RCNN, Fast RCNN, Faster RCNN

Presented by: Roi Shikler & Gil Elbaz
Advisor: Prof. Michael Lindenbaum

Topics of the lecture:

- Problem statement
- Review of slow R-CNN
- Review of Fast R-CNN
- Review of Faster R-CNN
- Compare with other methods
- Take away

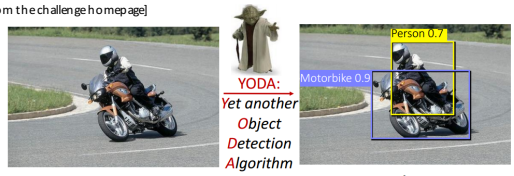
Topics of the lecture:

➡ **Problem statement**

- Review of slow R-CNN
- Review of Fast R-CNN
- Review of Faster R-CNN
- Compare with other methods
- Take away

PASCAL Visual Object Classes

"The main goal of this challenge is to **recognize objects** from a number of visual object classes in realistic scenes"
[from the challenge homepage]

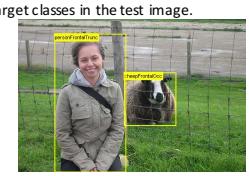


Input *Desired* output

PASCAL Visual Object Classes

The challenge face the participants in front of two problems:

- **Classification**: For each of the twenty classes, predicting presence/absence of an example of that class in the test image.
- **Detection (localization)**: Predicting the bounding box and label of each object from the twenty target classes in the test image.

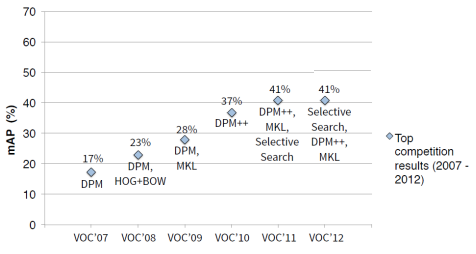


PASCAL Visual Object Classes

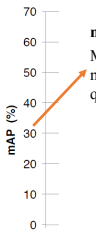
The twenty object classes that have been selected are:

- *Person:* person
- *Animal:* bird, cat, cow, dog, horse, sheep
- *Vehicle:* airplane, bicycle, boat, bus, car, motorbike, train
- *Indoor:* bottle, chair, dining table, potted plant, sofa, TV/monitor

PASCAL VOC detection history



PASCAL VOC detection history

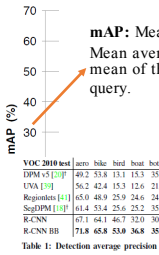


mAP: Mean Average Precision.

Mean average precision for a set of queries is the mean of the average precision scores for each query.

$$mAP = \frac{\sum_{q=1}^{Q} AvgP(q)}{Q}$$

PASCAL VOC detection history



mAP: Mean Average Precision.

Mean average precision for a set of queries is the mean of the average precision scores for each query.

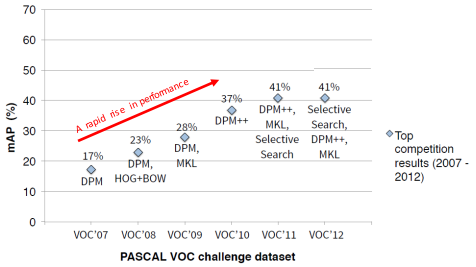
$$mAP = \frac{\sum_{q=1}^{Q} AvgP(q)}{Q}$$

P=TP/(TP+FP)

