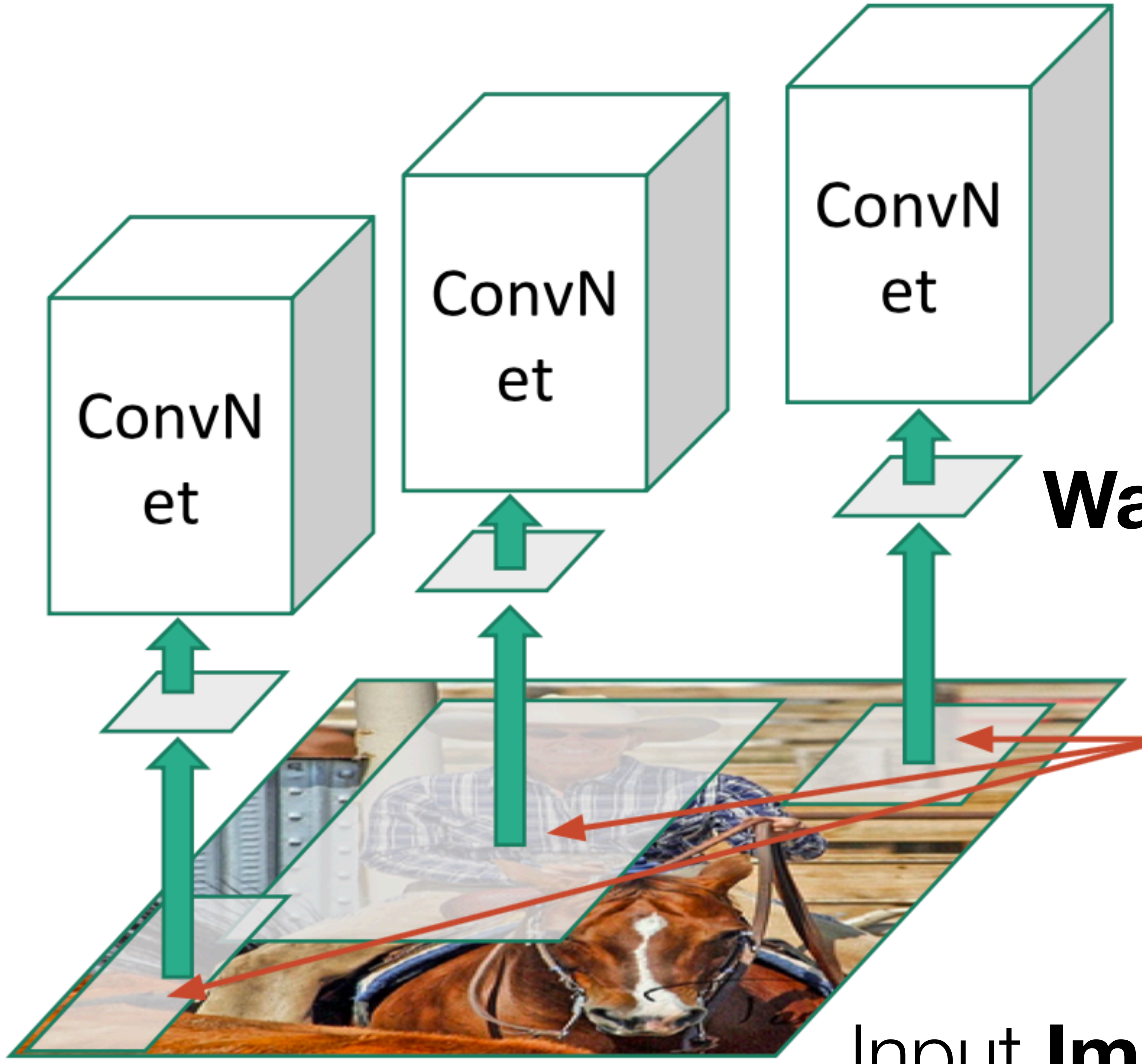
Regions of Interest from a proposal method (~2k)

Input **Image**

* image from Ross Girshick

R-CNN

[Girshick et al, CVPR 2014]



Forward each region through a **CNN**

Warped image regions

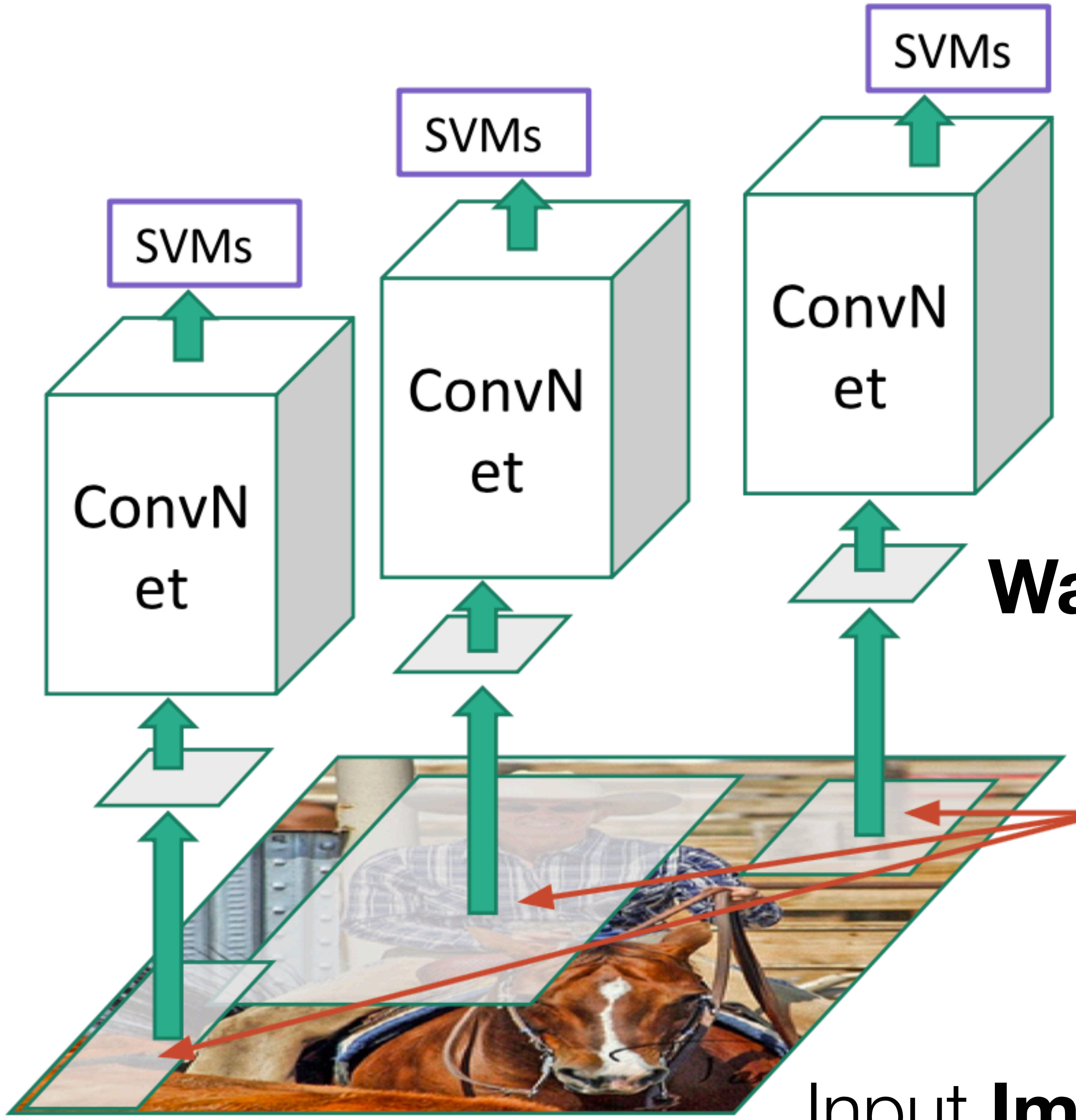
Regions of Interest from a proposal method (~2k)

Input **Image**

* image from Ross Girshick

R-CNN

[Girshick et al, CVPR 2014]



Classify regions with SVM

Forward each region through a **CNN**

Warped image regions

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

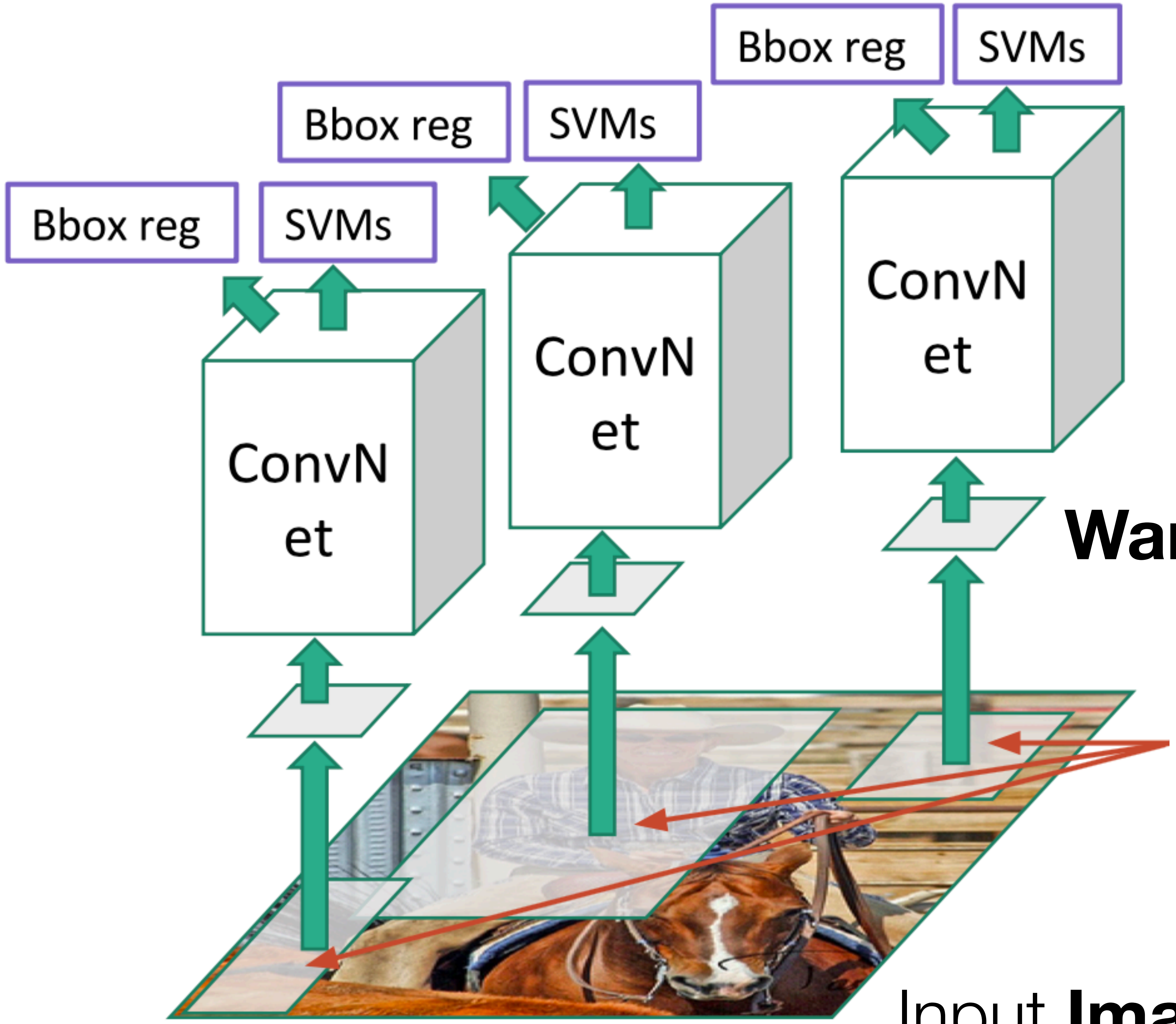
Input **Image**

* image from Ross Girshick

R-CNN

Linear Regression for bounding box offsets

[Girshick et al, CVPR 2014]



