CS484/684

Topic 12

Transfer Learning
+
Localization and Detection

Most slides are from Fei-Fei Li, Justin Johnson, Andrej Karpathy, Serena Yeung, Jia-Bin Huang

Transfer Learning

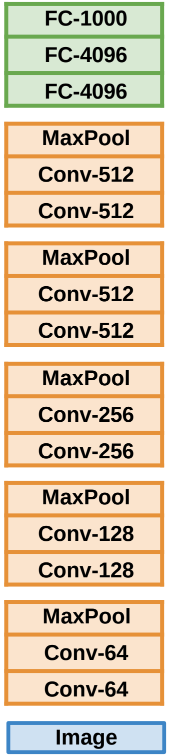
- Improve learning in a new task through transfer of knowledge from a related task that has already been learned
- Often used when your own dataset is not large enough
  - cannot train large CNN with little data
- Two major strategies
  - Use trained CNN as a fixed feature extractor
  - Fine tune trained CNN for your task
- References

  - Donahue et al, “DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition”, ICML 2014
  - Razavian et al, “CNN Features Off-the-Shelf: An Astounding Baseline for Recognition”, CVPR Workshops 2014
  - Oquab et al, “Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks”, CVPR 2014

Transfer Learning Classification: Fixed Feature Extraction

1. Train on Imagenet or load off internet

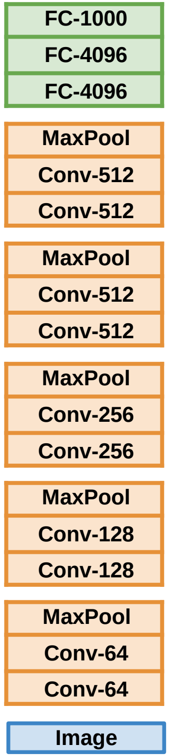
VGG



Caffe: https://github.com/BVLC/caffe/wiki/Model-Zoo
TensorFlow: https://github.com/tensorflow/models
PyTorch: https://github.com/pytorch/vision

Transfer Learning Classification: Fixed Feature Extraction

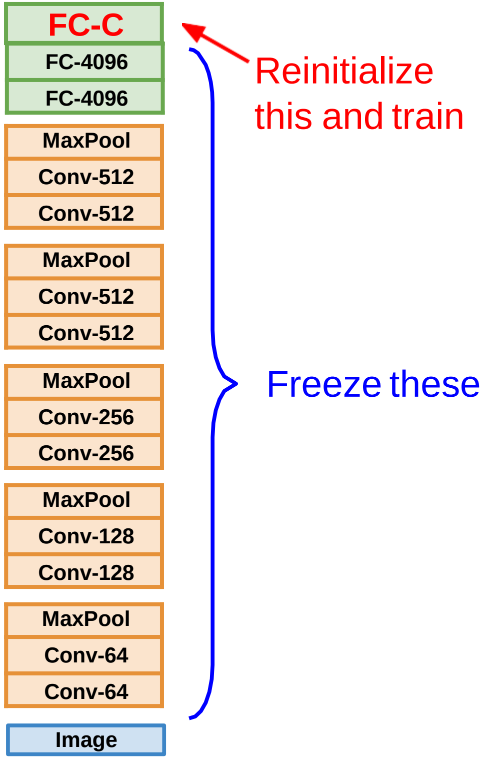
1. CNN pretrained on Imagenet
1000 classes



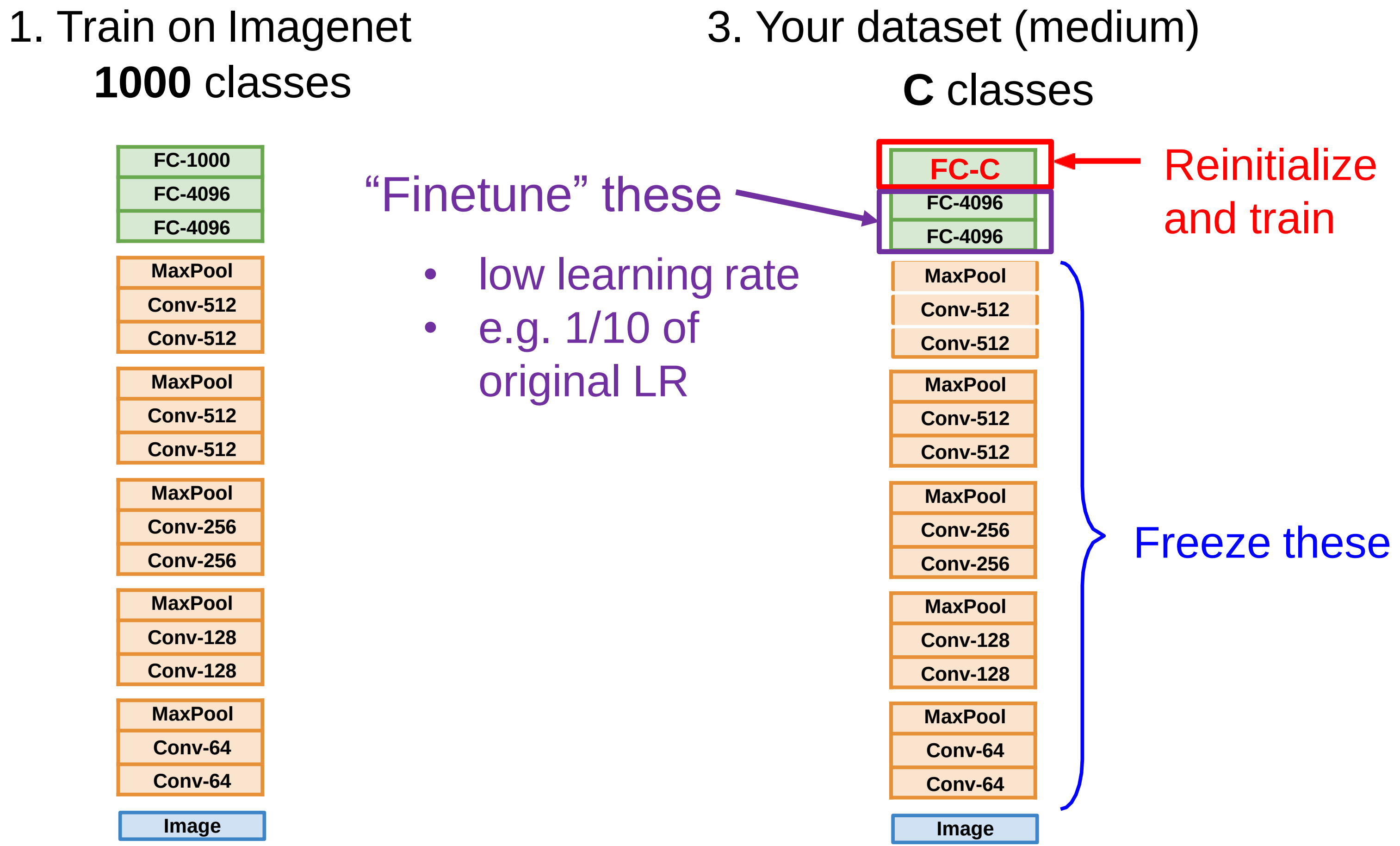
4096 features extracted from pretrained CNN

Interpretation:
Linear classifier is trained on top of fixed features

2. Your dataset (small)
C classes



Transfer Learning Classification: Fine Tuning

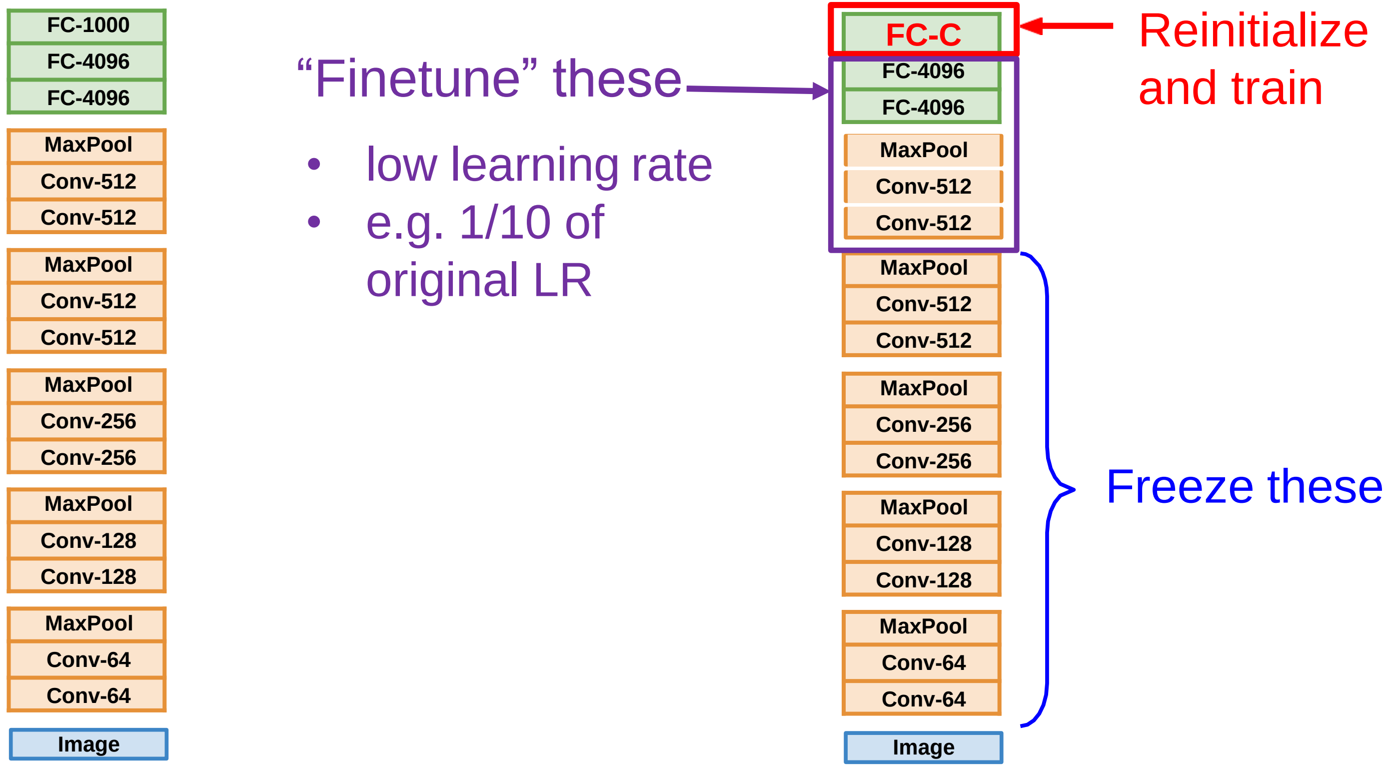


- low learning rate
- e.g. 1/10 of original LR

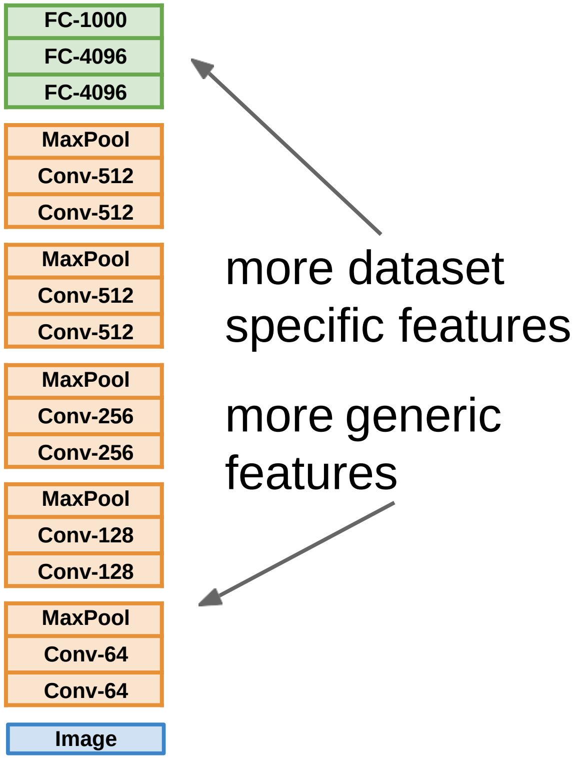
Transfer Learning Classification: Fine Tuning