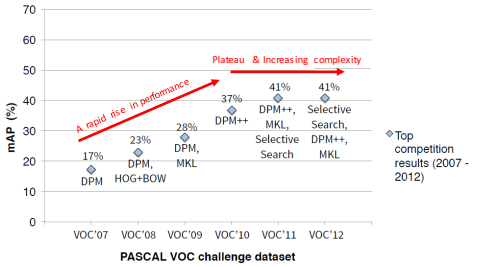
VOC 2010 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
DPM v5 [20]†	49.2	53.8	13.1	15.3	35.5	53.4	49.7	27.0	17.2	28.8	14.7	17.8	46.4	51.2	47.7	10.8	34.2	20.7	43.8	38.3	33.4
UVA [39]	56.2	42.4	15.3	12.6	21.8	49.3	36.8	46.1	12.9	32.1	30.0	36.5	43.5	52.9	32.9	15.3	41.1	31.8	47.0	44.8	35.1
Regionlets [41]	65.0	48.9	25.9	24.6	24.5	56.1	54.5	51.2	17.0	28.9	30.2	35.8	40.2	55.7	43.5	14.3	43.9	32.6	54.0	45.9	39.7
SegDPM [18]†	61.4	53.4	25.6	25.2	35.5	51.7	50.6	50.8	19.3	33.8	26.8	40.4	48.3	54.4	47.1	14.8	38.7	35.0	52.8	43.1	40.4
R-CNN	67.1	64.1	46.7	32.0	30.5	56.4	57.2	65.9	27.0	47.3	40.9	66.6	57.8	65.9	53.6	26.7	56.5	38.1	52.8	50.2	50.2
R-CNN BB	**71.8**	**65.8**	**53.0**	**36.8**	**35.9**	**59.7**	**60.0**	**60.9**	**27.9**	**50.6**	**41.4**	**70.0**	**62.0**	**69.0**	**58.1**	**29.5**	**59.4**	**39.3**	**61.2**	**52.4**	**53.7**

Table 1: Detection average precision (%) on VOC 2010 test.

PASCAL VOC detection history



PASCAL VOC detection history



PASCAL VOC detection history

PASCAL VOC detection history

Topics of the lecture:

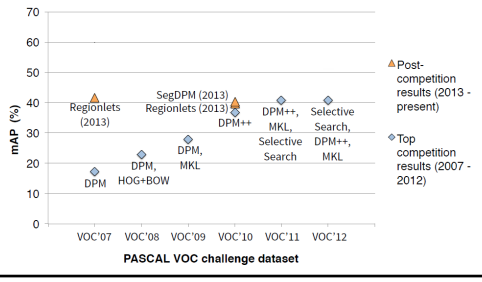
- Problem statement
- **Review of slow R-CNN**
- Review of Fast R-CNN
- Review of Faster R-CNN
- Compare with other methods
- Take away

R-CNN - Article (2013)

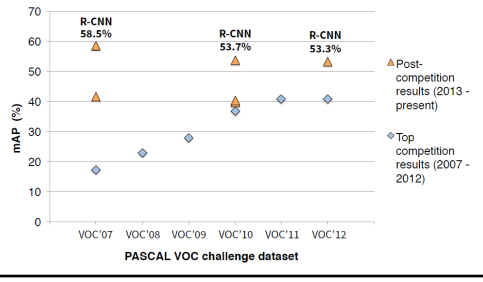
Rich feature hierarchies for accurate object detection and semantic segmentation
Tech report

Ross Girshick[1] Jeff Donahue[1,2] Trevor Darrell[1,2] Jitendra Malik[1]
[1]UC Berkeley and [2]ICSI
{rbg, jdonahue, trevor, malik}@eecs.berkeley.edu