Classify regions with SVM

Forward each region through a **CNN**

Warped image regions

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

Input **Image**

* image from Ross Girshick

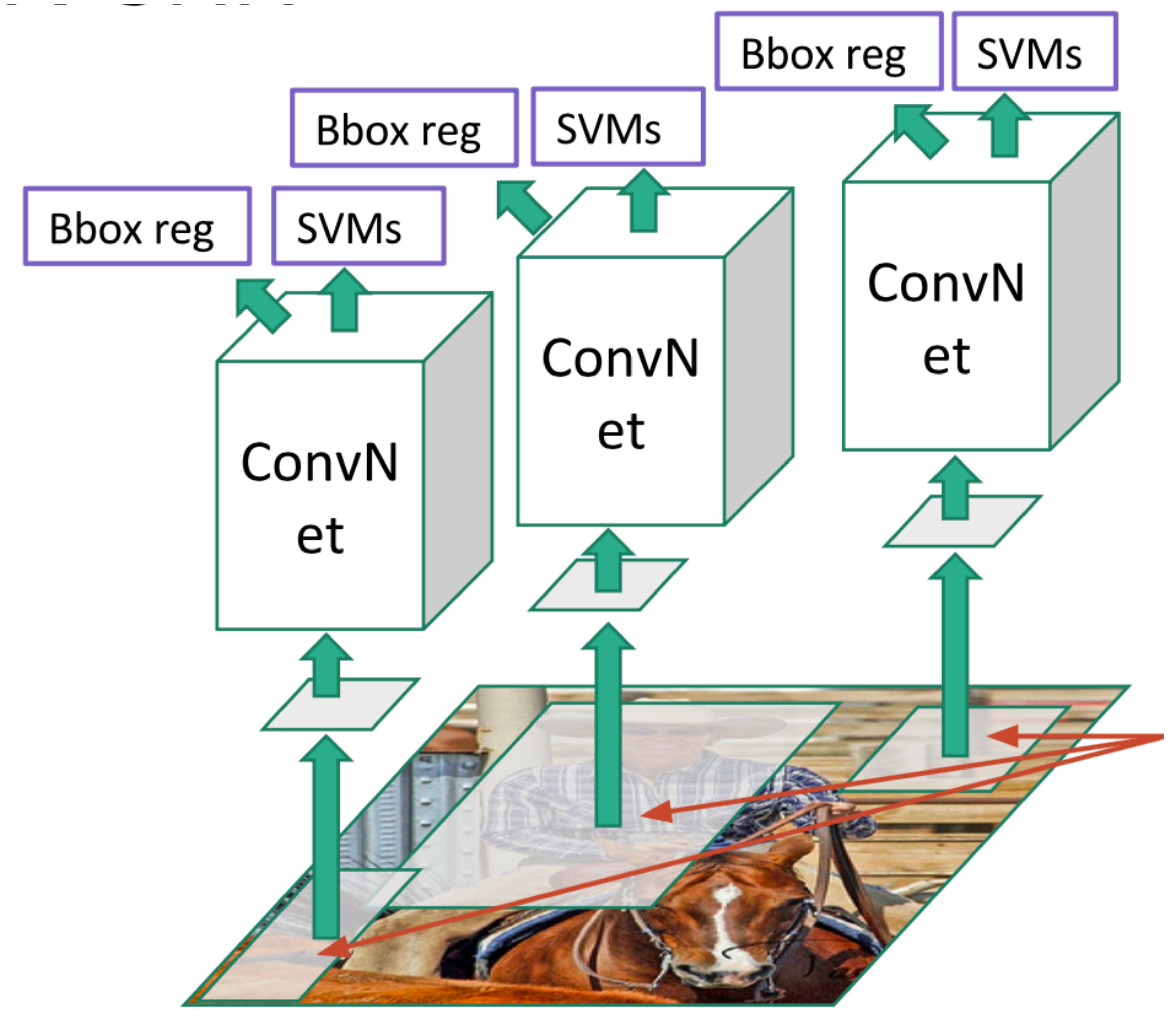
R-CNN: Training

[Girshick et al, CVPR 2014]

Fine-tuning ImageNet CNN on object proposal patches

— > 50% Intersection-over-Union overlap with GT considered "object" others "background"

— batches of 128 (**32 positives, 96 negatives**)



* image from Ross Girshick

R-CNN: Issues

[Girshick et al, CVPR 2014]

Ad-hoc training objectives

— Fine-tune network with softmax objective (**log** loss)

— Train post-hoc linear SVM (**hinge** loss)

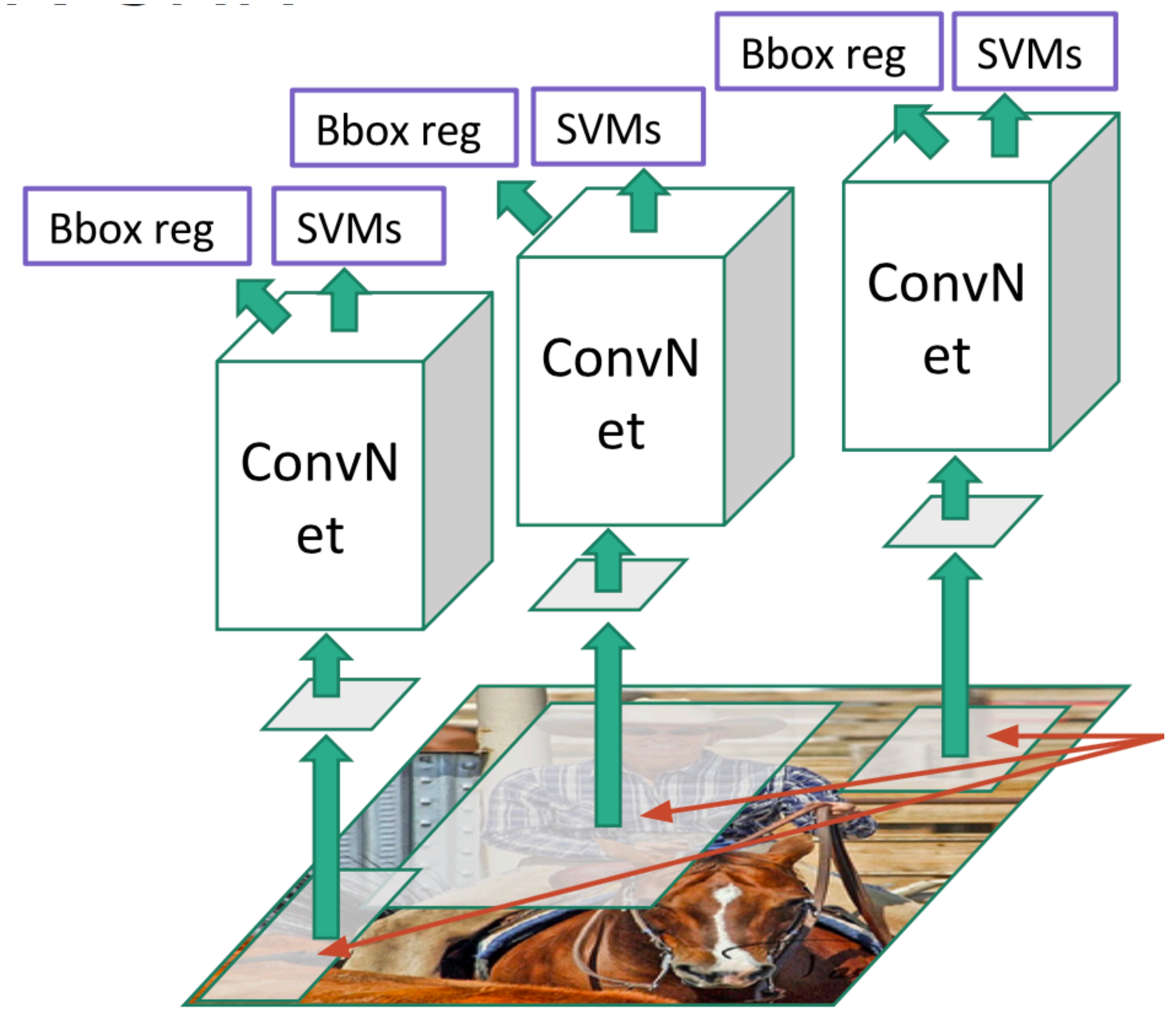
— Train post-hoc bounding-box regression (**least squares**)

Training is slow

— 84 hours and takes a lot of disk space

Inference / **Detection is slow**

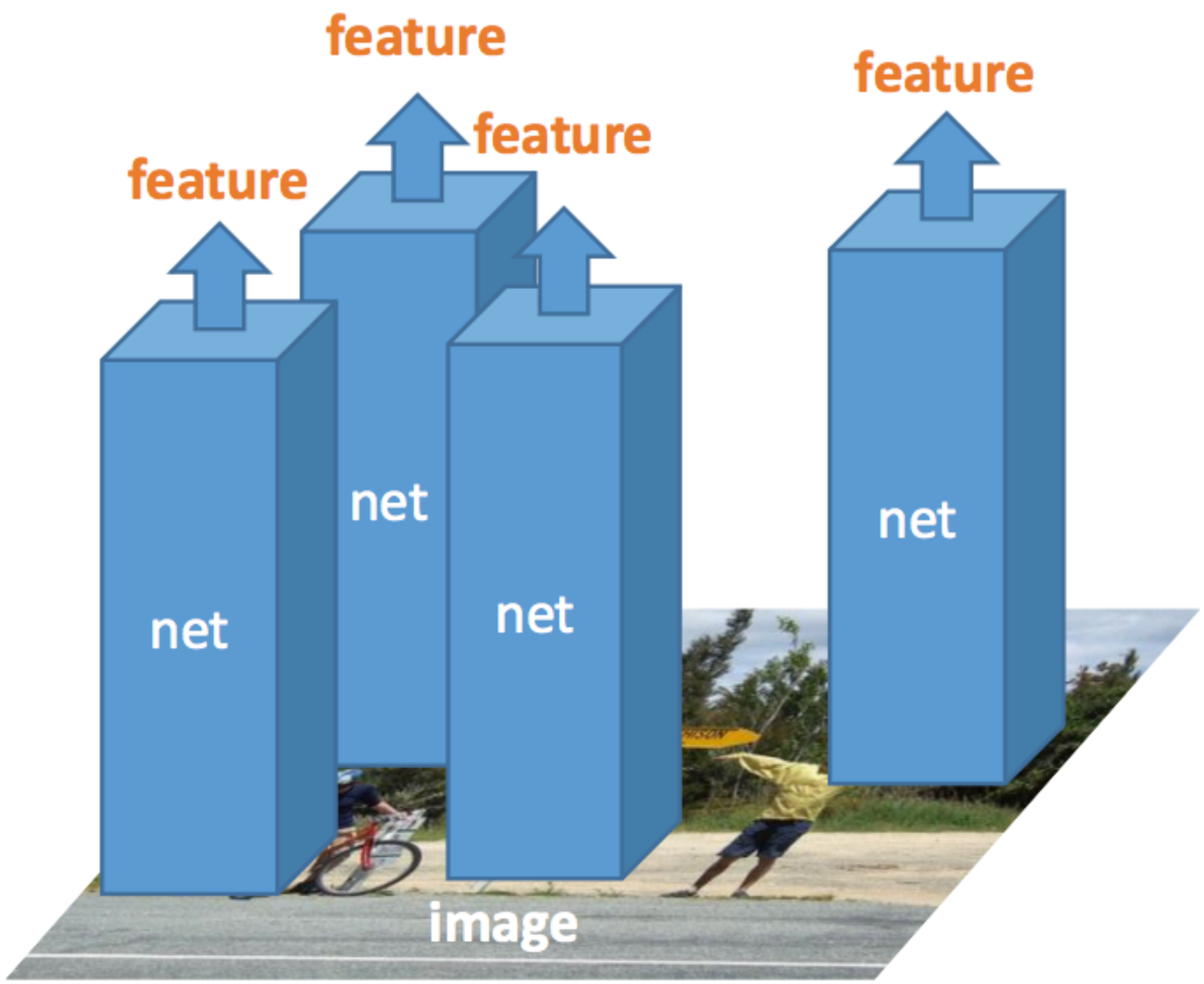
— 47 sec / image with VGG16 [Simonyan et al, ICLR 2015]



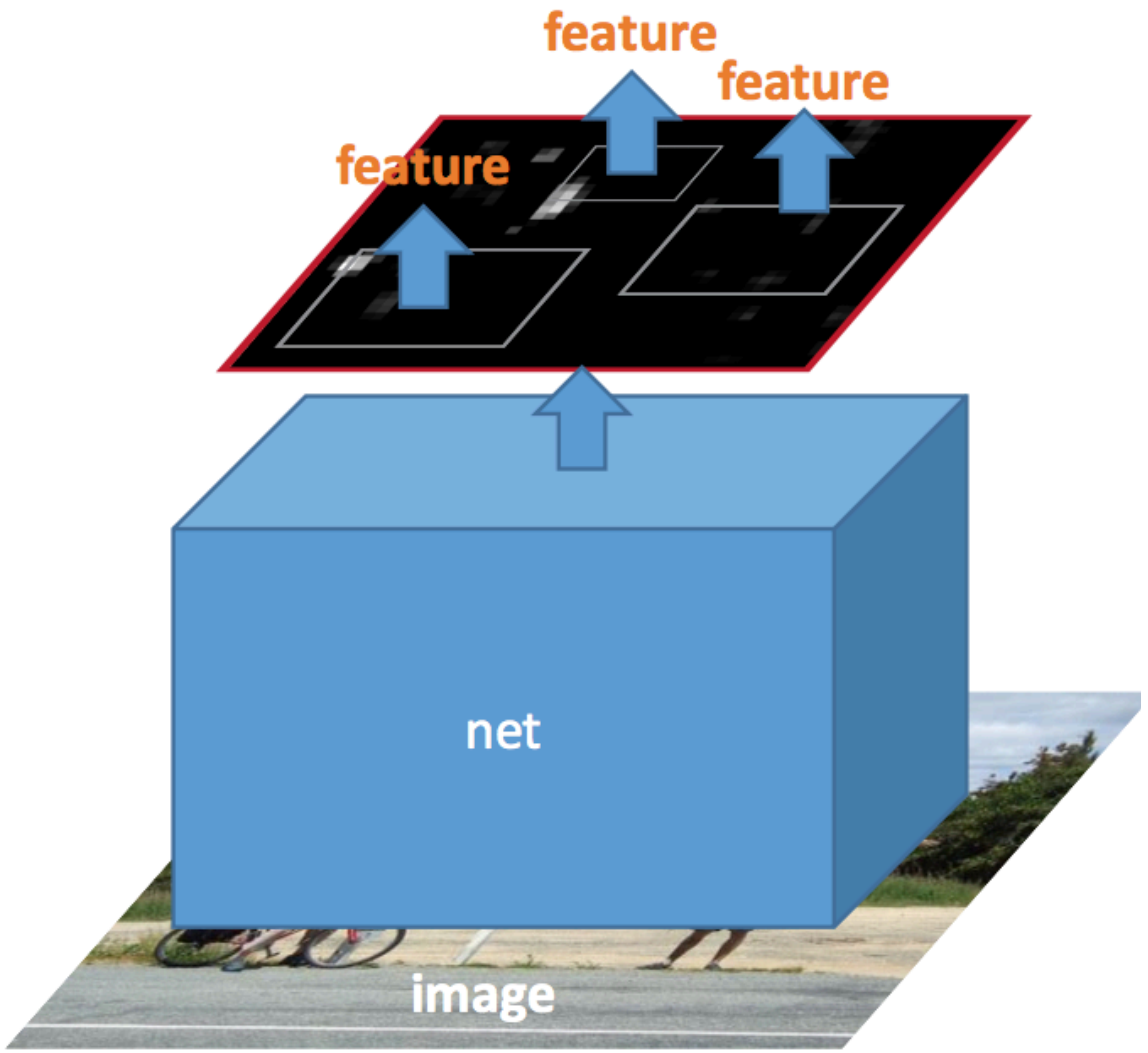
* image from Ross Girshick

R-CNN vs. SPP

[He et al, ECCV 2014]



