G-CNN: an Iterative Grid Based Object Detector

Magyar Najibi
Univ. Maryland

Mohammad Rastegari
Univ. Maryland

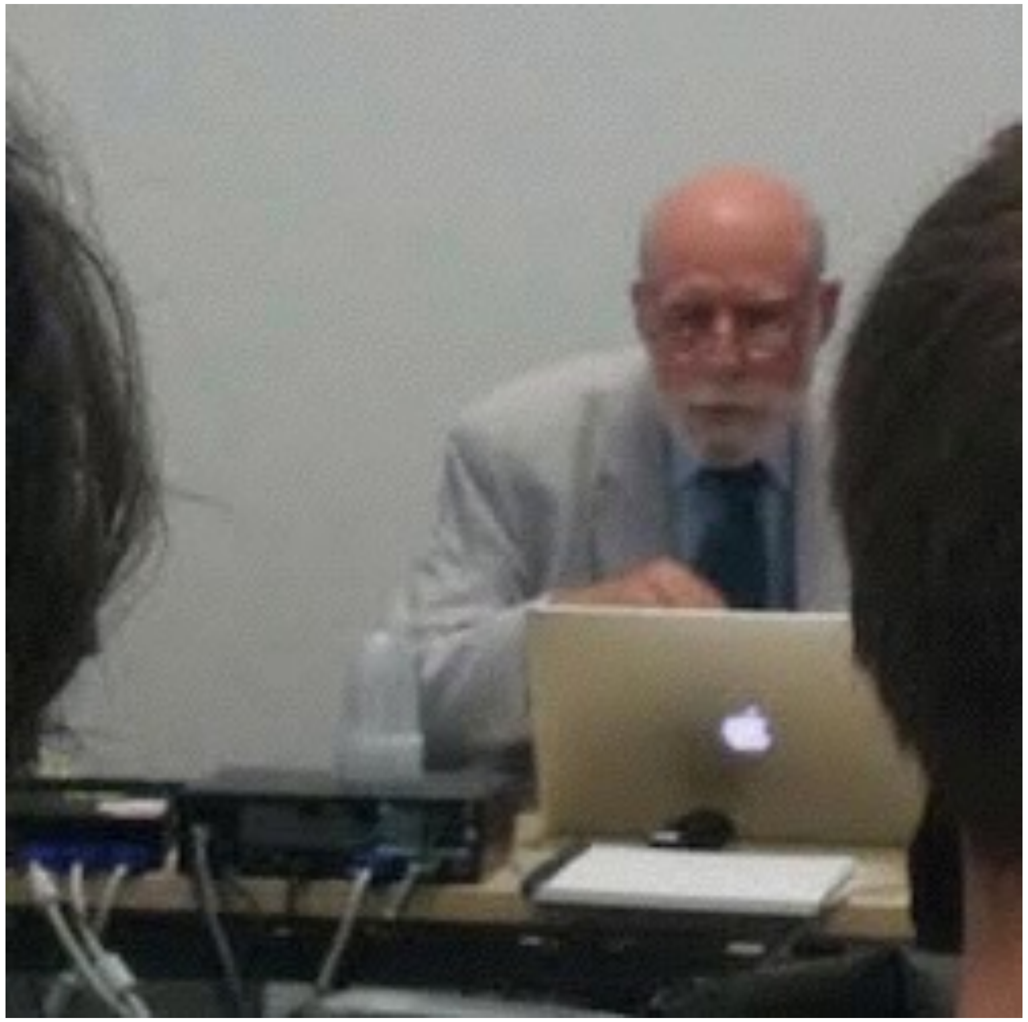
Larry S. Davis
Univ. Maryland

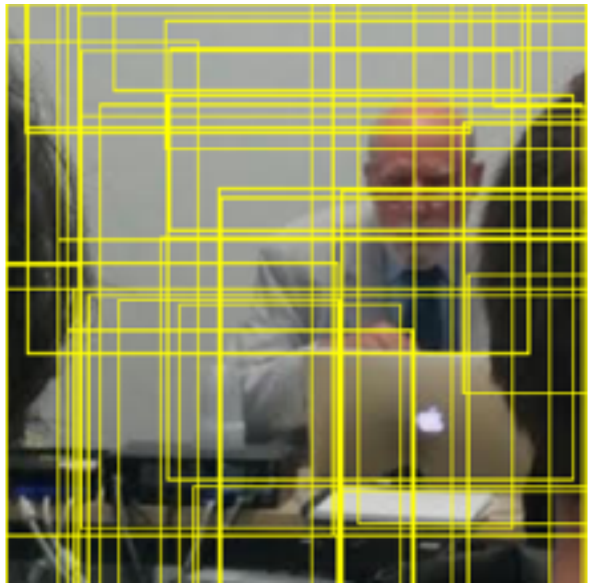
CVPR, 2016

VGG Reading Group - Sam Albanie

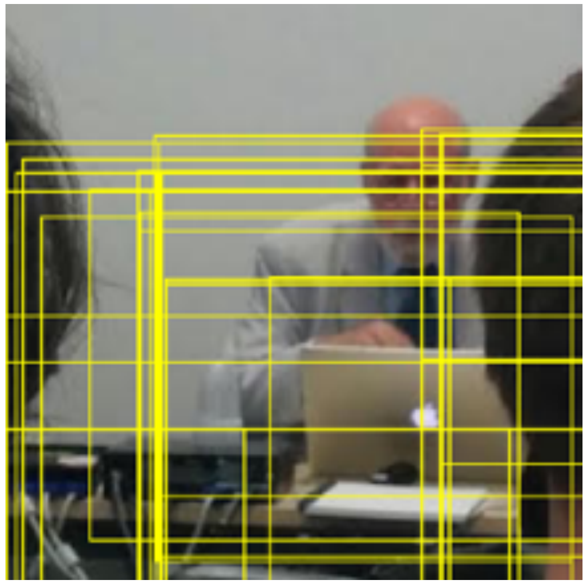
Motivation: Proposals are Expensive

Example: find the father of the internet





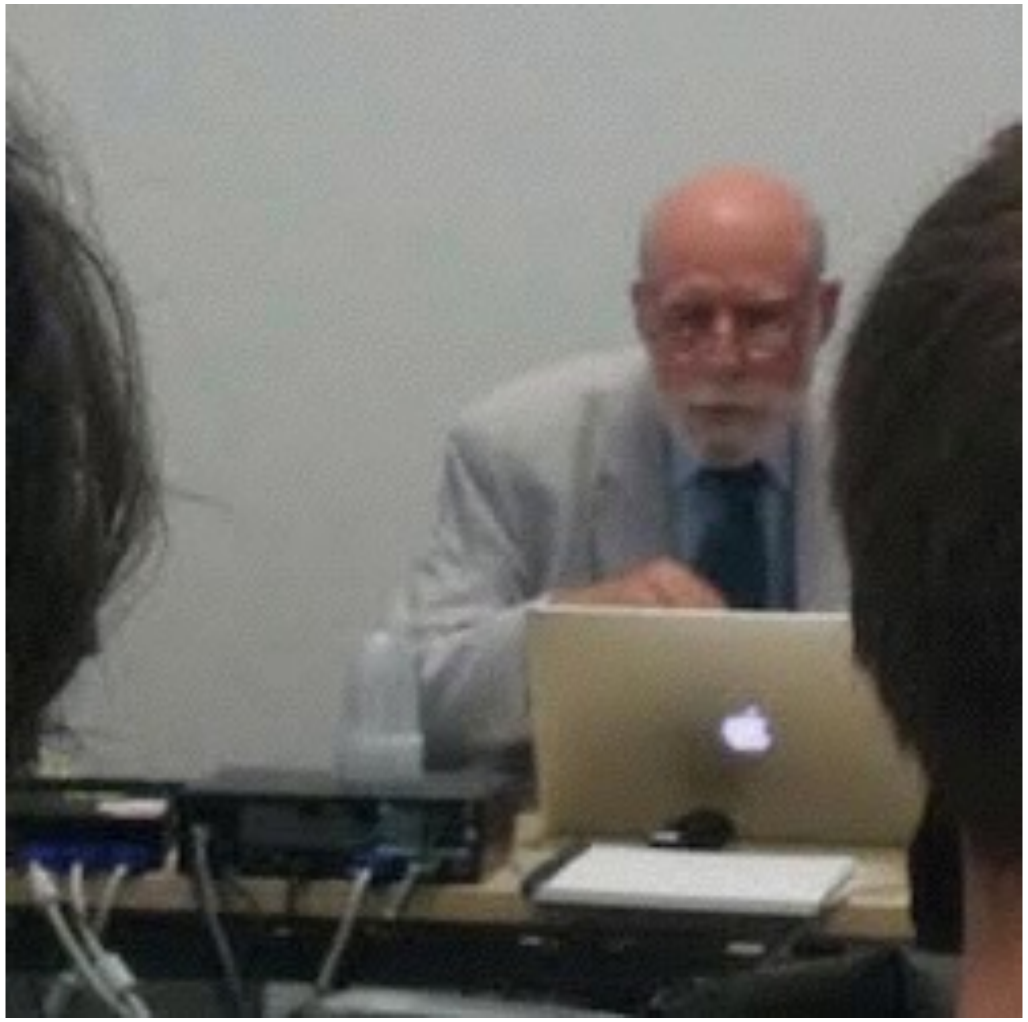
Selective Search
2.24 seconds



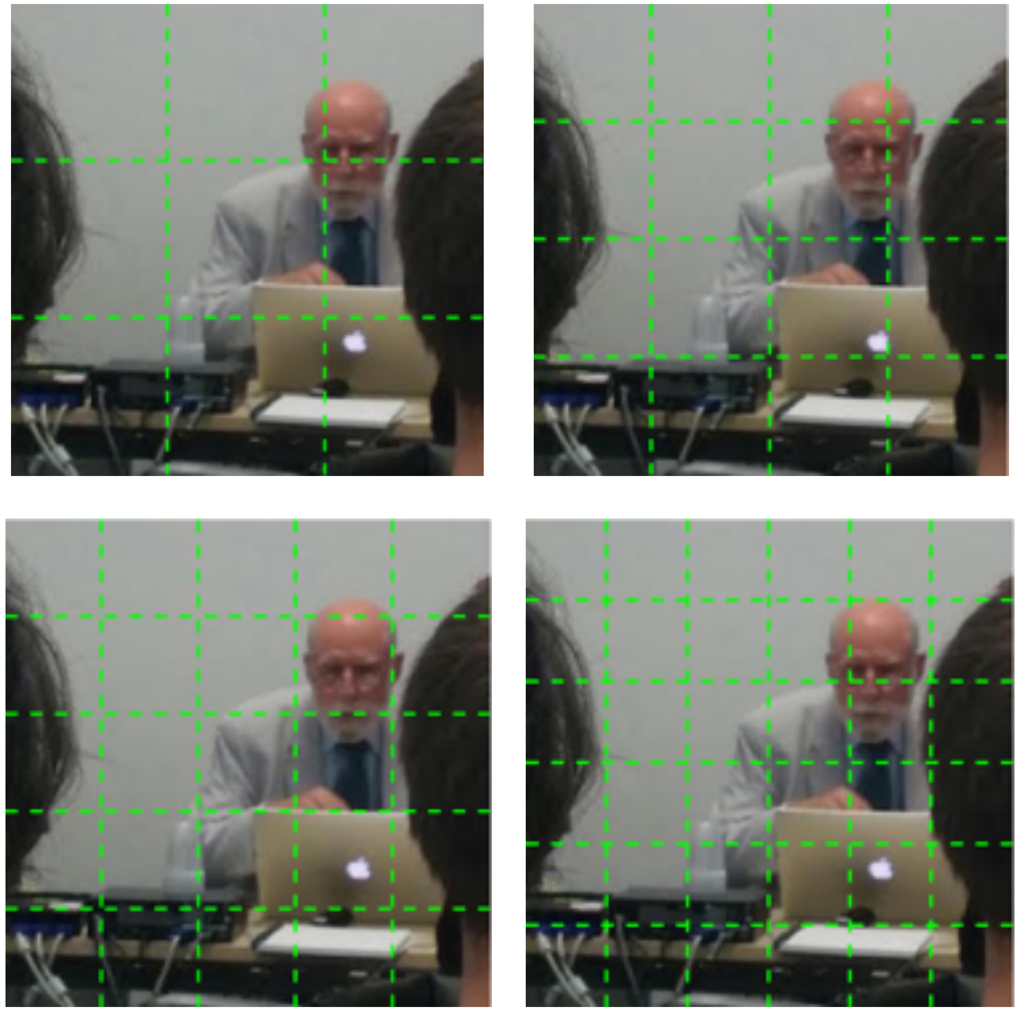
EdgeBoxes
0.38 seconds

Motivation: Proposals are Expensive

