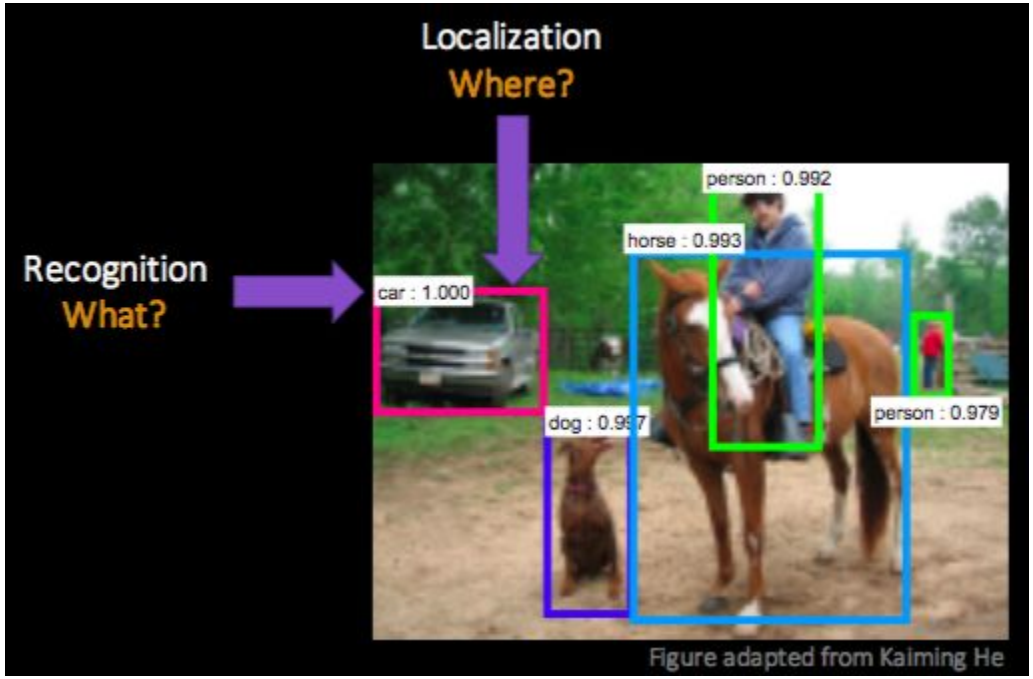
Fast R-CNN

Ross Girshick
Facebook AI Research (FAIR)
Work done at Microsoft Research

Presented by:
Nick Joodi | Doug Sherman

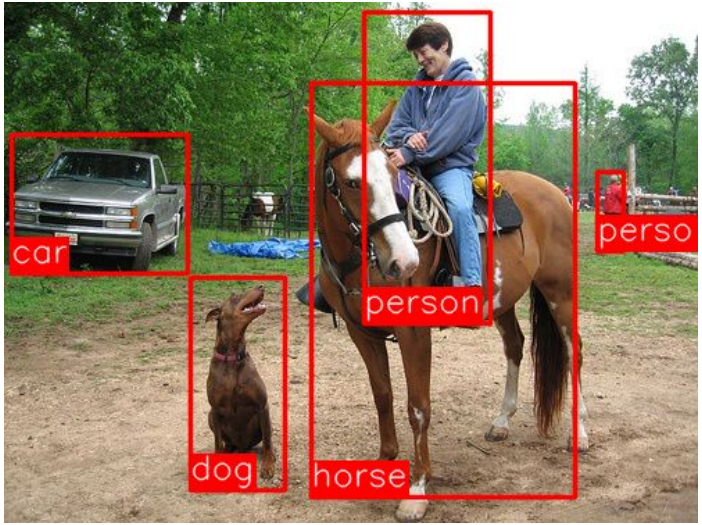
Fast Region-based ConvNets (R-CNNs)



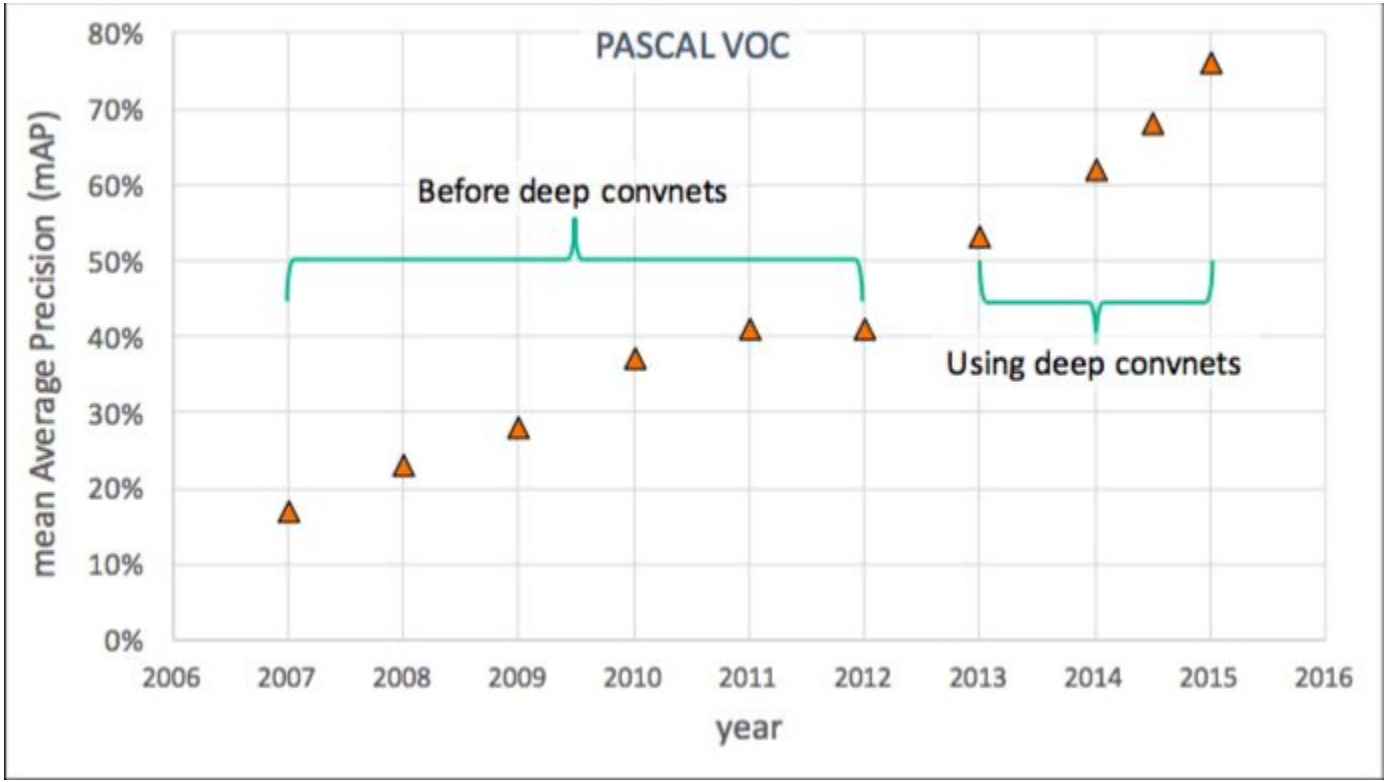
Sorry about the black BG, Girshick's slides were all black.

The Pascal Visual Object Classes Challenge

- Overview
  - **Classification, Detection**, Segmentation
  - For each image:
    - Does it contain the class? → classification
    - Where is it? → detection via bounding box
- Evaluation
  - Mean Average Precision (mAP)
    - Participants submitted results in the form of confidence
    - Produce Precision Recall curves
      - Average precision for each class -
      - Take mean to get mAP

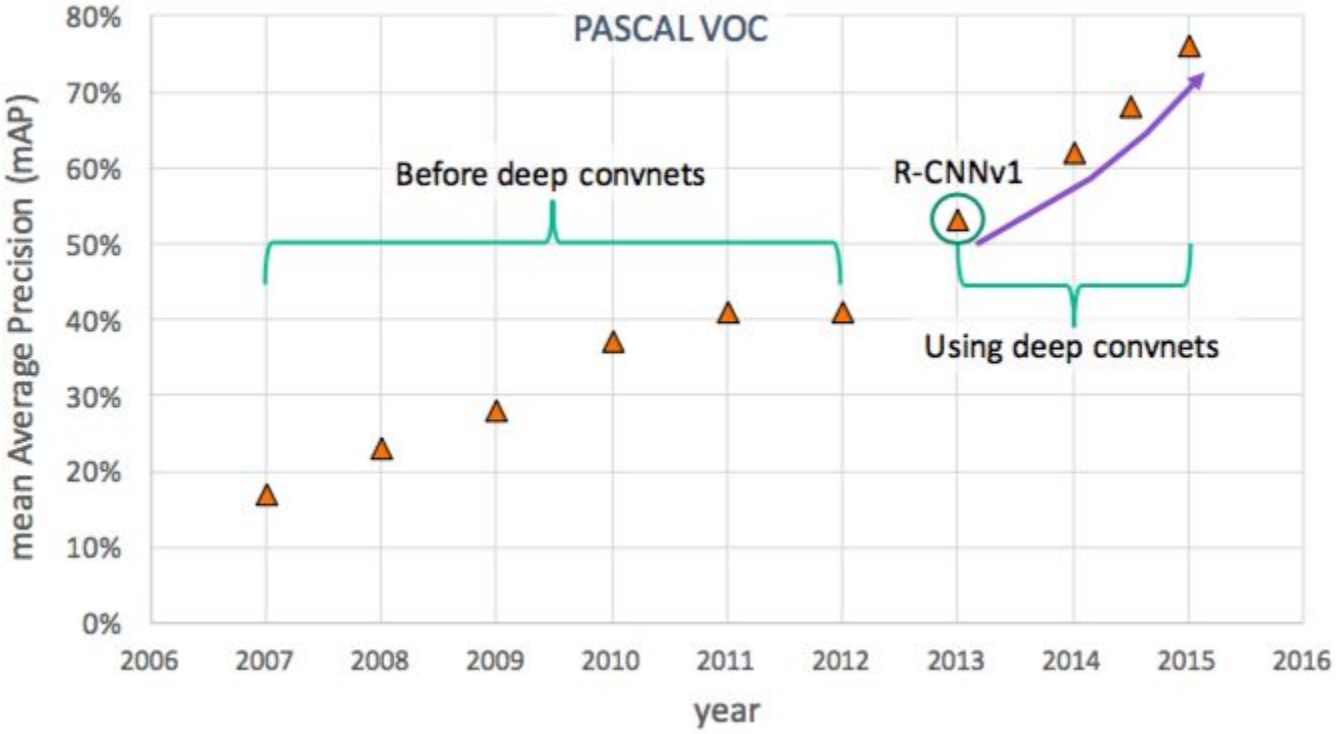


Object detection renaissance (2013-Present)