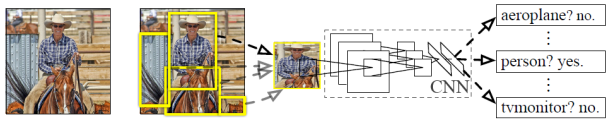
PASCAL VOC detection history



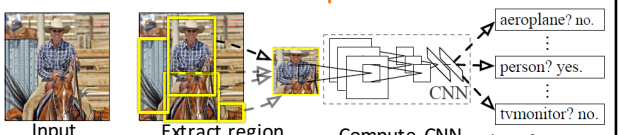
PASCAL VOC detection history



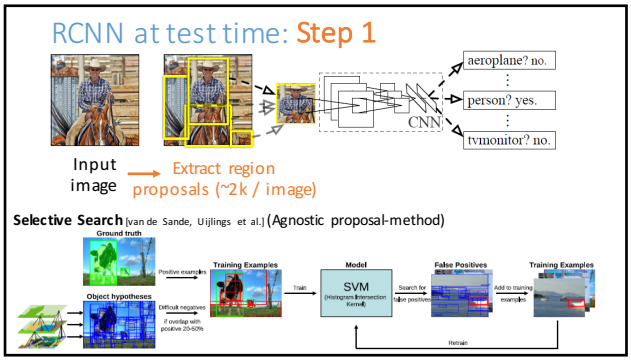
R-CNN: Regions with CNN features



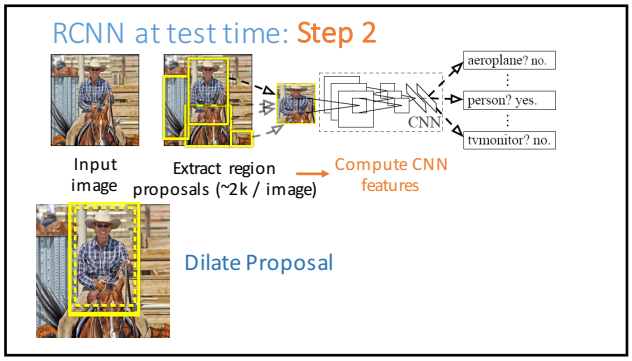
RCNN at test time: Step 1



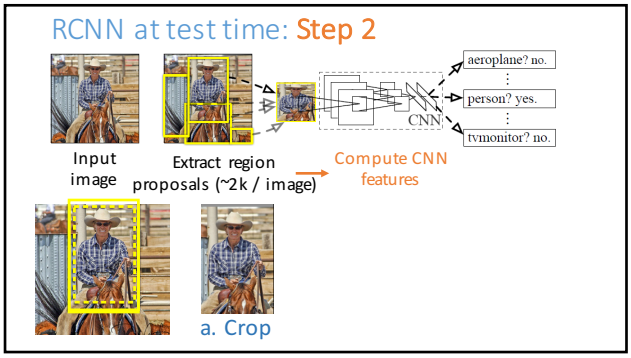
RCNN at test time: Step 1



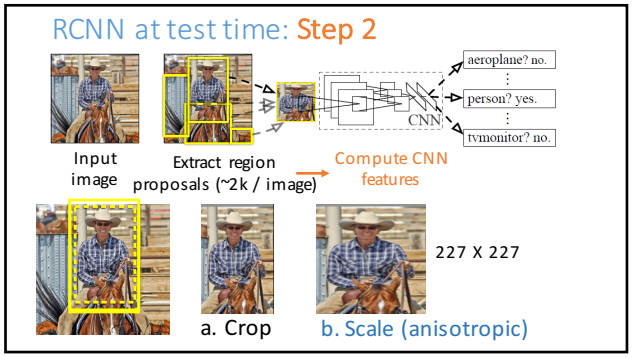
RCNN at test time: Step 2



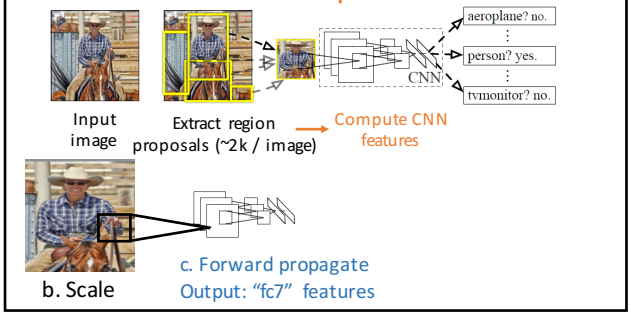
RCNN at test time: Step 2



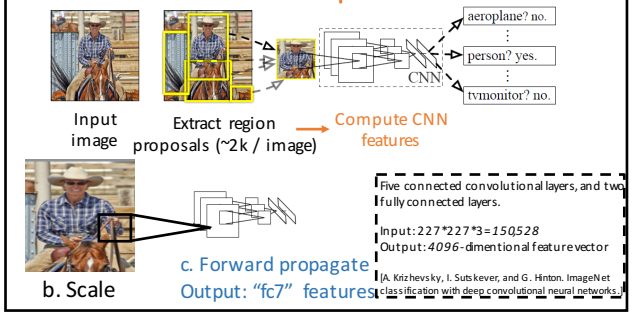
RCNN at test time: Step 2



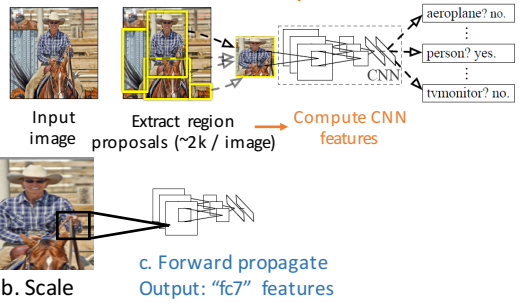
RCNN at test time: Step 2



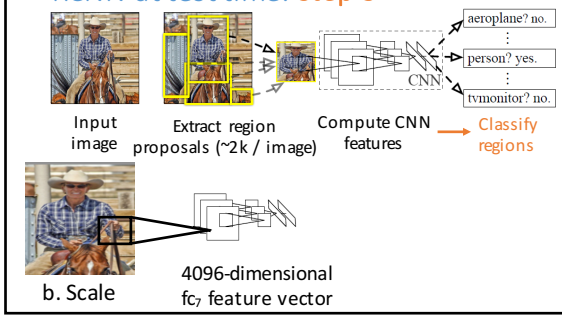
RCNN at test time: Step 2



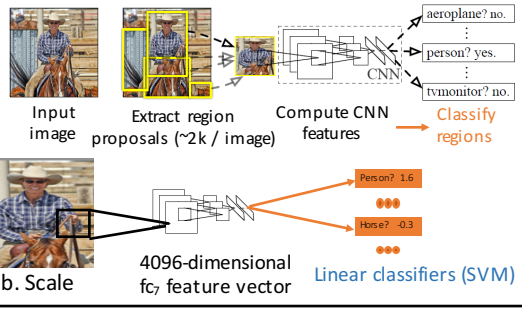
RCNN at test time: Step 2



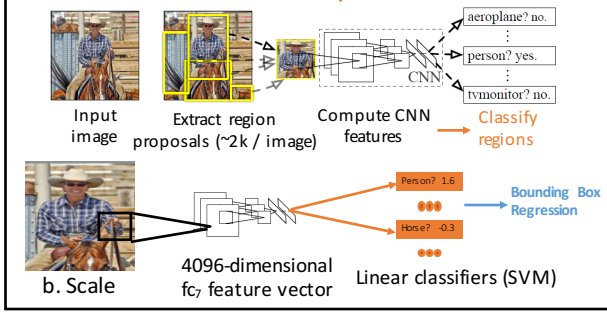
RCNN at test time: Step 3



