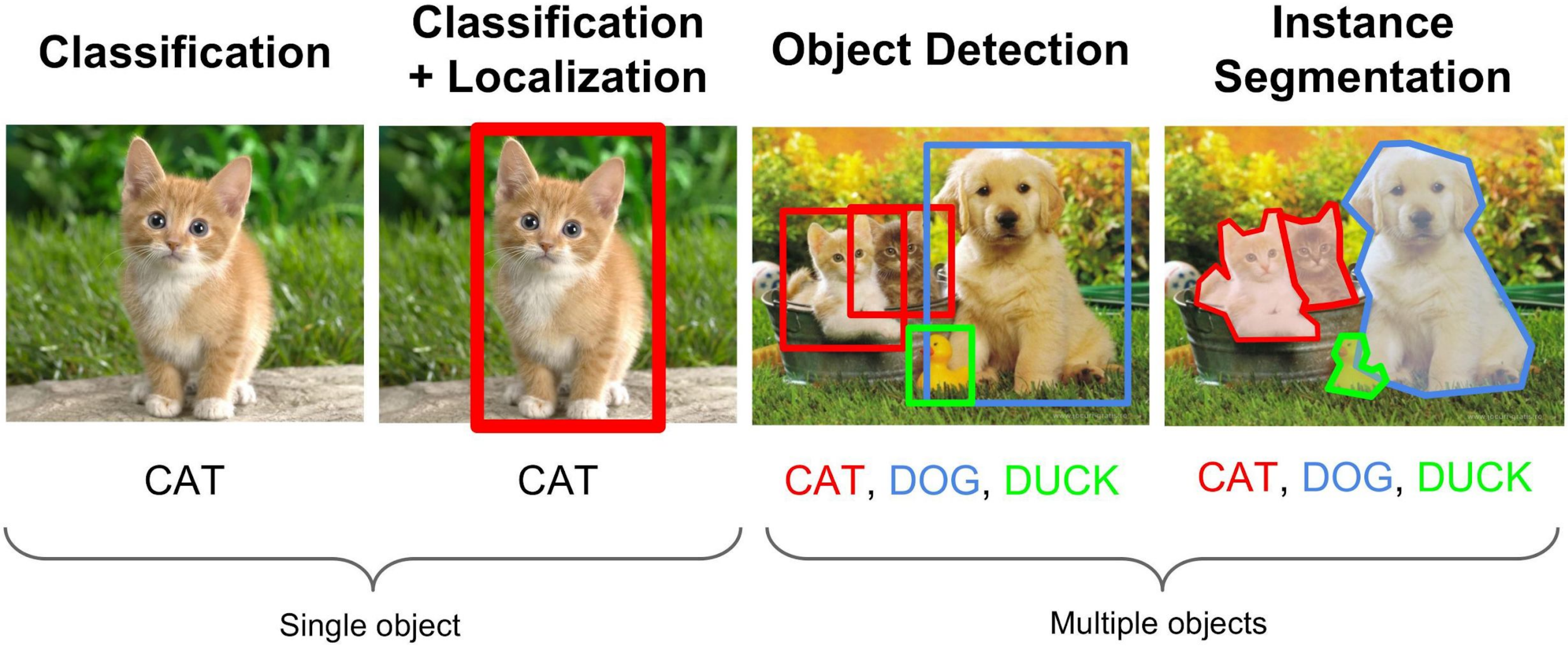
Object detection with R-CNN family

Computer vision

Comparison between image classification, object detection and instance segmentation.



Classification

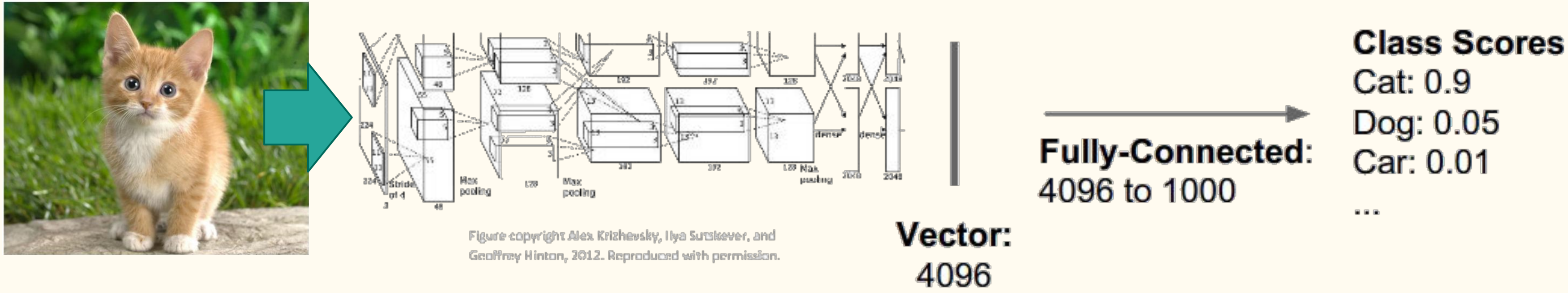


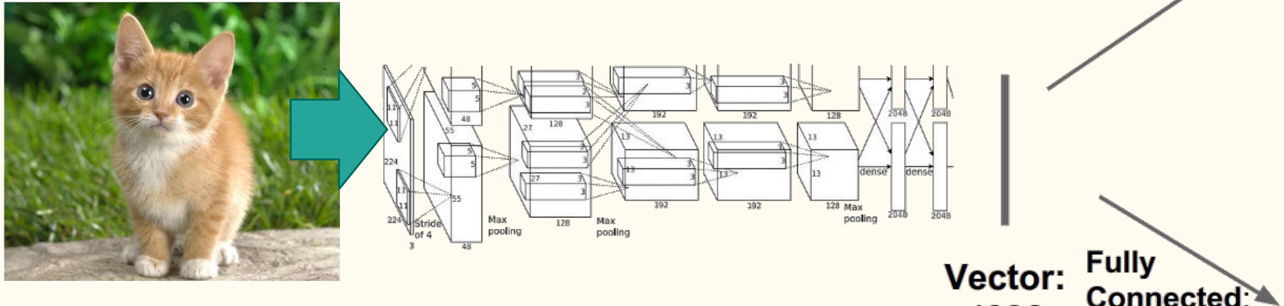
Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Vector: 4096

Fully-Connected: 4096 to 1000

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

Classification + Localization



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

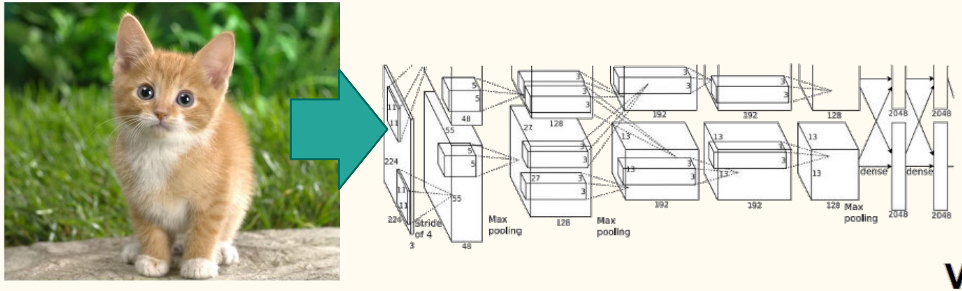
Fully Connected:
4096 to 1000

Vector:
4096

Fully Connected:
4096 to 4

Box Coordinates
(x, y, w, h)

Classification + Localization



Vector: 4096

Fully Connected: 4096 to 1000

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

Softmax Loss

Correct label:
Cat

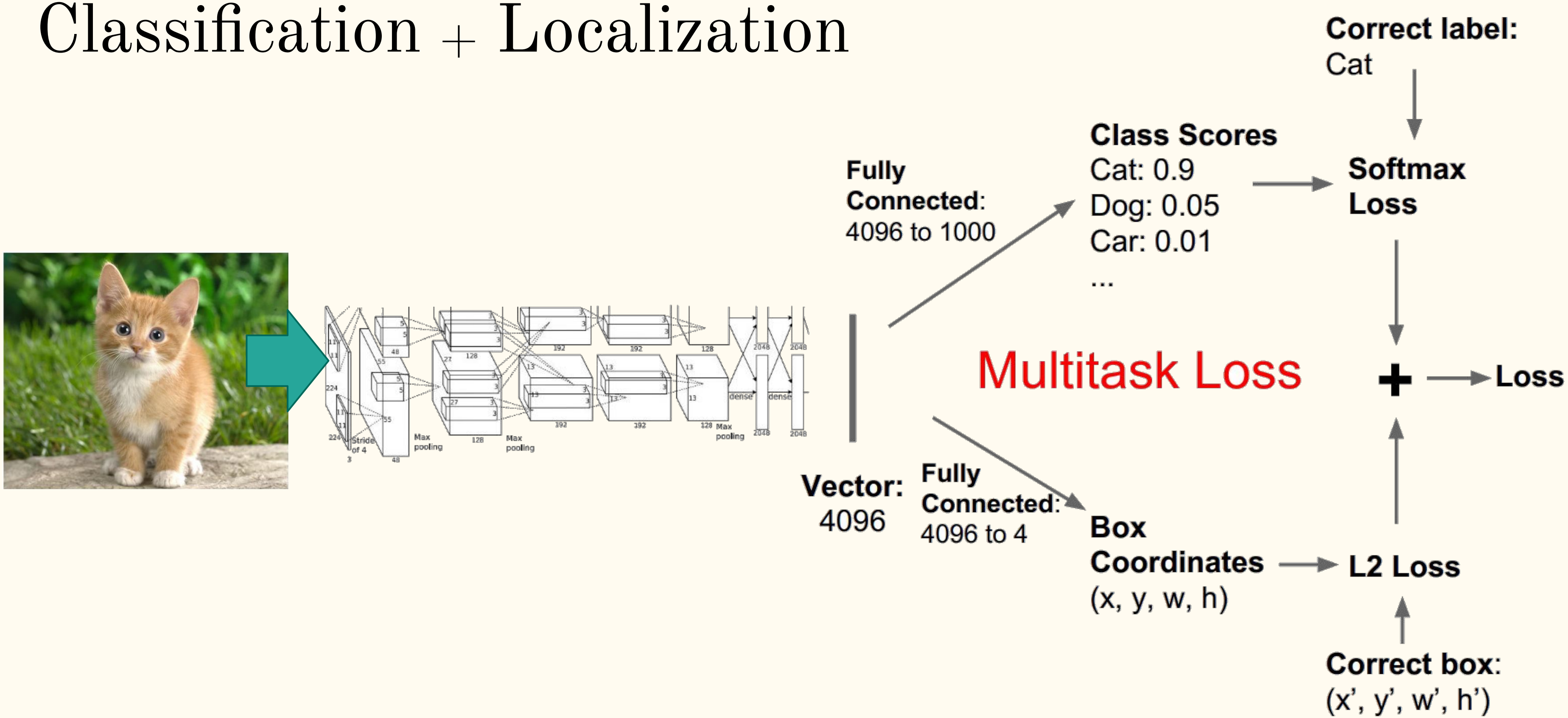
Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