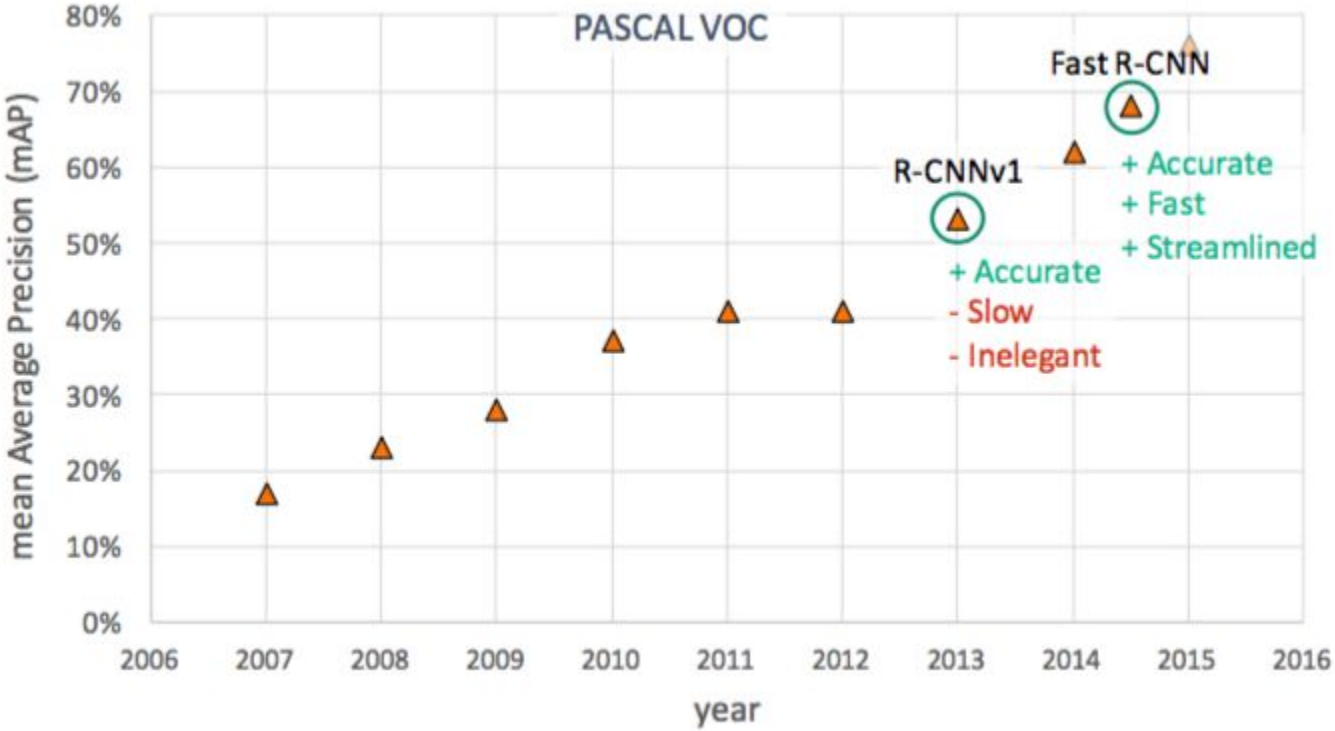
Adapted from Fast R-CNN [R. Girshick (2015)]

Object detection renaissance (2013-Present)



Adapted from Fast R-CNN [R. Girshick (2015)]

Object detection renaissance (2013-Present)



Adapted from Fast R-CNN [R. Girshick (2015)]

Agenda

1. Pre-existing Models
   a. **“Slow” R-CNN**
   b. SPP-net
2. Ways to improve
   a. SGD Mini-Batch
   b. New Loss Function
3. Fast R-CNN
   a. Architecture
   b. Results & Future Work

Region-based convnets (R-CNNs)

- R-CNN (aka “slow R-CNN”) [Girshick et al. CVPR14]
- SPP-net [He et al. ECCV14]

Slow R-CNN

Girshick et al. CVPR14.

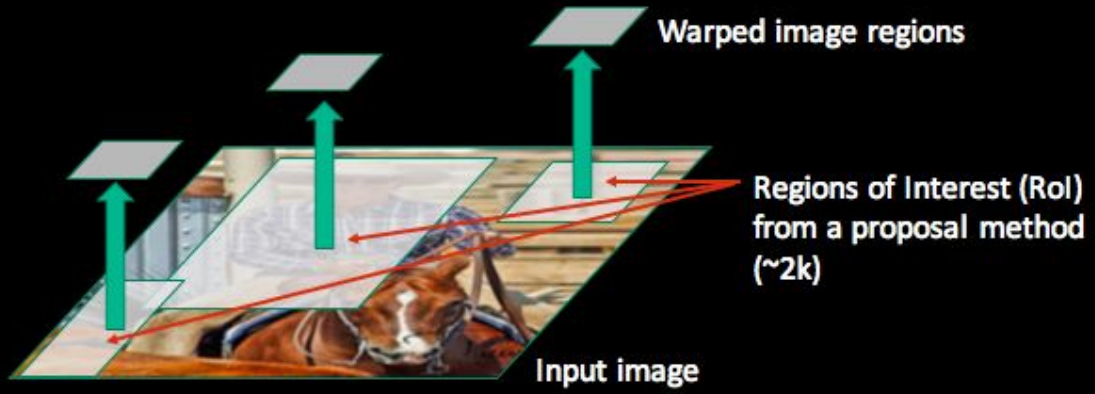
Adapted from Fast R-CNN [R. Girshick (2015)]

Slow R-CNN

Girshick et al. CVPR14.

Adapted from Fast R-CNN [R. Girshick (2015)]

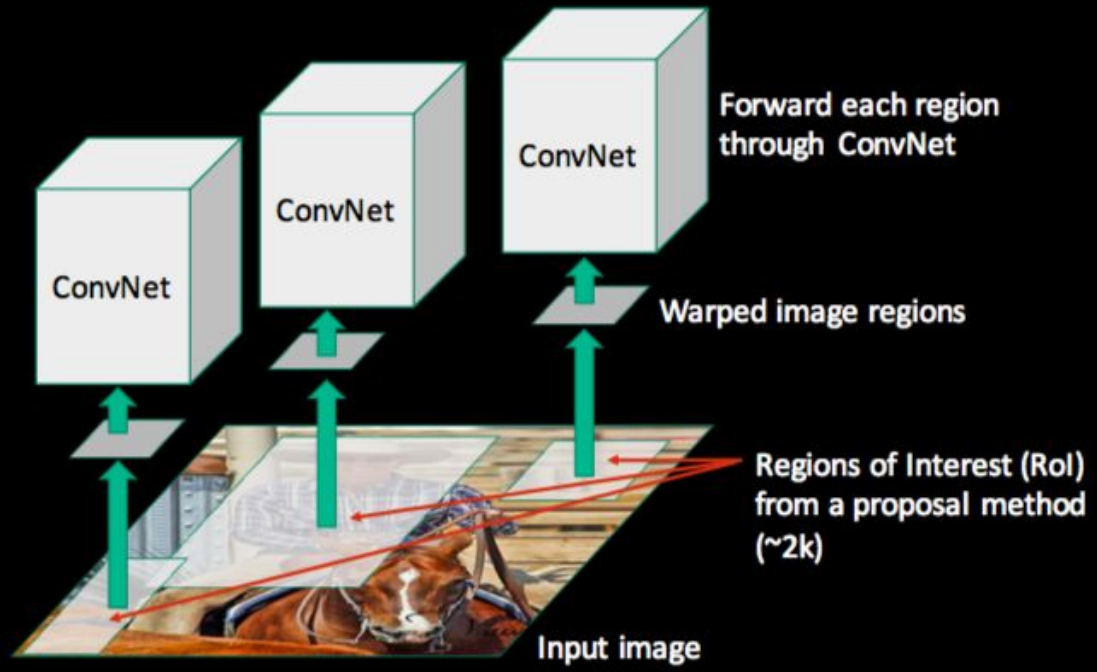
Slow R-CNN



Girshick et al. CVPR14.