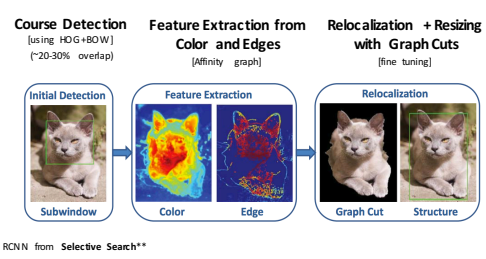
RCNN at test time: Step 3



RCNN at test time: Step 4



Step 4: Object proposal refinement



In RCNN from **Selective Search

Step 4: Object proposal refinement

Review of the slow R-CNN training pipeline

Steps for training a slow R-CNN detector:

1. [offline] *M* ← **Pre-train a ConvNet** for ImageNet classification

Review of the slow R-CNN training pipeline

Steps for training a slow R-CNN detector:

1. [offline] *M* ← **Pre-train a ConvNet** for ImageNet classification
2. *M'* ← **Fine-tune** *M* for object *detection* (softmax classifier)

Review of the slow R-CNN training pipeline

Steps for training a slow R-CNN detector:

1. [offline] *M* ← **Pre-train a ConvNet** for ImageNet classification
2. *M'* ← **Fine-tune** *M* for object *detection* (softmax classifier)
3. *F* ← **Cache feature** vectors to disc using *M'*