1. Train on Imagenet
1000 classes

3. Your dataset (medium large)
C classes

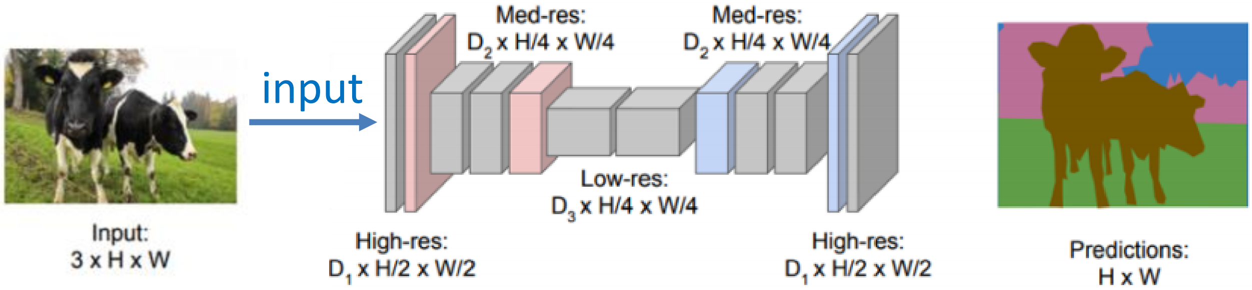


Transfer Learning Classification Overview

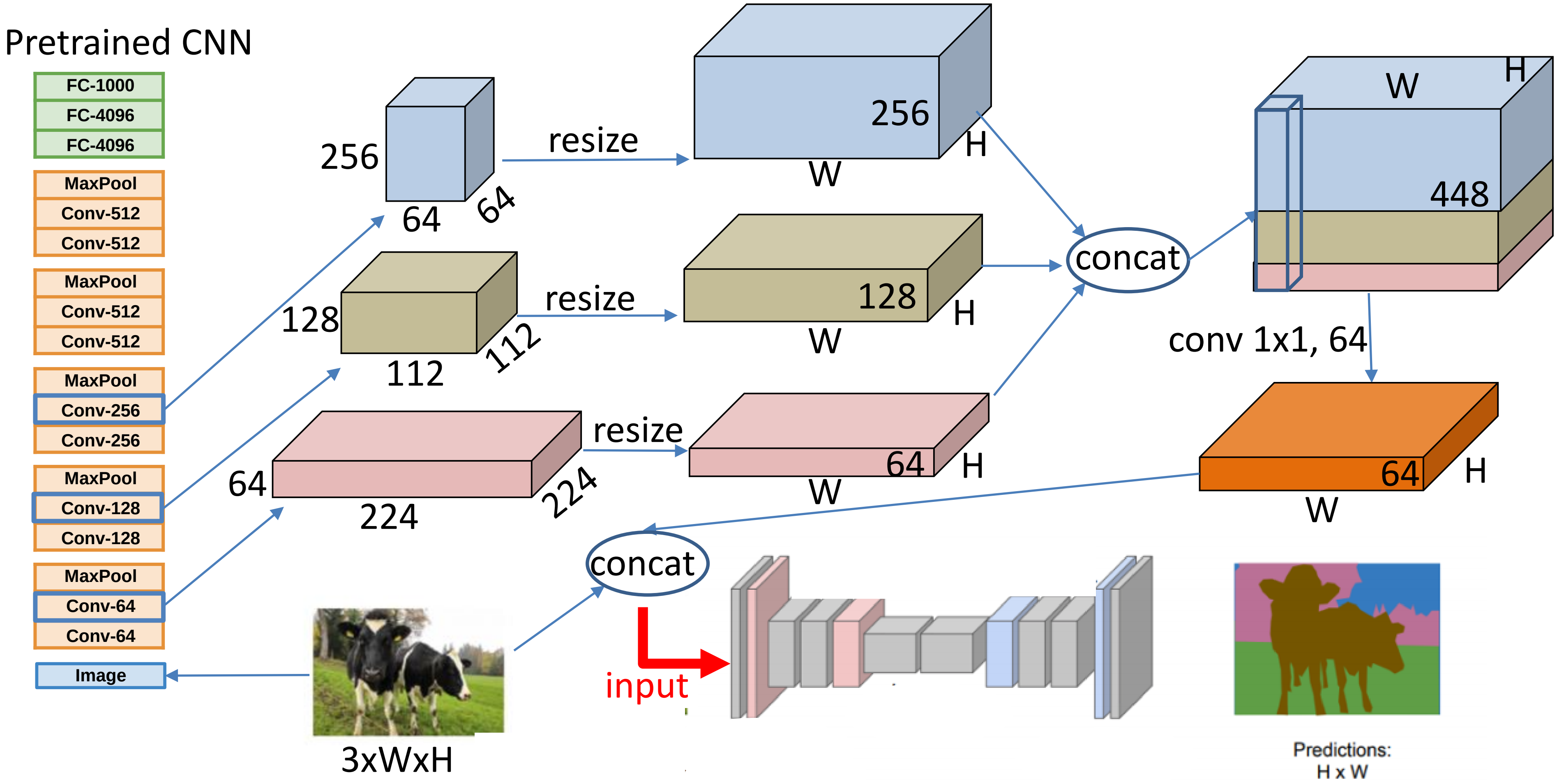


	very similar dataset	**very different dataset**
very little data	use linear classifier on top layer	most difficult case, try linear classifier from earlier stages
quite a lot of data	finetune a few layers	finetune a larger number of layers

Transfer Learning for Semantic Segmentation



Transfer Learning for Semantic Segmentation

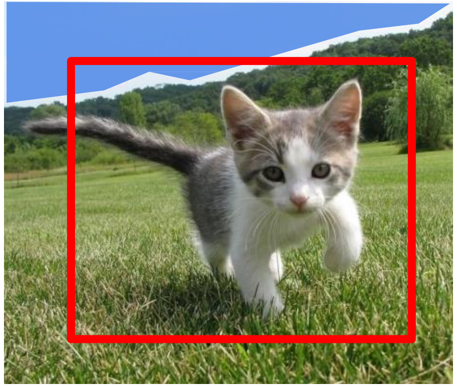


Classification vs. Localization

Classification

CAT

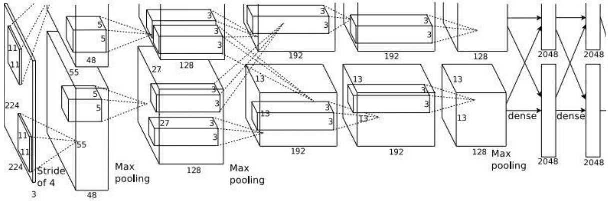
Classification + Localization



CAT

Classification + Localization

This image is CC0 public domain



Vector
4096

Fully Connected
4096 to 1000

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

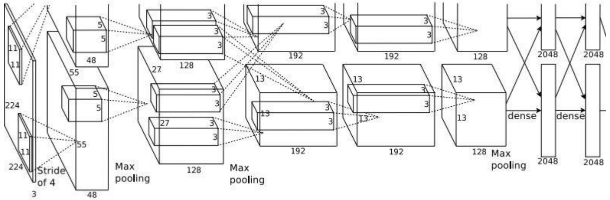
Fully Connected
4096 to 4

Box Coords
(x, y, w, h)

Treat localization as a regression problem!

Classification + Localization

This image is CC0 public domain



Vector
4096

Fully Connected
4096 to 1000

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

Correct label
Cat

Softmax Loss

Fully Connected
4096 to 4

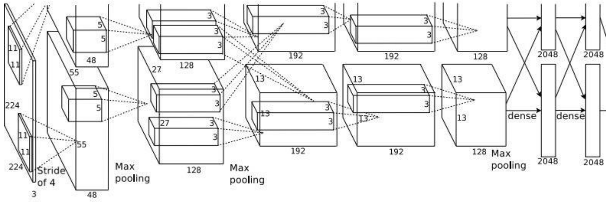
Box Coords
(x, y, w, h)

L2 Loss

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

Treat localization as a regression problem!

Classification + Localization



Vector
4096

Fully Connected:
4096 to 1000

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

Correct label
Cat

Softmax Loss

Multitask Loss

+ → **Loss**

Fully Connected:
4096 to 4

Box Coords
(x, y, w, h)

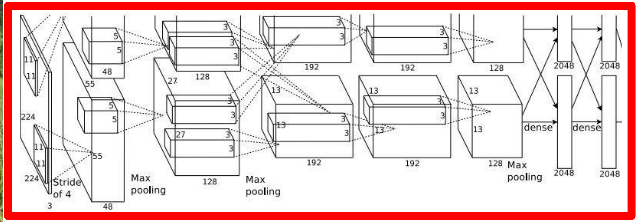
L2 Loss

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

Treat localization as a regression problem!

Classification + Localization

This image is CC0 public domain



Often pretrained on ImageNet
(Transfer learning)

Vector
4096

Fully Connected:
4096 to 1000

Class Scores

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

Fully Connected:
4096 to 4

Box Coords
(x, y, w, h)

Multitask Loss

Correct label:
Cat

Softmax Loss

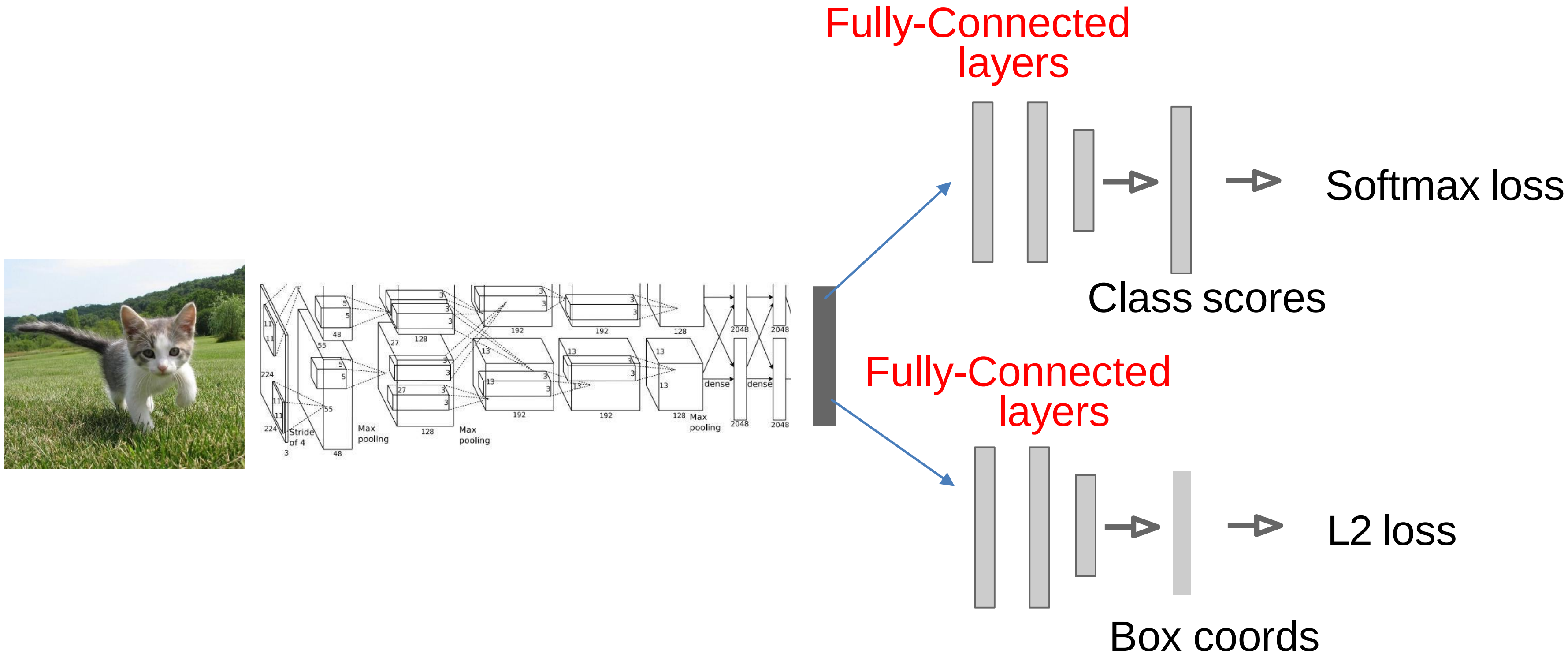
+ → **Loss**

L2 Loss

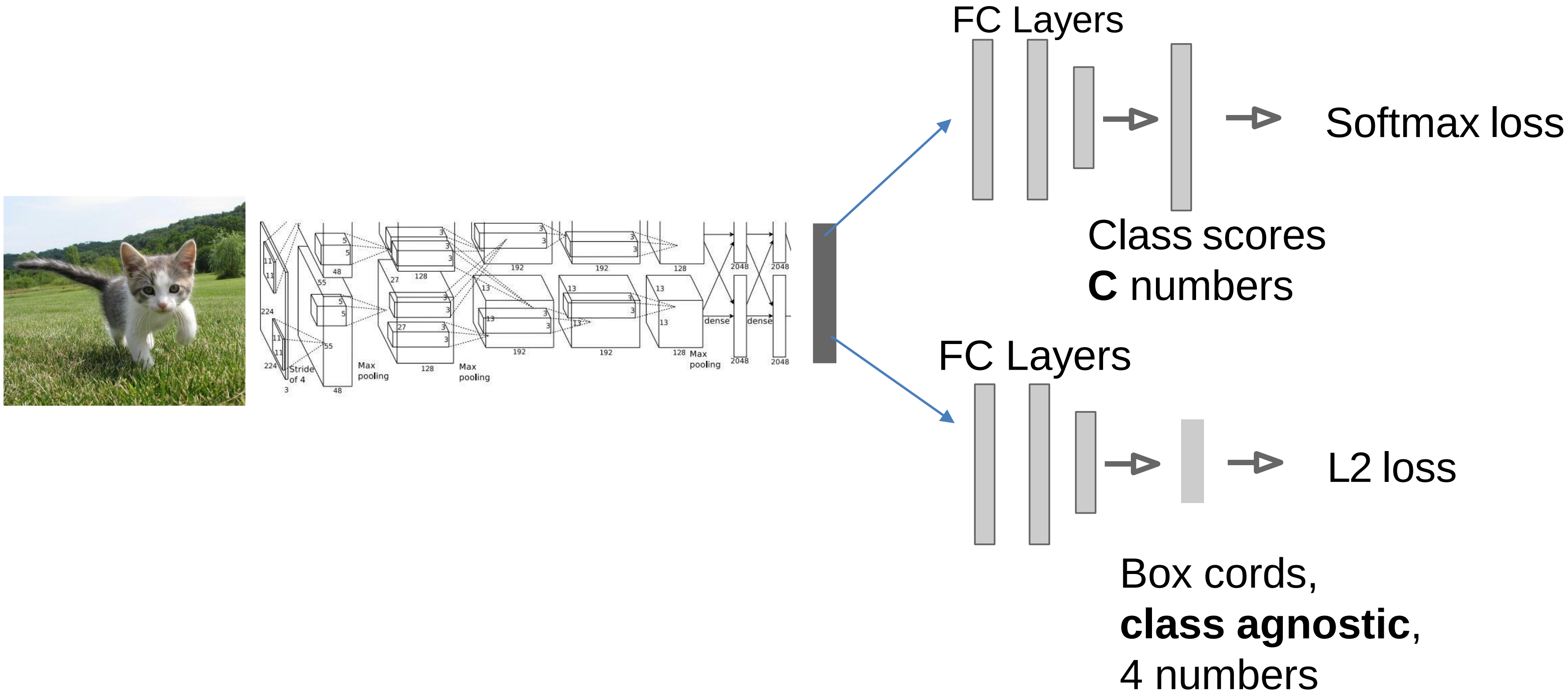
Correct box:
(x', y', w', h')

Treat localization as a regression problem!