Adapted from Fast R-CNN [R. Girshick (2015)]

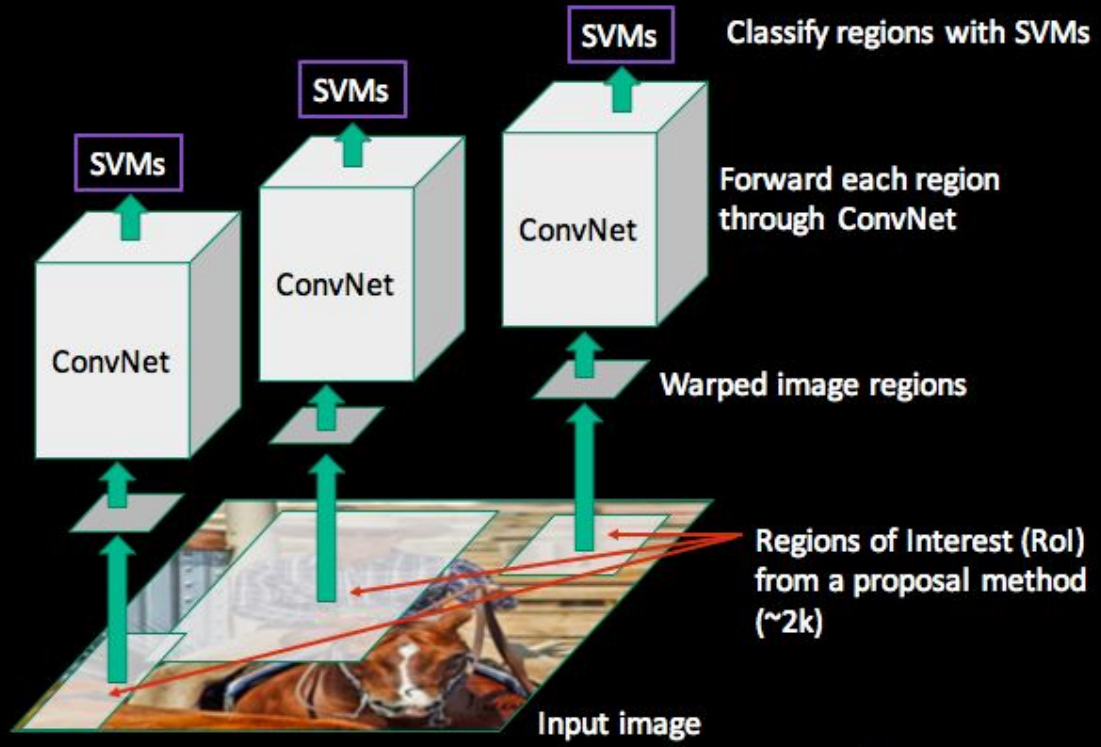
Slow R-CNN



Girshick et al. CVPR14.

Adapted from Fast R-CNN [R. Girshick (2015)]

Slow R-CNN

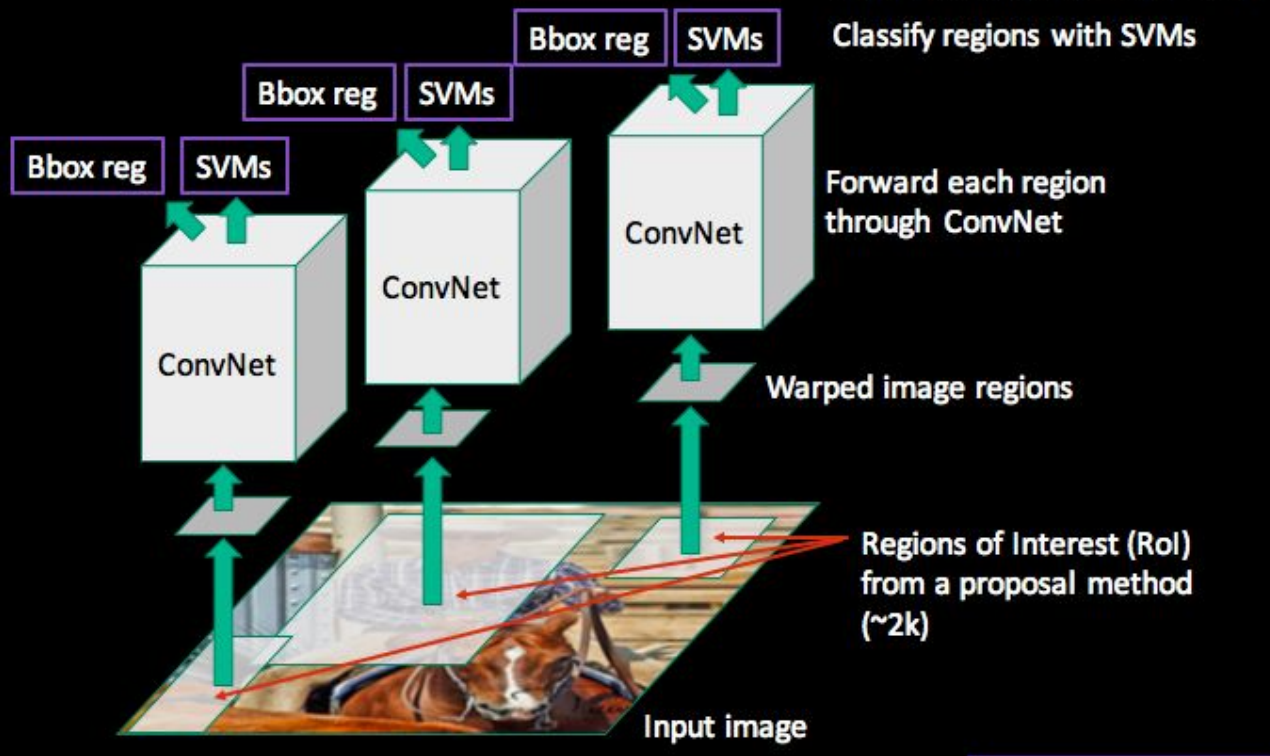


Girshick et al. CVPR14.

Post hoc component

Adapted from Fast R-CNN [R. Girshick (2015)]

Slow R-CNN



Girshick et al. CVPR14.

Post hoc component

Adapted from Fast R-CNN [R. Girshick (2015)]

What's wrong with slow R-CNN?

- Ad hoc training objectives
  - Fine-tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressors (L_2 loss)

What's wrong with slow R-CNN?

- Ad hoc training objectives
  - Fine-tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressors (L_2 loss)
- Training is slow (84h), takes a lot of disk space

What's wrong with slow R-CNN?

- Ad hoc training objectives
  - Fine-tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressors (L_2 loss)
- Training is slow (84h), takes a lot of disk space
- Inference (detection) is slow
  - 47s / image with VGG16 [Simonyan & Zisserman. ICLR15]
  - Fixed by SPP-net [He et al. ECCV14]

Agenda

1. Pre-existing Models
   a. “Slow” R-CNN
   b. SPP-net
2. Ways to improve
   a. SGD Mini-Batch
   b. New Loss Function
3. Fast R-CNN
   a. Architecture
   b. Results & Future Work

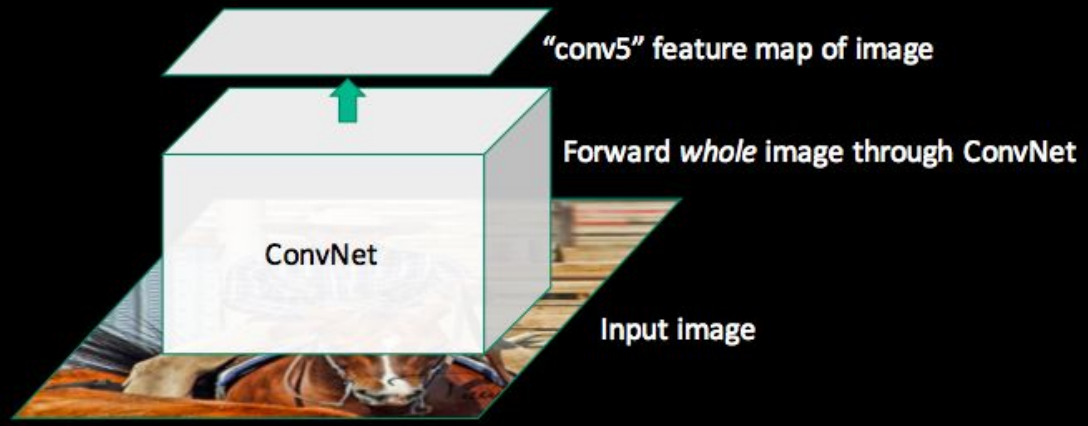
SPP-net

Input image

He et al. ECCV14.

Adapted from Fast R-CNN [R. Girshick (2015)]

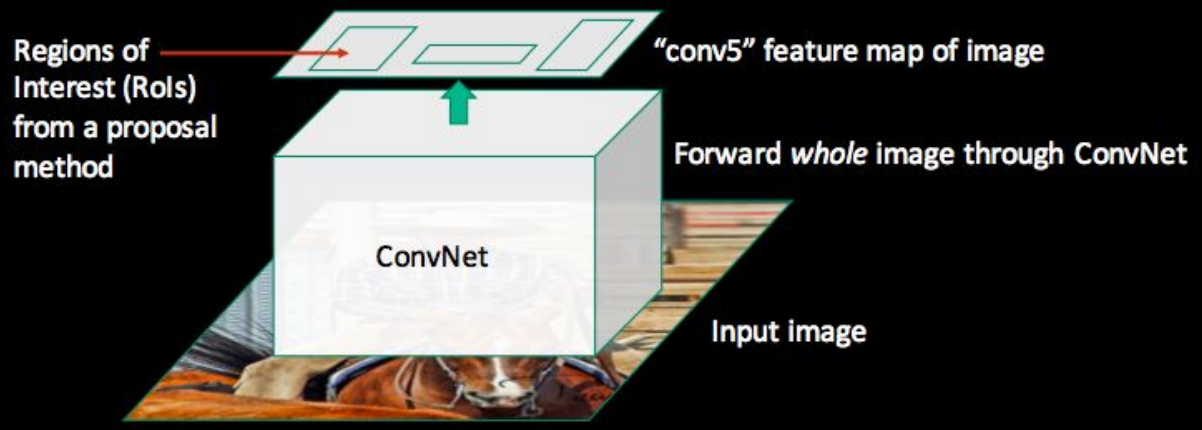
SPP-net



He et al. ECCV14.