Review of the slow R-CNN training pipeline

Steps for training a slow R-CNN detector:

1. [offline] *M* ← **Pre-train a ConvNet** for ImageNet classification
2. *M'* ← **Fine-tune** *M* for object *detection* (softmax classifier)
3. *F* ← **Cache feature** vectors to disc using *M'*
4. **Train** post hoc **linear SVMs** on *F* for object *classification*

* "post hoc" means the parameters are learned after the ConvNet is fixed

Review of the slow R-CNN training pipeline

Steps for training a slow R-CNN detector:

1. [offline] *M* ← **Pre-train a ConvNet** for ImageNet classification
2. *M'* ← **Fine-tune** *M* for object *detection* (softmax classifier)
3. *F* ← **Cache feature** vectors to disc using *M'*
4. **Train** post hoc **linear SVMs** on *F* for object *classification*
5. **Train** post hoc **linear bounding-box** regressors on *F*

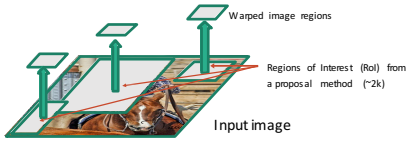
* "post hoc" means the parameters are learned after the ConvNet is fixed

Review slow R-CNN

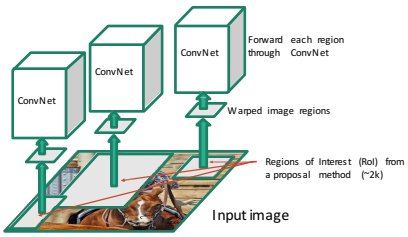
Input image

Review slow R-CNN

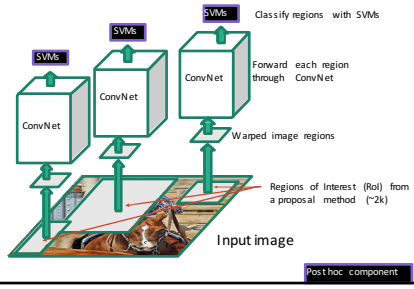
Review slow R-CNN



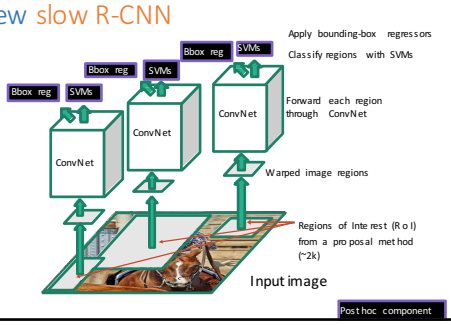
Review slow R-CNN



Review slow R-CNN



Review slow R-CNN



What's wrong with slow R-CNN?

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 (squared loss)

What's wrong with slow R-CNN?

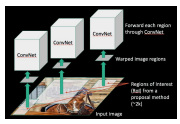
- Ad hoc training objectives
  - Fine tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post hoc bounding-box regressors (squared loss)
- Training is slow, takes a lot of disk space
  - The training process takes about 84h

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 (squared loss)
- Training is slow, takes a lot of disk space
  - The training process takes about 84h
- Detection (inference) is slow
  - About 47s per image
  - ~2k ConvNet forward passes per image

What's wrong with slow R-CNN?

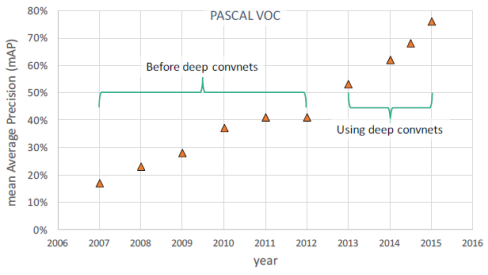
- Ad hoc training objectives
  - Fine tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post hoc bounding-box regressors (squared loss)
- Training is slow, takes a lot of disk space
  - The training process takes about 84h
- Detection (inference) is slow
  - About 47s per image
  - ~2k ConvNet forward passes per image