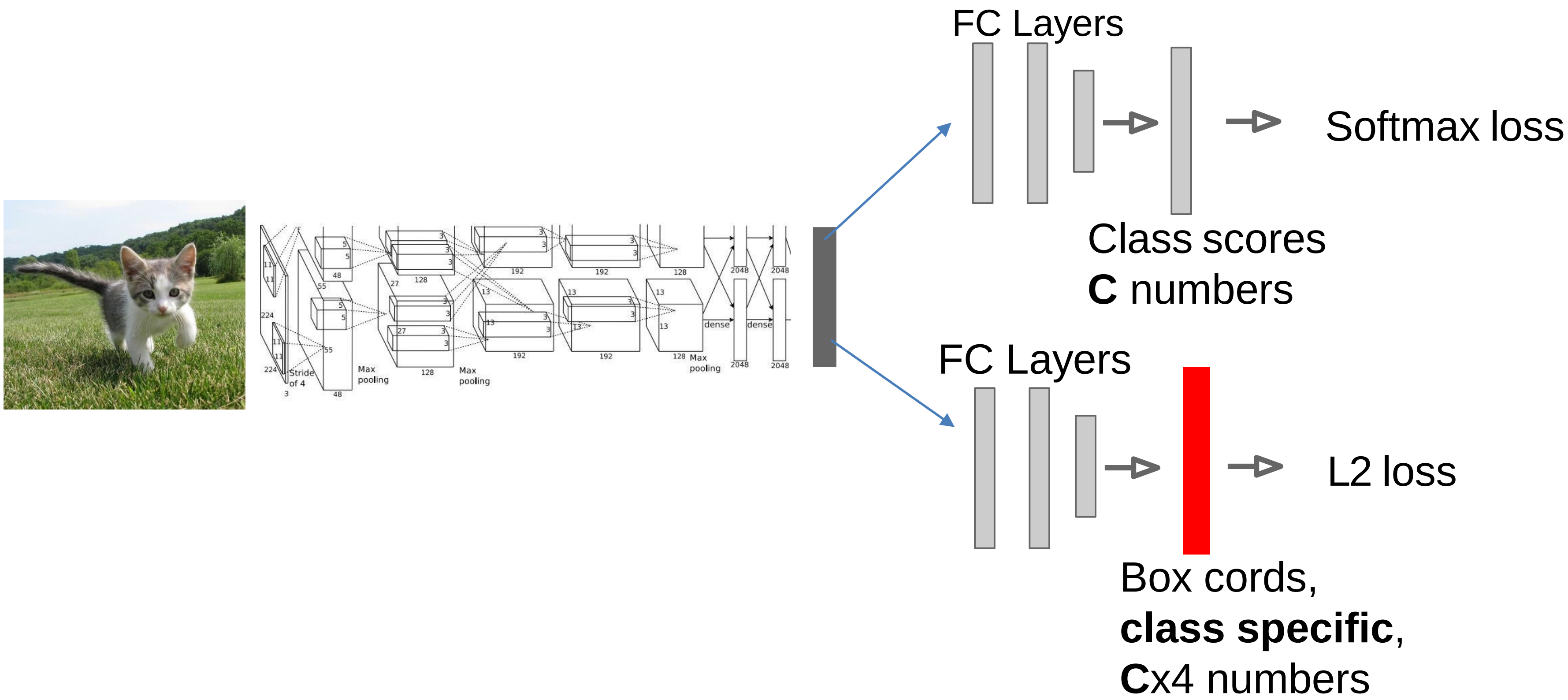
Classification + Localization: More FC Layers



Classification + Localization:
Class agnostic vs per class regression

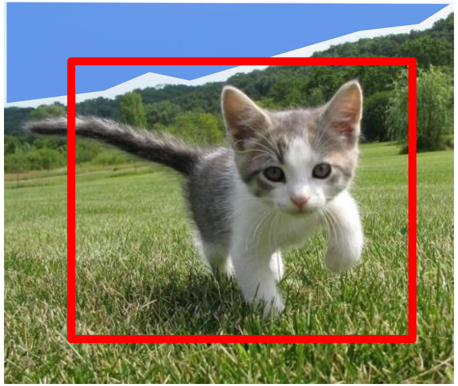


Classification + Localization: Class agnostic vs per class regression

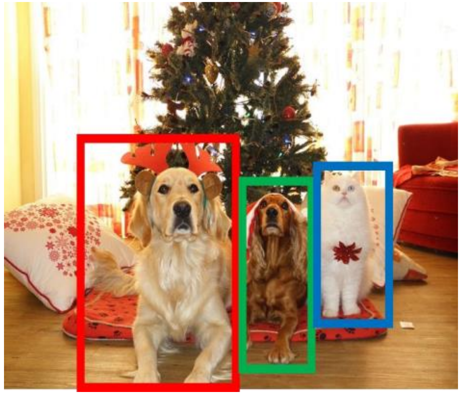


Localization vs. Object Detection

Localization



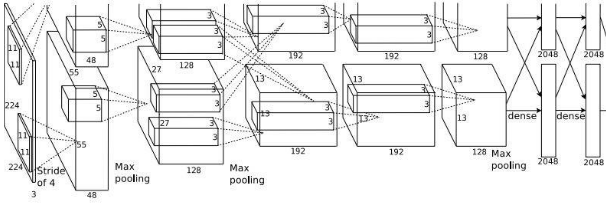
Detection



DOG, DOG, CAT

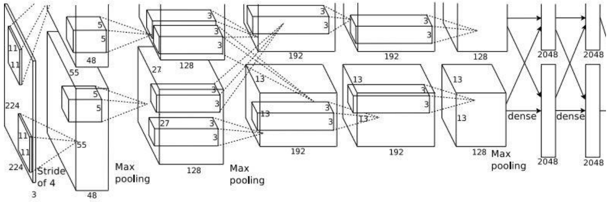
Object categories +
2D bounding boxes

Object Detection as Regression?



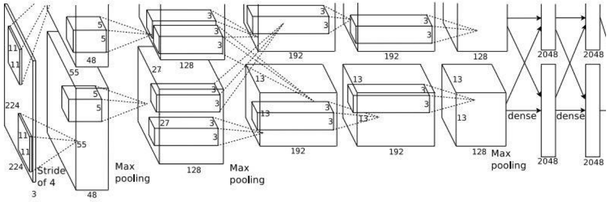
CAT: (x, y, w, h)

4 numbers



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

12 numbers



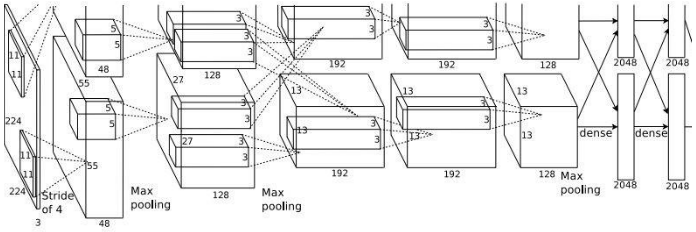
DUCK: (x, y, w, h)
DUCK: (x, y, w, h)

....

Many numbers

Each image needs a different number of outputs!

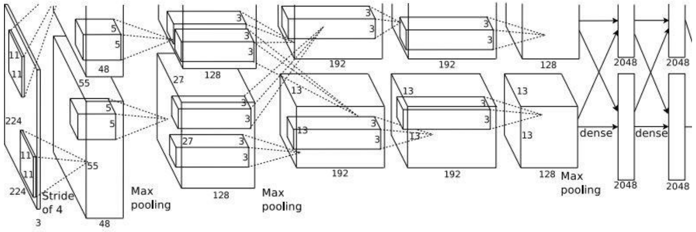
Object Detection as Classification: Sliding Window



Dog? NO
Cat? NO
Background? YES

- Apply a CNN to many different crops of the image
- Add an additional “background” class
- CNN classifies each crop as object or background

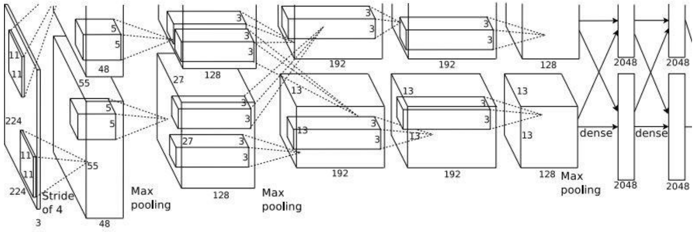
Object Detection as Classification: Sliding Window



Dog? YES
Cat? NO
Background? NO

- Apply a CNN to many different crops of the image
- Add an additional “background” class
- CNN classifies each crop as object or background

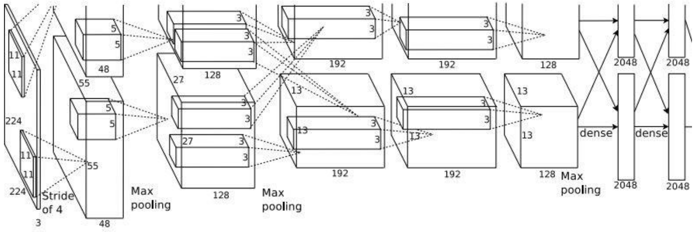
Object Detection as Classification: Sliding Window



Dog? YES
Cat? NO
Background? NO

- Apply a CNN to many different crops of the image
- Add an additional “background” class
- CNN classifies each crop as object or background

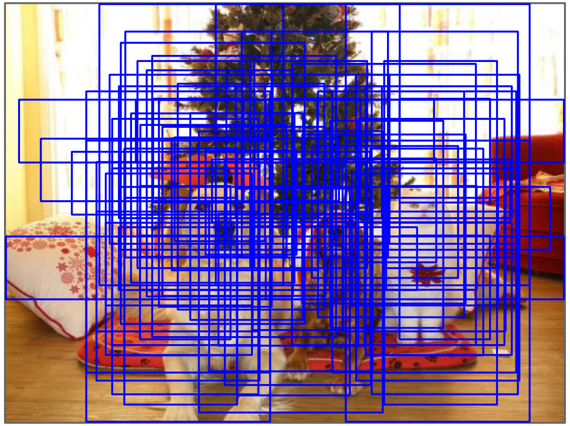
Object Detection as Classification: Sliding Window

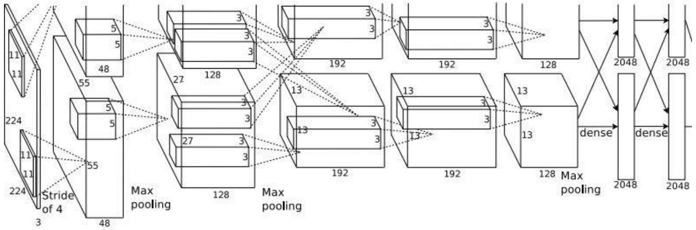


Dog? NO
Cat? YES
Background? NO

- Apply a CNN to many different crops of the image
- Add an additional "background" class
- CNN classifies each crop as object or background