Adapted from Fast R-CNN [R. Girshick (2015)]

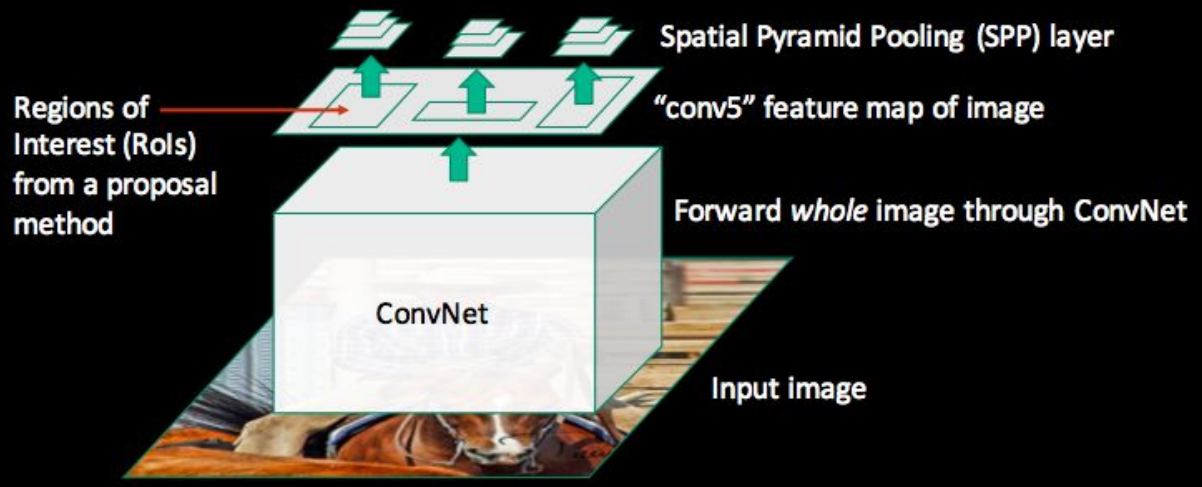
SPP-net



He et al. ECCV14.

Adapted from Fast R-CNN [R. Girshick (2015)]

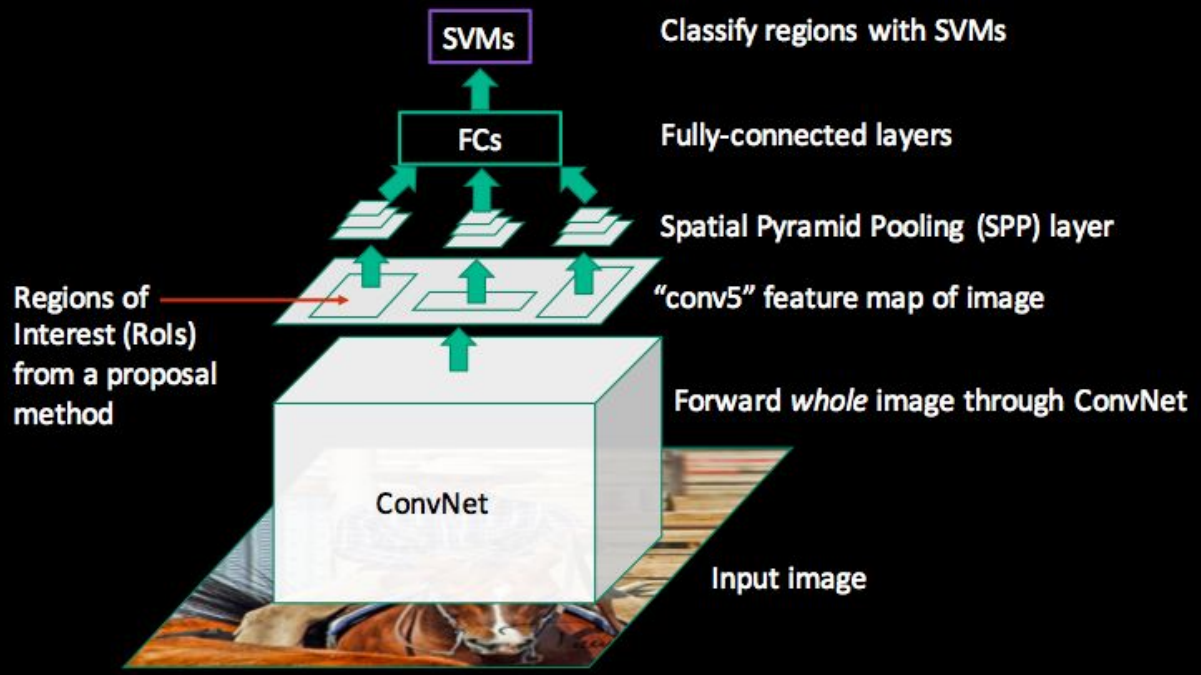
SPP-net



He et al. ECCV14.

Adapted from Fast R-CNN [R. Girshick (2015)]

SPP-net

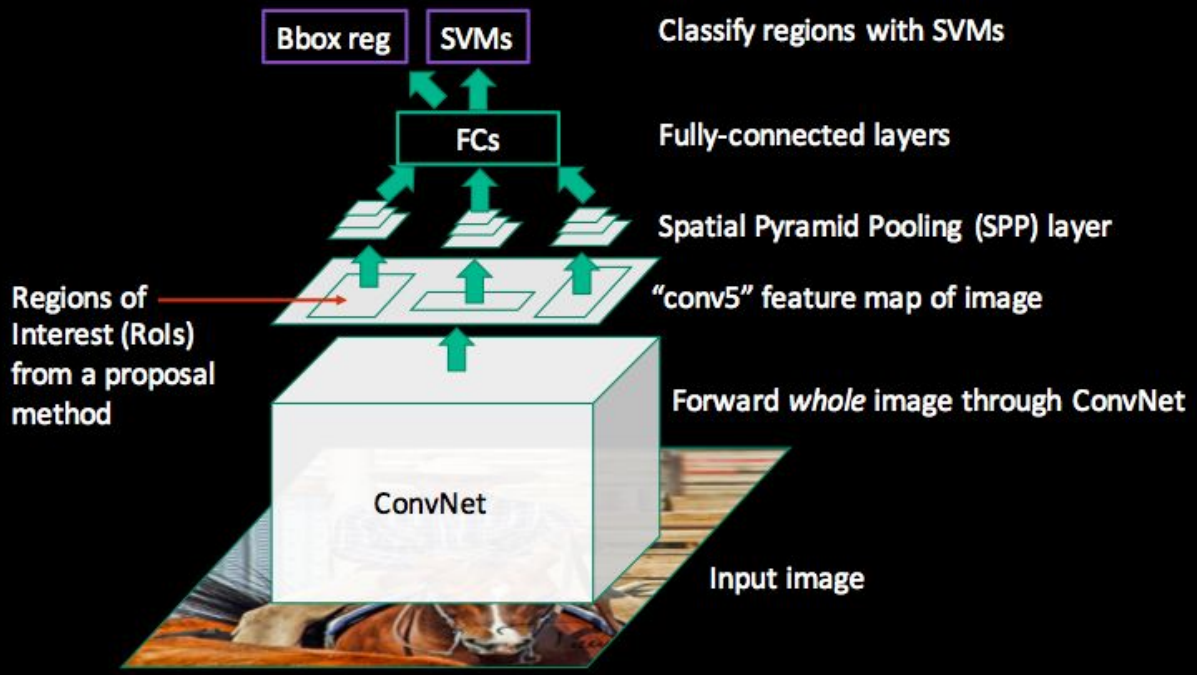


He et al. ECCV14.