L2 Loss

Correct box:
(x’, y’, w’, h’)

Classification + Localization



Correct label:
Cat

Fully Connected:
4096 to 1000

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

Softmax Loss

Multitask Loss

+ → Loss

Vector:
4096

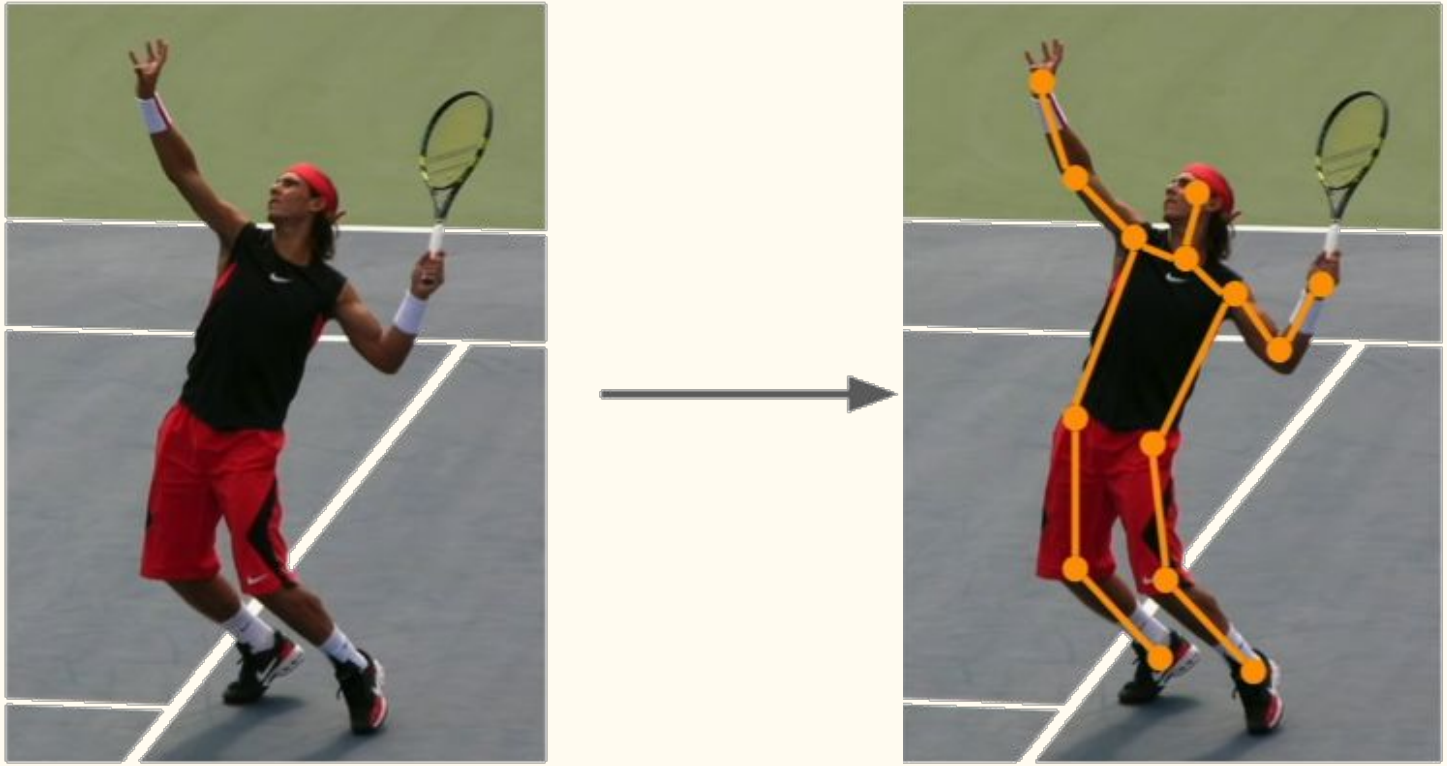
Fully Connected:
4096 to 4

Box Coordinates
(x, y, w, h)

L2 Loss

Correct box:
(x’, y’, w’, h’)

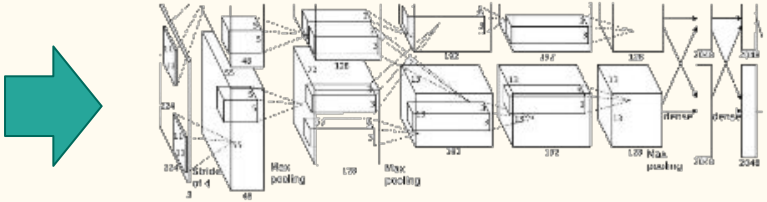
Aside: Human pose estimation



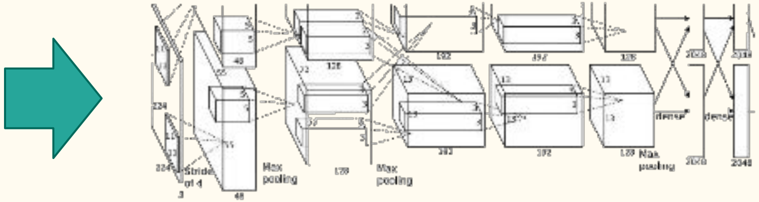
Represent pose as a set of 14 joint positions:

Left/right foot
Left/right knee
Left/right hip
Left/right shoulder
Left/right elbow
Left/right hand
Neck
Head top

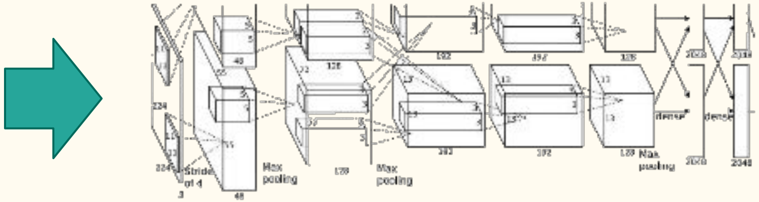
Object detection as regression?



CAT: (x, y, w, h)



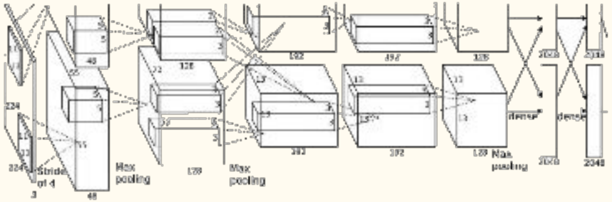
DOG: (x, y, w, h)
DUCK: (x, y, w, h)



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

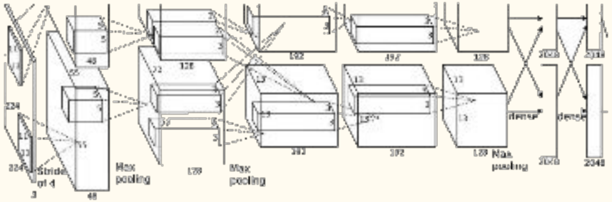
Object detection as regression?

Each image needs a different number of outputs!



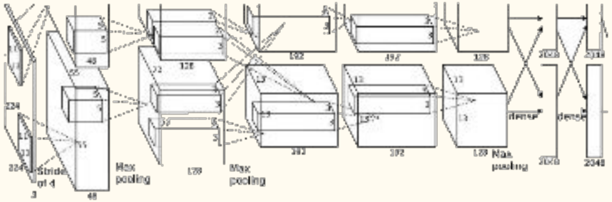
CAT: (x, y, w, h)

4 numbers



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

8 numbers



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

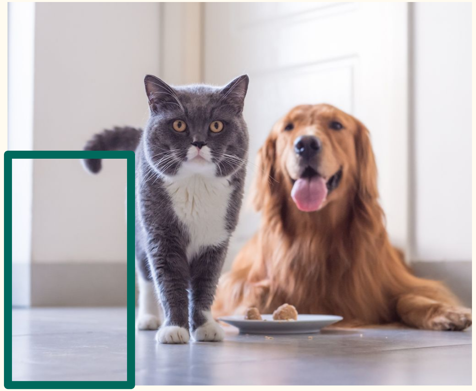
16+ numbers

Object detection as classification

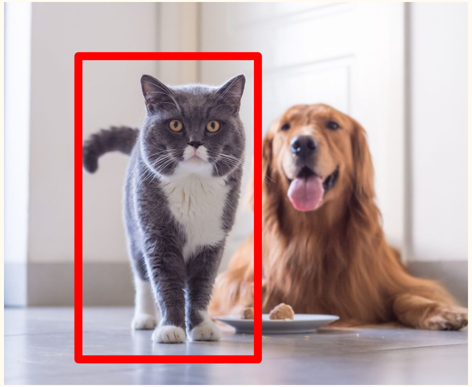
Sliding window

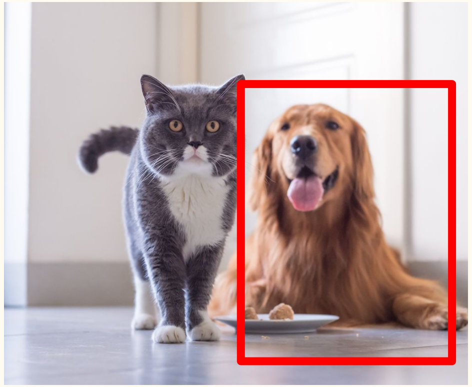
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background.