Object Detection as Classification: Sliding Window



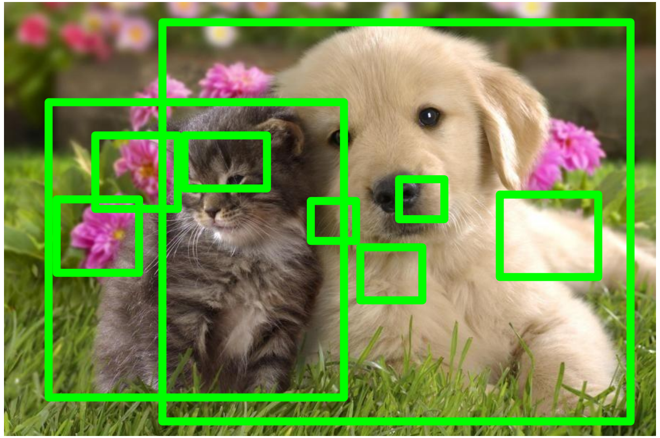


Dog? NO
Cat? YES
Background? NO

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

Region Proposals / Selective Search

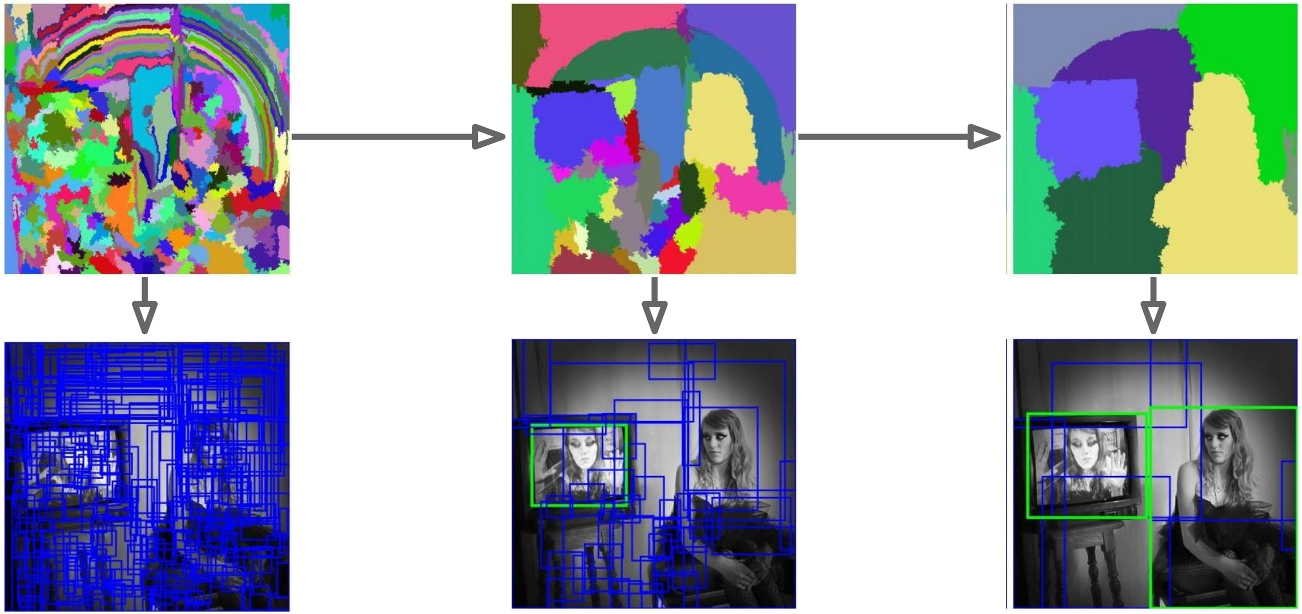
- Find “blobby” image regions that are likely to contain objects
  - Alexe et al, “Measuring the objectness of image windows”, TPAMI 2012
  - Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013
  - Cheng et al, “BING: Binarized normed gradients for objectness estimation at 300fps”, CVPR 2014
  - Zitnick and Dollar, “Edge boxes: Locating object proposals from edges”, ECCV 2014
- Relatively fast to run
  - e.g. Selective Search gives 2000 region proposals in a few seconds on CPU



Region Proposals: Selective Search

Bottom-up segmentation, merging regions at multiple scales

Convert regions to boxes



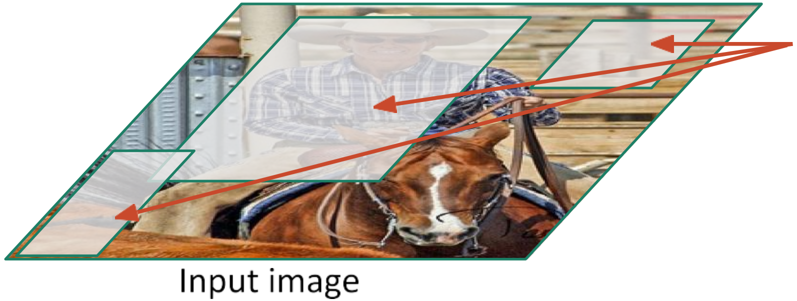
Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013

R-CNN

Input image

Girshick et al, “Rich feature hierarchies for accurate object detection and segmentation”, CVPR'14

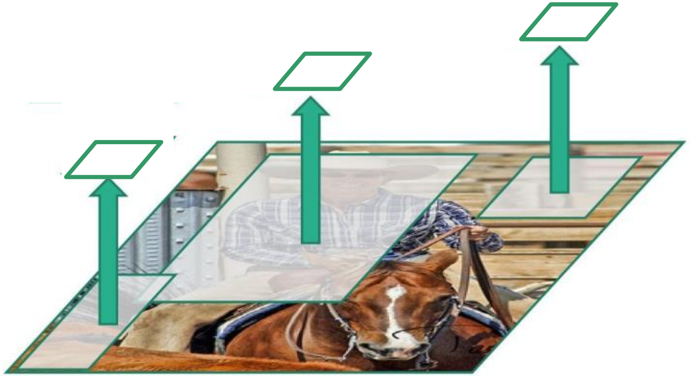
R-CNN



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

Girshick et al, “Rich feature hierarchies for accurate object detection and segmentation”, CVPR'14

R-CNN



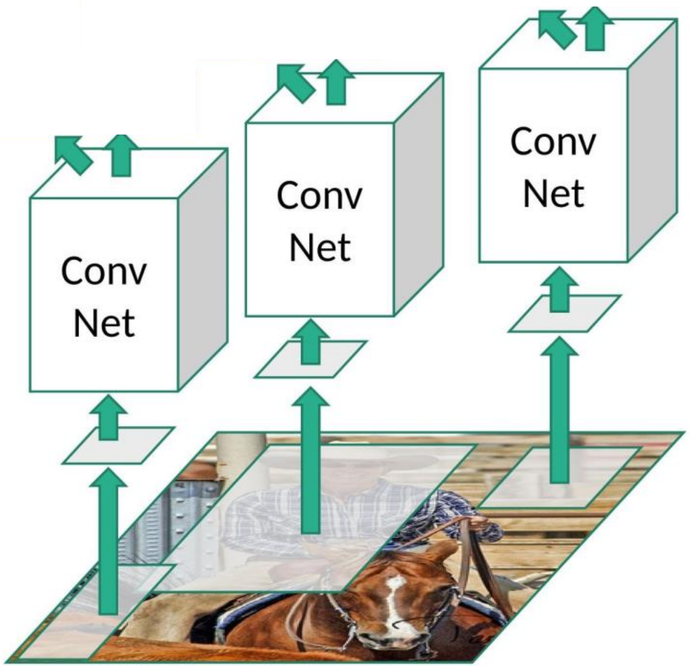
Warp ROI to the same size

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

Girshick et al, “Rich feature hierarchies for accurate object detection and segmentation”, CVPR'14

R-CNN

Sidenote: ConvNet pretrained on large classification dataset, then fine-tuned on proposal windows



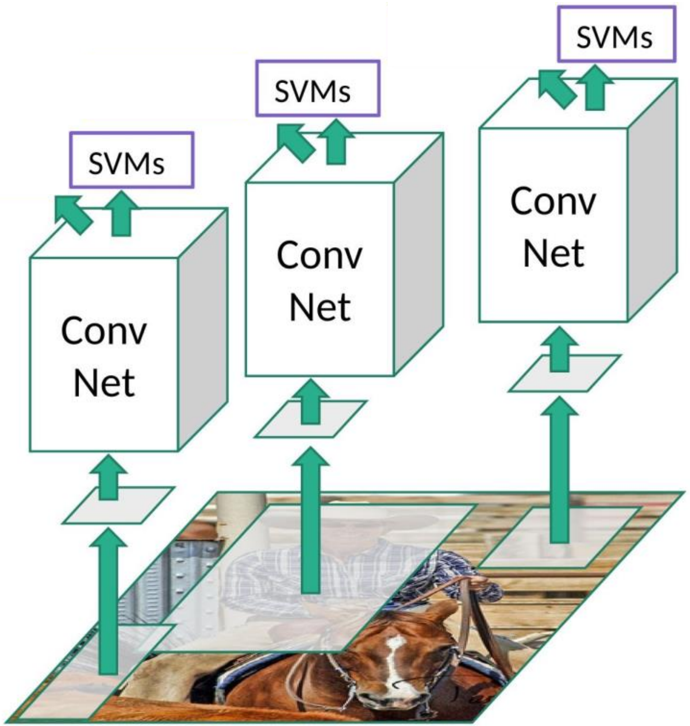
Forward each ROI through ConvNet and extract features

Warp ROI to the same size

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

Girshick et al, "Rich feature hierarchies for accurate object detection and segmentation", CVPR'14

R-CNN



Classify ROIs with SVM (type of linear classifier)

Forward each ROI through ConvNet and extract features

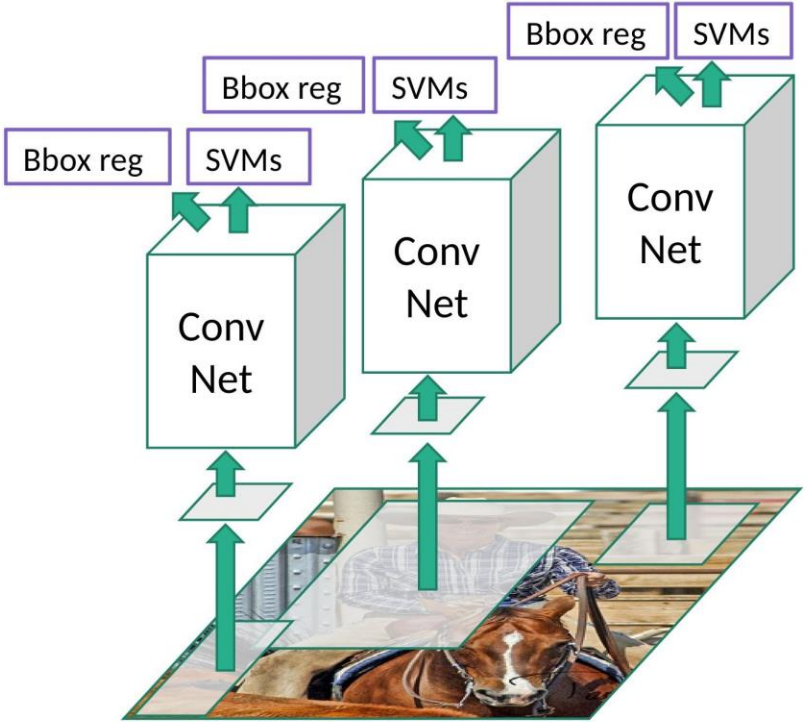
Warp ROI to the same size

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

Sidenote:
In original paper, separate SVM classifier worked better than softmax layer used for finetuning

Girshick et al, "Rich feature hierarchies for accurate object detection and segmentation", CVPR'14

R-CNN



Regression for bounding box offset

Classify ROIs with SVM (type of linear classifier)

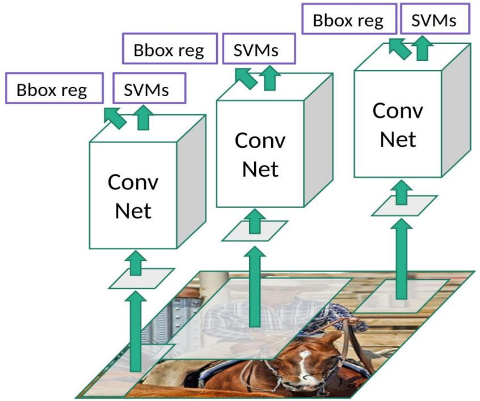
Forward each ROI through ConvNet and extract features

Warp ROI to the same size

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

Girshick et al, “Rich feature hierarchies for accurate object detection and segmentation”, CVPR'14

R-CNN: Problems



- Proposals are not learned, brittle
- Training is multi-stage pipeline
  - with each stage has its own loss function and is trained separately
  - i.e., box regression does not help classification, classification does not help regression
- Training is slow (84h), takes a lot of disk space
- Inference (detection) is slow
  - 47s / image with VGG16 [Simonyan & Zisserman. ICLR15]
  - Need to feed-forward 2000 proposals through ConvNet

Girshick et al, “Rich feature hierarchies for accurate object detection and segmentation”, CVPR'14