Post hoc component

Adapted from Fast R-CNN [R. Girshick (2015)]

SPP-net



Apply bounding-box regressors

Classify regions with SVMs

Fully-connected layers

Spatial Pyramid Pooling (SPP) layer

"conv5" feature map of image

Forward *whole* image through ConvNet

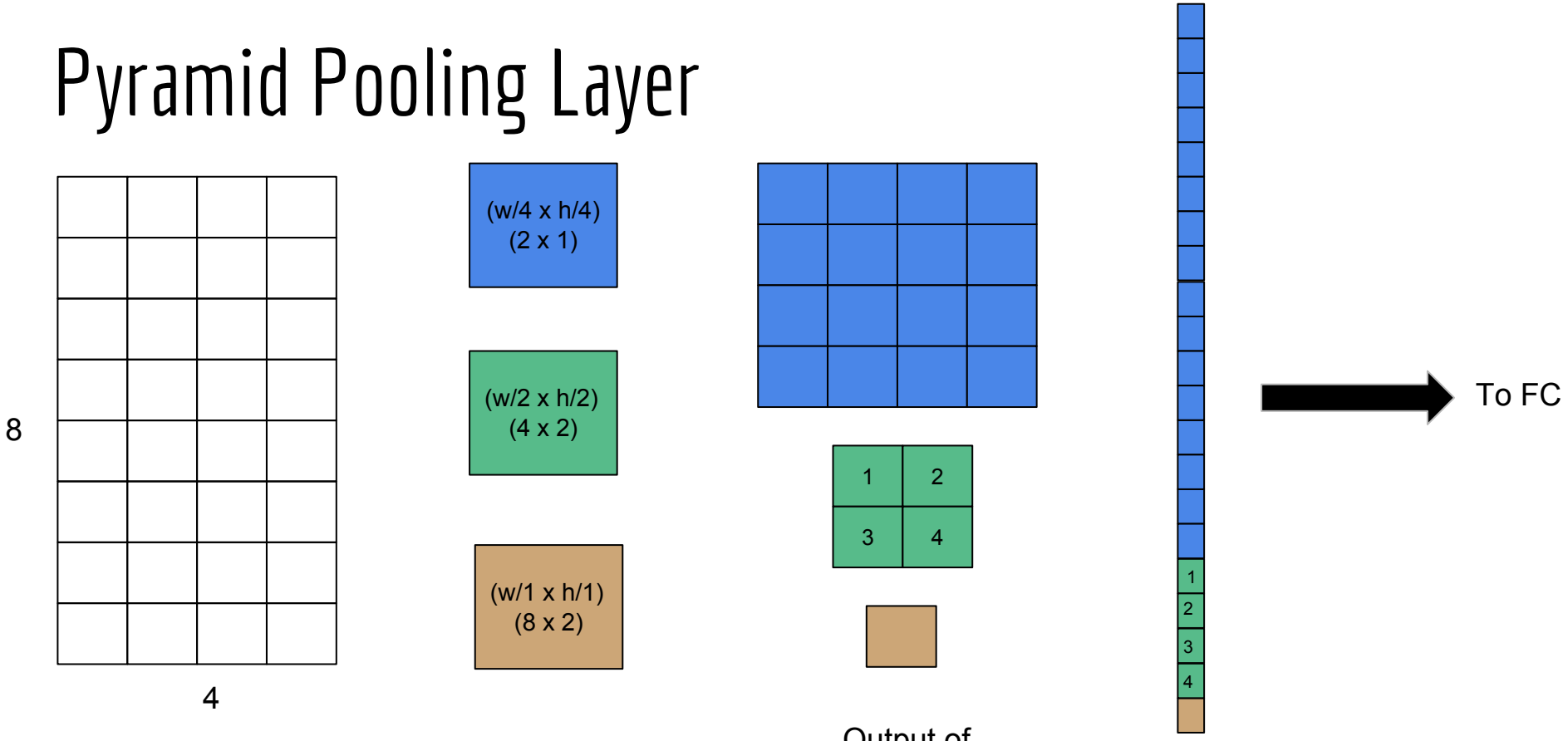
Input image

He et al. ECCV14.

Post hoc component

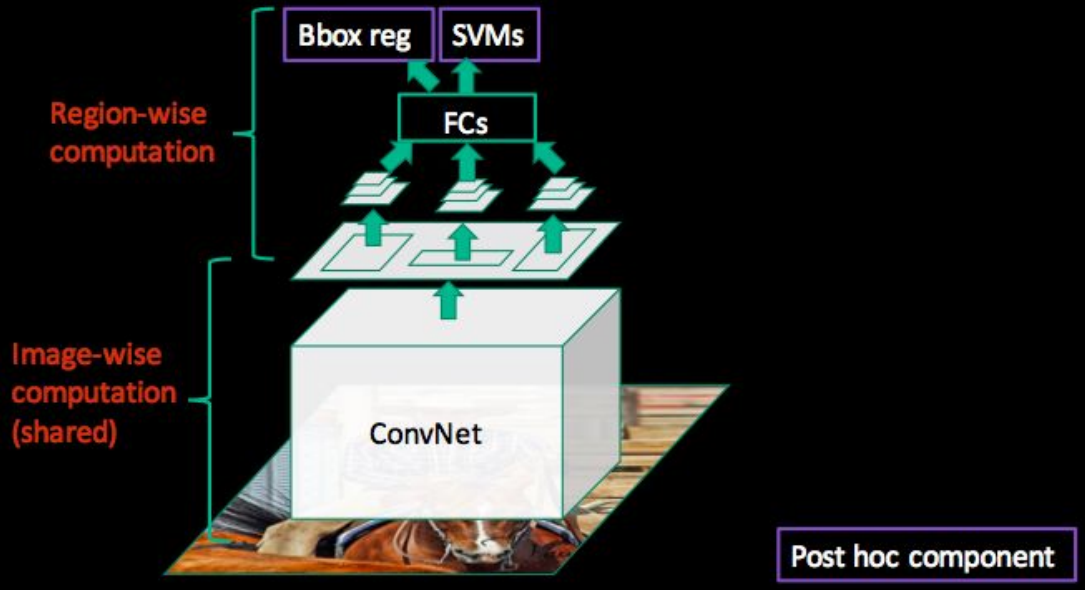
Adapted from Fast R-CNN [R. Girshick (2015)]

Pyramid Pooling Layer



What's good about SPP-net?

- Fixes one issue with R-CNN: makes testing fast

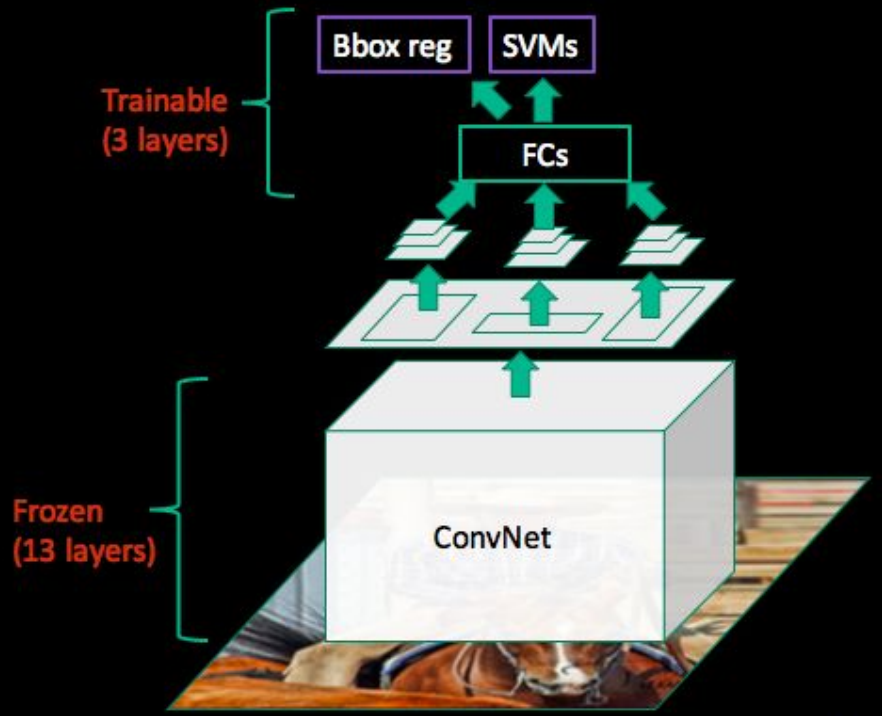


Adapted from Fast R-CNN [R. Girshick (2015)]

What's wrong with SPP-net?

- Inherits the rest of R-CNN's problems
  - Ad hoc training objective
  - Training is slow (25h), takes a lot of disk space
- Introduces a new problem: cannot update parameters below SPP layer during training

SPP-net: the main limitation



He et al. ECCV14.

Post hoc component

Adapted from Fast R-CNN [R. Girshick (2015)]

Agenda

1. Pre-existing Models
   a. “Slow” R-CNN
   b. SPP-net
2. Ways to improve
   a. SGD Mini-Batch
   b. New Loss Function
3. Fast R-CNN
   a. Architecture
   b. Results & Future Work

SGD Mini-Batch Method for RoIs

Slow R-CNN and SPP-net use region-wise sampling to make mini-batches

- Sample 128 example RoIs uniformly at random
- Examples will come from different images with high probability

Adapted from Fast R-CNN [R. Girshick (2015)]

SGD Mini-Batch Method for RoIs

Note the receptive field for one example RoI is often very large

- Worst case: the receptive field is the entire image

Adapted from Fast R-CNN [R. Girshick (2015)]

SGD Mini-Batch Method for RoIs

Worst case cost per mini-batch (crude model of computational complexity)

Input size for SPP-net

input size for slow R-CNN

128*600*1000 / (128*224 *224) = 12x more computation than slow R-CNN

Adapted from Fast R-CNN [R. Girshick (2015)]

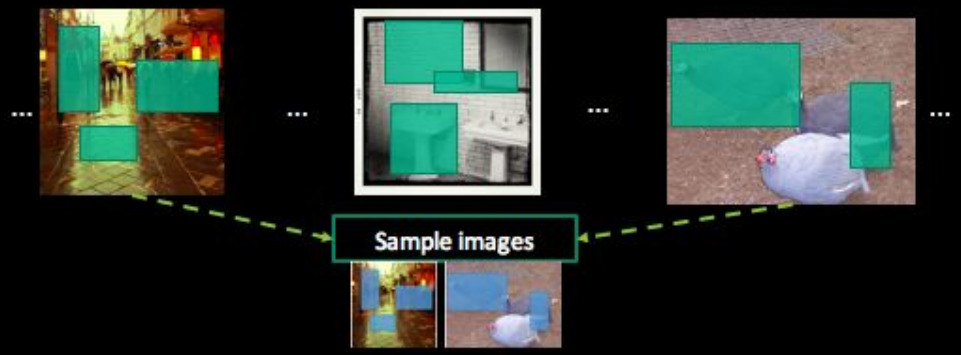
SGD Mini-Batch Method for RoIs

Solution: use hierarchical sampling to build mini-batches

Adapted from Fast R-CNN [R. Girshick (2015)]