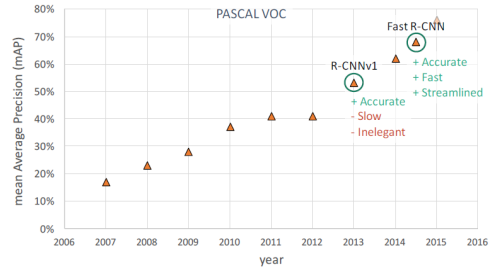
Topics of the lecture:

- ✓ Problem statement
- ✓ Review of slow R-CNN
- ➡ **Review of Fast R-CNN**
- Review of Faster R-CNN
- Compare with other methods
- Take away

PASCAL VOC detection history



PASCAL VOC detection history



Fast R-CNN objectives

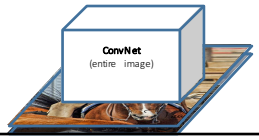
Fix most of what's wrong with slow R-CNN and SPP-net

- Train the detector in a single stage, end-to-end
  - No caching features to disk
  - No post hoc training steps
- Train all layers of the network

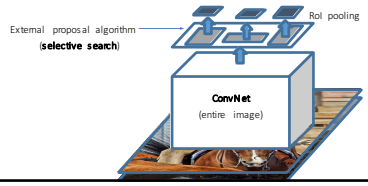
  [Training the conv layers is important for very deep networks]

Review of the fast R-CNN training pipeline

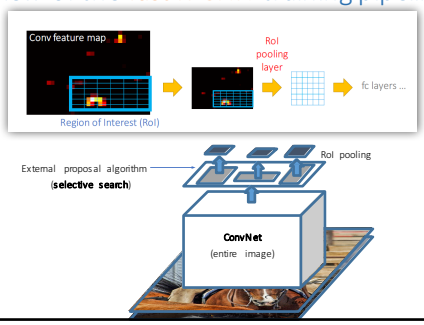
Review of the fast R-CNN training pipeline



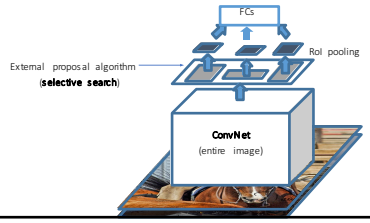
Review of the fast R-CNN training pipeline



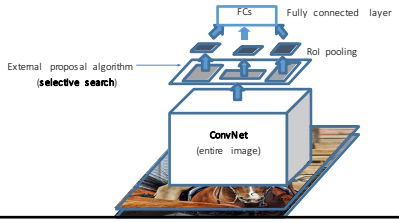
Review of the fast R-CNN training pipeline



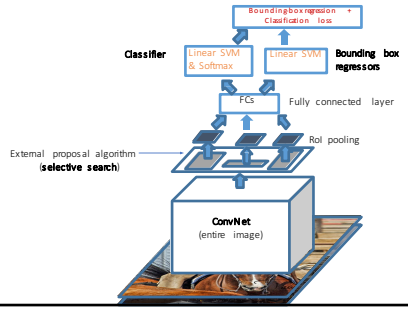
Review of the fast R-CNN training pipeline



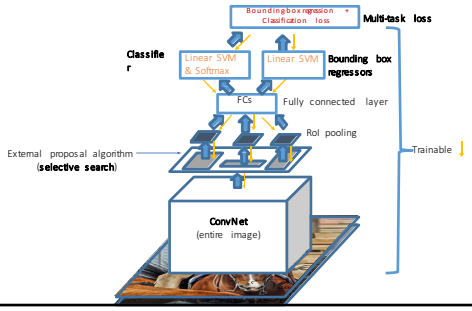
Review of the fast R-CNN training pipeline



Review of the fast R-CNN training pipeline



Review of the fast R-CNN training pipeline



Benefits of end-to-end training

- Faster training
  - No reading/writing features from/to disk
  - No training post hoc SVMs and bounding-box regressors
- Verified empirically: optimizing a single multi-task objective is more accurate than optimizing objectives independently