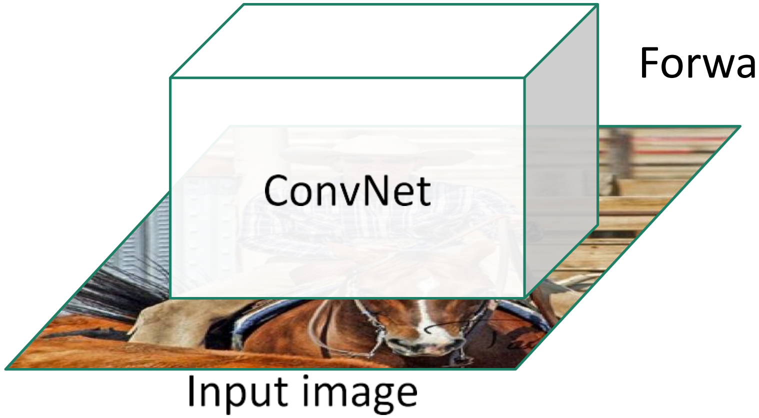
Fast R-CNN

Main idea: speedup via shared CNN computation

Input image

Girshick, “Fast R-CNN”, ICCV 2015

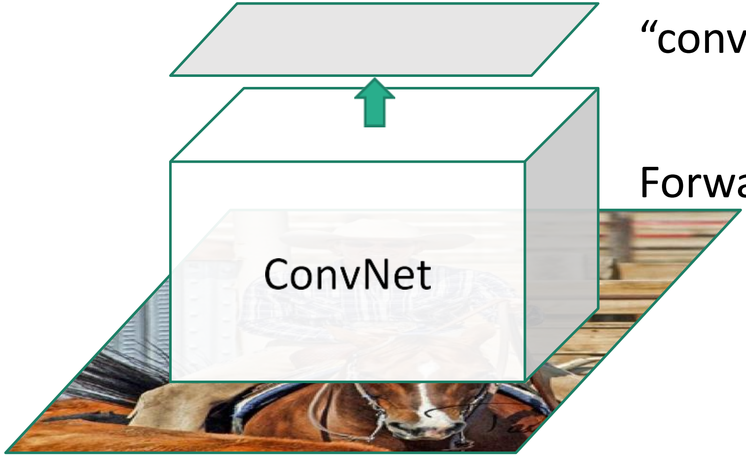
Fast R-CNN



Forward **the whole** image through ConvNet

Girshick, “Fast R-CNN”, ICCV 2015

Fast R-CNN

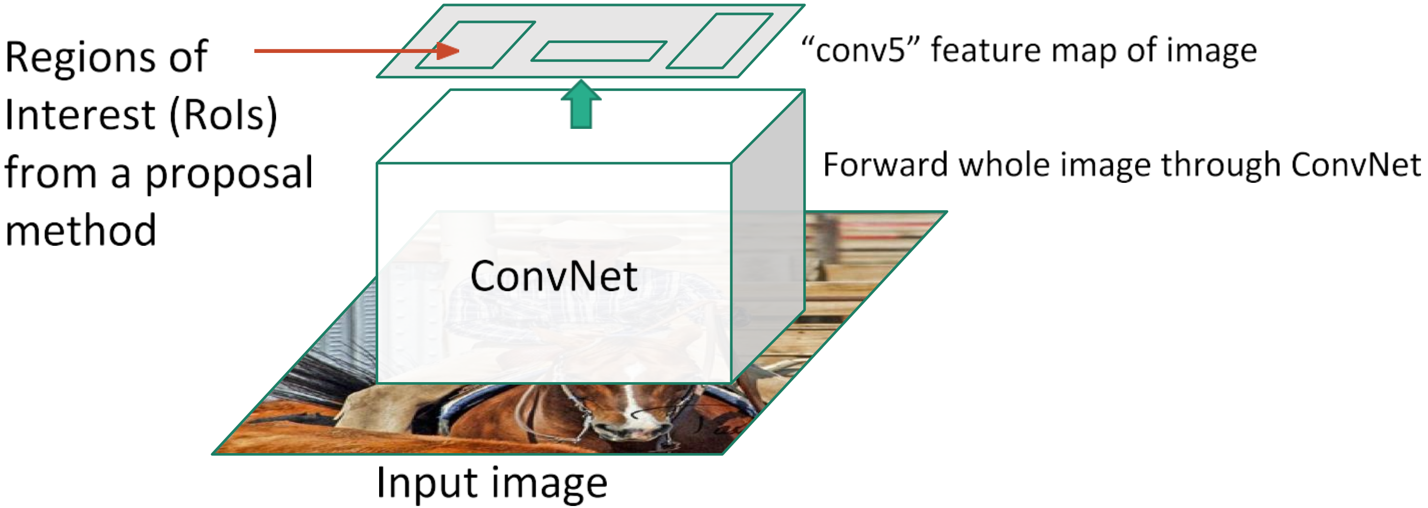


“conv5” feature map of the image

Forward **the whole** image through ConvNet

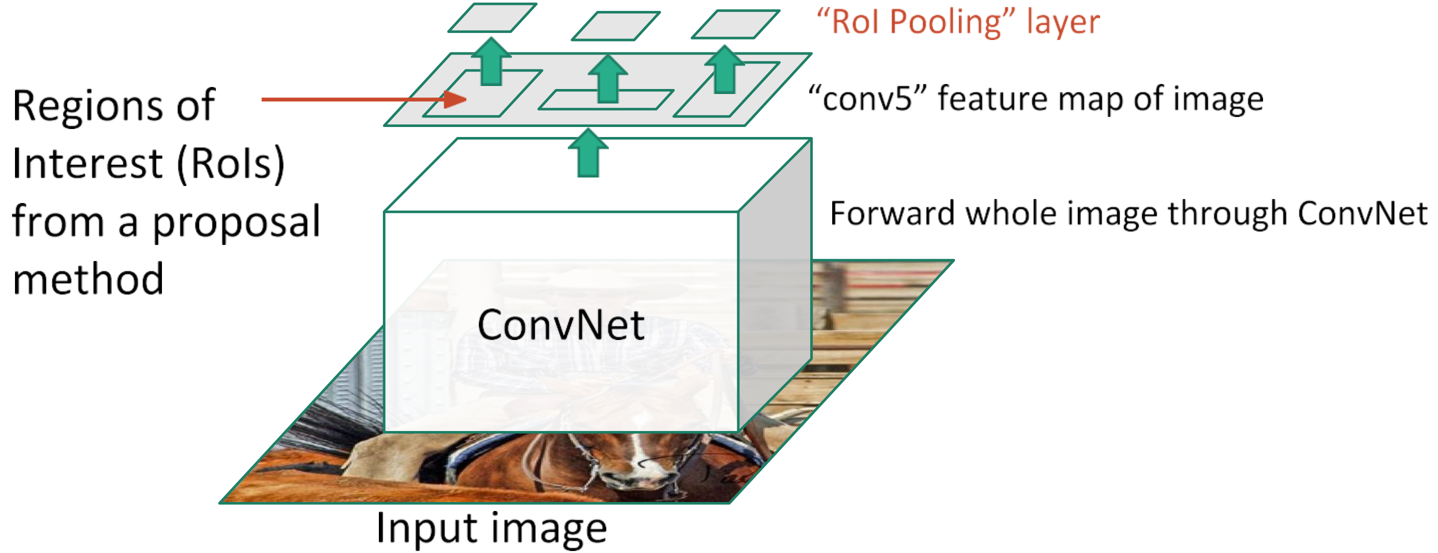
Girshick, “Fast R-CNN”, ICCV 2015

Fast R-CNN



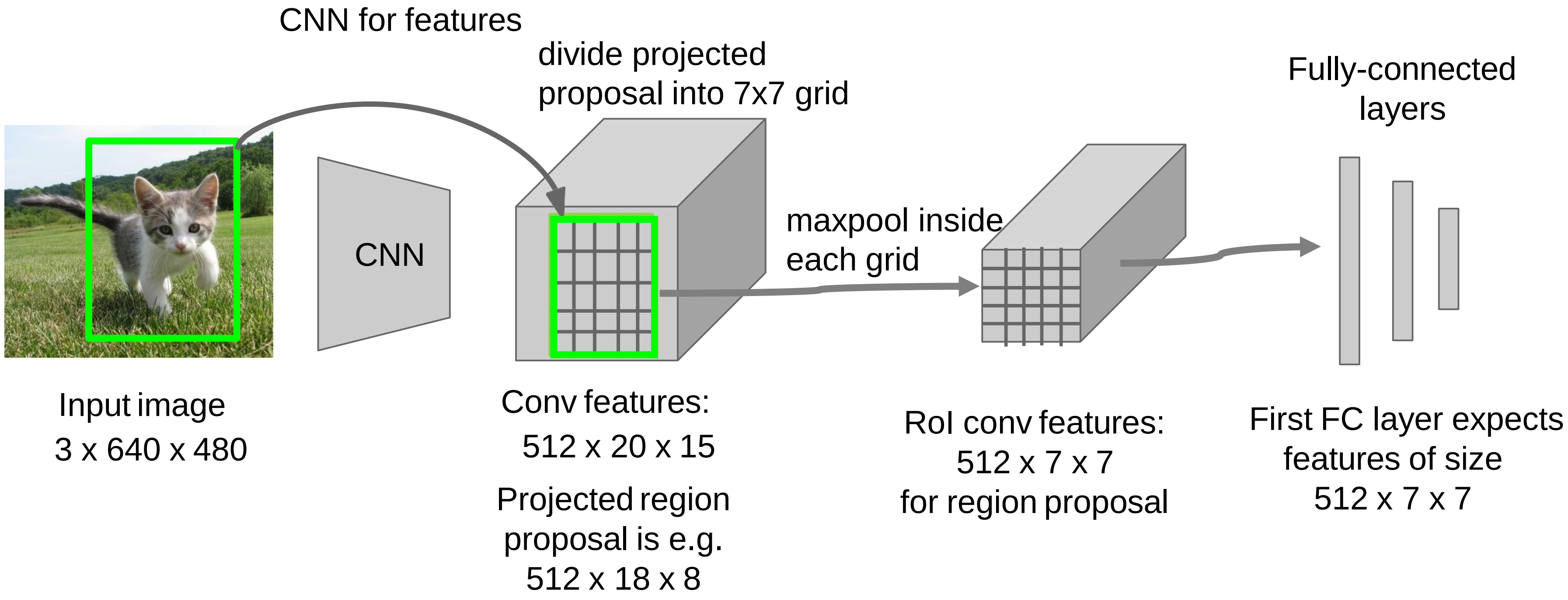
Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN



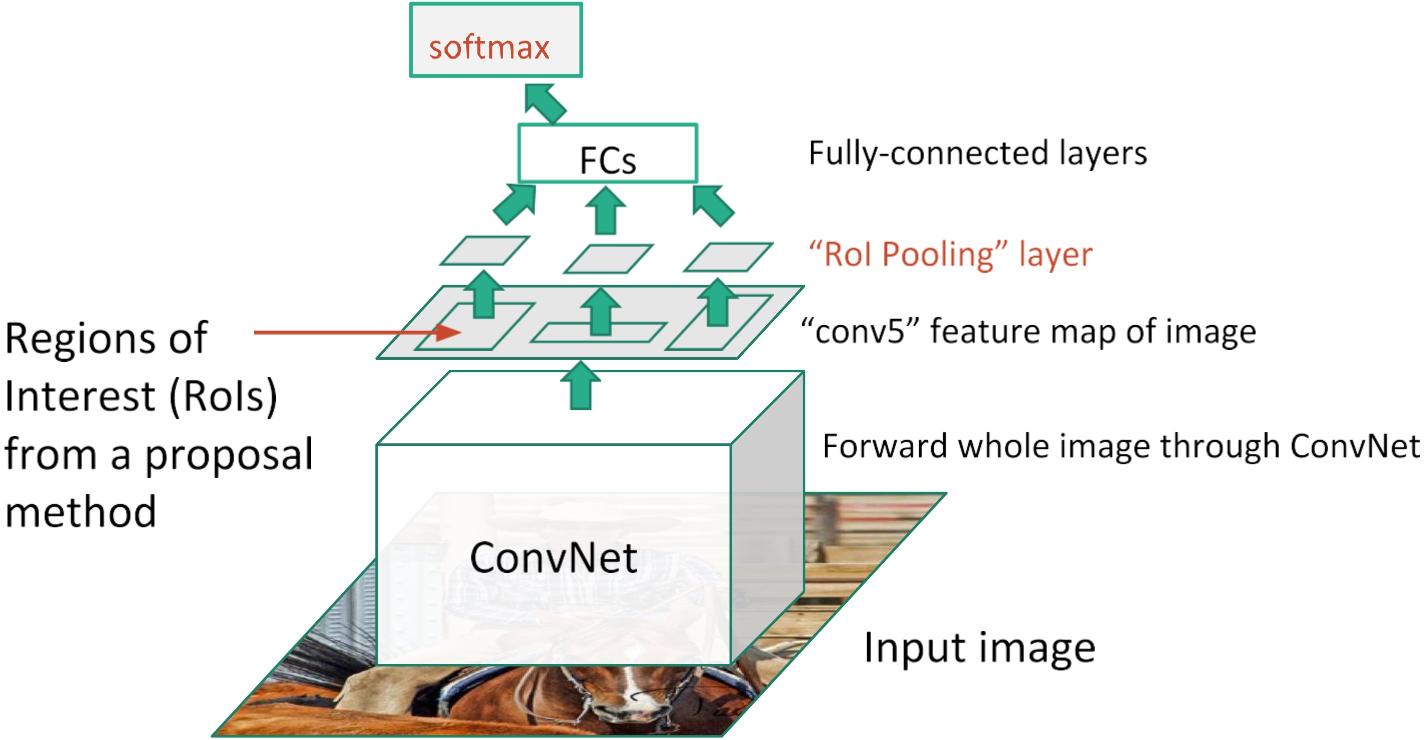
Girshick, "Fast R-CNN", ICCV 2015

Fast R-CNN: RoI Pooling



Girshick, “Fast R-CNN”, ICCV 2015

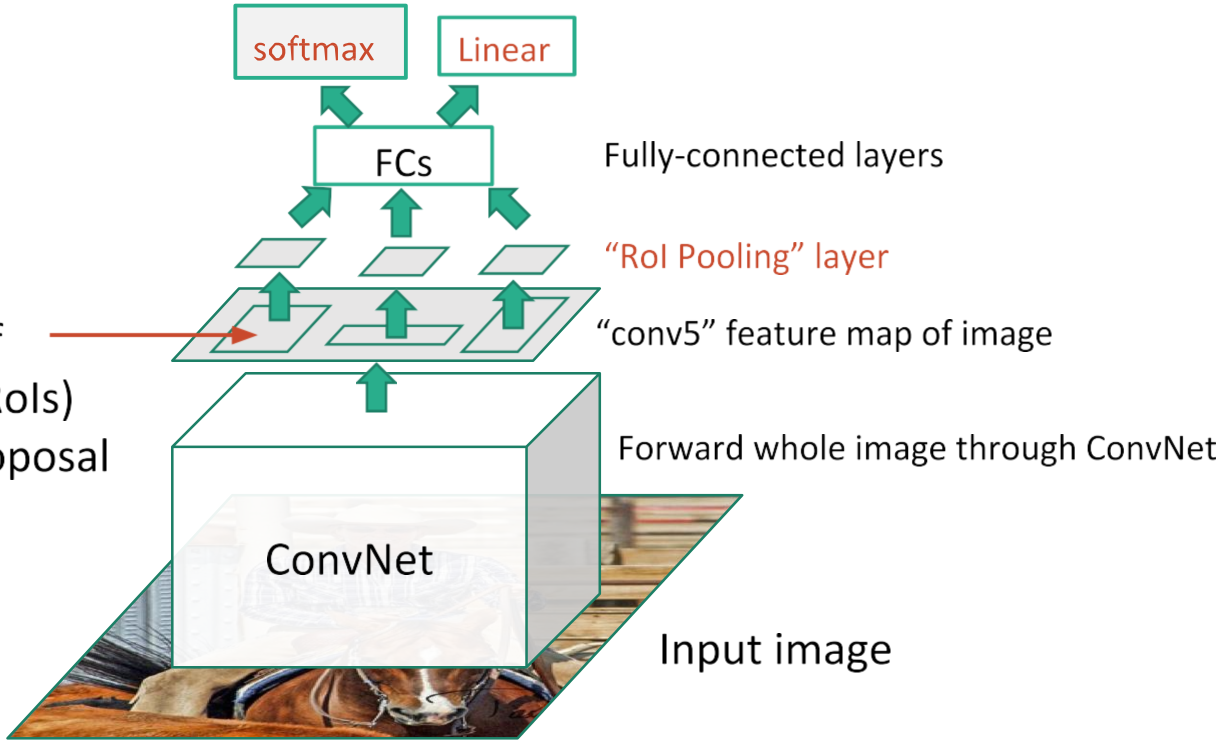
Fast R-CNN



Girshick, "Fast R-CNN", ICCV 2015

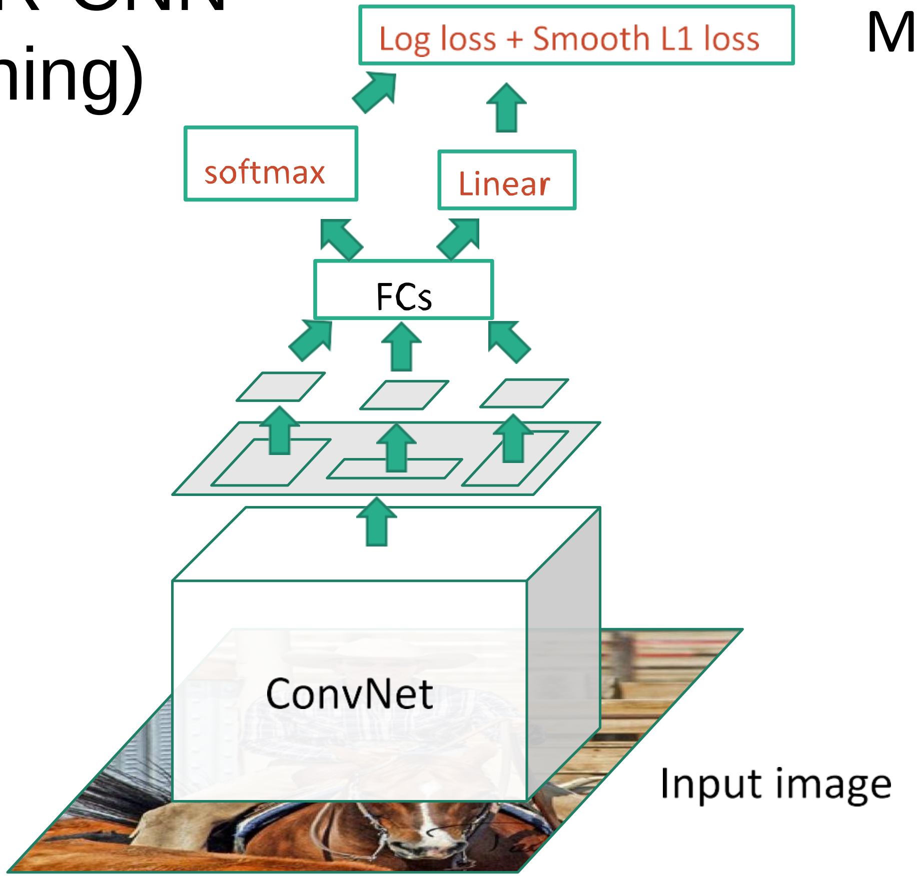
Fast R-CNN

Regions of Interest (RoIs) from a proposal method



Girshick, “Fast R-CNN”, ICCV 2015

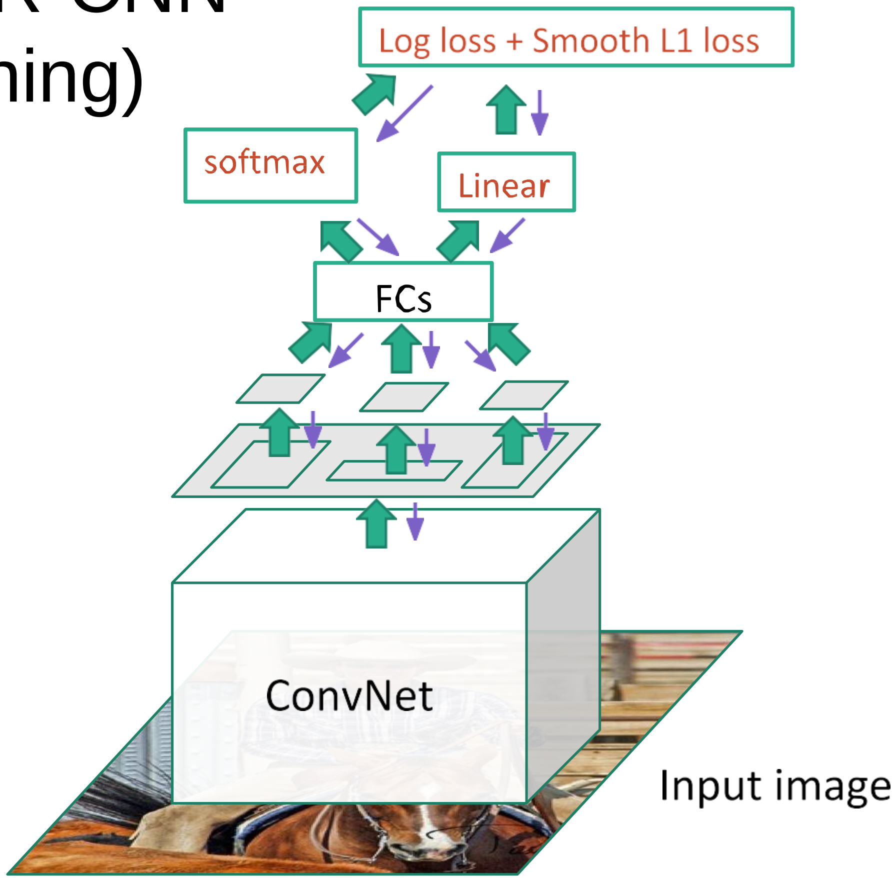
Fast R-CNN (Training)



Multi-task loss

Girshick, “Fast R-CNN”, ICCV 2015

Fast R-CNN (Training)



Multi-task loss

- single loss function
- end-to-end training
- box regression and classification can influence each other

Girshick, “Fast R-CNN”, ICCV 2015

Fast R-CNN Results

	R-CNN	**Fast R-CNN**
Training Time:	84 hours	**9.5 hours**
(Speedup)	1x	**8.8x**

Faster!

Using VGG-16 CNN on Pascal VOC 2007 dataset

Fast R-CNN Results

	R-CNN	Fast R-CNN
Training Time:	84 hours	**9.5 hours**
(Speedup)	1x	**8.8x**
Test time per image	47 seconds	**0.32 seconds**
(Speedup)	1x	**146x**

Faster!

FASTER!

Using VGG-16 CNN on Pascal VOC 2007 dataset

Fast R-CNN Results

		R-CNN	**Fast R-CNN**
Faster!	Training Time:	84 hours	**9.5 hours**
	(Speedup)	1x	**8.8x**
FASTER!	Test time per image	47 seconds	**0.32 seconds**
	(Speedup)	1x	**146x**
Better!	mAP (VOC 2007)	66.0	**66.9**

Using VGG-16 CNN on Pascal VOC 2007 dataset

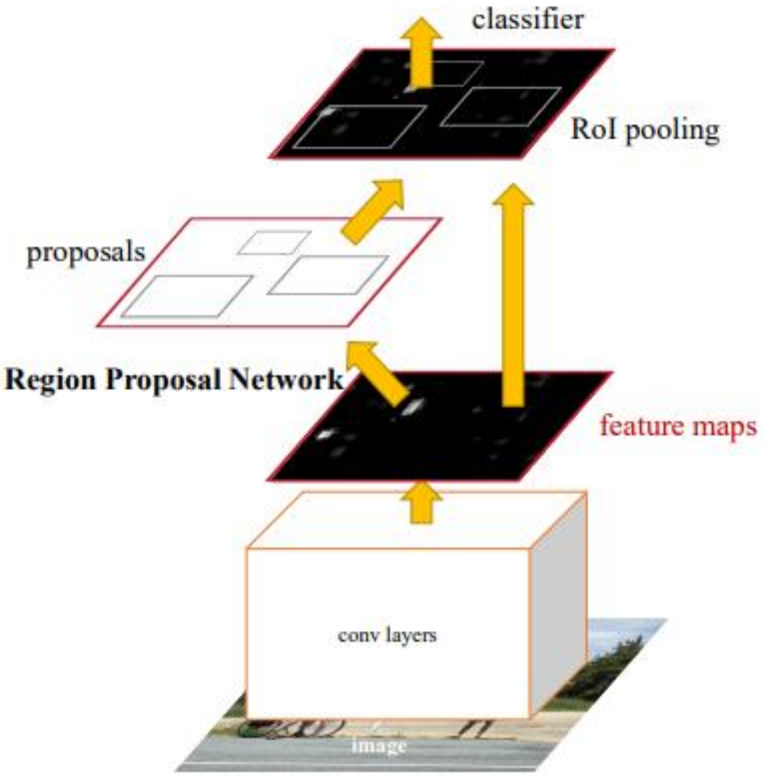
Fast R-CNN Problem

	R-CNN	Fast R-CNN
Test time per image	47 seconds	**0.32 seconds**
(Speedup)	1x	**146x**
Test time per image with Selective Search	50 seconds	**2 seconds**
(Speedup)	1x	**25x**

Test-time speeds don’t include region proposals

Faster R-CNN

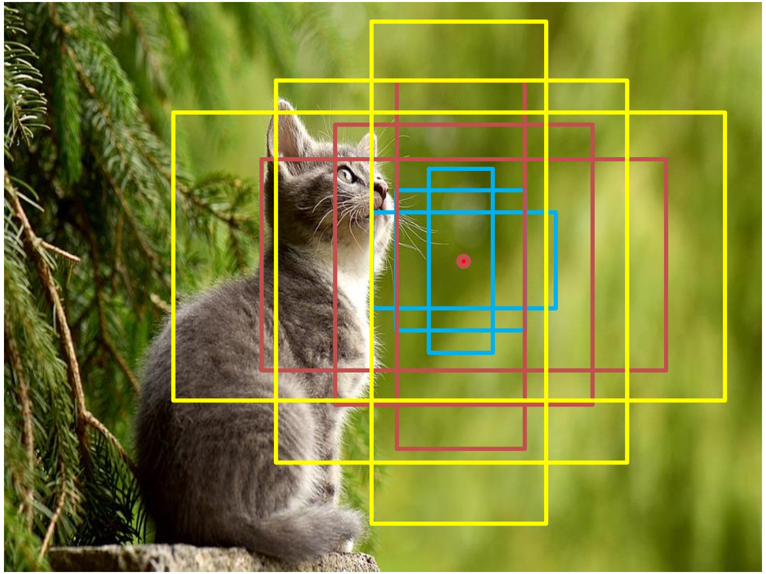
- Make CNN learn the proposals
  - Insert **Region Proposal Network (RPN)** to predict proposals from features
  - After RPN, as before, use
    - RoI Pooling
    - Classifier
    - Box regressor



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

Faster R-CNN: Region Proposal Network

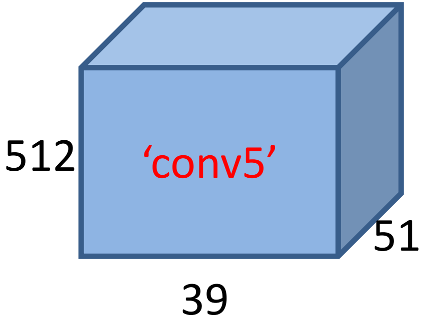
- Consider a fixed set of rectangles as possible proposals
- Choices in paper:
  - height and width ratios 1:1, 1:2, 2:1, at
  - 3 scales, 128x128, 256x256, 512x512
  - stride 16
  - Gives 17,901 boxes for 600x800 image

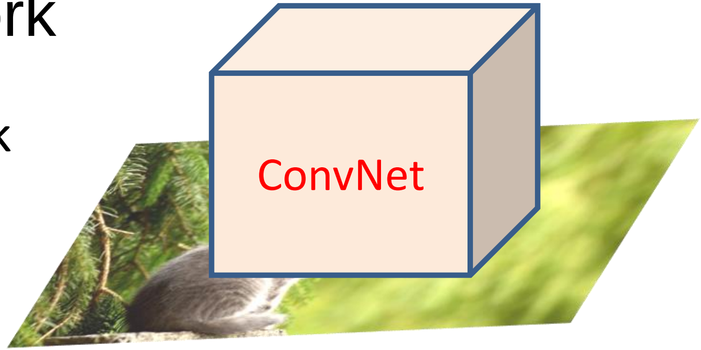


- Classify each box as proposal/not proposal
  - or object/no object
- How to classify efficiently?