Cat: NO
Dog: NO
Background: YES



Cat: YES
Dog: NO
Background: NO





Cat: NO
Dog: YES
Background: NO

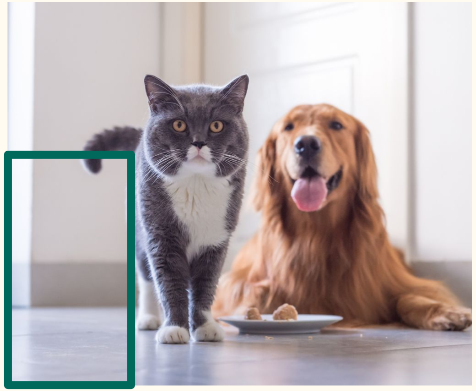
Object detection as classification

Sliding window

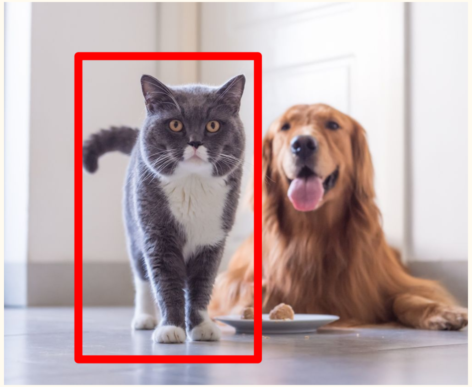
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background.

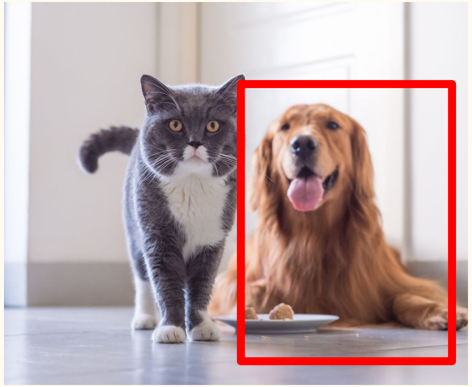
But is it good?

Cat: NO
Dog: NO
Background: YES



Cat: YES
Dog: NO
Background: NO





Cat: NO
Dog: YES
Background: NO

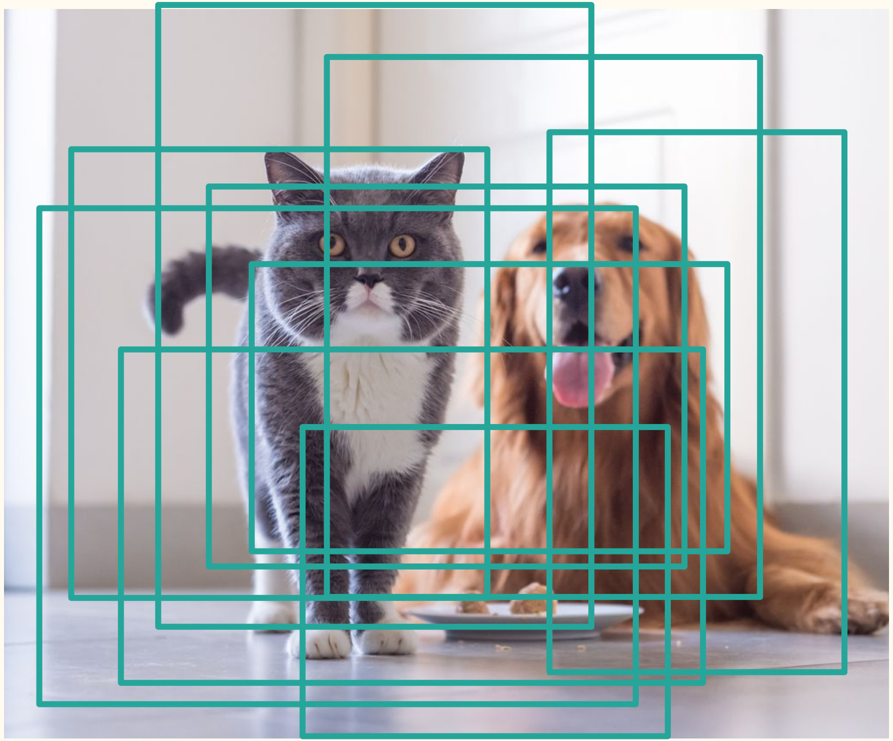
Object detection as classification

Sliding window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background.

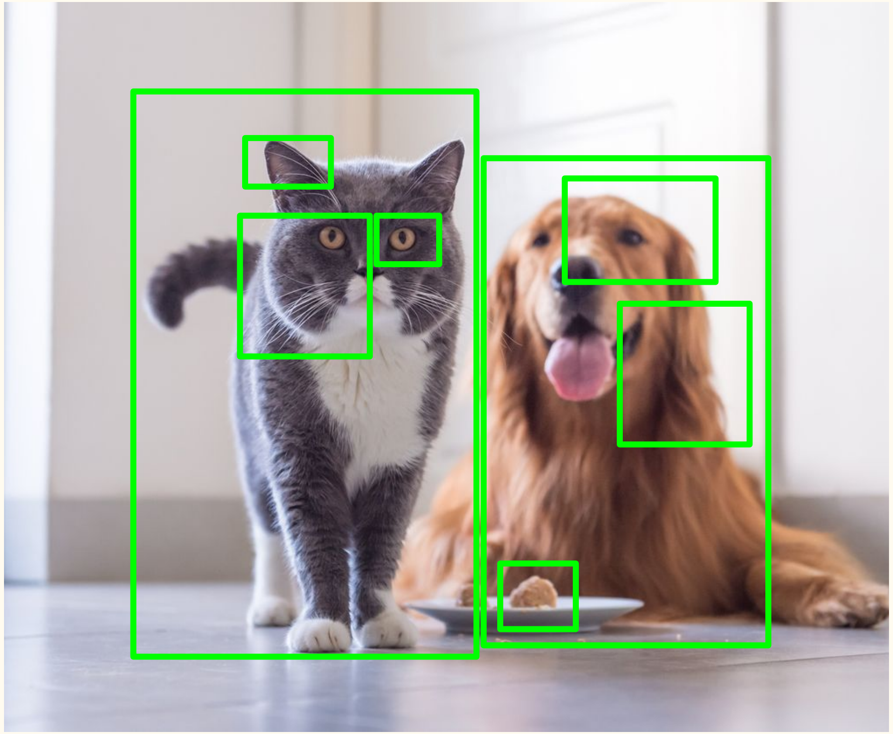
But is it good?

NO! 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
- Relatively fast to run; e.g. Selective Search gives 2000 region proposals in a few seconds on CPU