Example: find the father of the internet

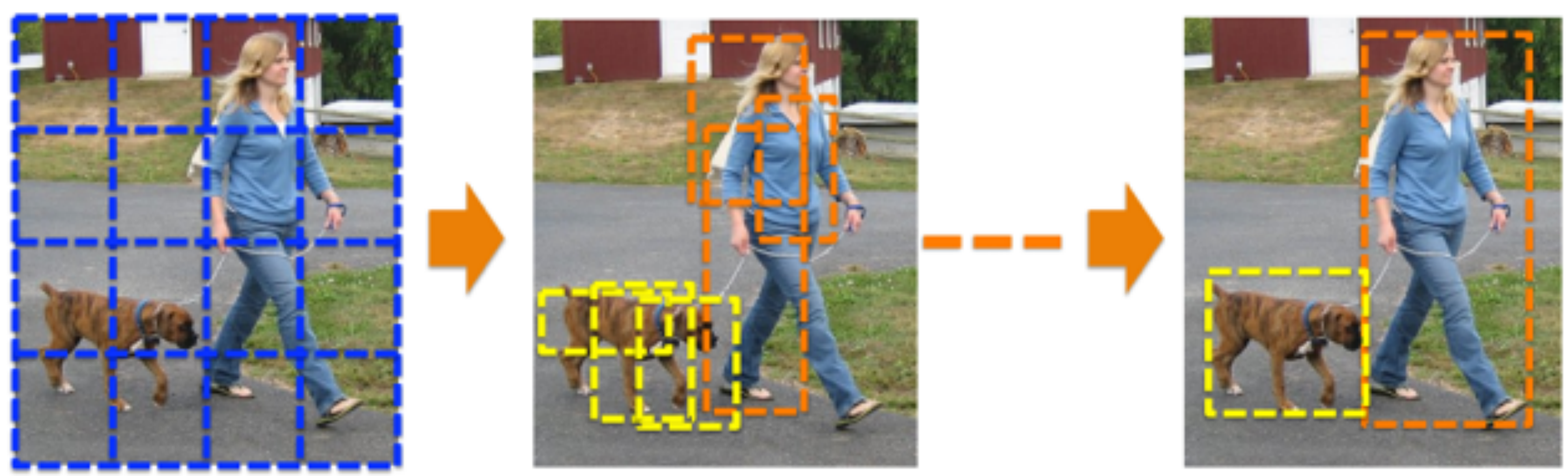


Cheaper Alternative: **grids**



Motivation: Keep accuracy with **iterations!**

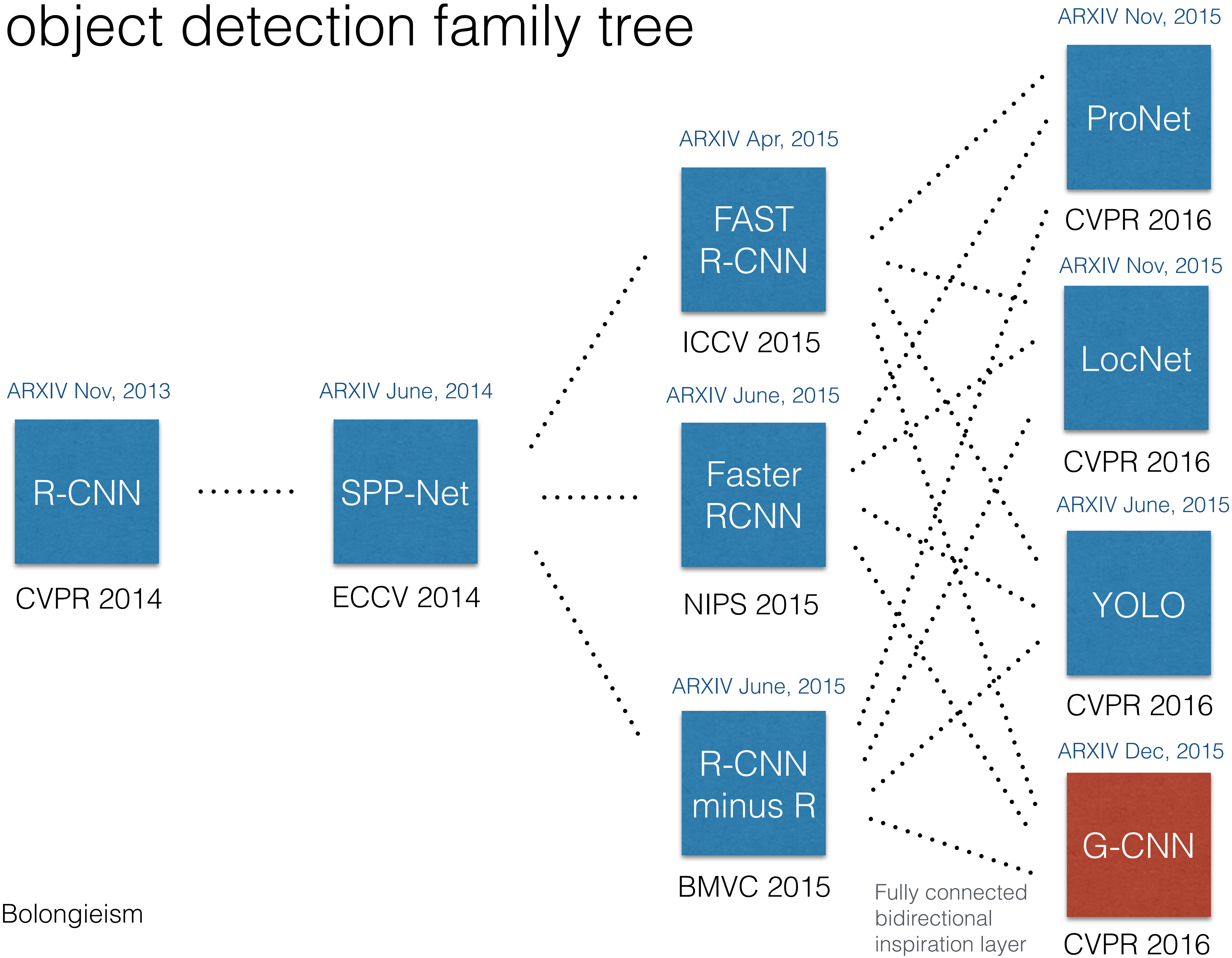
Downside of using **grids**: loss of accuracy



In G-CNN, high accuracy is achieved with grid proposals by using an **iterative** bounding box regression scheme

Inspired by IEF - move the work into the regression space!

Some members of the *postdeepluvian** object detection family tree



*Serge Bolongieism

Potentially interesting

R-CNN authors investigated iterative procedure:

“At test time, we score each proposal and predict its new detection window only once. In principle, we could iterate this procedure (i.e. re-score the newly predicted bounding box and then predict a new bounding box from it, and so on). However, we found that iterating does not improve results”

APPENDIX C - Rich Feature Hierarchies for accurate object detection and semantic segmentation

Discussion

How does it work?

Bounding Box Regression In Object Detection: Recap

Key idea: snakes are not the same shape as donkeys.
(i.e. once you have predicted the object category, you should be able to improve your bounding box)

Introduced in the DPM paper (geometric features)

Revisited in the R-CNN paper (CNN features)

Bounding Box Regression In Object Detection: R-CNN style

Training: The goal is to learn a mapping per category from a proposed box *P* to a ground truth box *G*. Inputs are *N* training pairs

$$\{(P_i, G_i)\}_{i=1,\ldots,N}, \text{ where } P^i = (P_x^i, P_y^i, P_w^i, P_h^i)$$

Parameterise mapping with linear functions $d_x(P), d_y(P), d_w(P), d_h(P)$ such that:

$$\hat{G}_x = P_w d_x(P) + P_x$$
$$\hat{G}_y = P_h d_y(P) + P_y$$

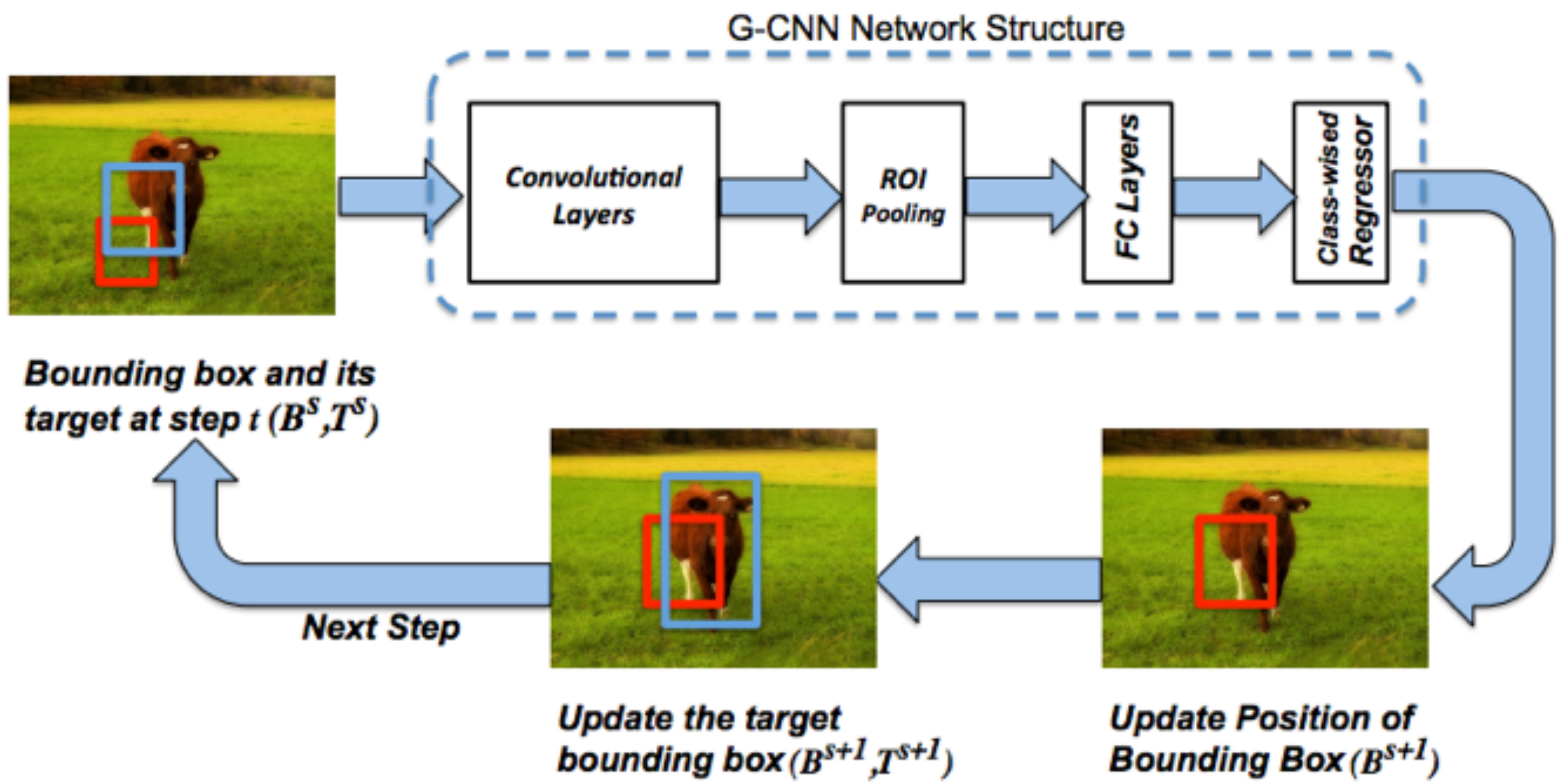
(scale invariant)

$$\hat{G}_w = P_w \exp(d_w(P)))$$
$$\hat{G}_h = P_h \exp(d_h(P)))$$

(log space)

The functions are learned with ridge regression.

Training Architecture



Notes: bounding box colours

Bounding Box Regression: The Nitty Gritty for G-CNN

Training - each bounding box with IoU > 0.2 assigned to one of ground truth boxes in the same image, based on its initial grid position.

The function is learned with piece-wise regression, using target boxes at step 1 < s < S_train

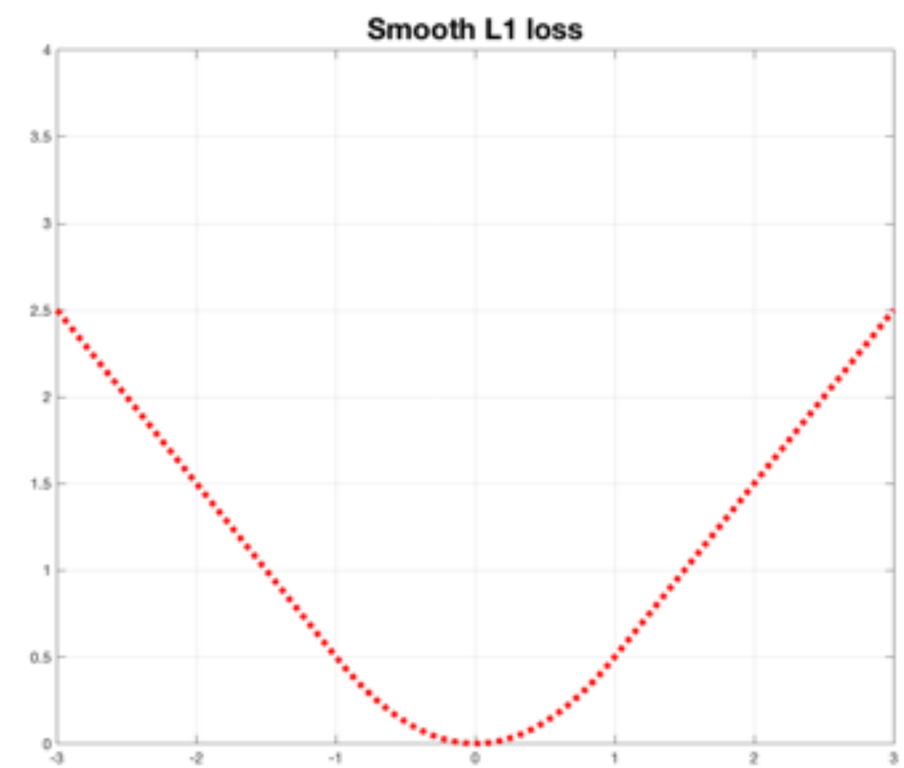
$$\Phi(\mathbf{B}_i^s, \mathbf{G}_i^*, s) = \mathbf{B}_i^s + \frac{\mathbf{G}_i^* - \mathbf{B}_i^s}{S_{\text{train}} - s + 1}$$