Faster R-CNN: Region Proposal Network

- Take layer "conv5" for Region Proposal Network

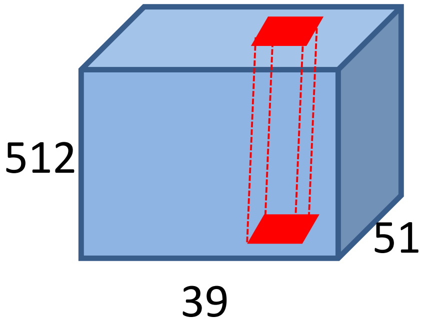




- Moving by 1 "spatial pixel" in this layer corresponds to moving by 16 pixels in the original image

Faster R-CNN: Region Proposal Network

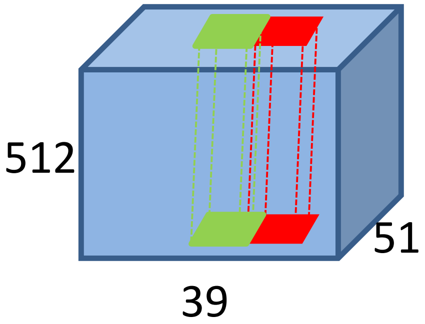
- Take layer “conv5” for Region Proposal Network

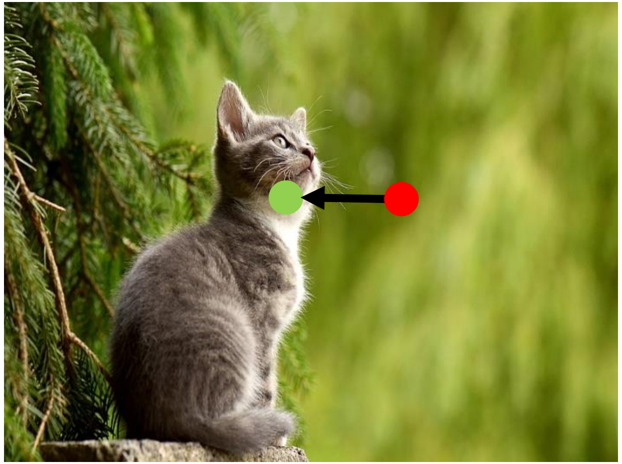


- Moving by 1 “spatial pixel” in this layer corresponds to moving by 16 pixels in the original image

Faster R-CNN: Region Proposal Network

- Take layer "conv5" for Region Proposal Network



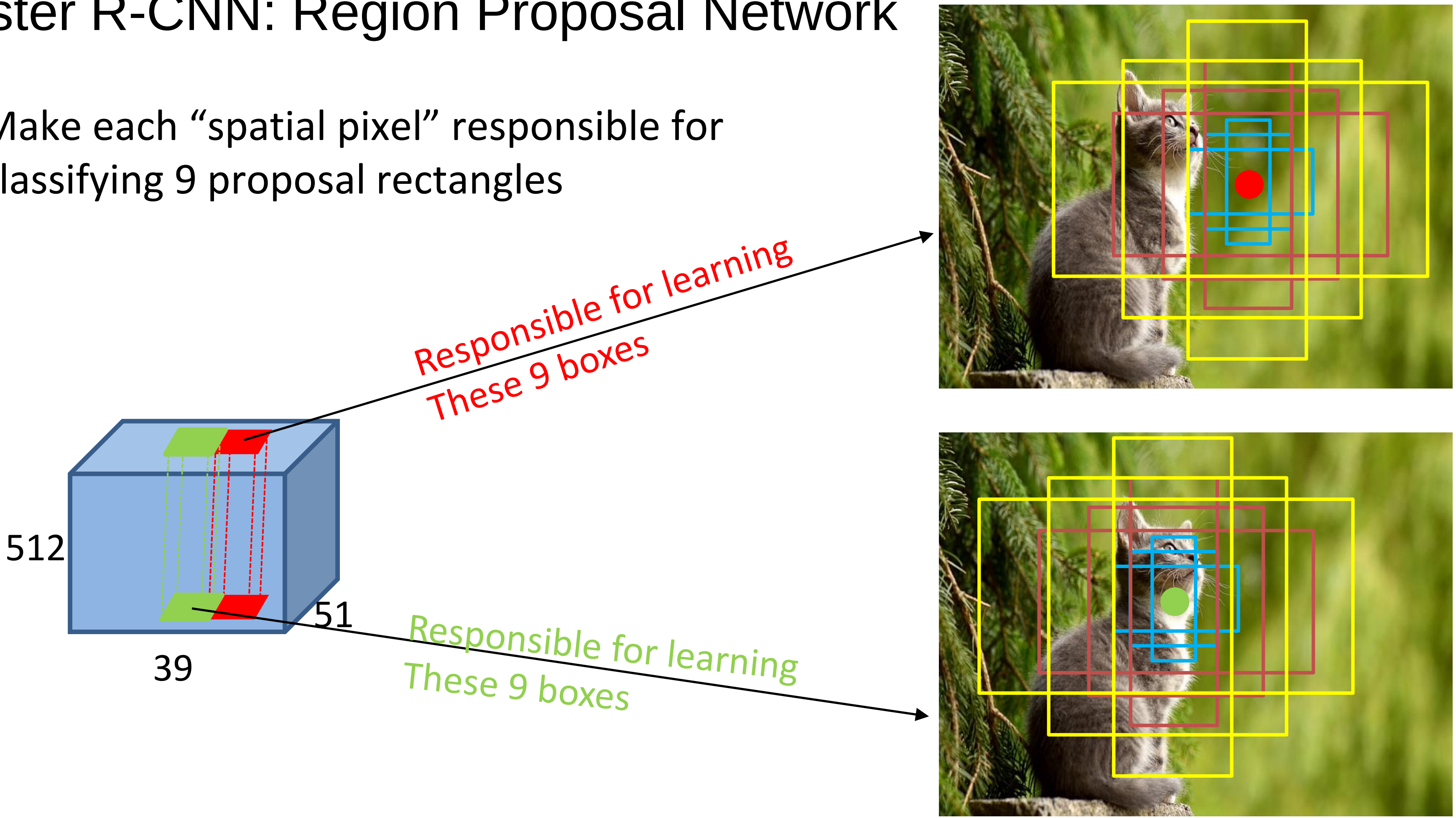


move by 16 pixels

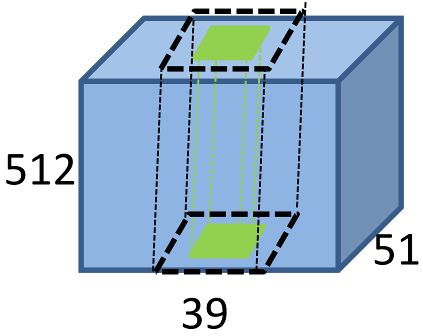
- Moving by 1 "spatial pixel" in this layer corresponds to moving by 16 pixels in the original image
- We have 9 possible proposal windows every 16 pixels

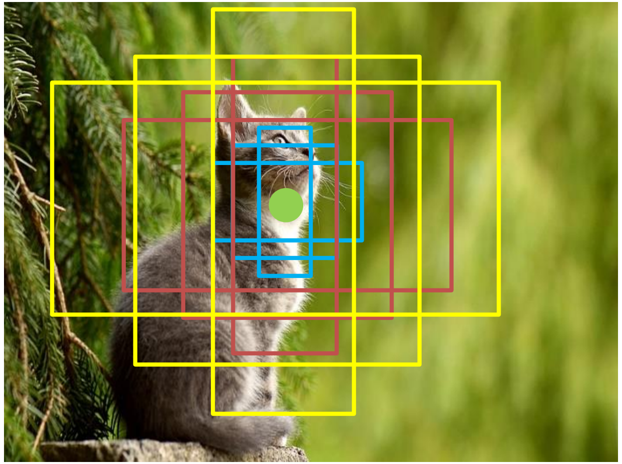
Faster R-CNN: Region Proposal Network

- Make each “spatial pixel” responsible for classifying 9 proposal rectangles



Faster R-CNN: Region Proposal Network

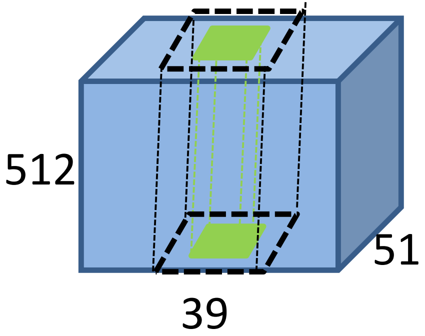


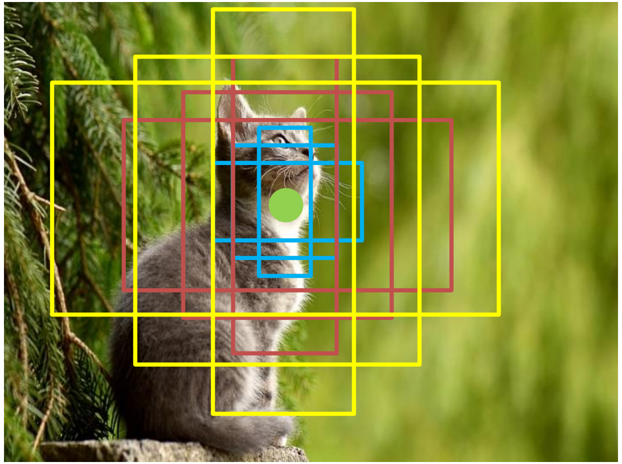


move by 16 pixels

- Already have 512 features for learning, but not enough
- Take 3 by 3 spatial window around green ‘pixel’ for learning

Faster R-CNN: Region Proposal Network



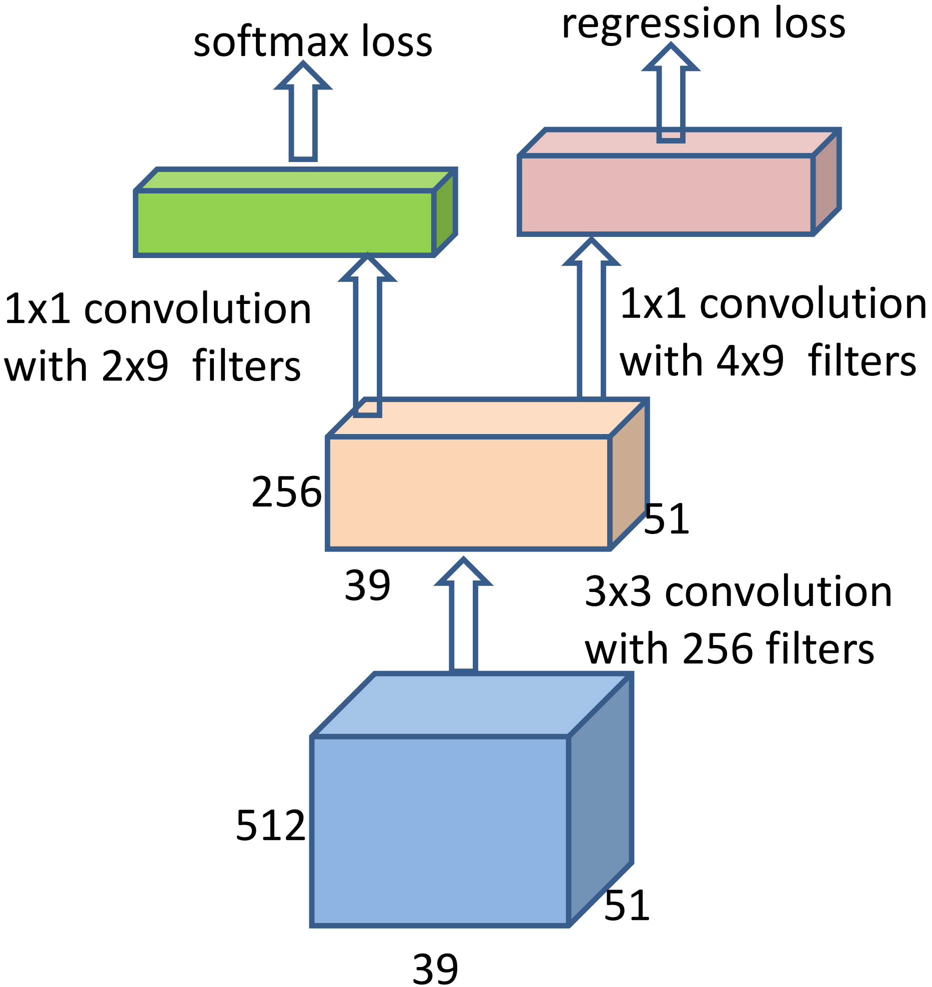


move by 16 pixels

- Already have 512 features for learning, but not enough
- Take 3 by 3 spatial window around green 'pixel' for learning
- Implemented as 3x3 convolutional layer for all pixels
  - with 256 filters, to get 256 new features per 'pixel'

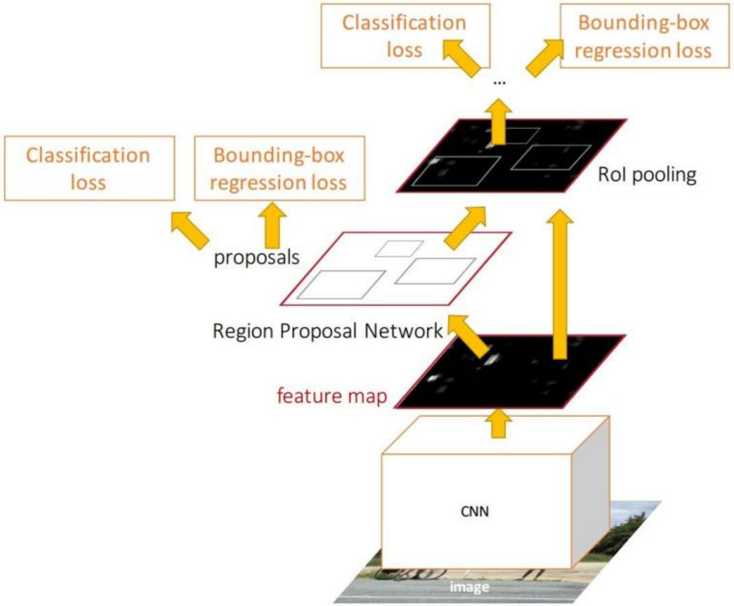
Faster R-CNN: Region Proposal Network

- Now need to classify each “pixel” as object or not object
  - for 9 different proposal boxes
- And get 4 box coordinates
  - for 9 different proposal boxes