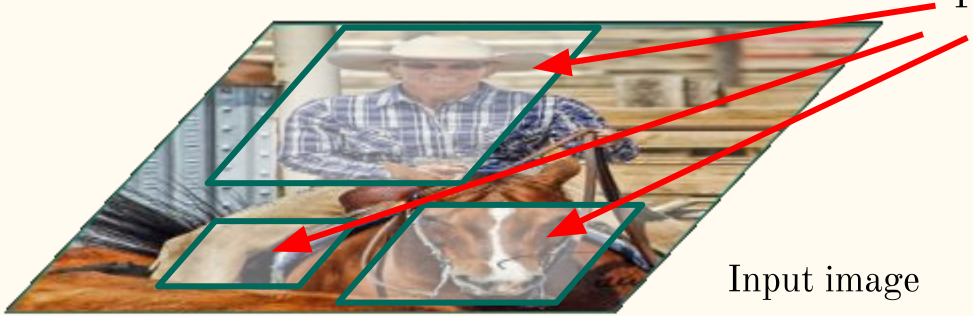


R-CNN

R-CNN

Input image

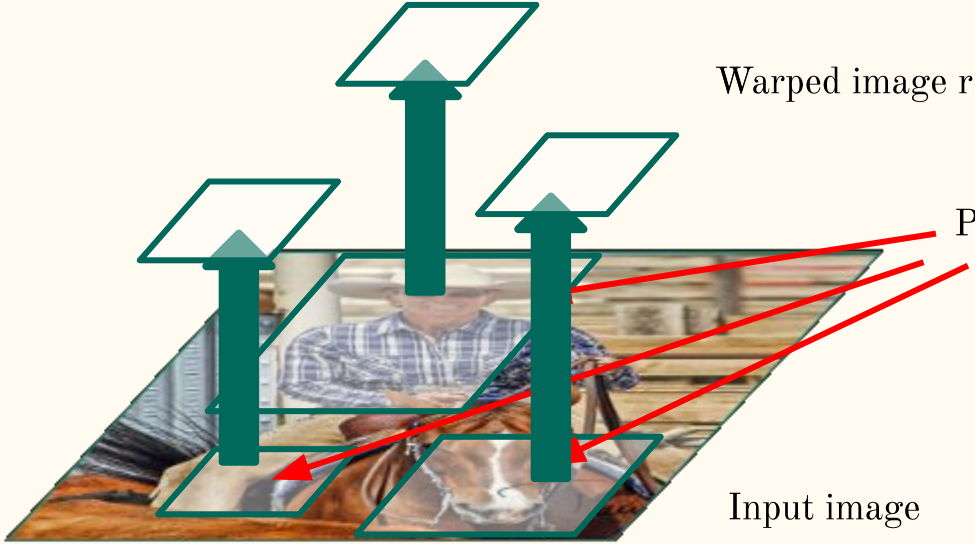
R-CNN



Proposed regions $\sim$ 2k

Input image

R-CNN

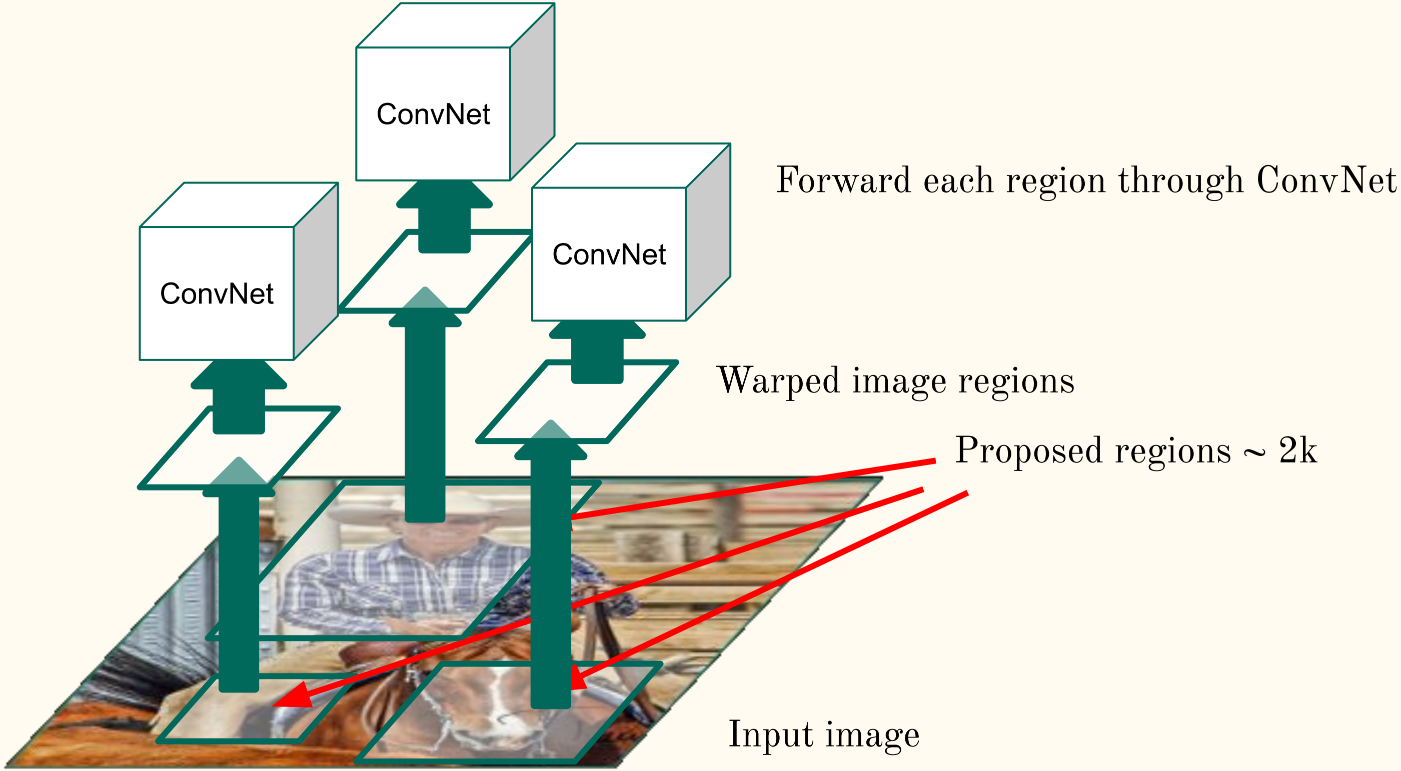


Warped image regions

Proposed regions $\sim$ 2k

Input image

R-CNN



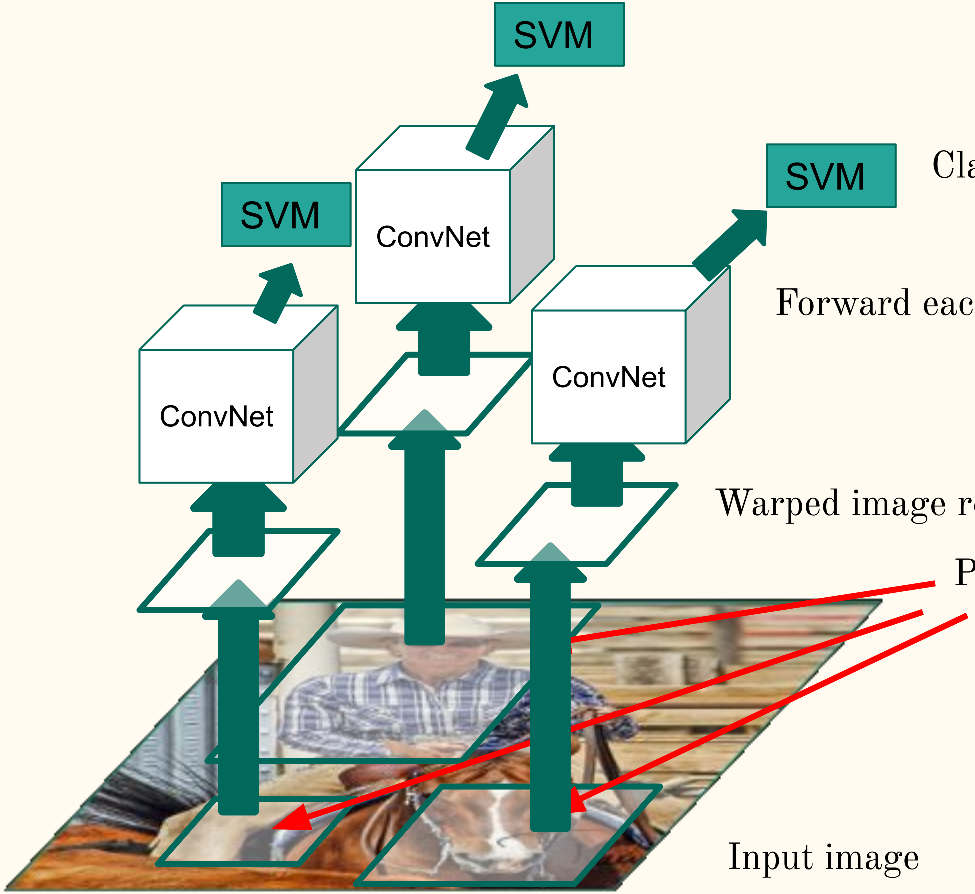
Forward each region through ConvNet

Warped image regions

Proposed regions ~ 2k

Input image

R-CNN



Classify each region with SVMs

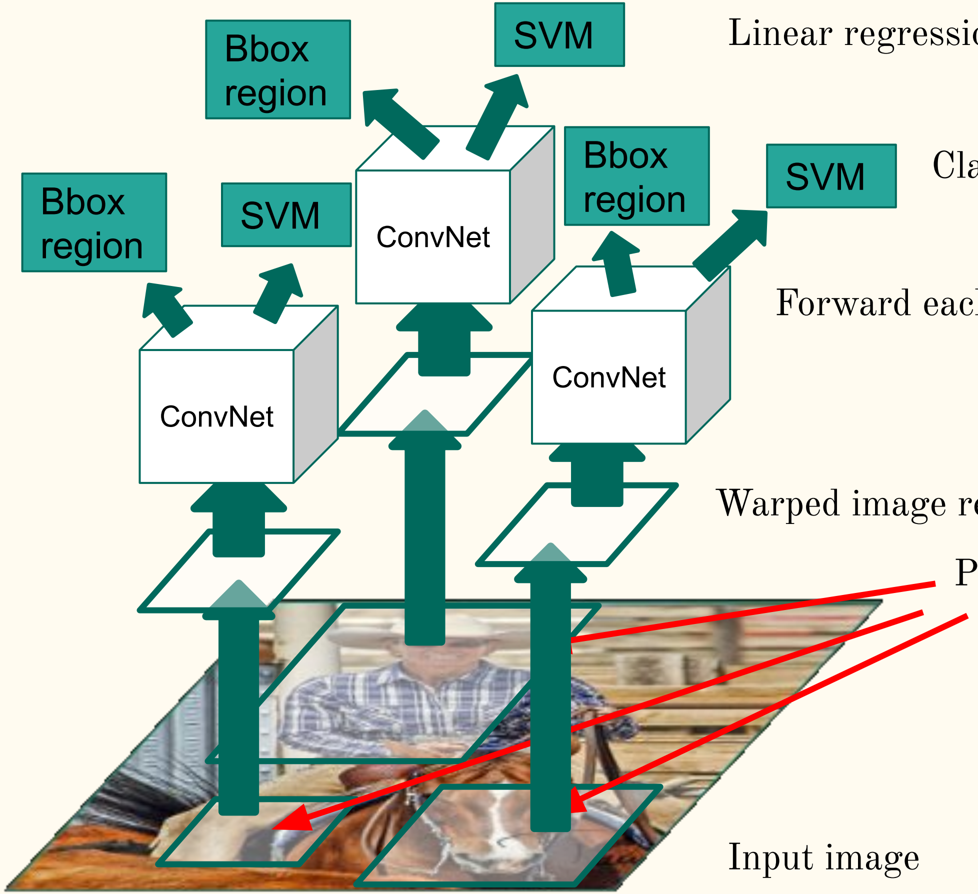
Forward each region through ConvNet

Warped image regions