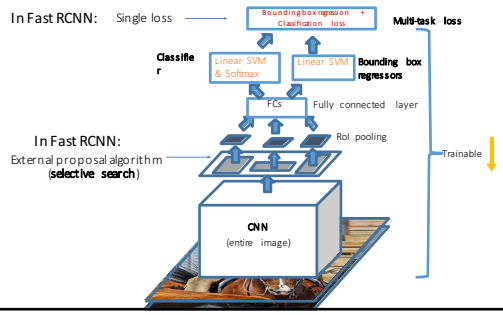
Slow R-CNN vs. Fast R-CNN

- Training time: 84 hours / 8.75 hours
- VOC07 test mAP: 66.0% / 68.1%
- Testing time per image: 47s / 0.32s
  - With selective search: 49s / 2.32s (+2s per image)

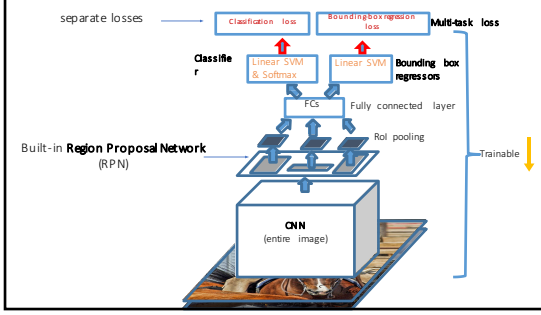
Topics of the lecture:

- ✓ Problem statement
- ✓ Review of slow R-CNN
- ✓ Review of Fast R-CNN
- ➡ **Review of Faster R-CNN**
- Compare with other methods
- Take away

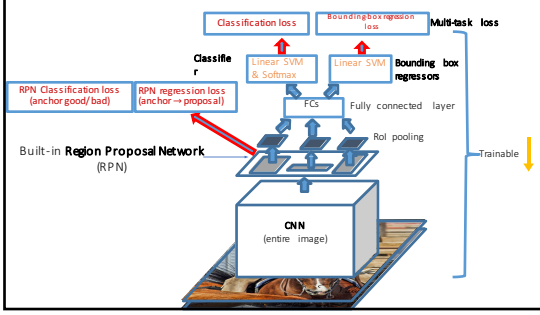
Review of the faster R-CNN



Review of the faster R-CNN



Review of the faster R-CNN



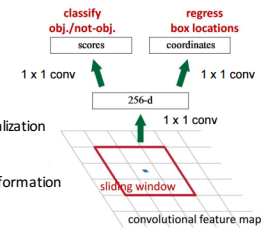
Faster R-CNN: Region Proposal Network

Slide a small window on the feature map.

Build a **small network** for:
- **Classifying** object or not-object
- Regressing bounding-box **localizations**

Position of the sliding window provides localization information with reference to the image

Box regression provides finer localization information with reference to this sliding window.



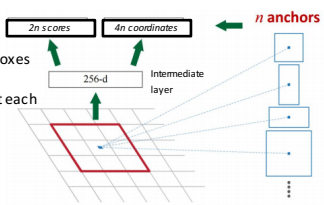
Faster R-CNN: Region Proposal Network

Use **N anchor boxes** at each location

Anchors are **translation invariant**

Regression gives offsets from anchor boxes

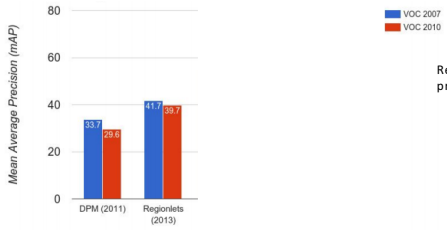
Classification gives the probability that each regressed box shows an object



Topics of the lecture:

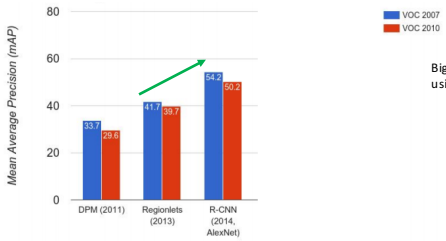
- ✓ Problem statement
- ✓ Review of slow R-CNN
- ✓ Review of Fast R-CNN
- ✓ Review of Faster R-CNN
- ➡ **Compare with other methods**
- Take away