R-CNN
2000 nets on image regions



SPP-net
1 net on full image

Fast **R-CNN**

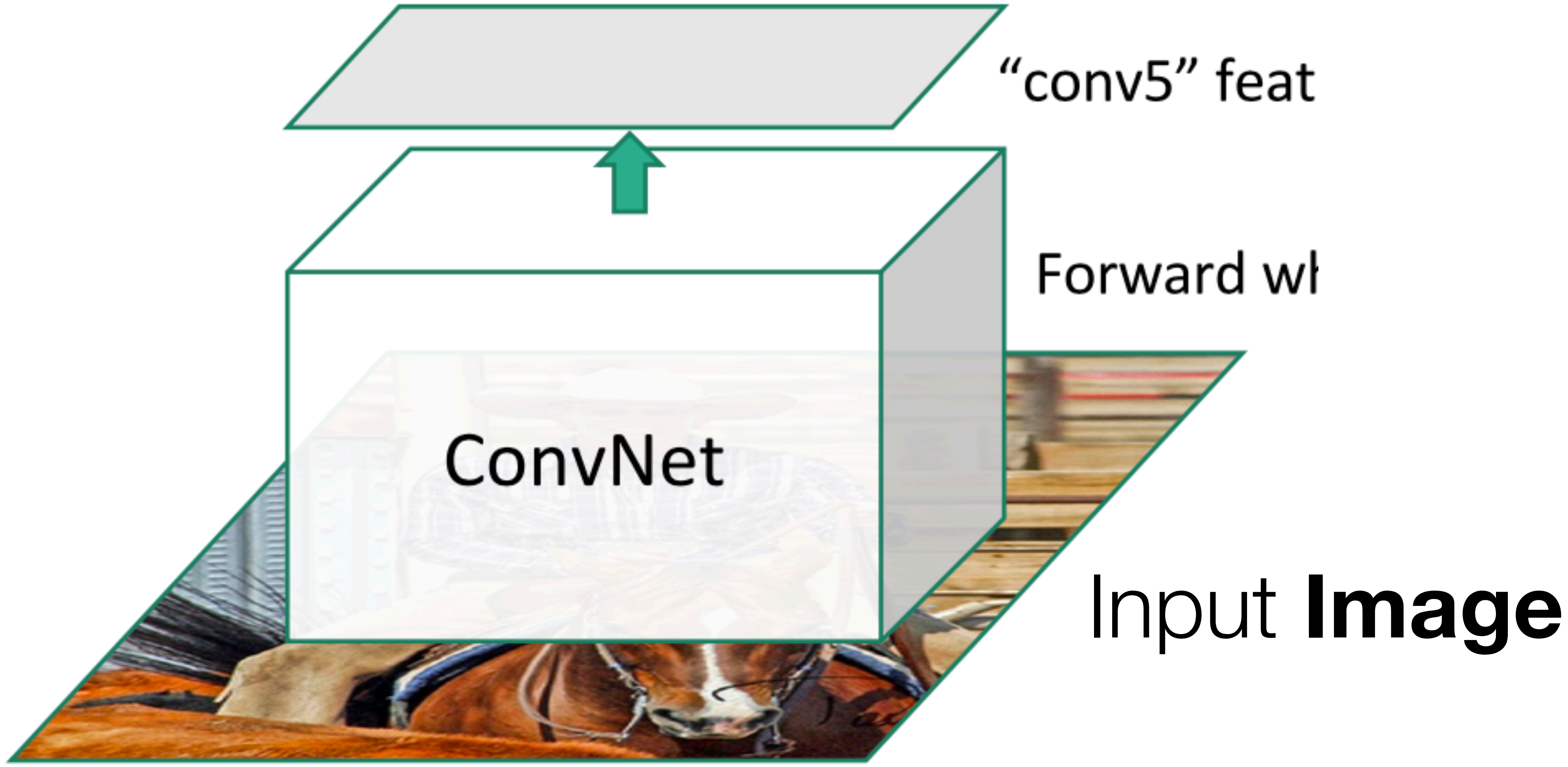
[Girshick et al, ICCV 2015]

Input **Image**

* image from Ross Girshick

Fast **R-CNN**

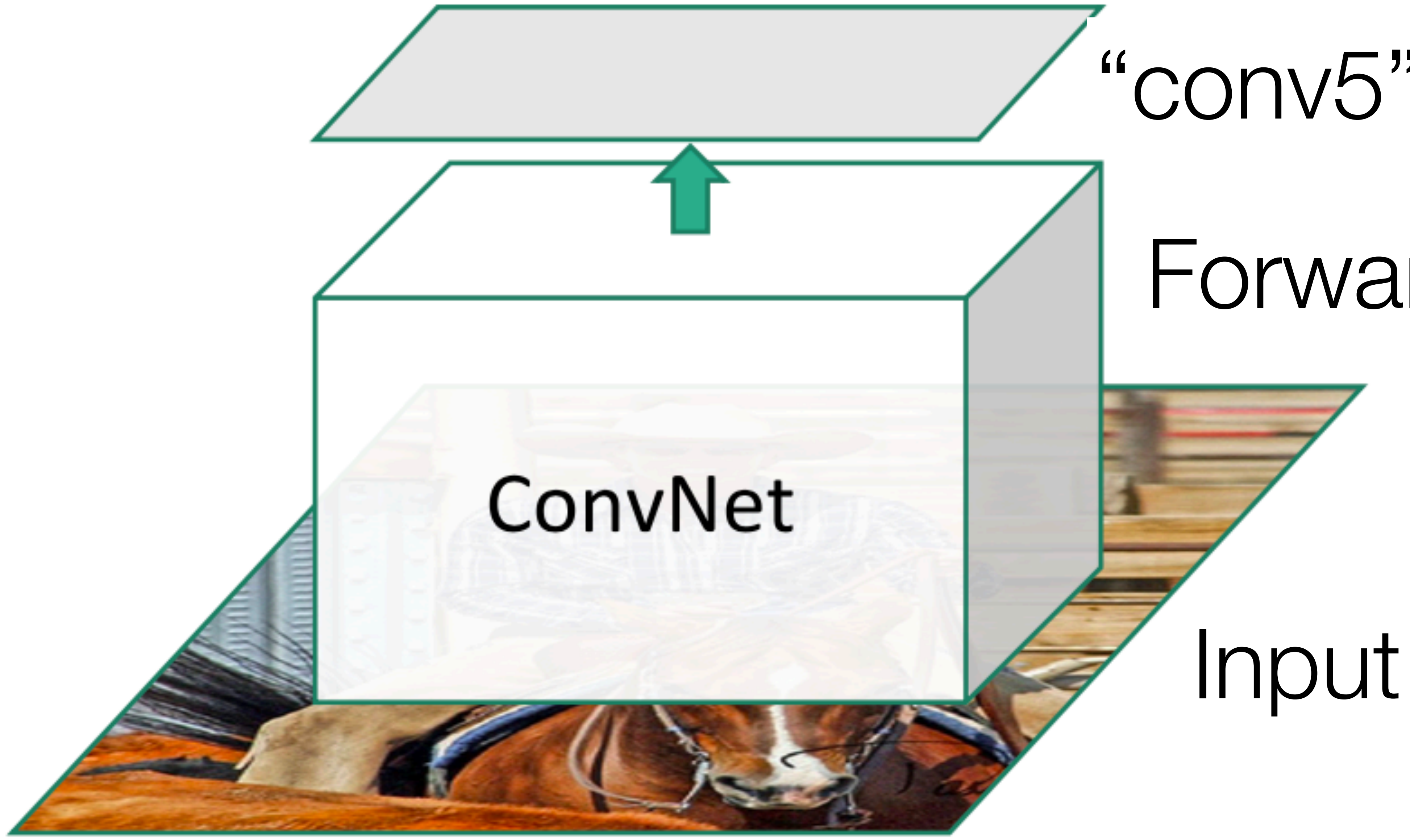
[Girshick et al, ICCV 2015]



* image from Ross Girshick

Fast **R-CNN**

[Girshick et al, ICCV 2015]



"conv5" feature map

Forward prop the **whole image** through CNN

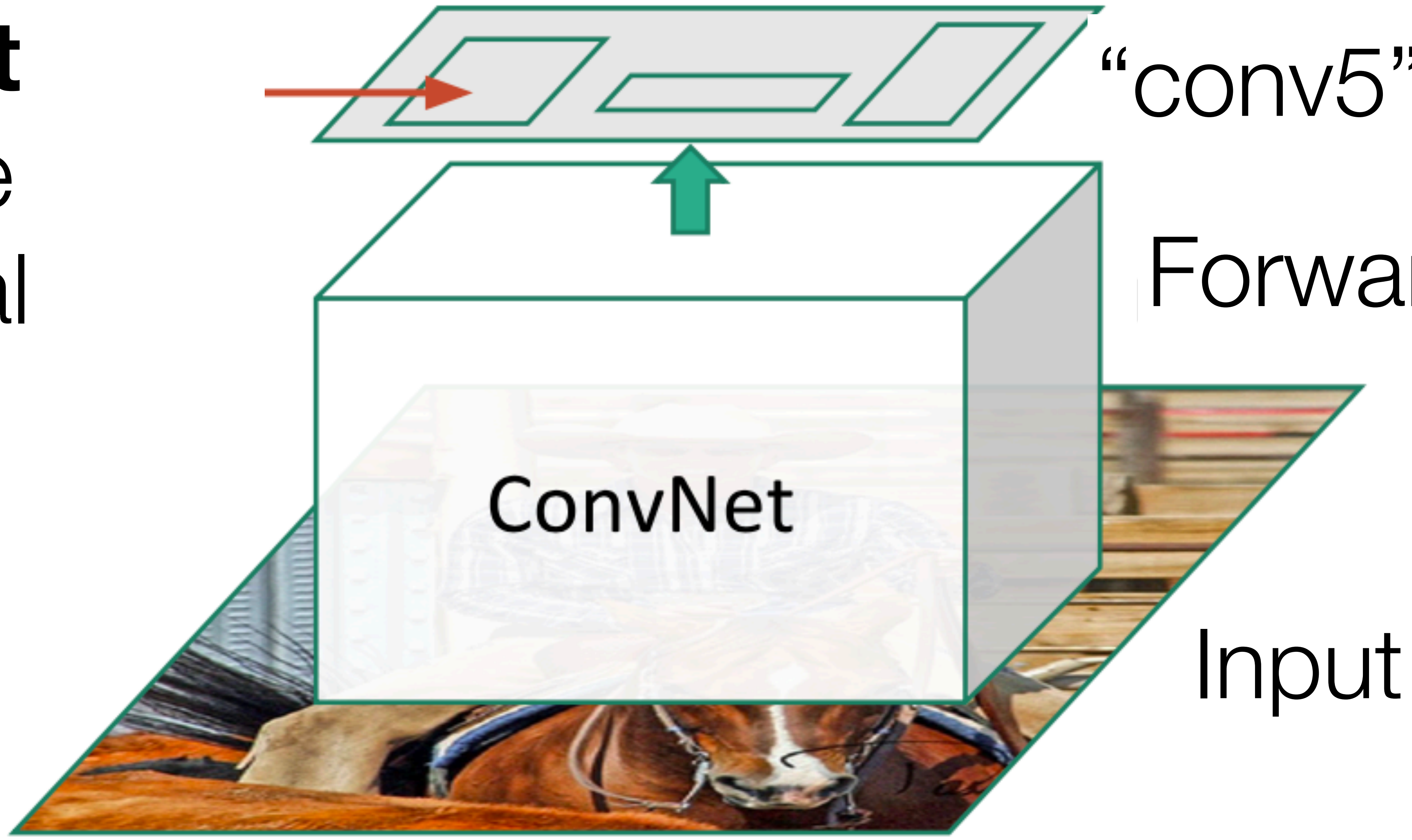
Input **Image**

* image from Ross Girshick

Fast **R-CNN**

[Girshick et al, ICCV 2015]