Proposed regions ~ 2k

Input image

R-CNN



Linear regression for bounding box offsets

Classify each region with SVMs

Forward each region through ConvNet

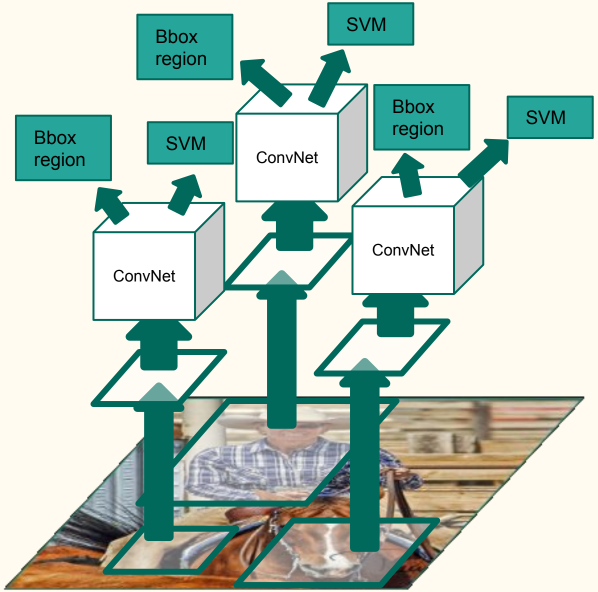
Warped image regions

Proposed regions ~ 2k

Input image

R-CNN problems

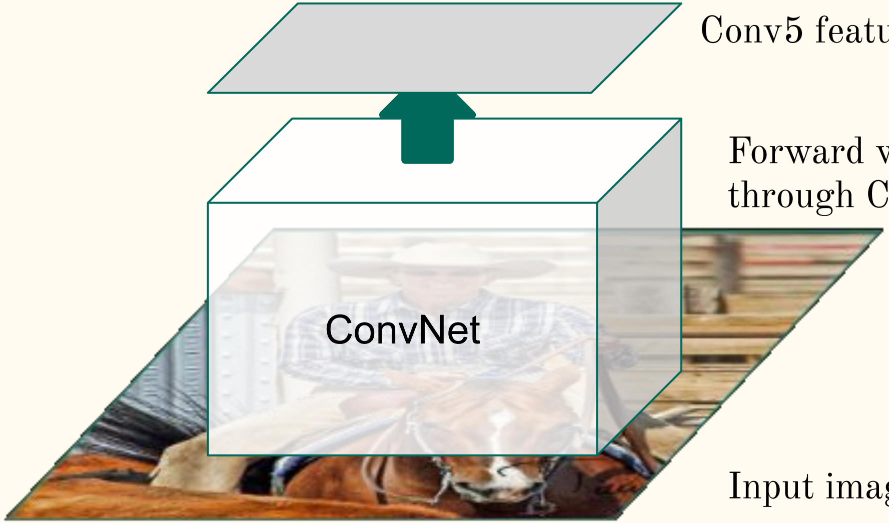
- Ad hoc training objectives
  - Fine-tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressions (least squares)
- Training is slow (84h), takes a lot of disk space
- Inference (detection) is slow
  - 47s / image with VGG16 [Simonyan & Zisserman. ICLR15]