SGD Mini-Batch Method for RoIs

Solution: use hierarchical sampling to build mini-batches

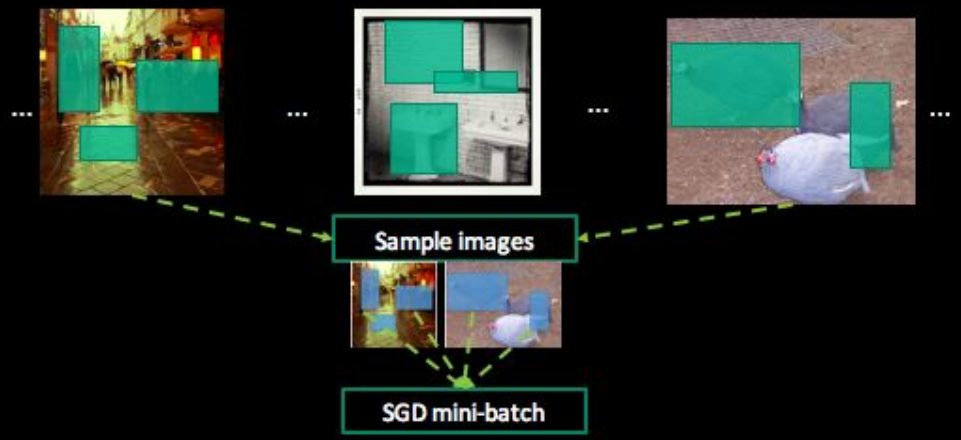


- Sample a small number of images (2)

Adapted from Fast R-CNN [R. Girshick (2015)]

SGD Mini-Batch Method for RoIs

Solution: use hierarchical sampling to build mini-batches



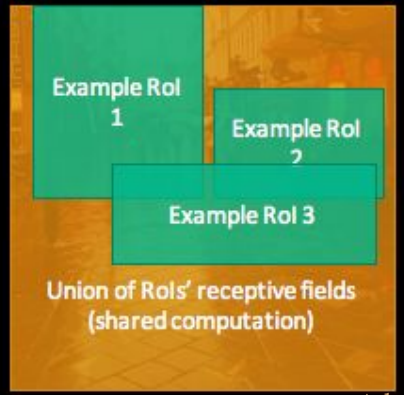
- Sample a small number of images (2)
- Sample many examples from each image (64)

Adapted from Fast R-CNN [R. Girshick (2015)]

SGD Mini-Batch Method for RoIs

Use the test-time trick from SPP-net during training

- Share computation between overlapping examples from the same image

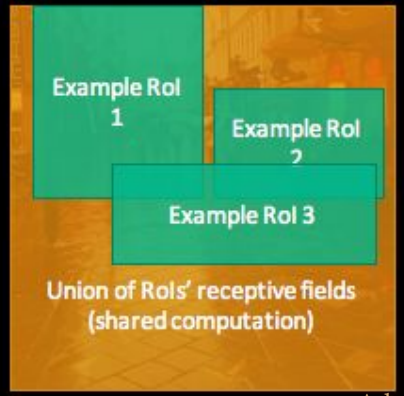


Adapted from Fast R-CNN [R. Girshick (2015)]

SGD Mini-Batch Method for RoIs

Cost per mini-batch compared to slow R-CNN (same crude cost model)

- $\underbrace{2*600*1000}_{\text{input size for Fast R-CNN}} / \underbrace{(128*224*224)}_{\text{input size for slow R-CNN}}$ = 0.19x less computation than slow R-CNN



Adapted from Fast R-CNN [R. Girshick (2015)]

Agenda

1. Pre-existing Models
   a. “Slow” R-CNN
   b. SPP-net
2. Ways to improve
   a. SGD Mini-Batch
   b. **New Loss Function**
3. Fast R-CNN
   a. Architecture
   b. Results
   c. Future Work

Revised loss function

For the classification *For the bounding box*

$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1]L_{\text{loc}}(t^u, v),$$

p: Predicted RoI Classification
u: True RoI Classification

$t^u = (t_x, t_y, t_w, t_h)$: Predicted Bounding Box
$v = (v_x, v_y, v_w, v_h)$: True Bounding Box

λ : Controls the balance between the two losses

Revised loss function

$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1] L_{\text{loc}}(t^u, v),$$

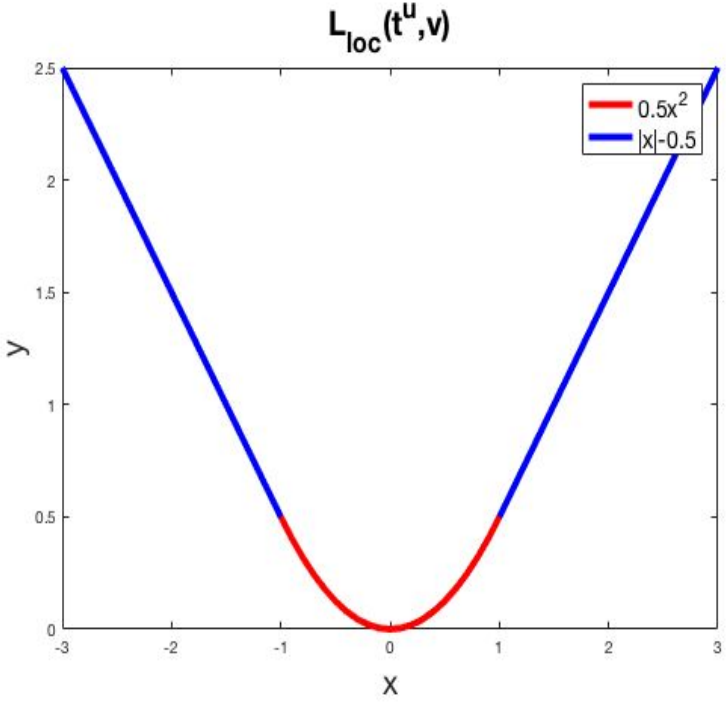
$$L_{\text{loc}}(t^u, v) = \sum_{i \in \{\text{x,y,w,h}\}} \text{smooth}_{L_1}(t_i^u - v_i),$$

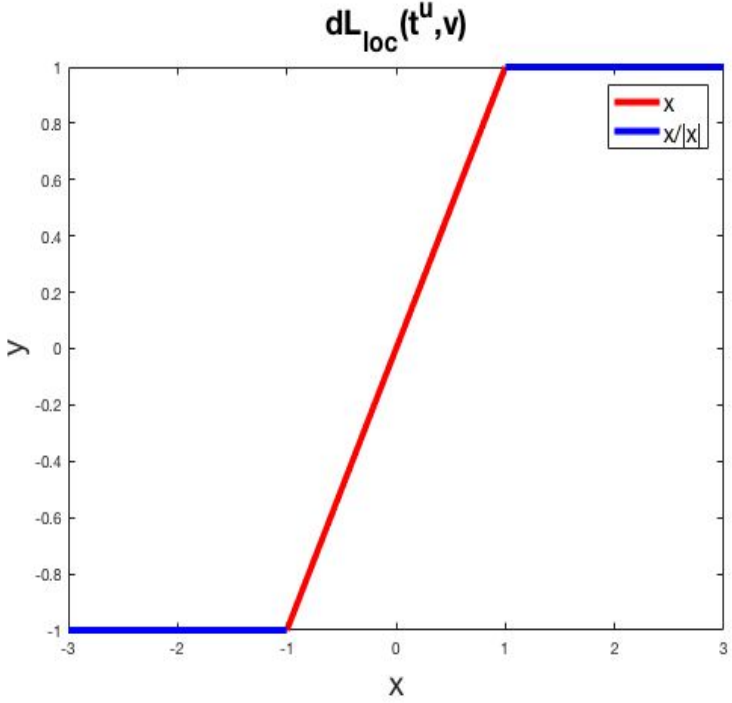
in which

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$

Revised loss function

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$





Smooth: *Continuously* Differentiable

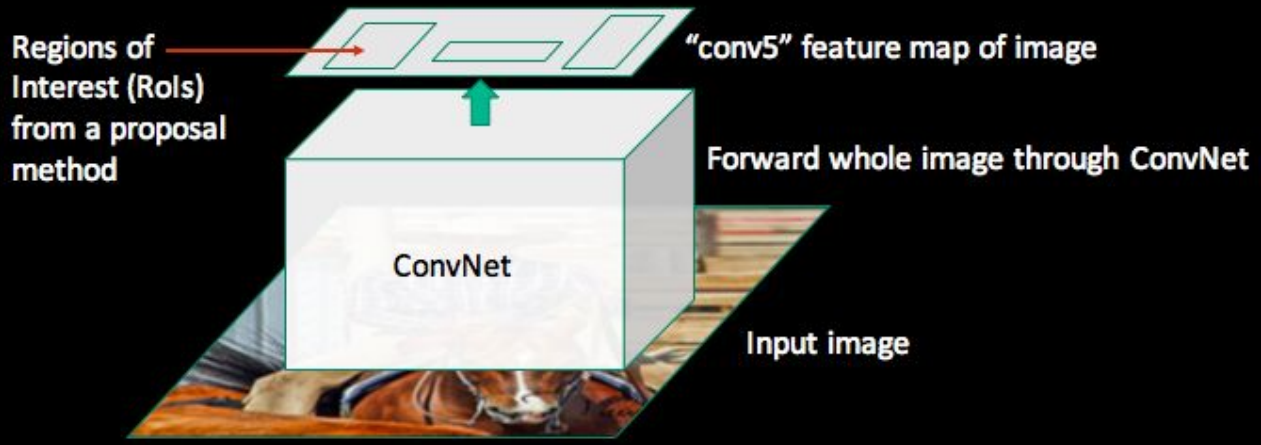
Agenda

1. Pre-existing Models
   a. “Slow” R-CNN
   b. SPP-net
2. Ways to improve
   a. SGD Mini-Batch
   b. New Loss Function
3. Fast R-CNN
   a. **Architecture**
   b. Results & Future Work