Regions of Interest from the proposal method



“conv5” feature map

Forward prop the **whole image** through CNN

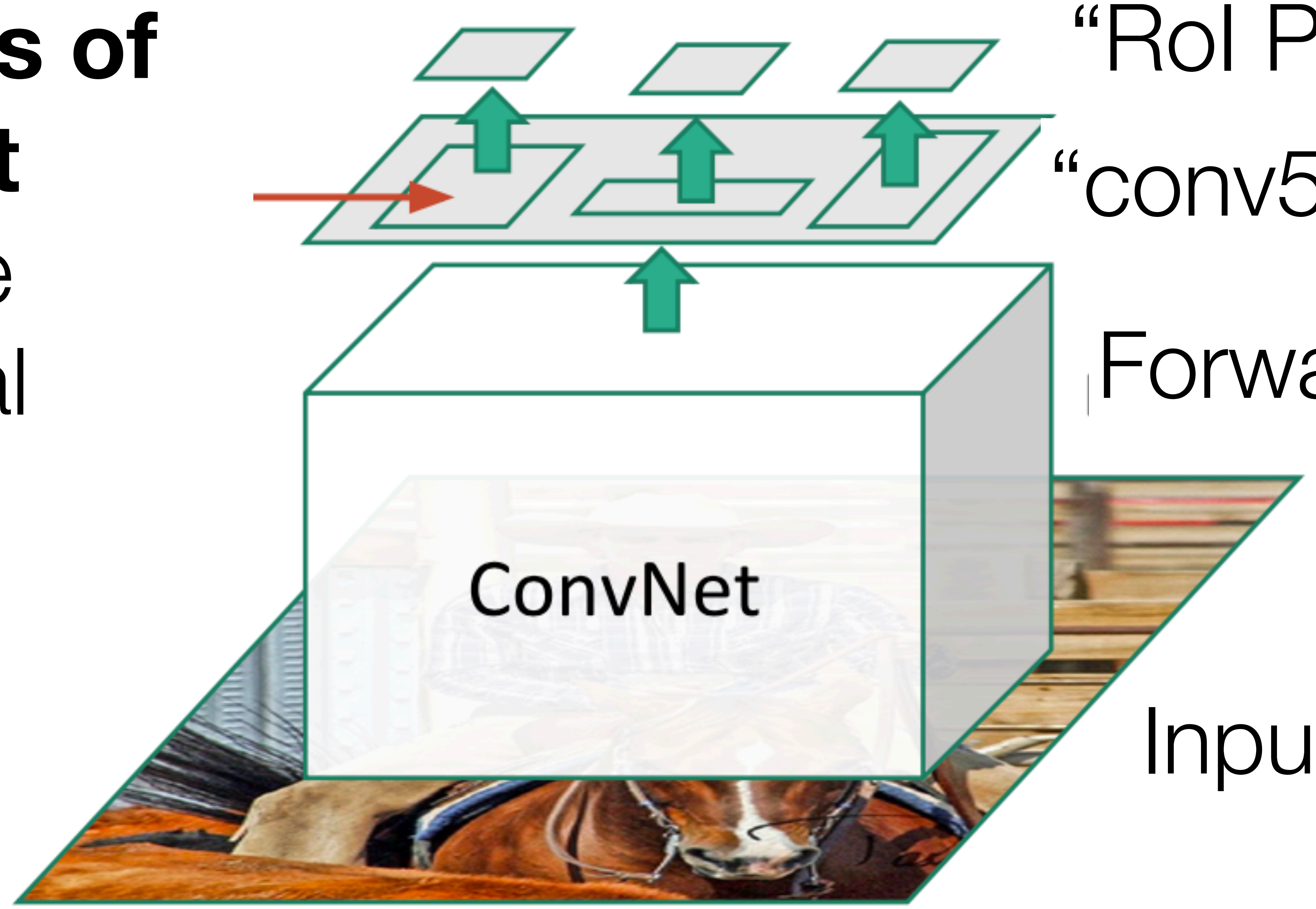
Input **Image**

* image from Ross Girshick

Fast **R-CNN**

[Girshick et al, ICCV 2015]

Regions of Interest from the proposal method



“RoI Pooling” layer

“conv5” feature map

Forward prop the **whole image** through CNN

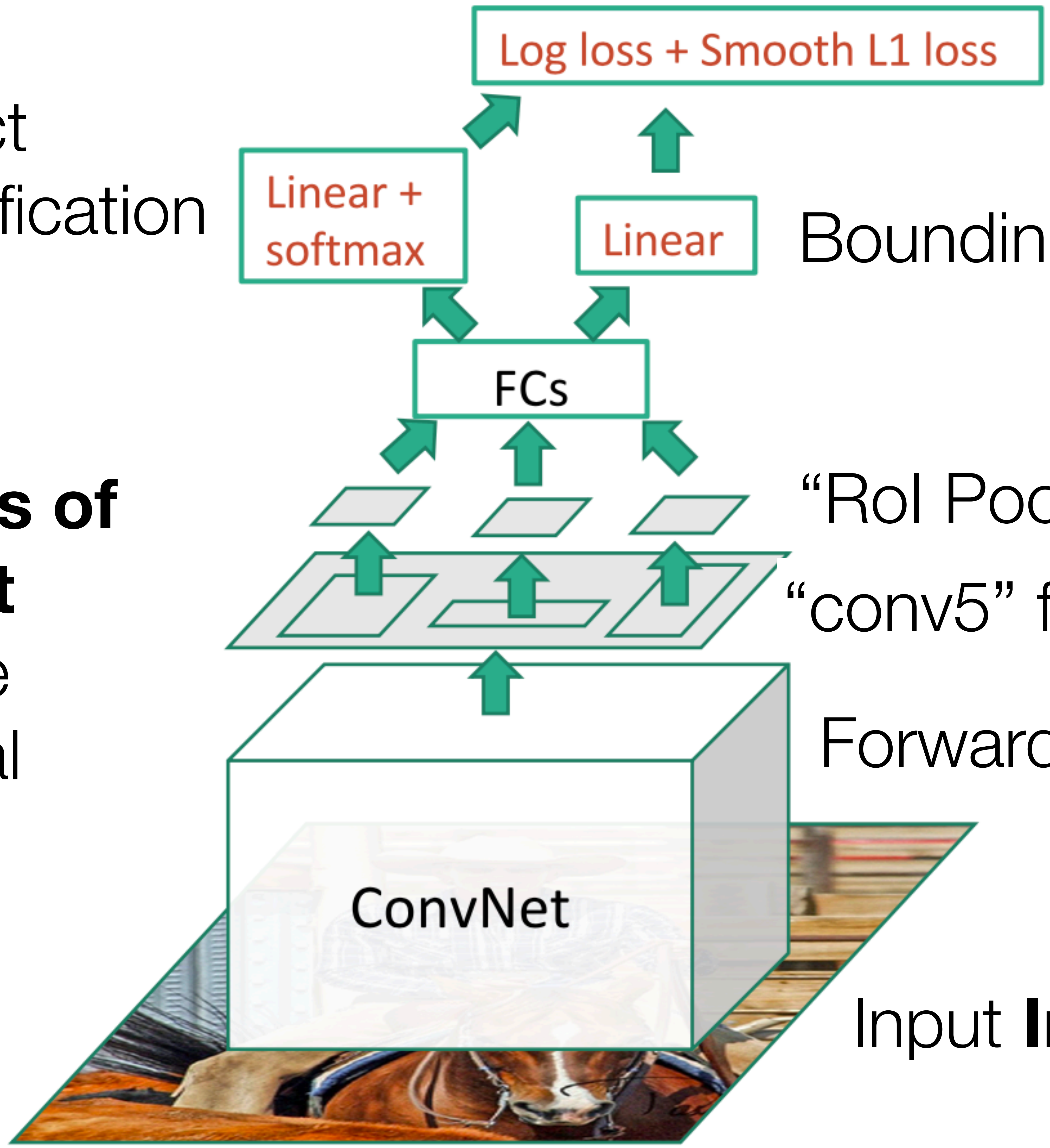
Input **Image**

Girshick, “Fast R-C
Figure copyright R

* image from Ross Girshick

Fast **R-CNN**

[Girshick et al, ICCV 2015]



Multi-task loss

Object classification

Bounding box regression

"RoI Pooling" layer

"conv5" feature map

Regions of Interest from the proposal method

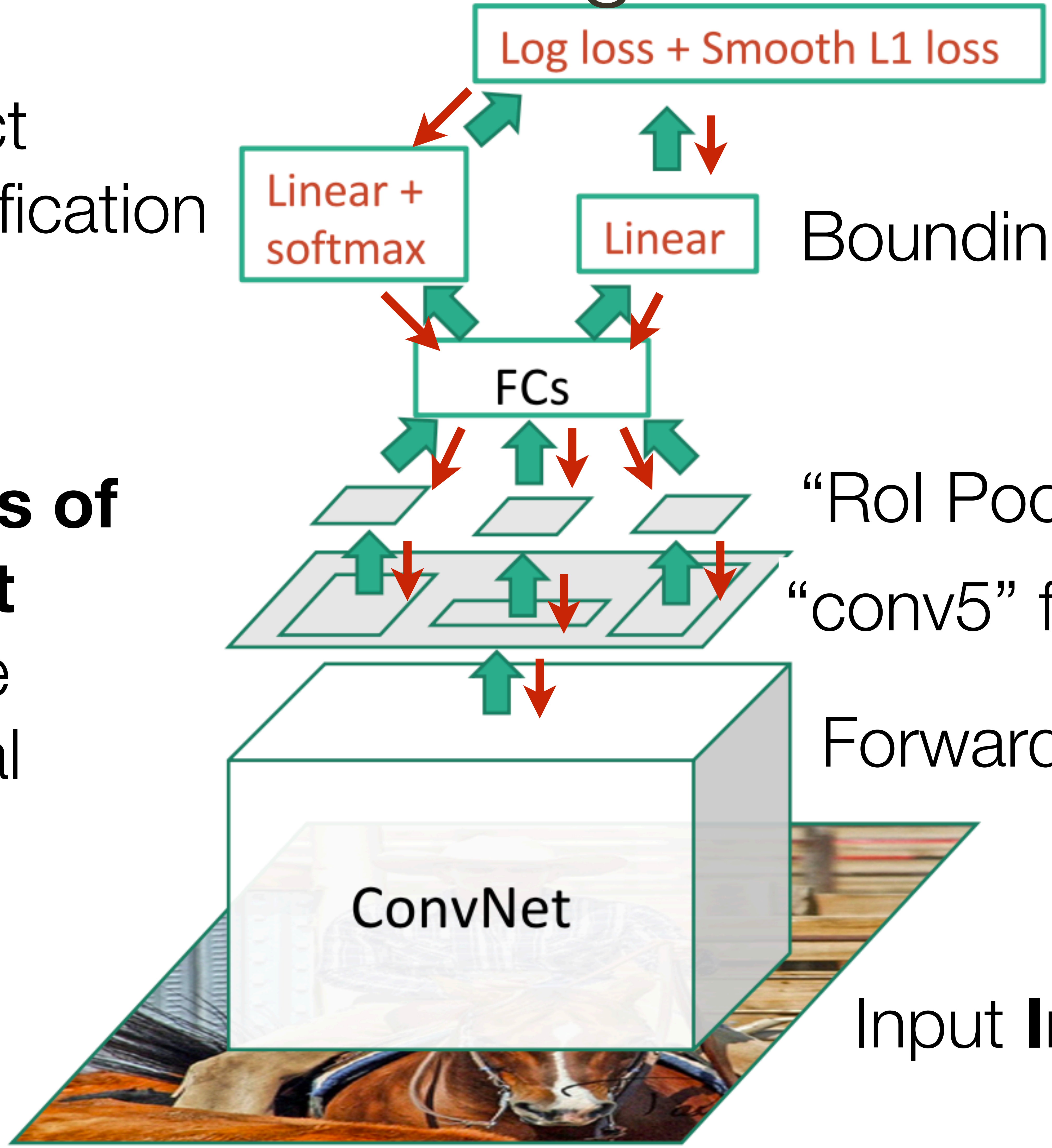
Forward prop the **whole image** through CNN

Input **Image**

* image from Ross Girshick

Fast **R-CNN:** Training

[Girshick et al, ICCV 2015]