Bounding Box Regression: The Nitty Gritty for G-CNN

Loss function:

$$L(\{\mathbf{B}_i\}) = \sum_{s=1}^{S_{\text{train}}} \sum_{i=1}^{N} [I(\mathbf{B}_i^1 \notin \mathcal{B}_{BG}) \times L_{\text{reg}}(\delta_{i,l_i}^s - \Delta(\mathbf{B}_i^s, \Phi(\mathbf{B}_i^s, \mathcal{A}(\mathbf{B}_i^s), s)))]$$

L_reg is the smooth L1 loss from Fast R-CNN:



Bounding Box Regression: The Nitty Gritty for G-CNN

For efficiency during training, approximate *predicted update*

$$\mathbf{B}_i^s = \mathbf{B}_i^{s-1} + \boldsymbol{\Delta}^{-1}(\delta_{i,l_i}^{s-1})$$

with *the perfect update*

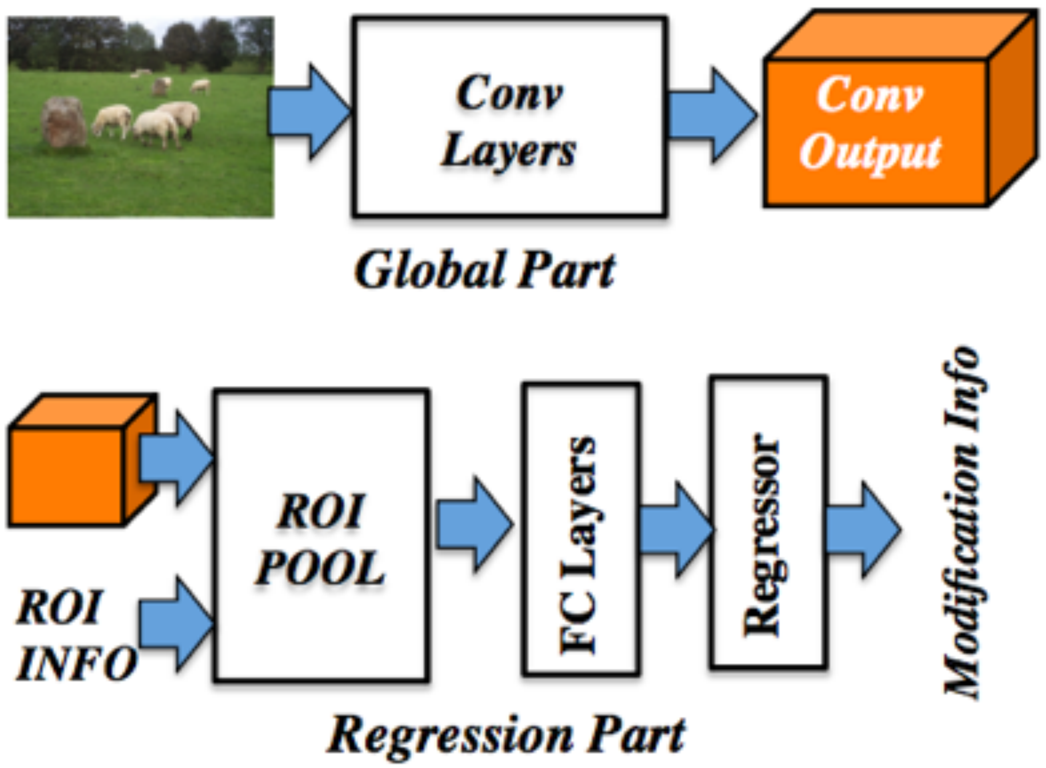
$$\mathbf{B}_i^s = \Phi(\mathbf{B}_i^{s-1}, \mathbf{G}_i^*, s-1)$$

Optimisation

SGD ftw.

Note: sampling biases early iteration steps

Test-time architecture



Comparison to R-CNN: *N_proposal* vs *(S_test x N_grid)*

Demo

0
50
100
150
200
250
300
0
100
200
300
400
www.pferdefotoarchiv.de

Experiments

Config

2x2 5x5 10x10

Training overlaps:
[0.9, 0.8, 0.7]

Test overlaps:
[0.7, 0.5, 0]

Regression network is trained for S = 3 steps

Experiment 1:VOC 2007

VOC 2007	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
FR-CNN [6]	**66.4**	**71.6**	**53.8**	43.3	24.7	69.2	**69.7**	**71.5**	31.1	**63.4**	59.8	62.2	73.1	65.9	**57**	26	**52**	56.4	67.8	**57.7**	57.1
G-CNN(3) [ours]	63.2	68.9	51.7	41.8	**27.2**	69.1	67.7	69.2	31.8	60.6	**60.8**	63.9	**75.5**	67.3	54.9	26.1	51.2	**57.2**	69.6	56.8	56.7
G-CNN(5) [ours]	65	68.5	52	**44.9**	24.5	**69.3**	69.6	68.9	**34.6**	60.3	58.1	**64.6**	75.1	**70.5**	55.2	**28.5**	50.7	56.8	**70.2**	56.1	**57.2**

Each network was based on Alexnet, trained on VOC 2007 *trainval* set and evaluated on the *test* set.