Fast R-CNN

Input image

Fast R-CNN

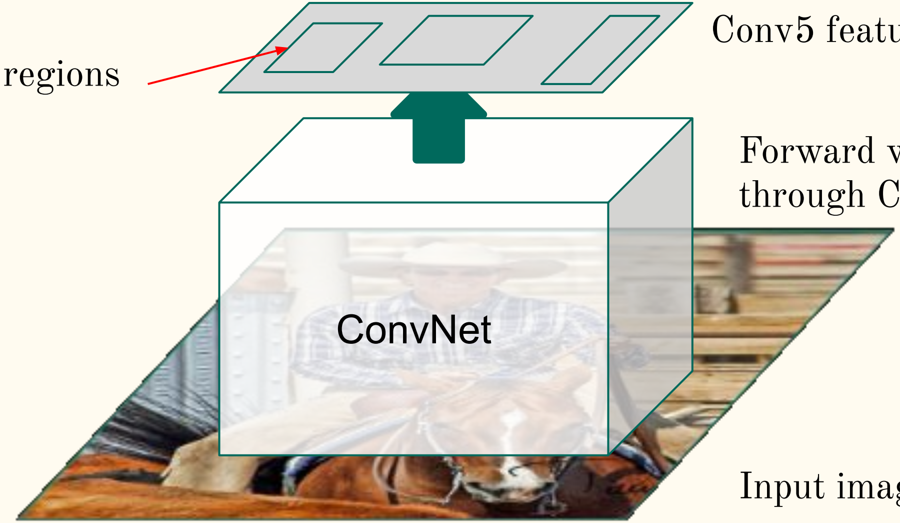


Conv5 feature maps of image

Forward whole image
through ConvNet

Input image

Fast R-CNN



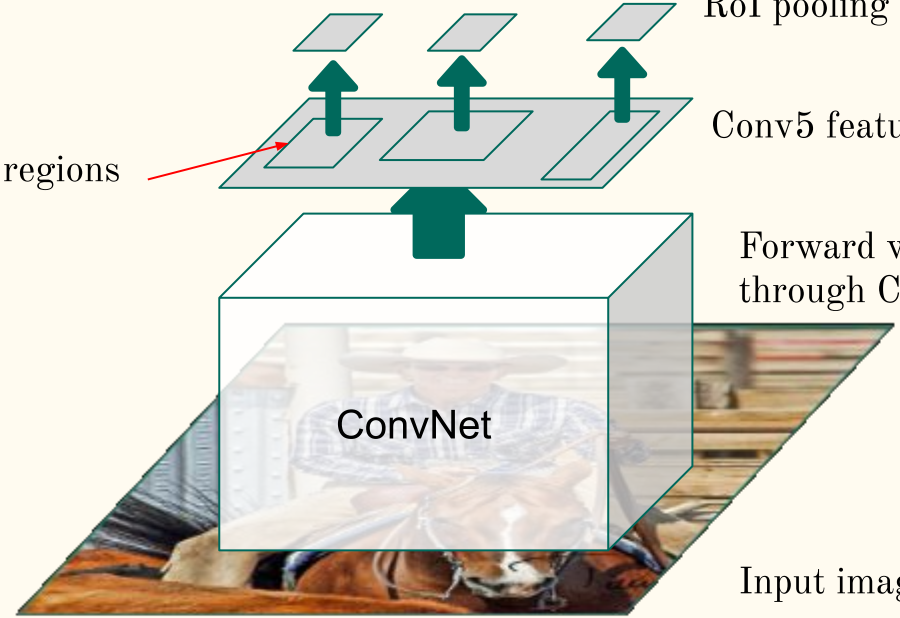
Proposed regions

Conv5 feature maps of image

Forward whole image through ConvNet

Input image

Fast R-CNN



RoI pooling layer

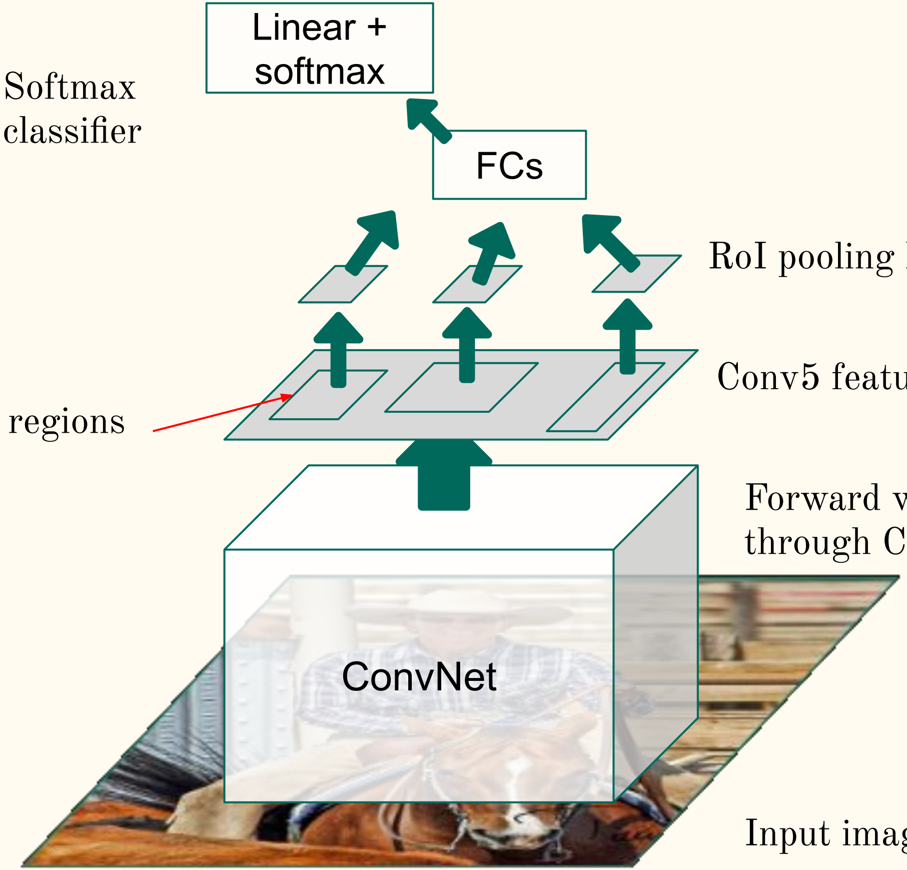
Conv5 feature maps of image

Proposed regions

Forward whole image
through ConvNet

Input image

Fast R-CNN



Linear + softmax

Softmax classifier

FCs

RoI pooling layer

Conv5 feature maps of image

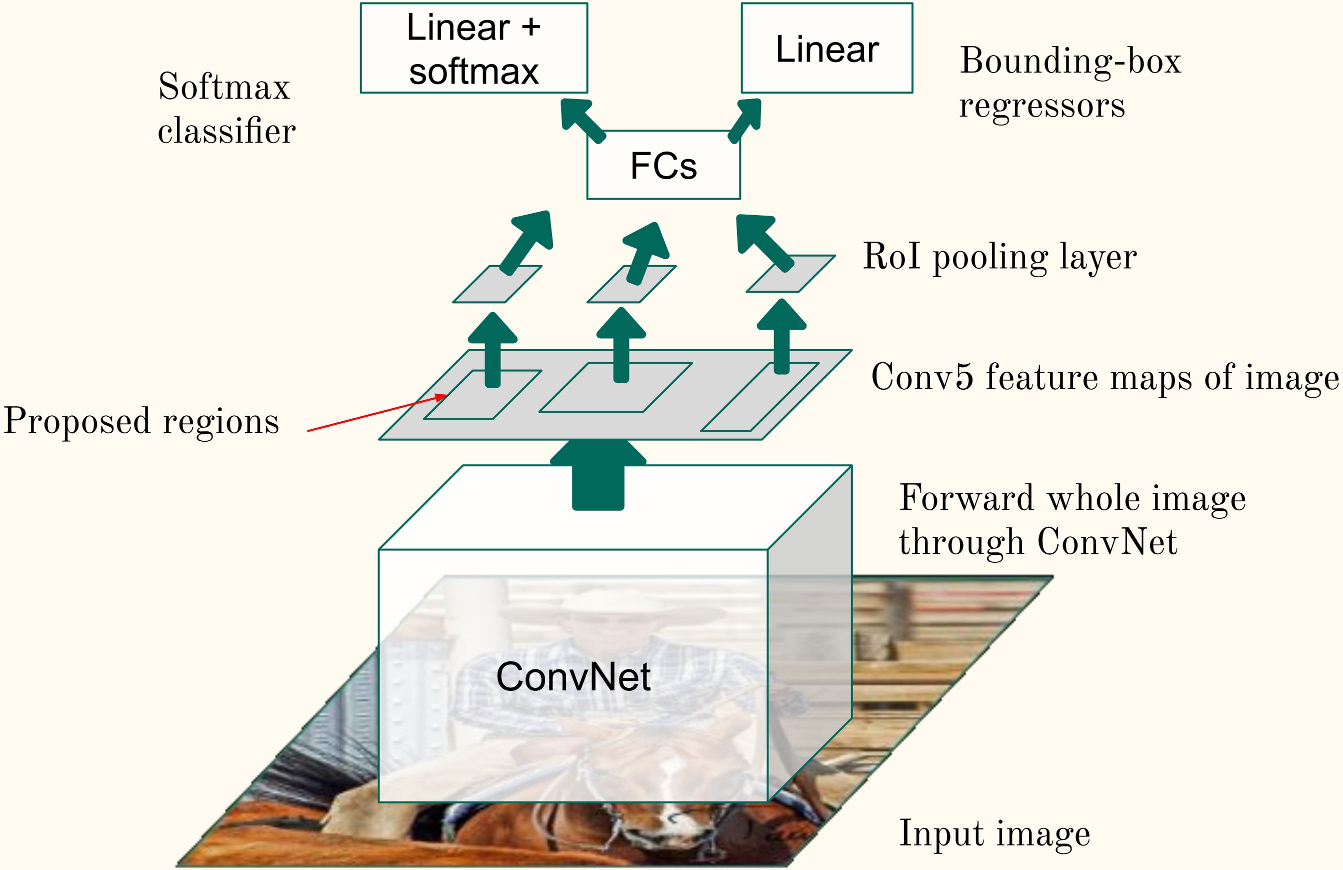
Proposed regions

ConvNet

Forward whole image through ConvNet

Input image

Fast R-CNN



Linear + softmax

Linear

Softmax classifier

Bounding-box regressors

FCs

RoI pooling layer

Conv5 feature maps of image

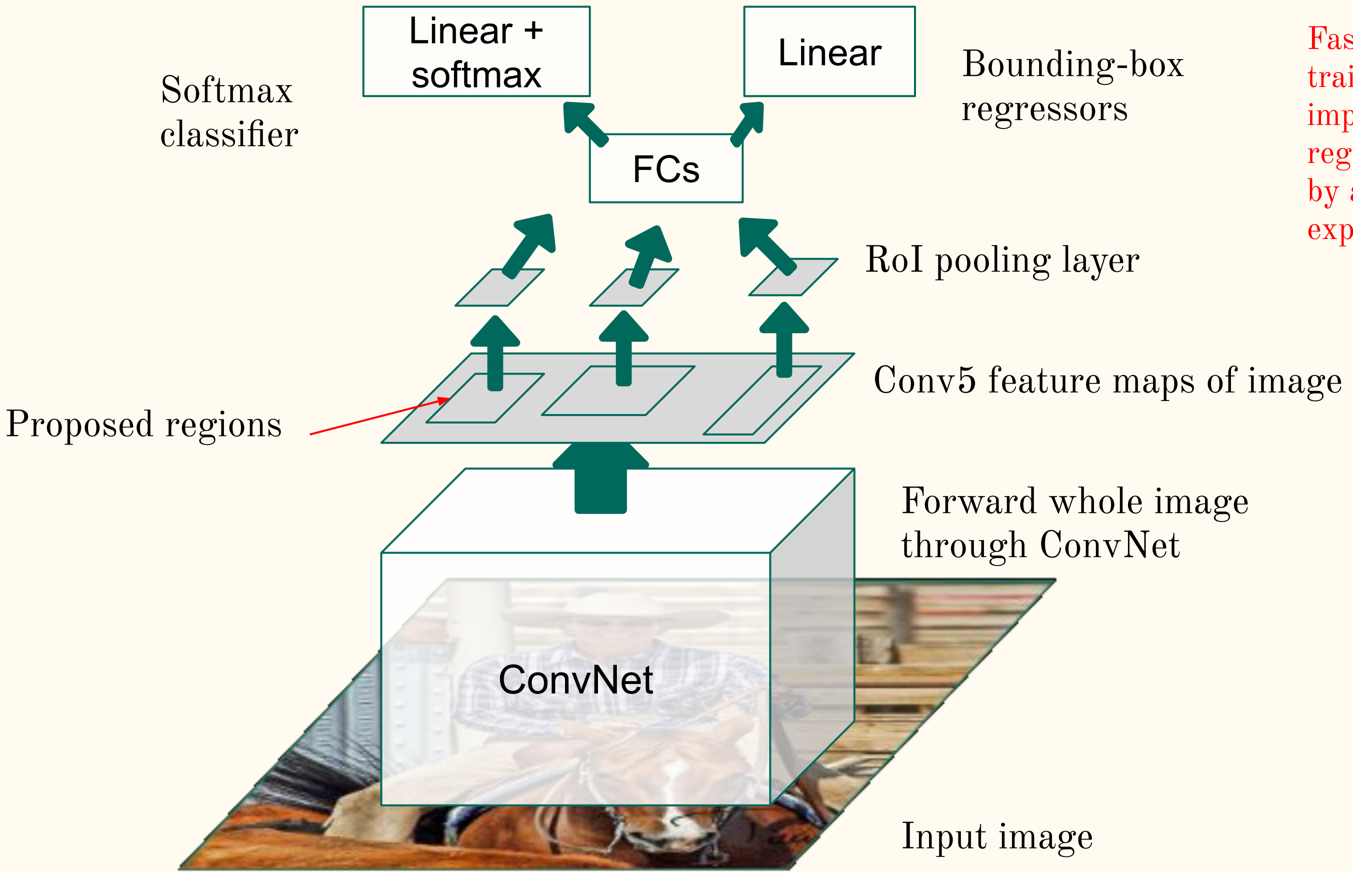