Multi-task loss

Object classification

Bounding box regression

"RoI Pooling" layer

"conv5" feature map

Forward prop the **whole image** through CNN

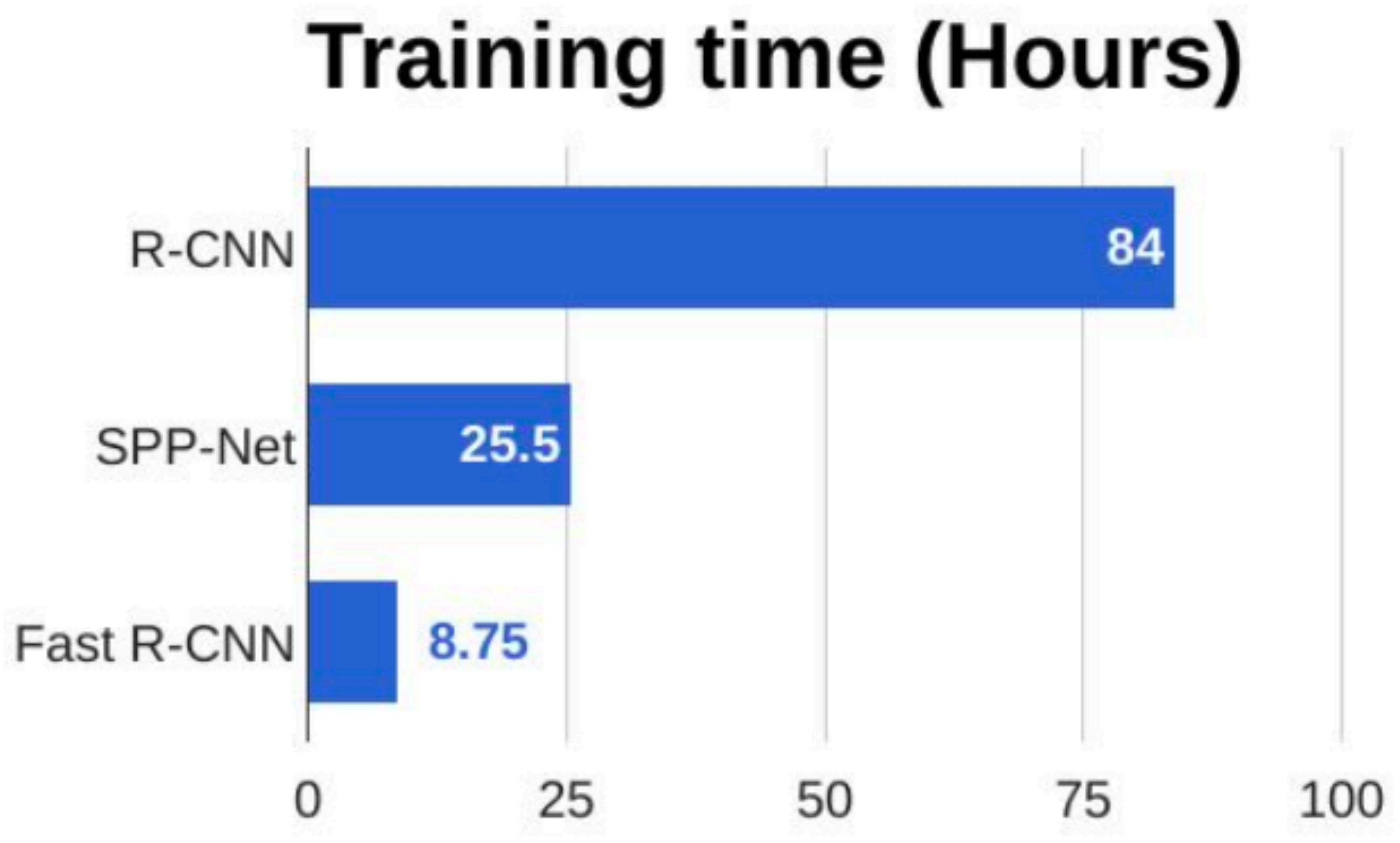
Input **Image**

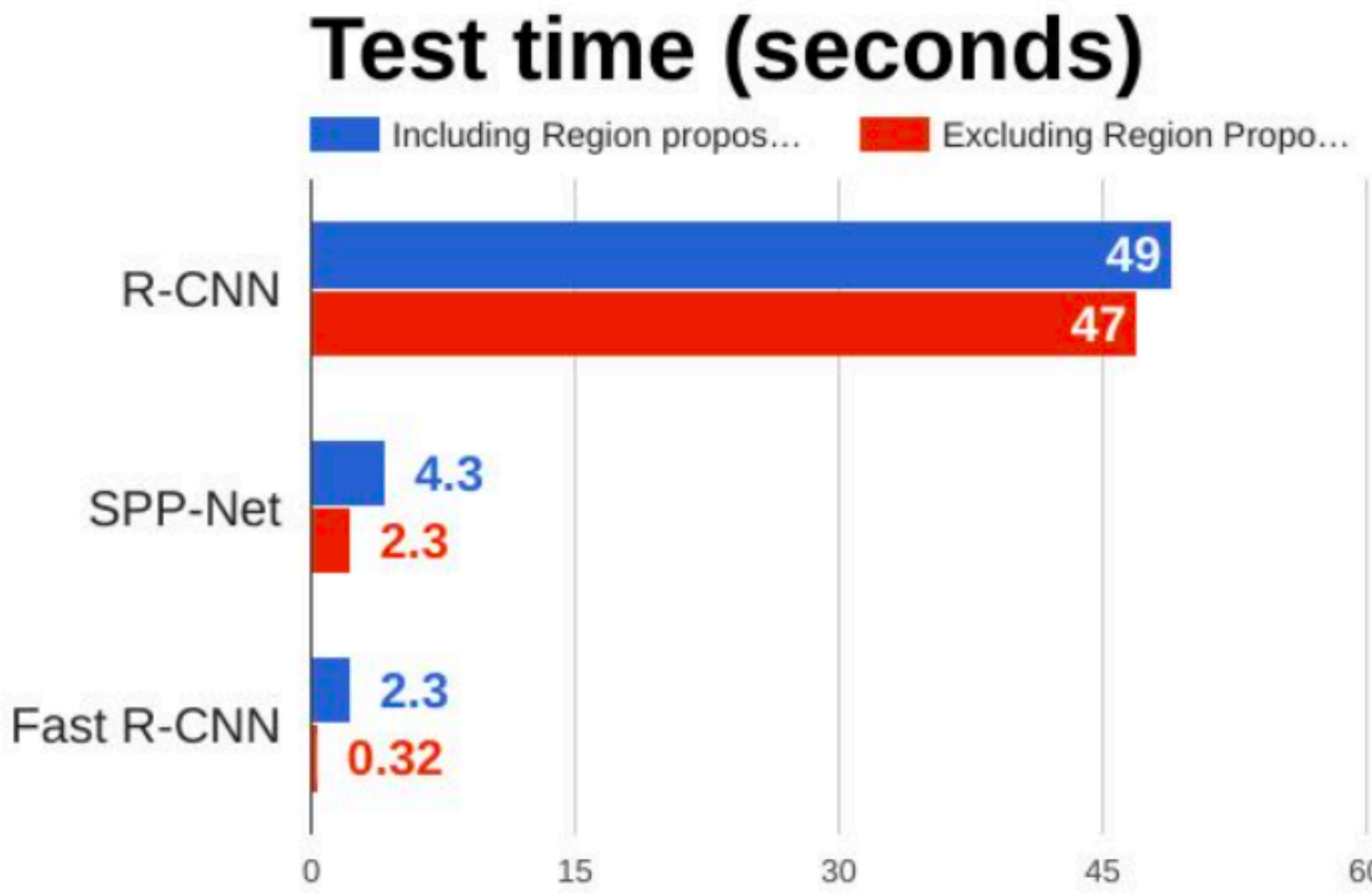
Regions of Interest from the proposal method

* image from Ross Girshick

R-CNN vs. **SPP** vs. **Fast R-CNN**

[Girshick et al, CVPR 2014]
[Girshick et al, ICCV 2015]
[He et al, ECCV 2014]





Observation: Performance dominated by the region proposals at this point!

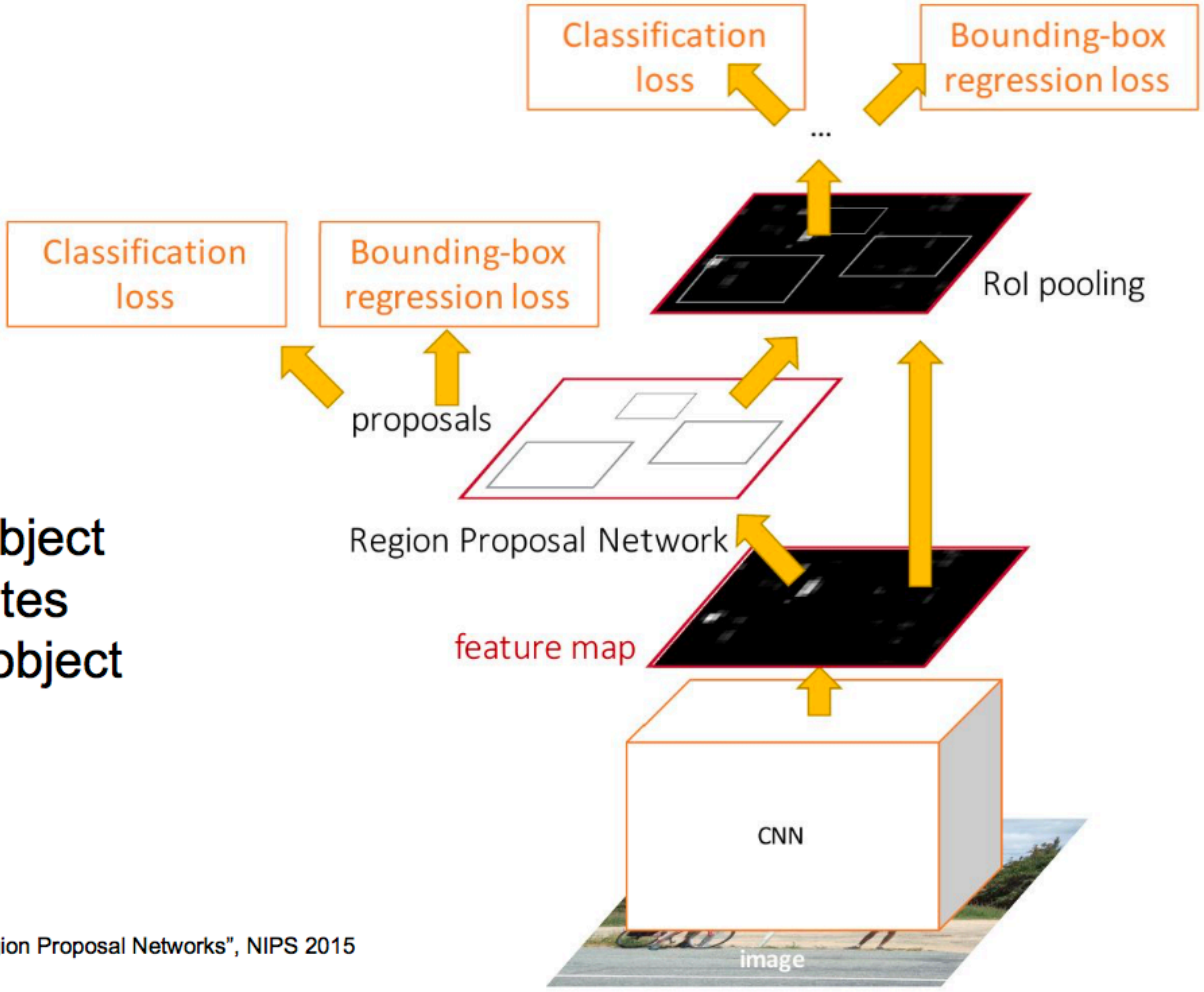
Faster **R-CNN**

Make CNN do proposals!

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

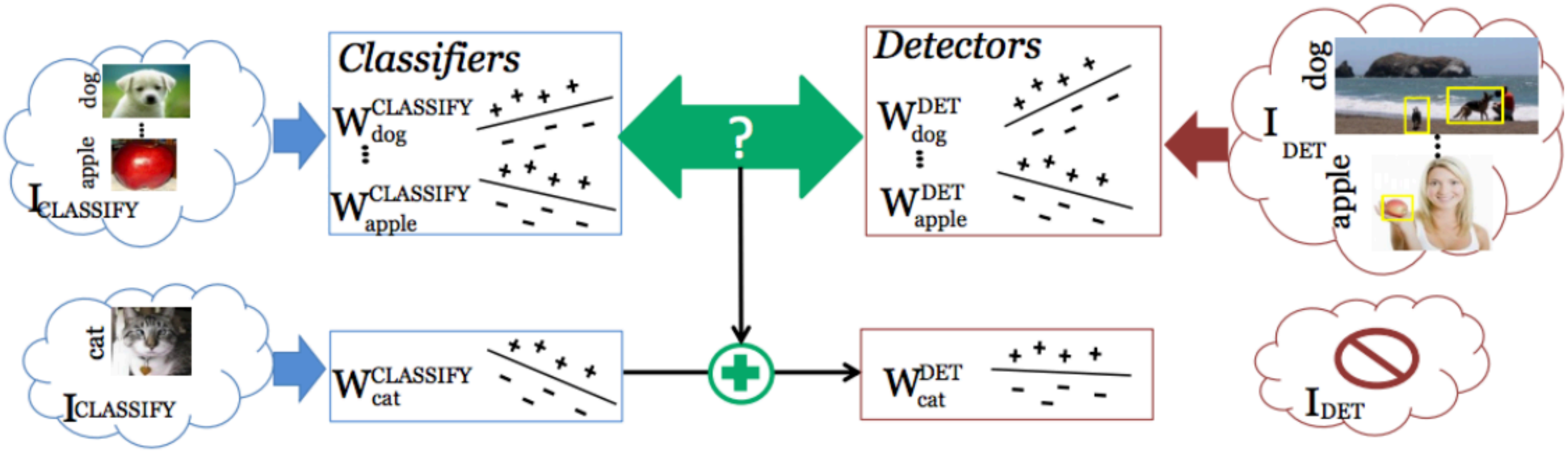
Jointly train with 4 losses:

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



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

LSDA: Large Scale Detection through Adaptation



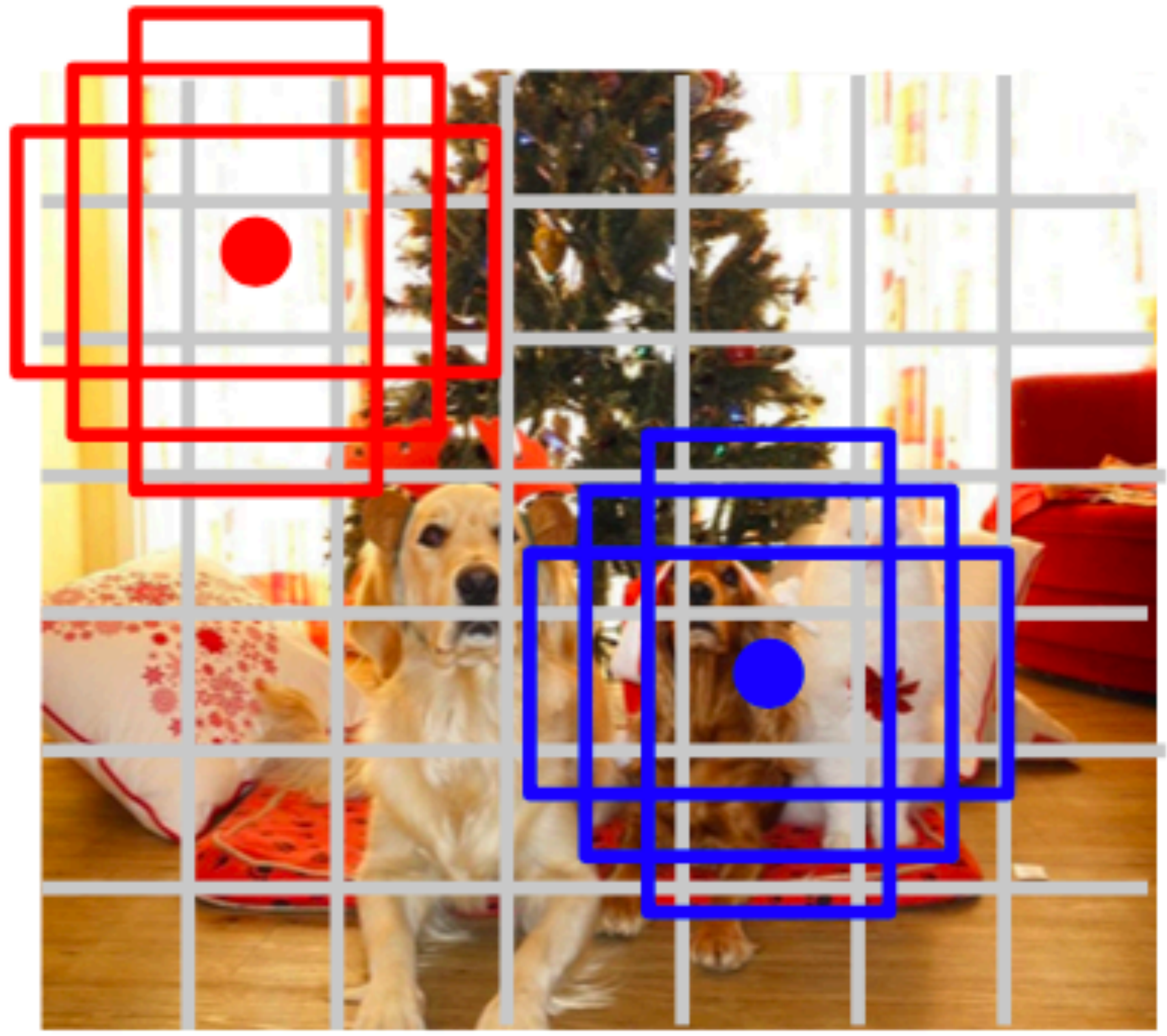
$$\mathbf{W}_{cat}^{DETECT} = \mathbf{W}_{cat}^{CLASIFY} + \delta \mathbf{W}_{cat}$$

[Hoffman et al, NIPS 2014]

YOLO: You Only Look Once

[Redmon et al, CVPR 2016]

Input image
3 x H x W



Divide image into grid
7 x 7

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

Within each grid cell:

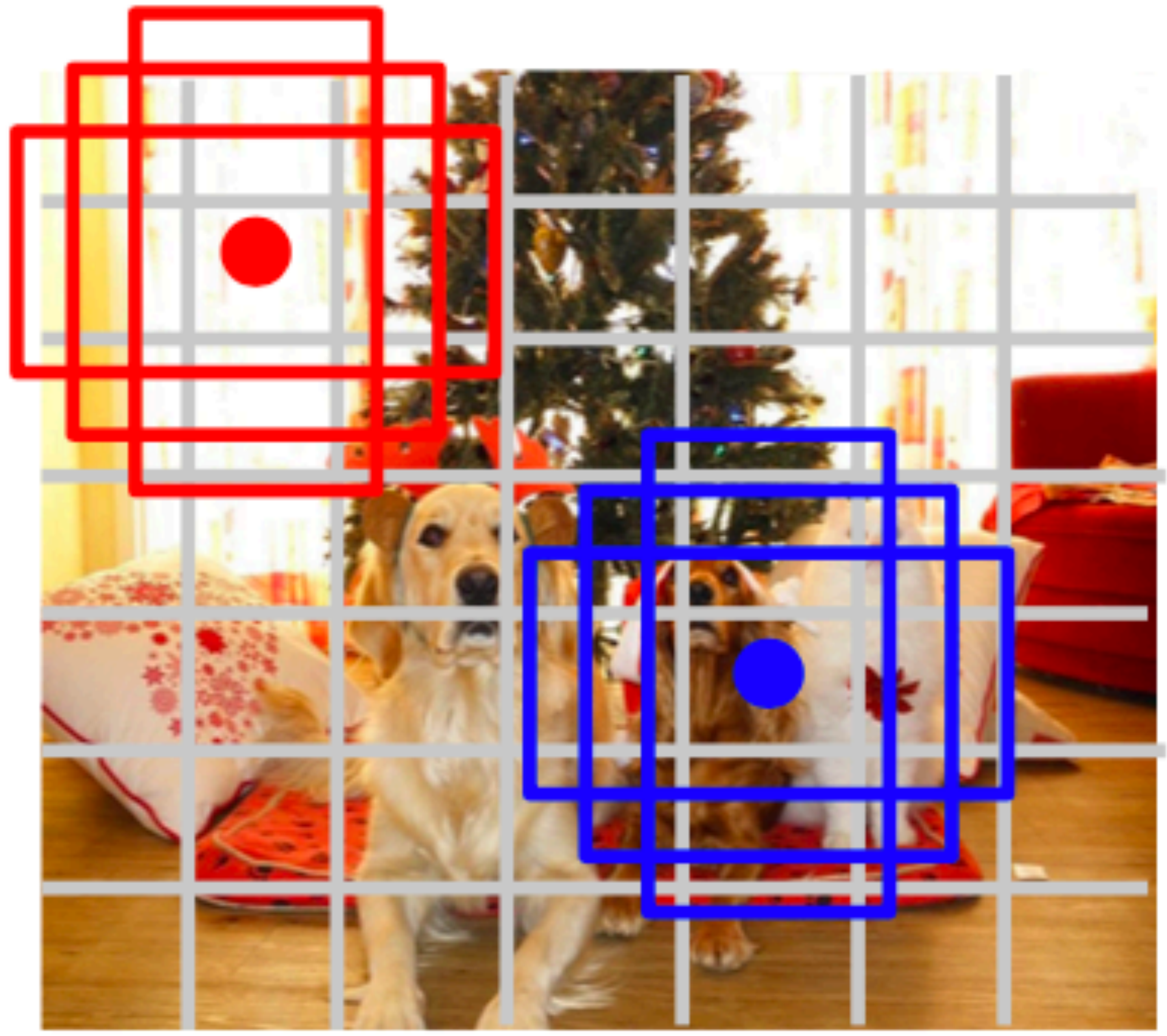
- Regress from each of the B base boxes to a final box with 5 numbers:
  (dx, dy, dh, dw, confidence)
- Predict scores for each of C classes (including background as a class)

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

YOLO: You Only Look Once

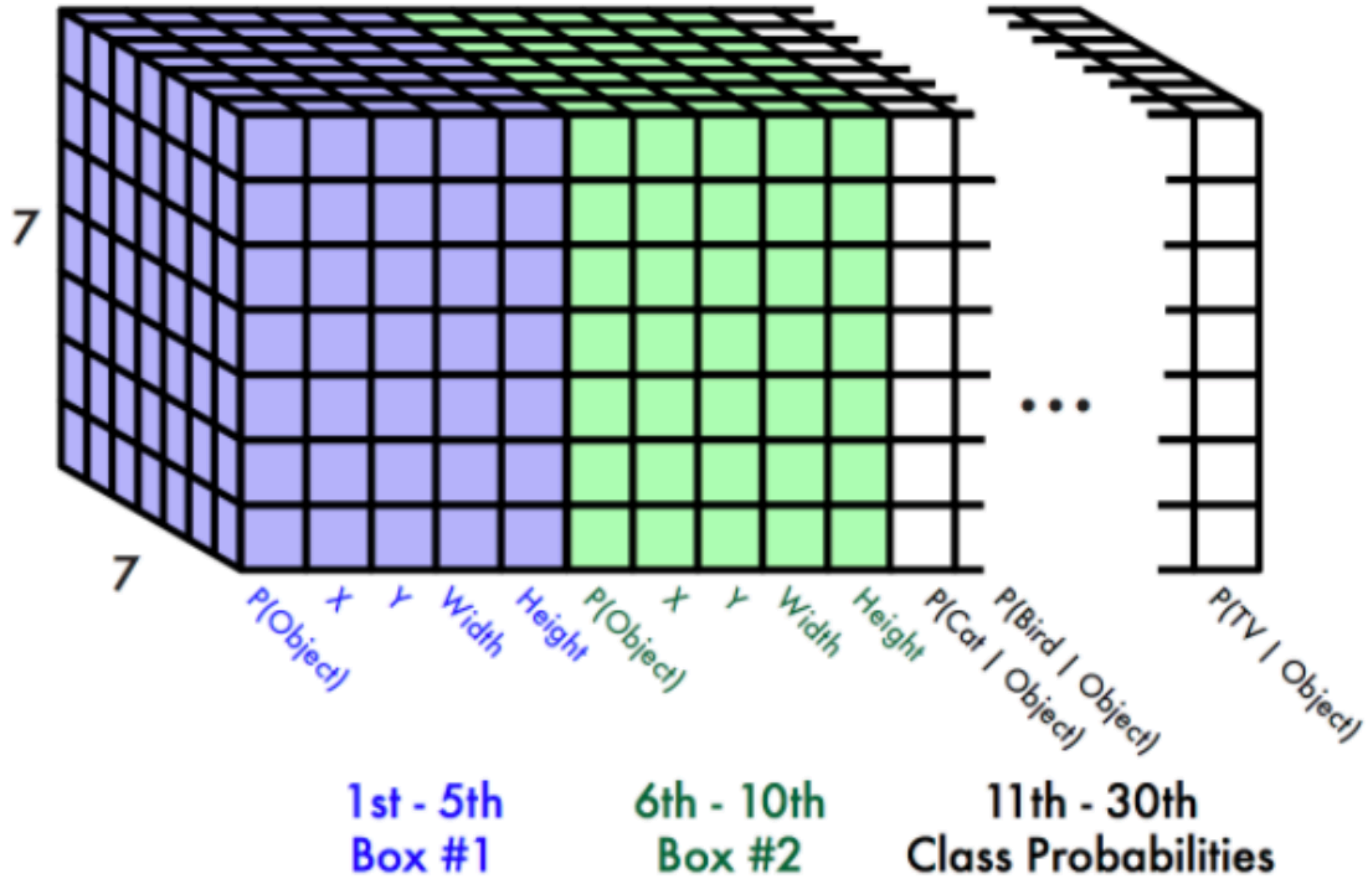
[Redmon et al, CVPR 2016]

Input image
3 x H x W



Divide image into grid
7 x 7

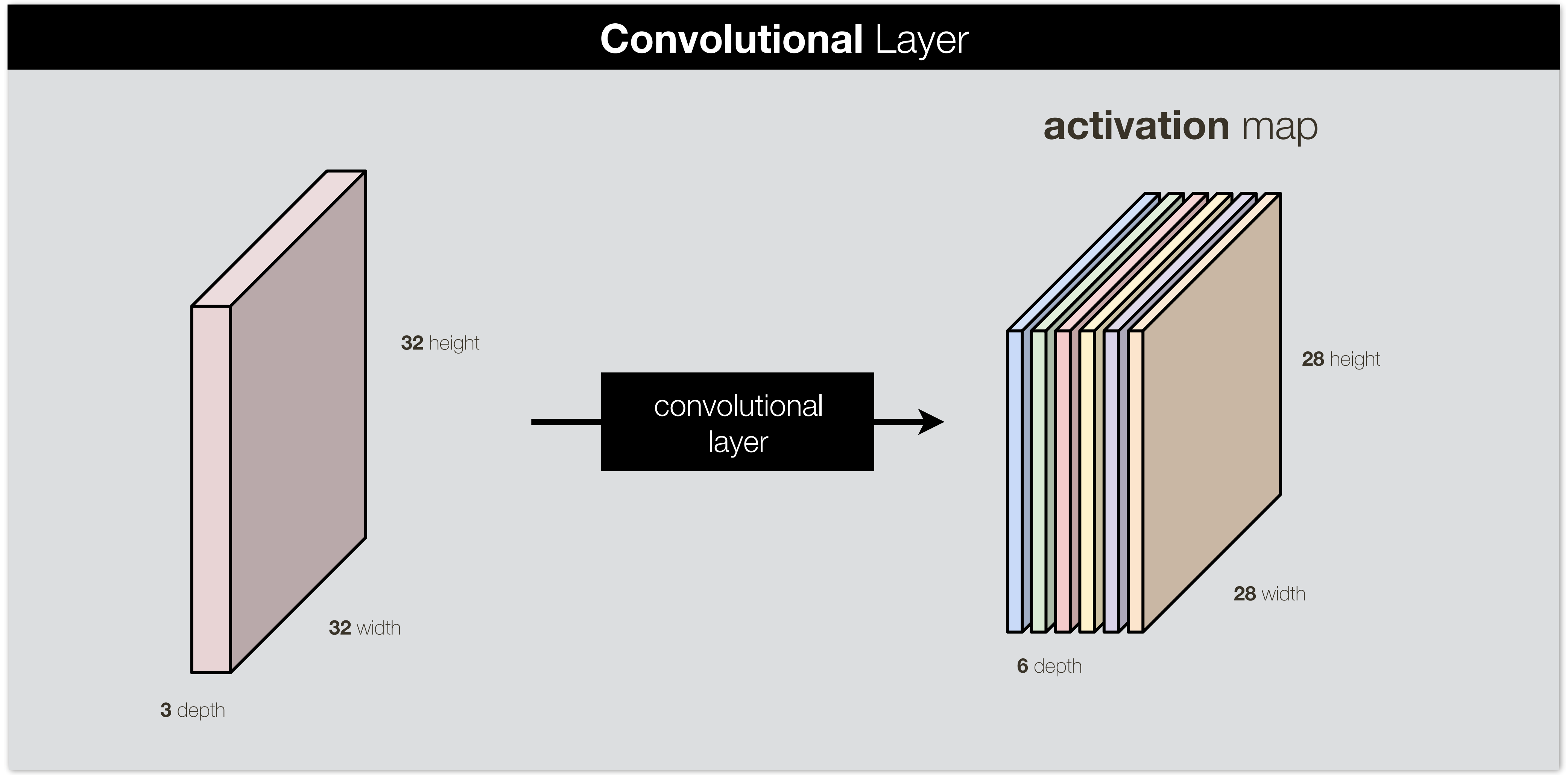
Image a set of **base boxes** centered at each grid cell
Here B = 3