FR-CNN := One step at test time + approx 2000 initial boxes (SS)

G-CNN(3) := Three steps at test time + approx 1500 initial boxes

G-CNN(5) := Five steps at test time + approx 180 initial boxes

Experiment 2:VOC 2007

VOC 2007	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
SPPnet BB[9]	73.9	72.3	62.5	51.5	44.4	74.4	73.0	74.4	42.3	73.6	57.7	70.3	74.6	74.3	54.2	34.0	56.4	56.4	67.9	**73.5**	63.1
R-CNN BB[8]	73.4	77.0	63.4	45.4	**44.6**	75.1	78.1	79.8	40.5	**73.7**	62.2	79.4	78.1	73.1	64.2	**35.6**	**66.8**	67.2	70.4	71.1	66.0
FR-CNN[6]	**74.5**	**78.3**	**69.2**	**53.2**	36.6	77.3	78.2	**82.0**	40.7	72.7	**67.9**	**79.6**	79.2	73.0	**69.0**	30.1	65.4	**70.2**	**75.8**	65.8	**66.9**
G-CNN[ours]	68.3	77.3	68.5	52.4	38.6	**78.5**	**79.5**	81	**47.1**	73.6	64.5	77.2	**80.5**	**75.8**	66.6	34.3	65.2	64.4	75.6	66.4	66.8

Each network was based on VGG-16, trained on VOC 2007 *trainval* set and evaluated on the *test* set.

Claim: *G-CNN effectively moves small # of boxes to targets*

Experiment 3:VOC 2012

VOC 2012	train	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
R-CNN BB[8]	12	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3	62.4
YOLO[16]	12	71.5	64.2	54.1	35.3	23.3	61.0	54.4	78.1	35.3	56.9	40.9	72.4	68.6	68.0	62.5	26.0	51.9	48.8	68.7	47.2	54.5
FR-CNN[6]	12	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	65.7
FR-CNN[6]	07++12	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	68.4
G-CNN [ours]	12	82	74	68.2	49.5	38.9	74.4	68.9	85.4	40.6	70.9	50	85.5	77	77.4	67.9	33.7	67.6	60	77.6	60.8	65.5
G-CNN [ours]	07+12	82	76.1	69.3	49.9	40.1	75.2	69.5	86.3	42.3	72.3	50.8	84.7	77.8	77.2	68	38.1	68.4	59.8	79.1	61.9	66.4*

Each network was based on VGG-16 with the following training

12 := VOC2012 *trainval*,

07+12 := VOC2007 *trainval* + VOC2012 *trainval*

07++12 := VOC2007 *trainval/test* + VOC2012 *trainval*

Claim: *G-CNN provides best mAP without a proposal stage*

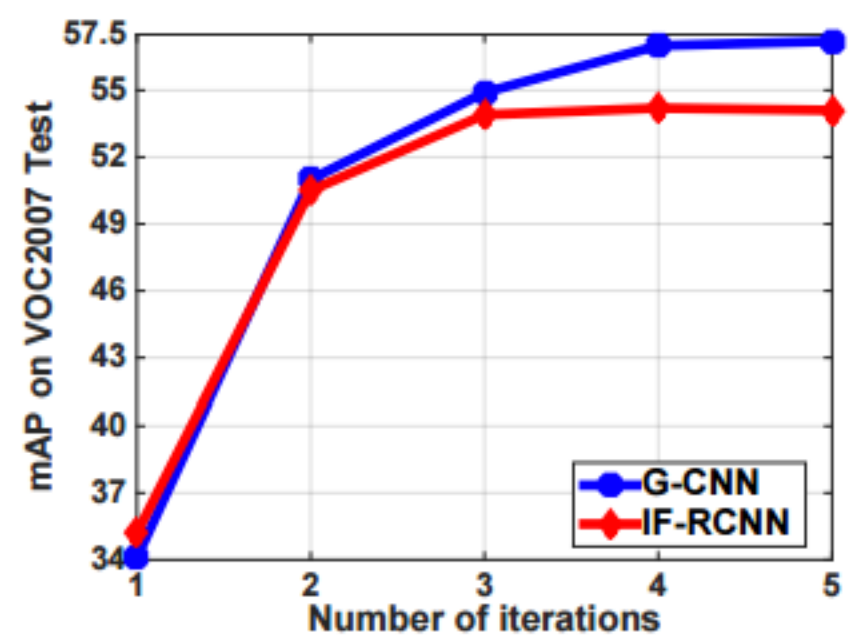
Experiment 4:VOC 2007

VOC 2007	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
IF-RCNN	51.3	67.1	51.6	33.7	**26.2**	67.8	66.3	**70.3**	31.5	56.3	55.9	62.6	74.7	64.6	**55.6**	22.2	46.5	54.3	67.4	55	54.1
1Step-Grid	59.6	63.3	**52.4**	40.2	20.9	68.1	67.1	68.6	29.7	59.6	**62.1**	63	70.7	64	53.2	23.4	50.1	56	63.5	53.9	54.5
G-CNN [ours]	**65**	**68.5**	52	**44.9**	24.5	**69.3**	69.6	68.9	**34.6**	**60.3**	58.1	**64.6**	**75.1**	**70.5**	55.2	**28.5**	**50.7**	**56.8**	**70.2**	**56.1**	**57.2**