Proposed regions

Forward whole image through ConvNet

ConvNet

Input image

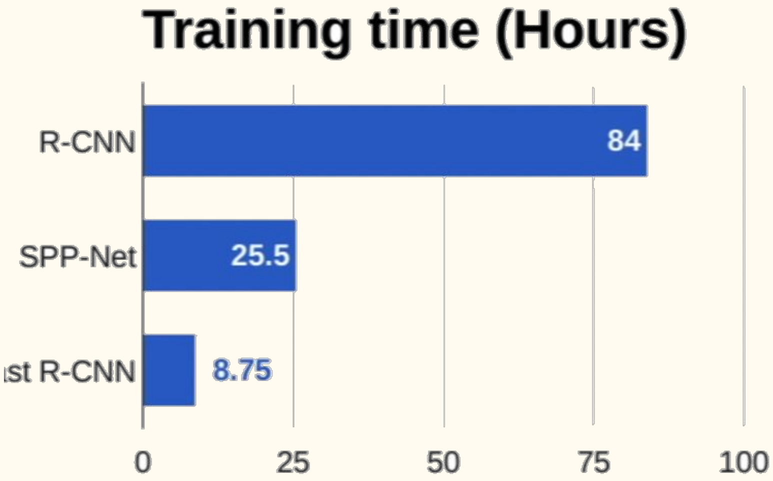
Fast R-CNN

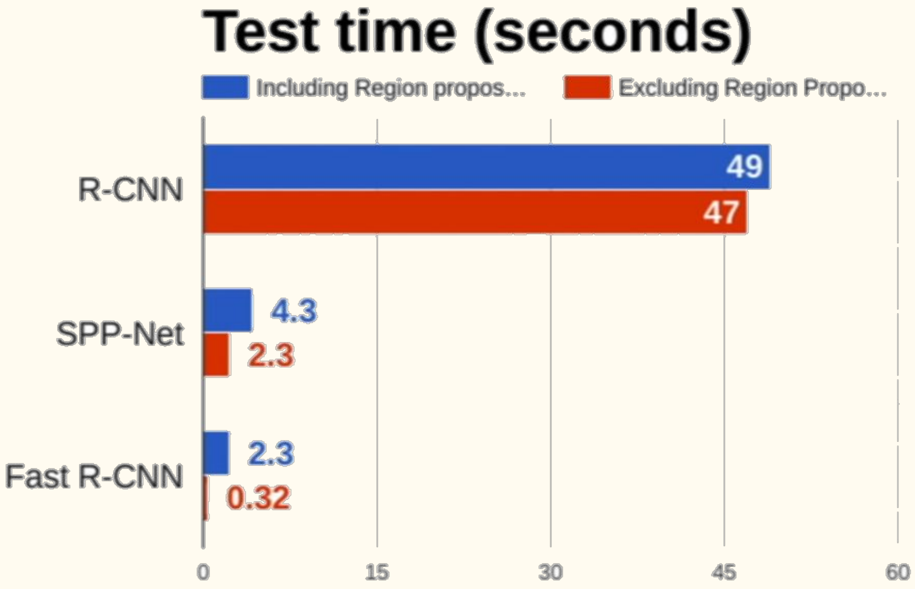


Problem: Speed Bottleneck

Fast R-CNN is much faster in both training and testing time. However, the improvement is not dramatic because the region proposals are generated separately by another model and that is very expensive.

R-CNN VS SPP-Net VS Fast R-CNN





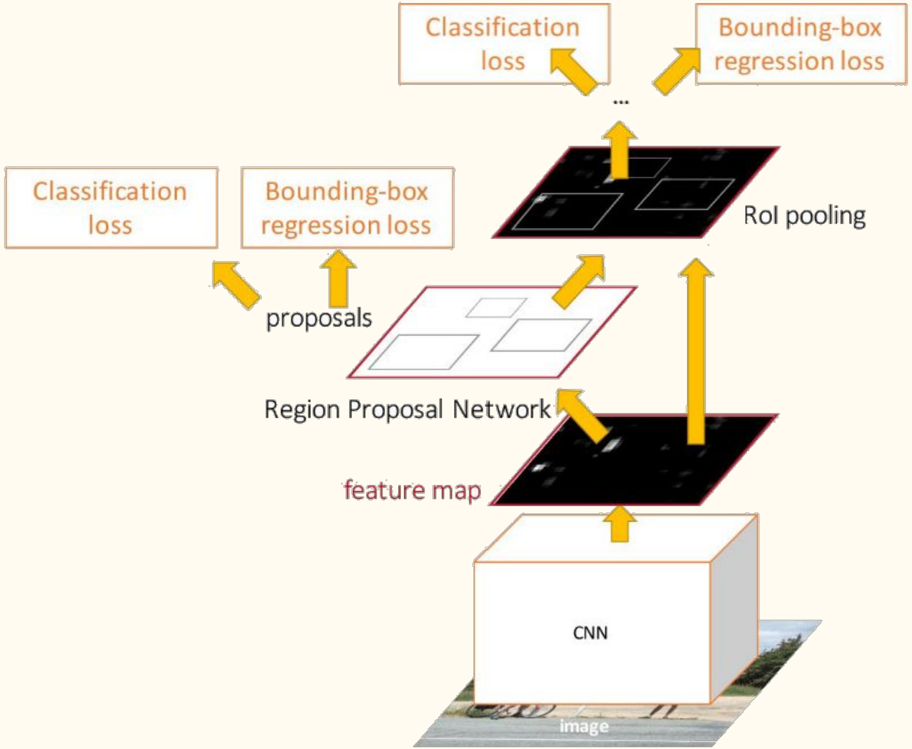
Faster R-CNN

Make CNN do proposals!

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

Jointly train with 4 losses:

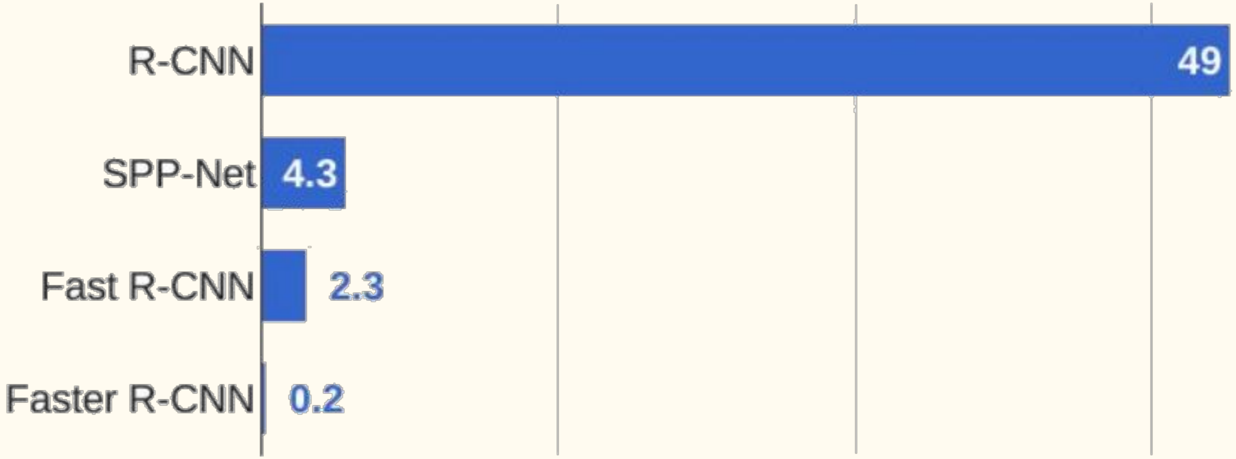
1. RPN classify object / not object
2. RPN regress box coordinates
3. Final classification score (object classes)
4. Final box coordinates



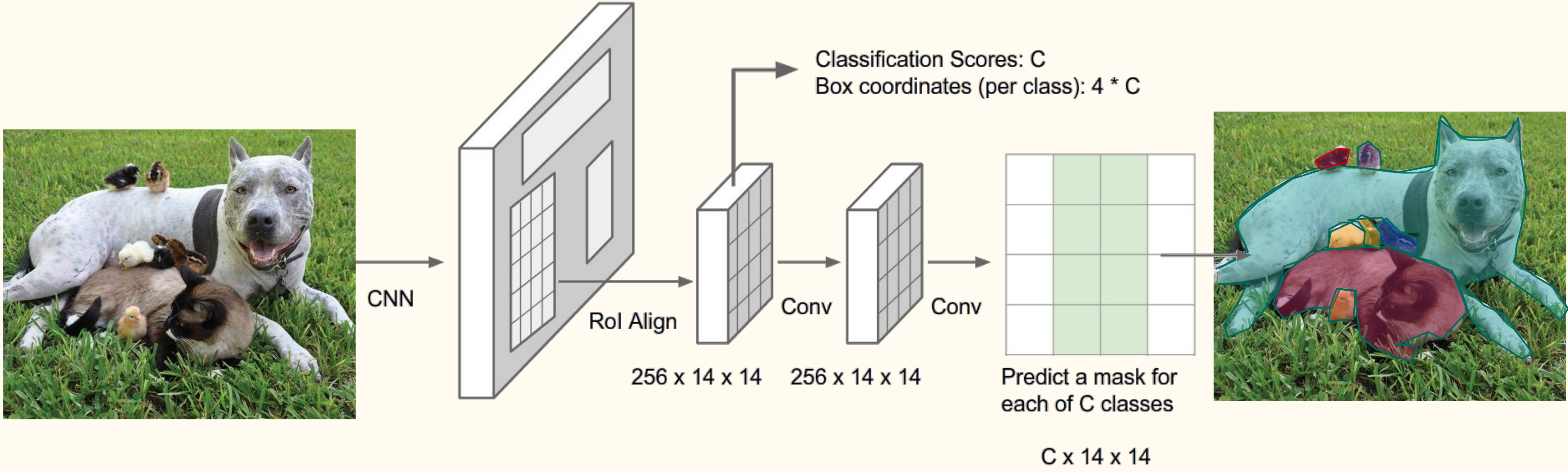
Faster R-CNN

Make CNN do proposals!