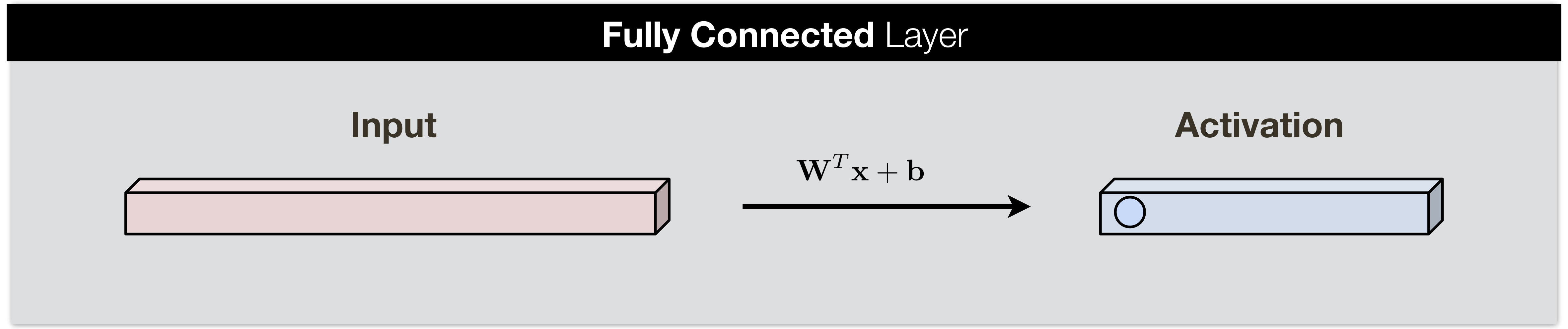


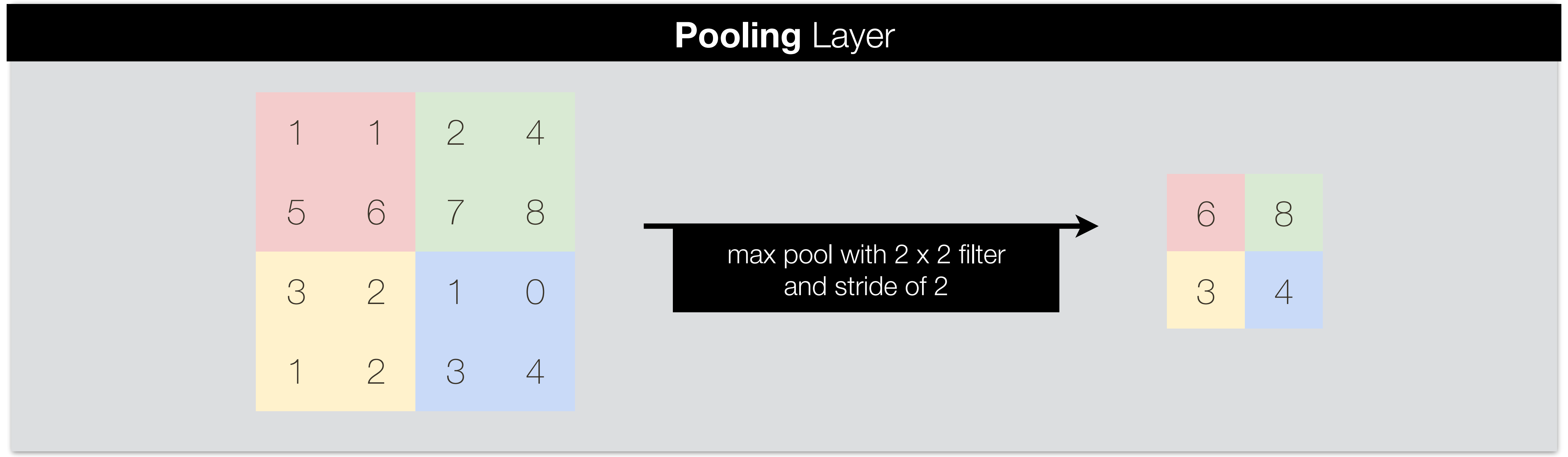
YOLO v2

http://pjreddie.com/yolo

Review of **CNNs**







Effective Techniques for **Training**

— **Regularization:** L1, L2, data augmentation

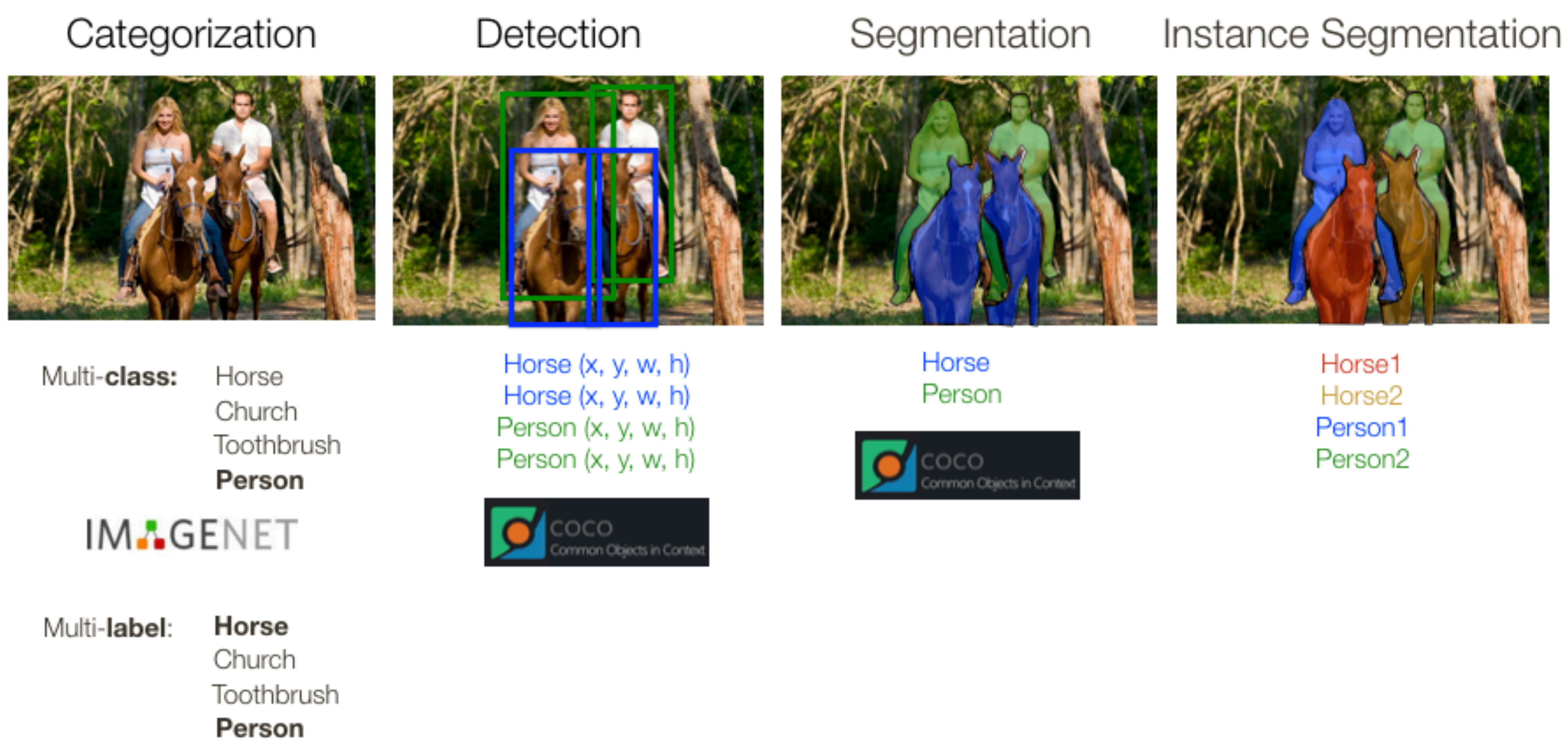
— **Transfer Learning:** fine-tuning networks

Vision **Applications** of CNNs

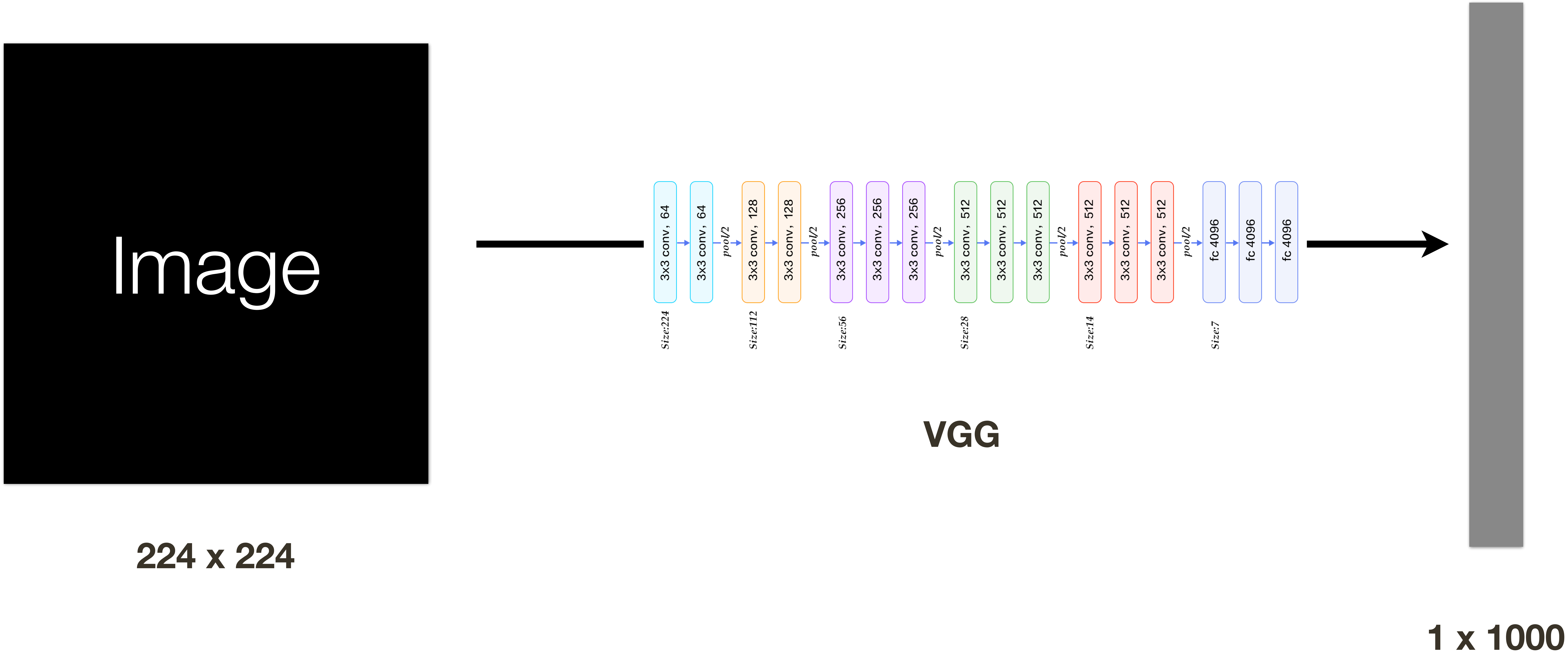
— **Classification:** AlexNet, VGG, GoogleLeNet, ResNet

— **Segmentation:** Fully convolutional CNNs

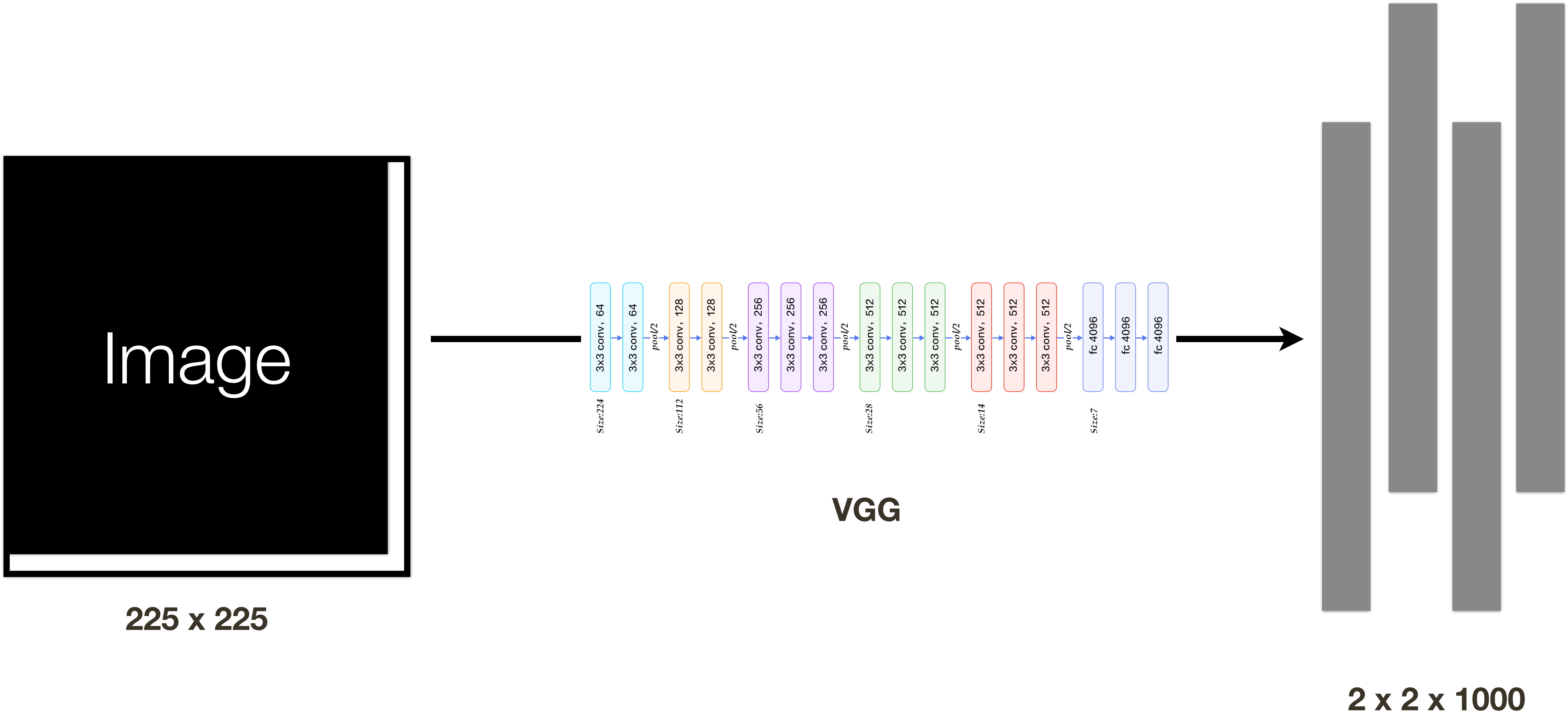
— **Detection:** R-CNN, Fast R-CNN, Faster R-CNN, YOLO