Each network was based on Alexnet, trained on VOC 2007 *trainval* set and evaluated on the *test* set, with five steps at test time

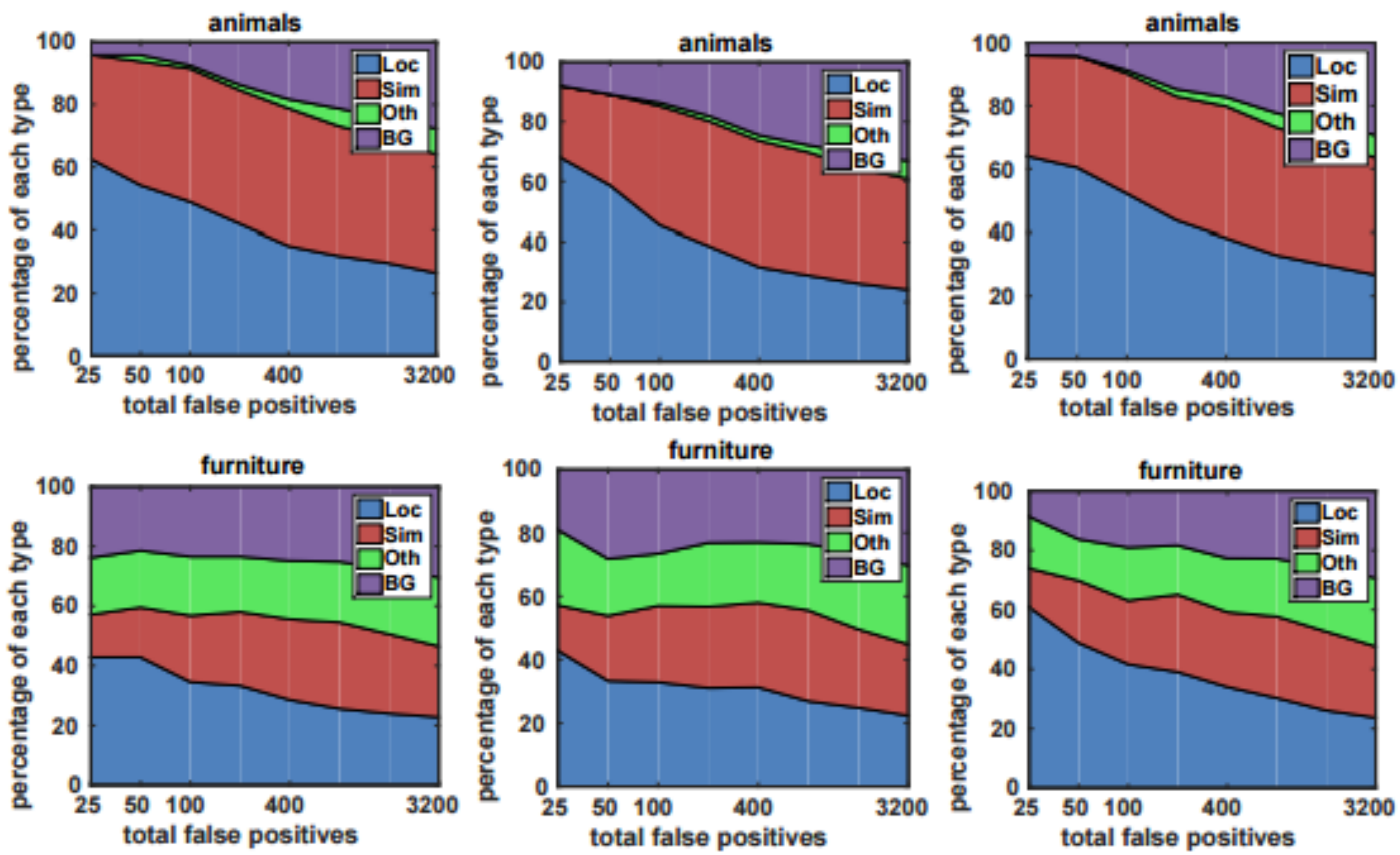
IF-FRCNN := Apply FR-CNN iteratively

1Step-Grid := Train G-CNN with all tuples in one step



Claim: *Stepwise training matters*

Analysis of Detection Results



(a) G-CNN (b) Fast R-CNN (c) 1Step-Grid

Claim: *Removing proposal stage did not hurt localisation*

Detection Run Time

Benchmarks with two K40 GPUs with VGG16 Net

Fast R-CNN: 0.5 fps

G-CNN: 3 fps

Rough comparison with current state of the art (VOC 2007 test set)

Different training sets give an idea of how well the model scales with additional data.
Table compiled July 2016

Model	Training	Speed (juice)	mAP
G-CNN	07	3fps (2xK40)	66.8
Faster R-CNN	07+12	5fps (K40)	73.2
SSD-300	07+12	58fps (TITAN X)	72.1
SSD-500	07+12	23fps (TITAN X)	75.1
R-FCN	07+12	6fps (K40)	80.5
Faster R-CNN	07+12+CO	5fps (K40)	85.6
R-FCN	07+12+CO	6fps (K40)	83.6

NOTE: By the time you are reading this, it is probably out of date…