Faster R-CNN: Training

- In the paper: Ugly pipeline
- Since publication: Joint training
- One network, four losses
  - RPN classification (anchor good / bad)
  - RPN regression (anchor -> proposal)
  - Fast R-CNN classification (over classes)
  - Fast R-CNN regression (proposal -> box)

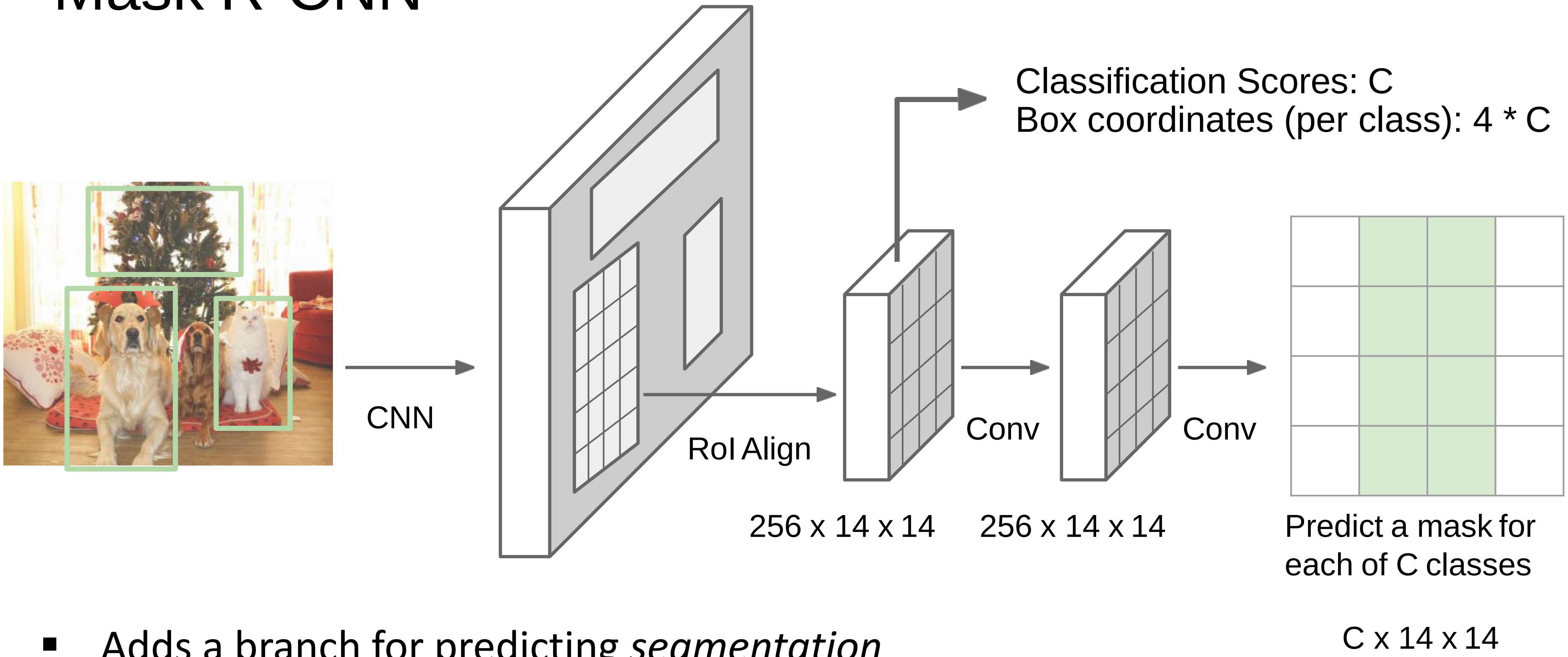


Slide credit: Ross Girschick

Faster R-CNN: Results

	R-CNN	**Fast R-CNN**	**Faster R-CNN**
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
(Speedup)	1x	25x	**250x**
mAP (VOC 2007)	66.0	**66.9**	**66.9**

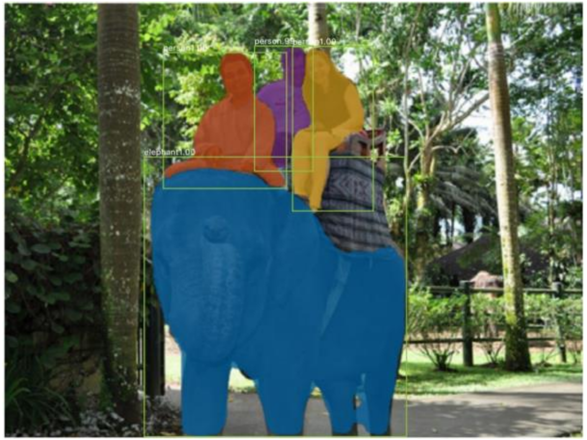
Mask R-CNN

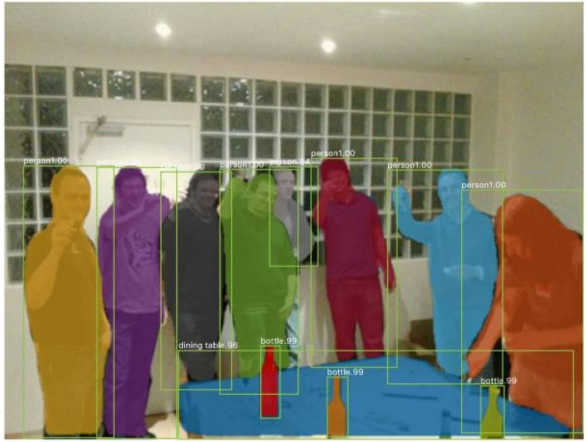


- Adds a branch for predicting *segmentation* masks on each RoI
- Also changes how RoI pooling implemented
- Computational overhead is small

He et al, “Mask R-CNN”, arXiv 2017

Mask R-CNN: Very Good Results!





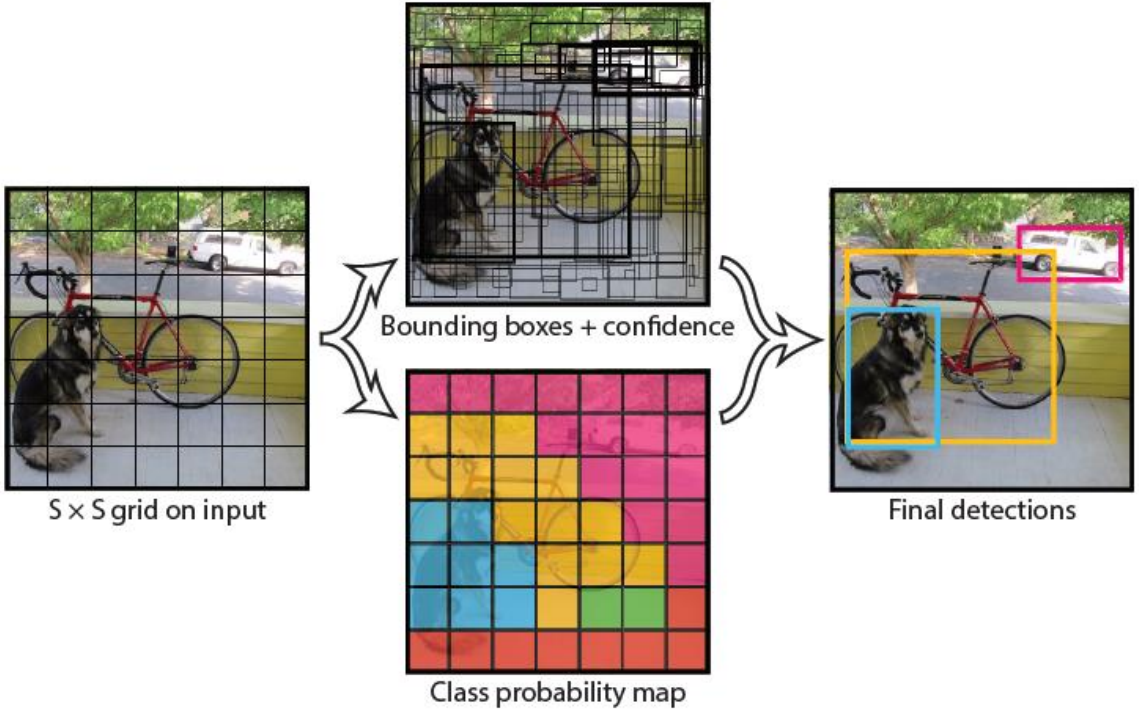
He et al, “Mask R-CNN”, arXiv 2017

Mask R-CNN
Also does pose

He et al, “Mask R-CNN”, arXiv 2017

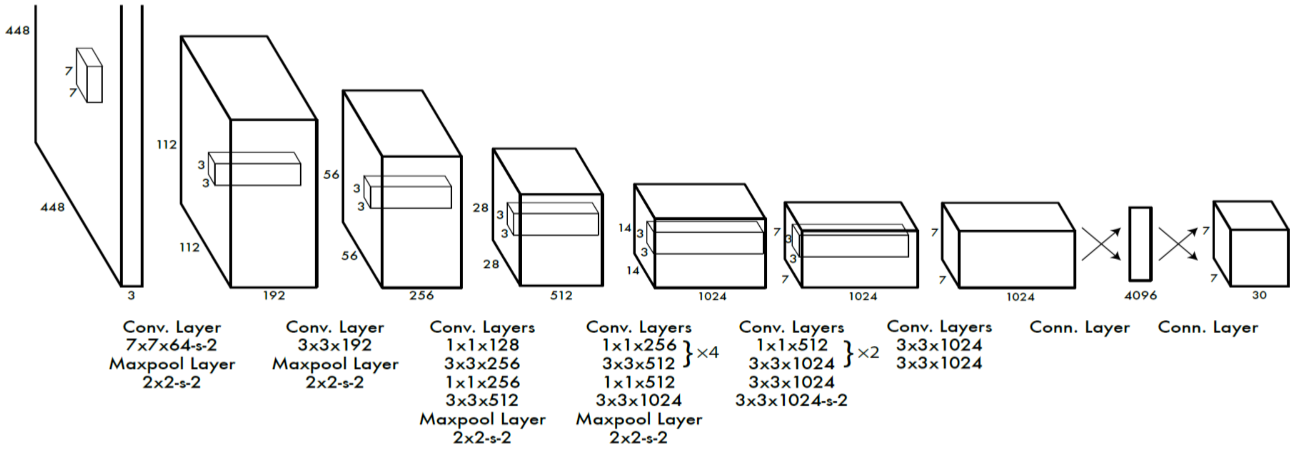
YOLO- You Only Look Once

Idea: No bounding box proposals. Predict a class and a box for every location in a grid.



https://arxiv.org/abs/1506.02640 Redmon et al. CVPR 2016.

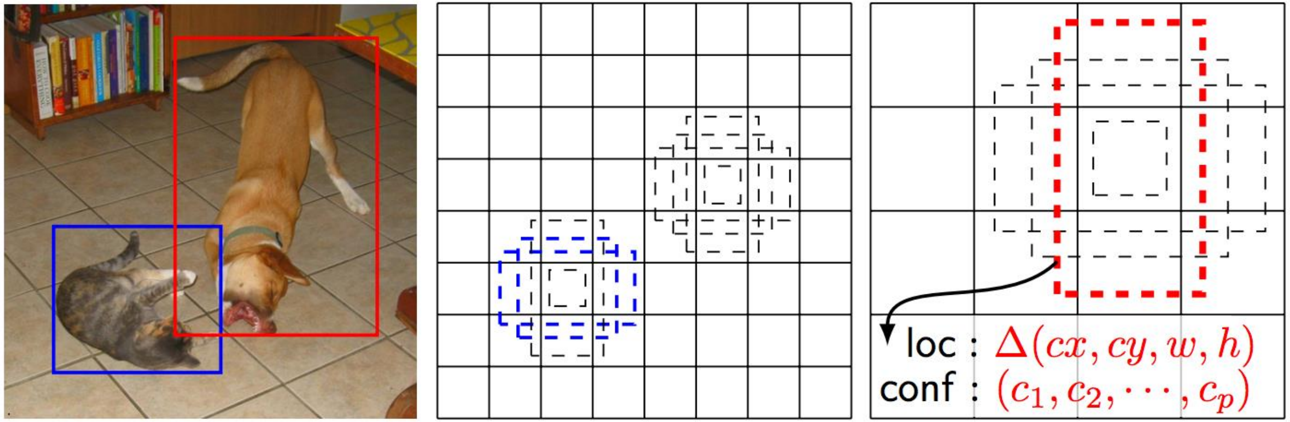
YOLO- You Only Look Once



Divide the image into 7x7 cells.
Each cell trains a detector.
The detector needs to predict the object's class distributions.
The detector has 2 bounding-box predictors to predict bounding-boxes and confidence scores.

https://arxiv.org/abs/1506.02640 Redmon et al. CVPR 2016.

SSD: Single Shot Detector



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

Idea: Similar to YOLO, but denser grid map, multiscale grid maps. + Data augmentation + Hard negative mining + Other design choices in the network.

Liu et al. ECCV 2016.

Object Detection: Impact of Deep Learning