Fast R-CNN

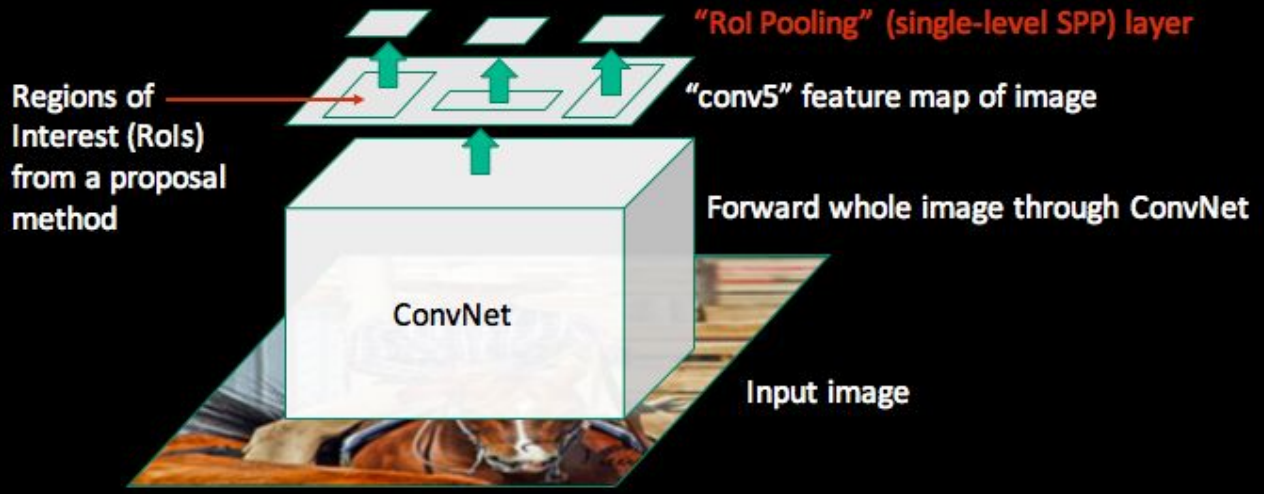
- Fast test-time, like SPP-net
- One network, trained in one stage
- Higher mean average precision than slow R-CNN and SPP-net

Fast R-CNN (test time)



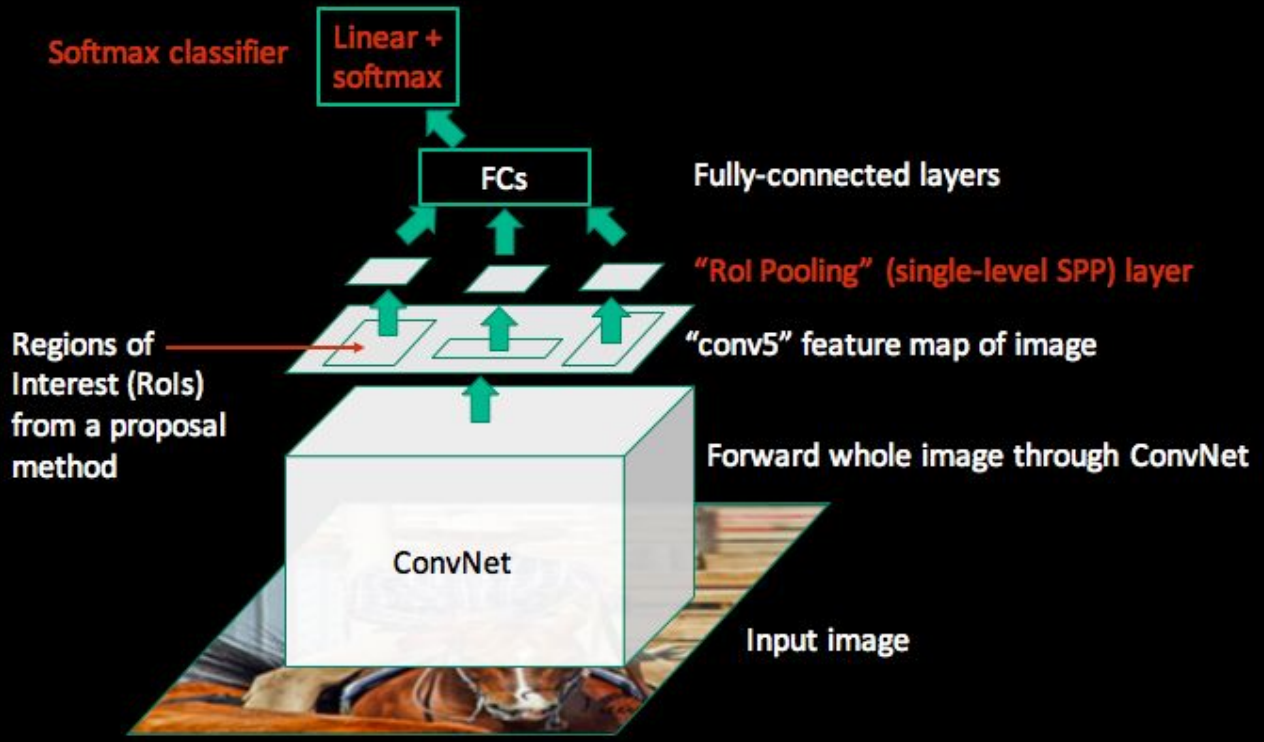
Adapted from Fast R-CNN [R. Girshick (2015)]

Fast R-CNN (test time)



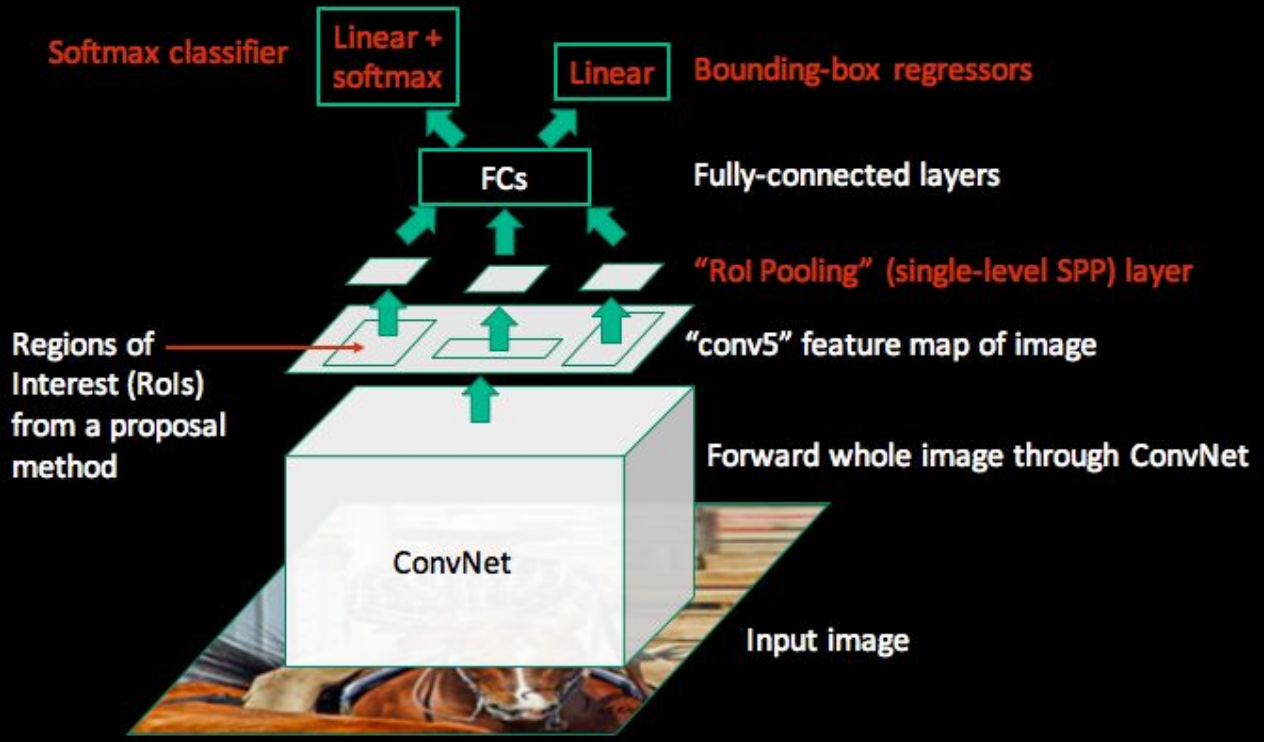
Adapted from Fast R-CNN [R. Girshick (2015)]