~ 7 FPS



Mask R-CNN



CNN

RoI Align

256 x 14 x 14

Conv

256 x 14 x 14

Conv

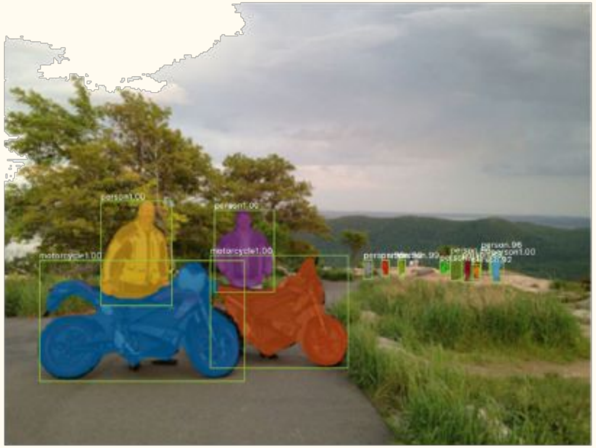
Classification Scores: C
Box coordinates (per class): 4 * C

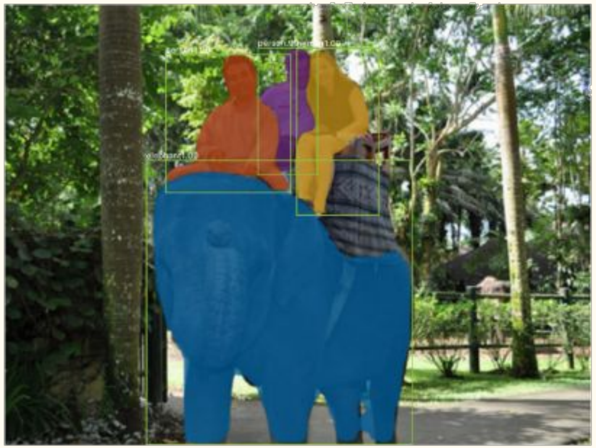
Predict a mask for each of C classes

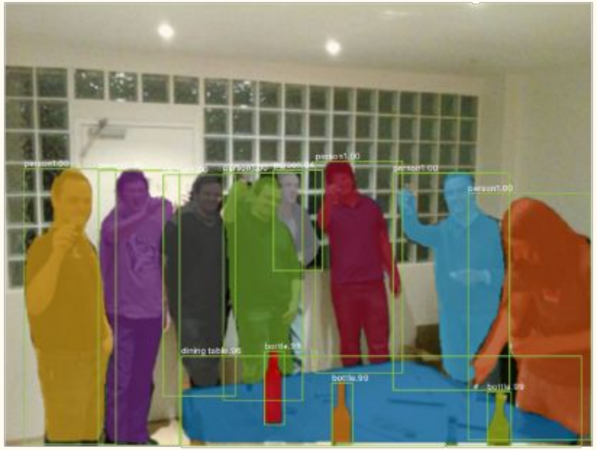
C x 14 x 14

https://arxiv.org/abs/1703.06870

Mask R-CNN: Very good results

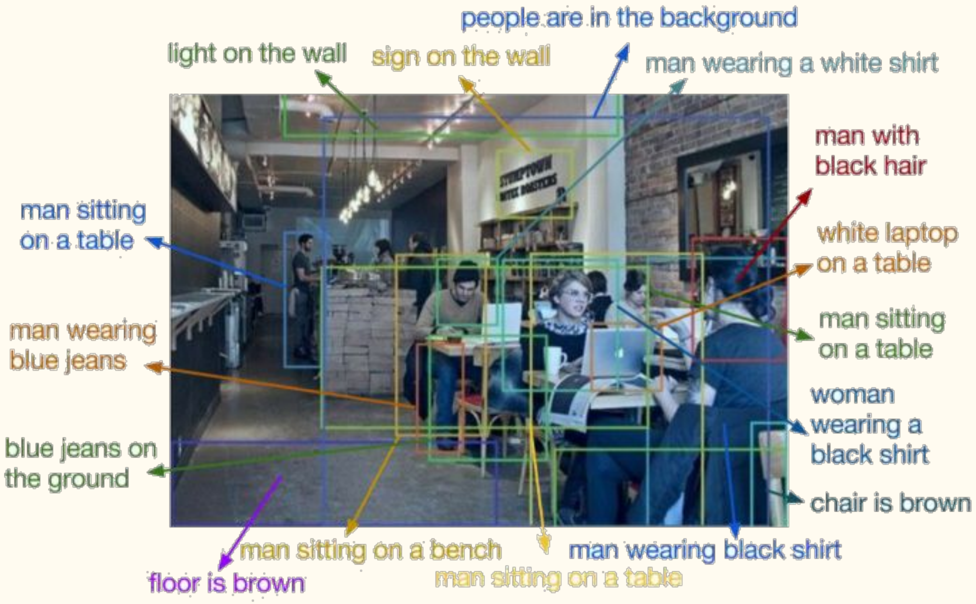


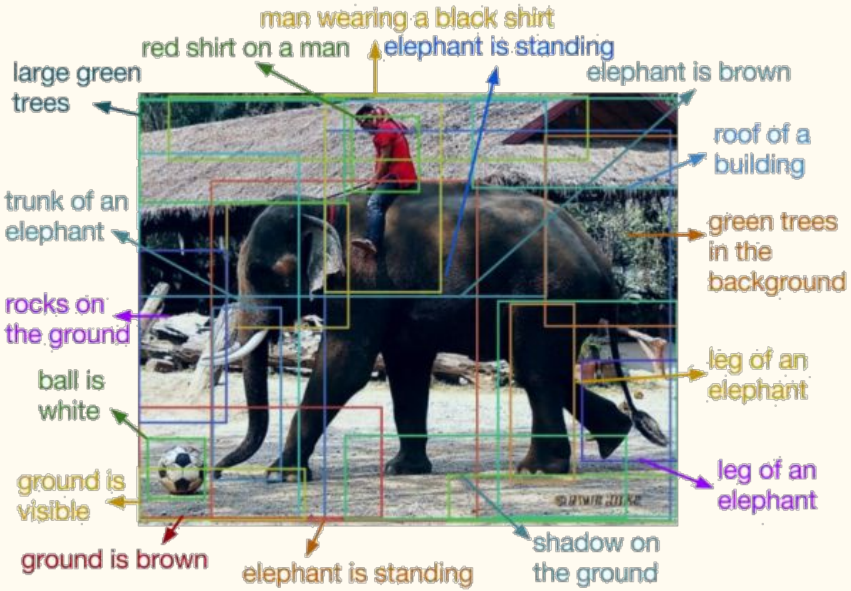




https://arxiv.org/abs/1703.06870

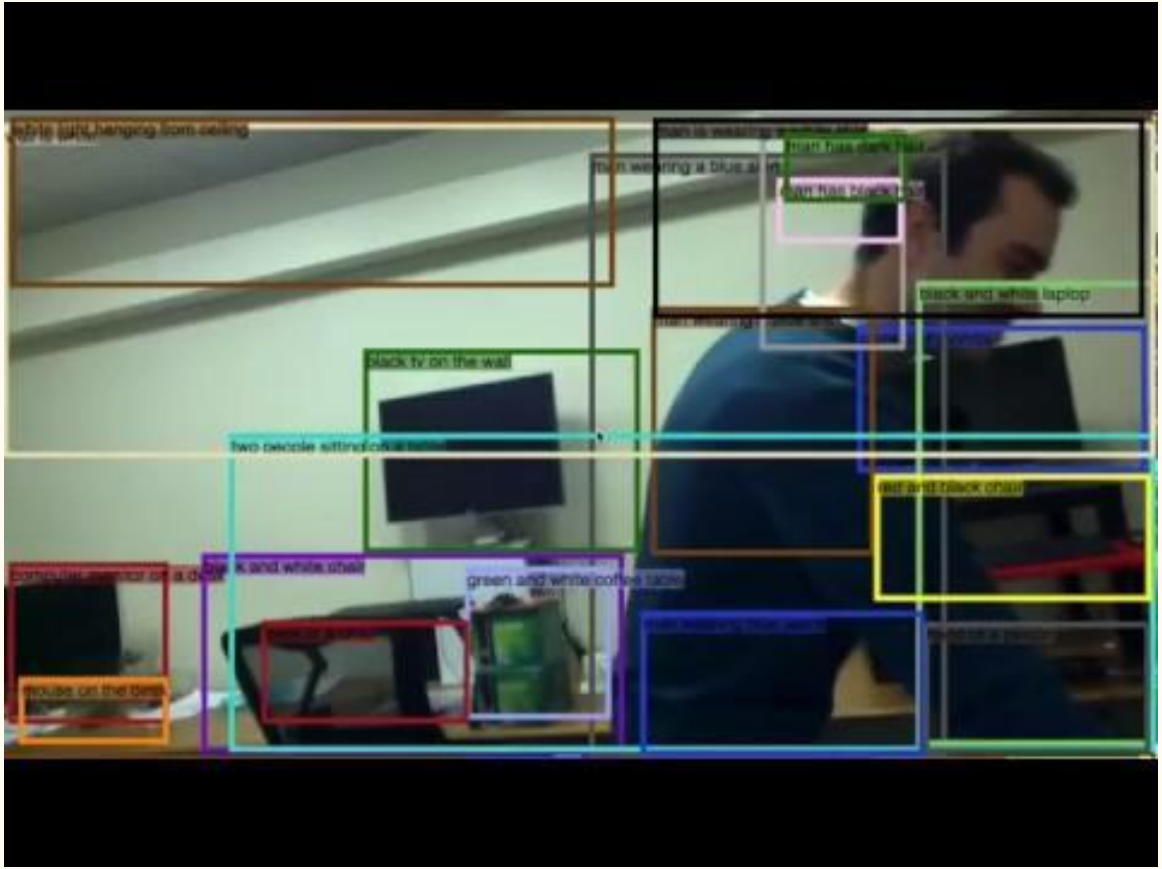
Aside: Object Detection + Captioning= Dense Captioning





https://arxiv.org/abs/1511.07571

Aside: Object Detection + Captioning= Dense Captioning



https://arxiv.org/abs/1511.07571

T.HANKS!