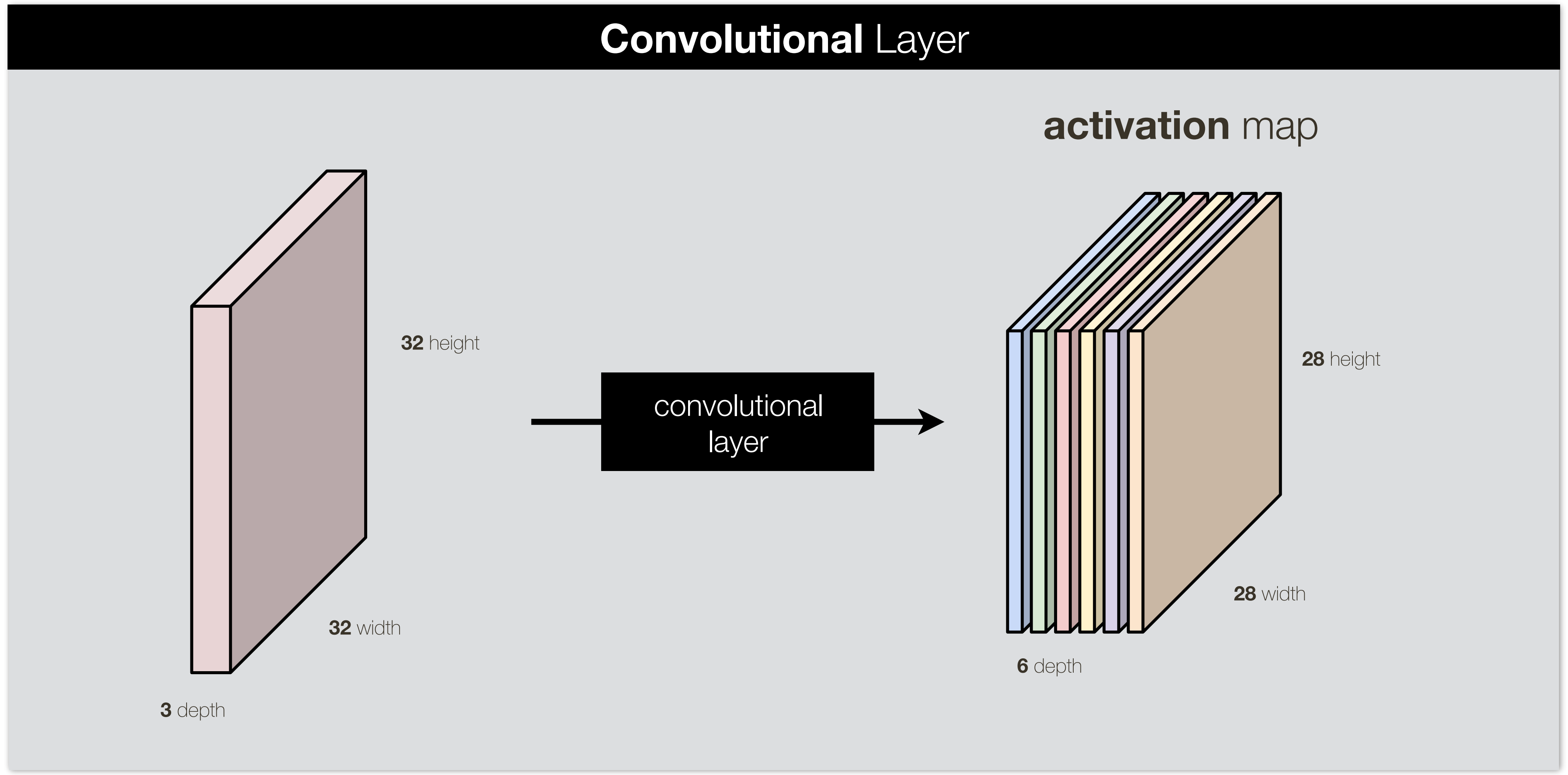
Any **CNN** Could be **Fully Convolutional**

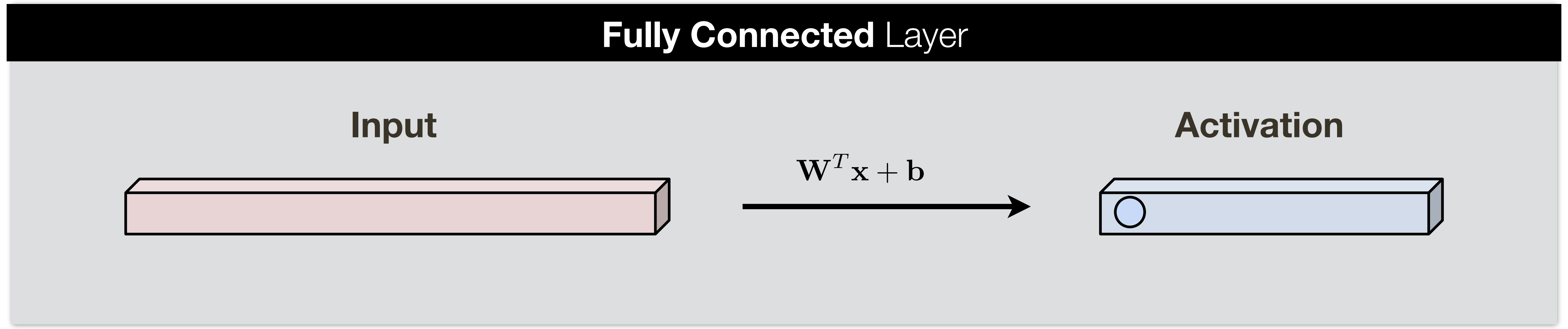


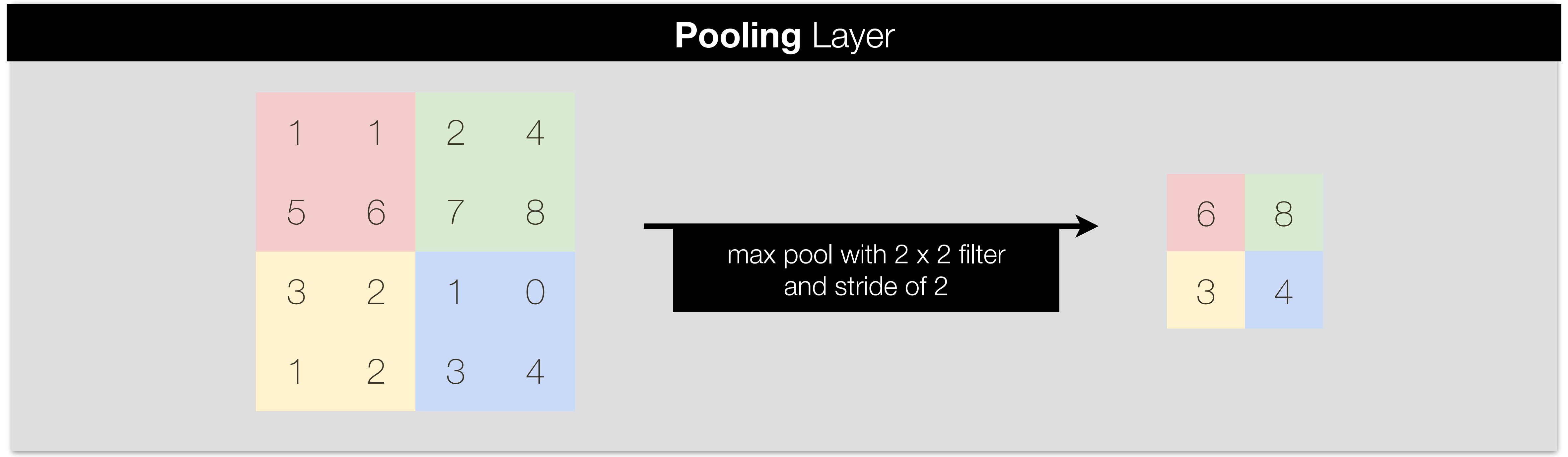
Any **CNN** Could be **Fully Convolutional**



Review of **CNNs**



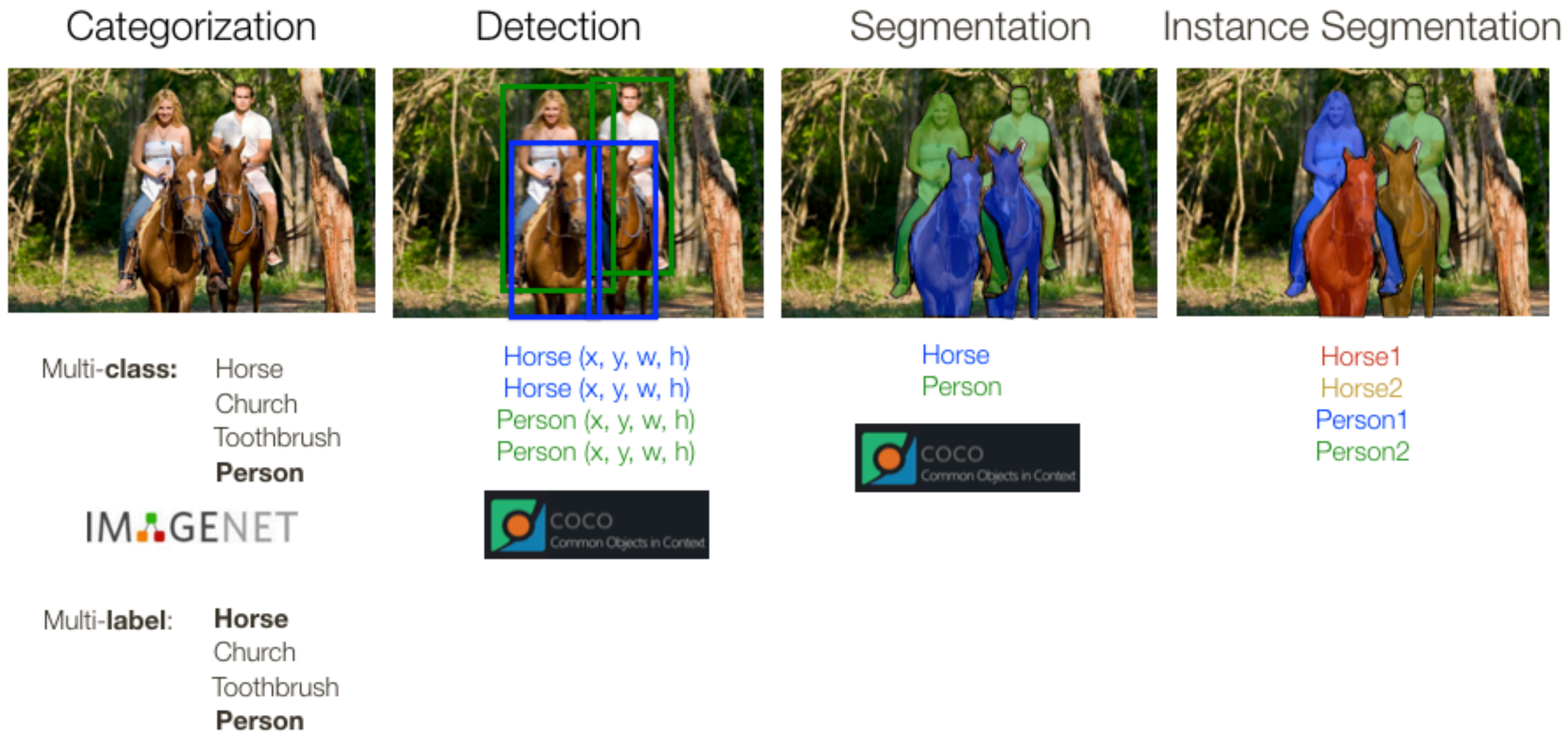




Effective Techniques for **Training**

— **Regularization:** L1, L2, data augmentation

— **Transfer Learning:** fine-tuning networks



Vision **Applications** of CNNs

— **Classification:** AlexNet, VGG, GoogleLeNet, ResNet

— **Segmentation:** Fully convolutional CNNs

— **Detection:** R-CNN, Fast R-CNN, Faster R-CNN, YOLO