Fast R-CNN (test time)



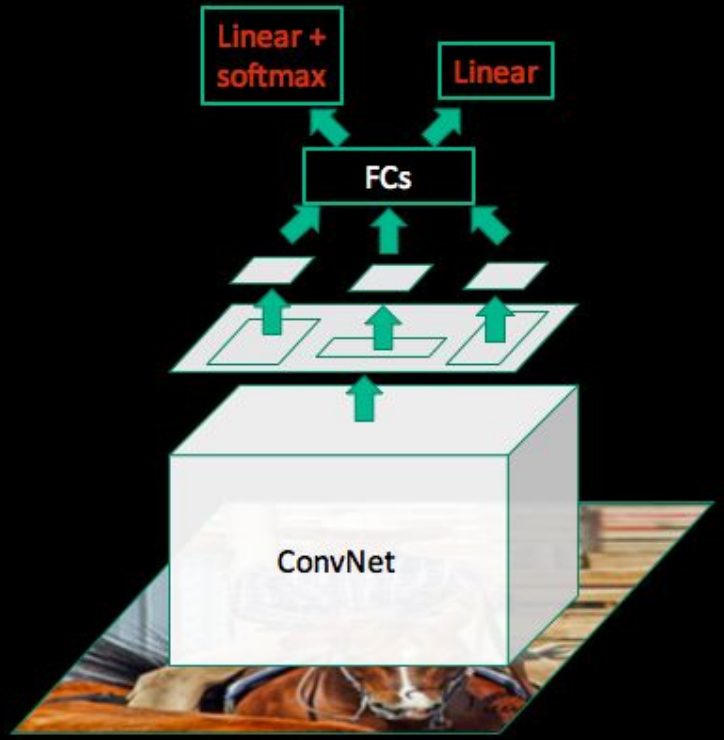
Adapted from Fast R-CNN [R. Girshick (2015)]

Fast R-CNN (test time)



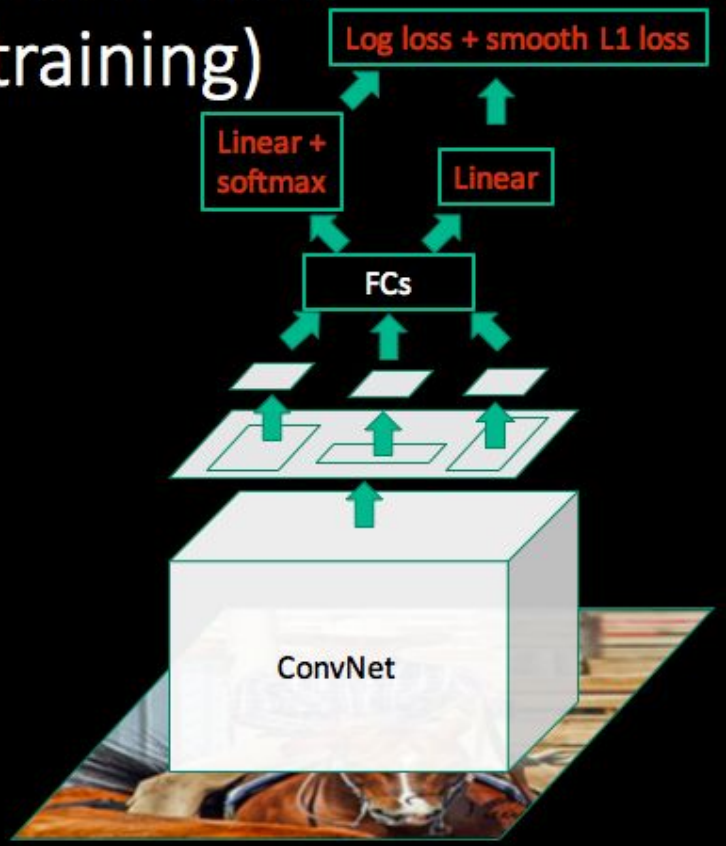
Adapted from Fast R-CNN [R. Girshick (2015)]

Fast R-CNN (training)



Adapted from Fast R-CNN [R. Girshick (2015)]

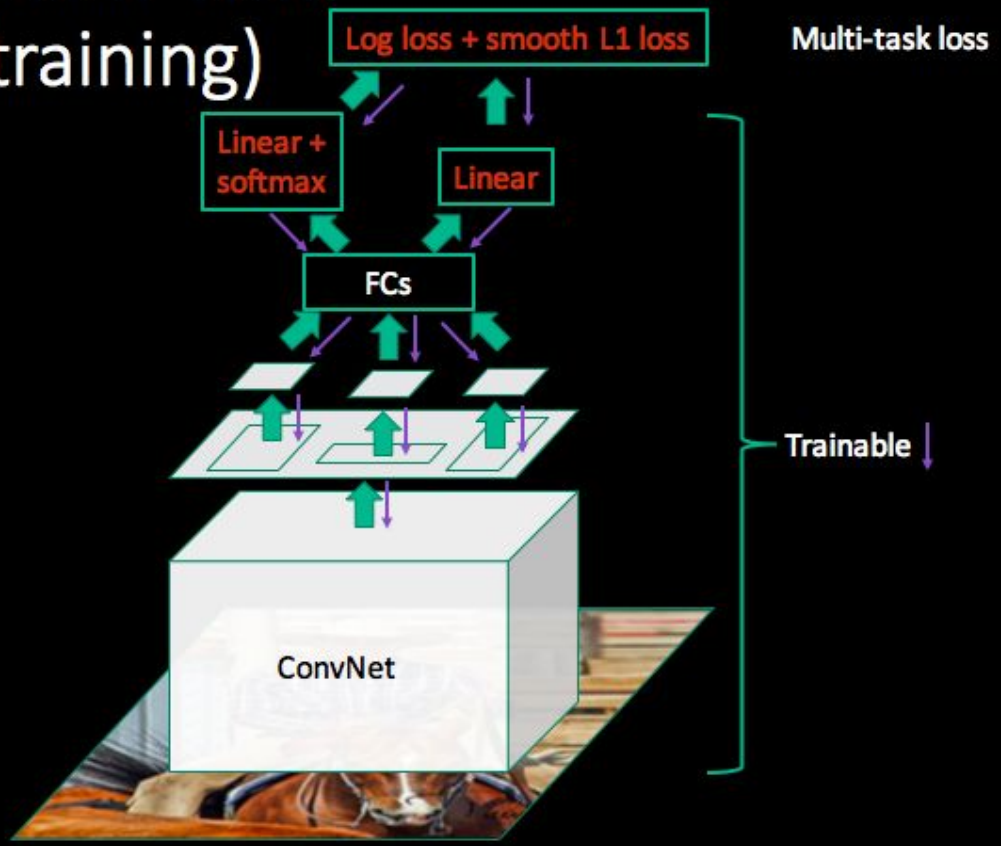
Fast R-CNN (training)



Multi-task loss

Adapted from Fast R-CNN [R. Girshick (2015)]

Fast R-CNN (training)



Adapted from Fast R-CNN [R. Girshick (2015)]

Agenda

1. Pre-existing Models
   a. “Slow” R-CNN
   b. SPP-net
2. Ways to improve
   a. SGD Mini-Batch
   b. New Loss Function
3. Fast R-CNN
   a. Architecture
   b. Results & Future Work

Main results

	Fast R-CNN	R-CNN [1]	SPP-net [2]
Train time (h)	**9.5**	84	25
- Speedup	**8.8x**	1x	3.4x
Test time / image	**0.32s**	47.0s	2.3s
Test speedup	**146x**	1x	20x
mAP	**66.9%**	66.0%	63.1%

Timings exclude object proposal time, which is equal for all methods.
All methods use VGG16 from Simonyan and Zisserman.

[1] Girshick et al. CVPR14.
[2] He et al. ECCV14.

Adapted from Fast R-CNN [R. Girshick (2015)]

Main results

	Fast R-CNN	R-CNN [1]	SPP-net [2]
Train time (h)	9.5	84	25
- Speedup	8.8x	1x	3.4x
Test time / image	**0.32s**	47.0s	2.3s
Test speedup	**146x**	1x	20x
mAP	66.9%	66.0%	63.1%

Timings exclude object proposal time, which is equal for all methods.
All methods use VGG16 from Simonyan and Zisserman.

[1] Girshick et al. CVPR14.
[2] He et al. ECCV14.

Adapted from Fast R-CNN [R. Girshick (2015)]

Main results

	Fast R-CNN	R-CNN [1]	SPP-net [2]
Train time (h)	9.5	84	25
- Speedup	8.8x	1x	3.4x
Test time / image	0.32s	47.0s	2.3s
Test speedup	146x	1x	20x
mAP	**66.9%**	66.0%	63.1%

Timings exclude object proposal time, which is equal for all methods.
All methods use VGG16 from Simonyan and Zisserman.

[1] Girshick et al. CVPR14.
[2] He et al. ECCV14.

Adapted from Fast R-CNN [R. Girshick (2015)]

What's still wrong?

- Out-of-network region proposals
  - Selective search: 2s / img; EdgeBoxes: 0.2s / img
- Fortunately, this has already been solved

S. Ren, K. He, R. Girshick & J. Sun. "Faster R'CNN: Towards Real'Time Object Detection with Region Proposal Networks." NIPS (2015).

Fast R-CNN take-aways

- End-to-end training of deep ConvNets for object detection
- Fast training times
- Open source for easy experimentation
- A large number of ImageNet detection and COCO detection methods are built on Fast R-CNN