R-CNN Results



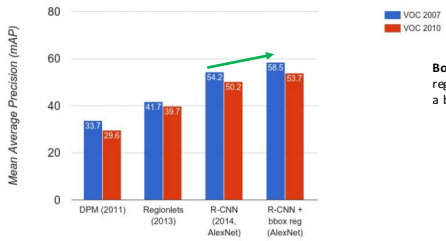
Results by using pre-CNN methods

R-CNN Results



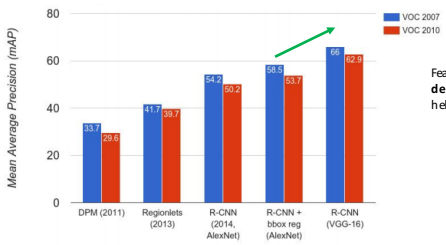
Big improvement using **CNN**

R-CNN Results



Bounding box regression shows a bit improvement

R-CNN Results



Features from **deeper network** helps a lot

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**
mAP (VOC 2007)	66.0	**66.9**
Test time per image with Selective Search	50 seconds	**2 seconds**
(Speedup)	1x	**25x**

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**

Faster R-CNN Results

YOLO – You Only Look Once

Detection as Regression

Faster than Faster R-CNN, but not as good

Real-Time Detectors	Train	mAP	FPS
100Hz DPM [30]	2007	16.0	100
30Hz DPM [30]	2007	26.1	30
Fast YOLO	2007+2012	52.7	**155**
YOLO	2007+2012	**63.4**	45
Less Than Real-Time			
Fastest DPM [37]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[27]	2007+2012	73.2	7
Faster R-CNN ZF [27]	2007+2012	62.1	18

Faster R-CNN Results

Selective Search vs. Region Proposal Network

Table 7: Results on PASCAL VOC 2012 test set with Fast R-CNN detectors and VGG-16. For RPN, the train-time proposals for Fast R-CNN are 2000.

method	# box	data	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
SS	2000	12	65.7	80.3	74.7	66.9	46.9	37.7	73.9	68.6	87.7	41.7	71.1	51.1	86.0	77.8	79.8	69.8	32.1	65.5	63.8	76.4	61.7
SS	2000	07++12	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	**87.5**	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
RPN	300	12	67.0	82.3	76.4	71.0	48.4	45.2	72.1	72.3	87.3	42.2	73.7	50.0	86.8	78.7	78.4	77.4	34.5	70.1	57.1	77.1	58.9
RPN	300	07++12	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
RPN	300	COCO+07++12	**75.9**	**87.4**	**83.6**	**76.8**	**62.9**	**59.6**	**81.9**	**82.0**	**91.3**	**54.9**	**82.6**	**59.0**	**89.0**	**85.5**	**84.7**	**84.1**	**52.2**	**78.9**	65.5	**85.4**	**70.2**

Topics of the lecture:

- Problem statement
- Review of slow R-CNN
- Review of Fast R-CNN
- Review of Faster R-CNN
- Compare with other methods
- **Take away**

Take Away

- Classification and detection of objects in images is approaching **high quality** and **real-time frame rate** results.
- The **major breakthroughs** came from:
  1. Utilizing CNN instead of classical computer vision methods
  2. Making deeper neural networks with multiple objectives
  3. Switching all stages of algorithm with one unified network (end-to-end)
- These algorithms are all **available** and **applicable** to real world problems! Once you know how they work you can change and adjust them to your specific research needs.

Object Detection code links:

R-CNN (Cafffe+MATLAB):
https://github.com/rbgirshick/rcnn
[Probably don't use this; too slow]

Fast R-CNN (Caffe+MATLAB):
https://github.com/rbgirshick/fast-rcnn

Faster R-CNN (Caffe+MATLAB):
https://github.com/ShaoqingRen/faster_rcnn

Faster R-CNN (Caffe+Python):
https://github.com/rbgirshick/py-faster-rcnn

YOLO:
http://pjreddie.com/darknet/yolo/

Any Questions?