Object Detection and Faster RCNN
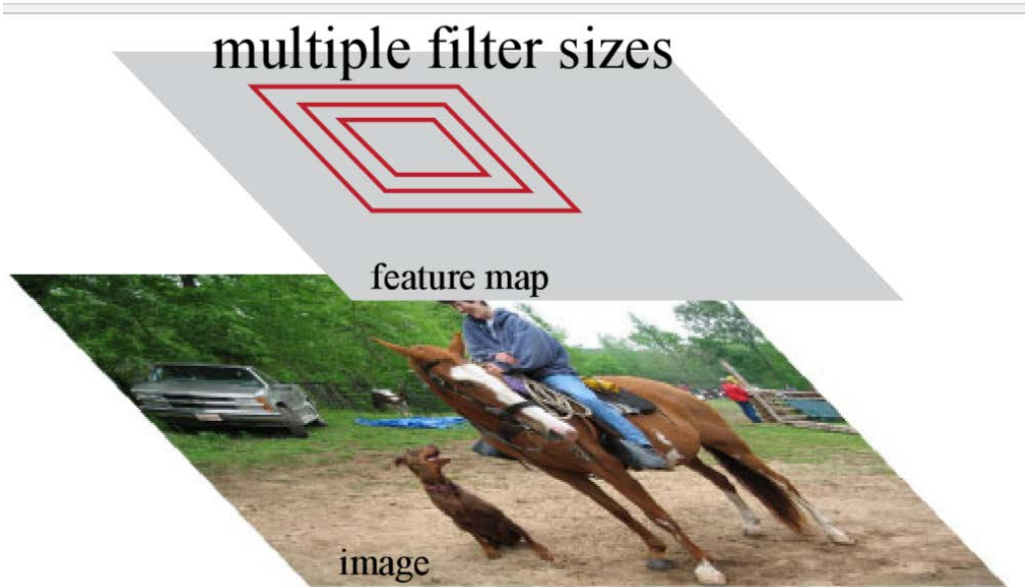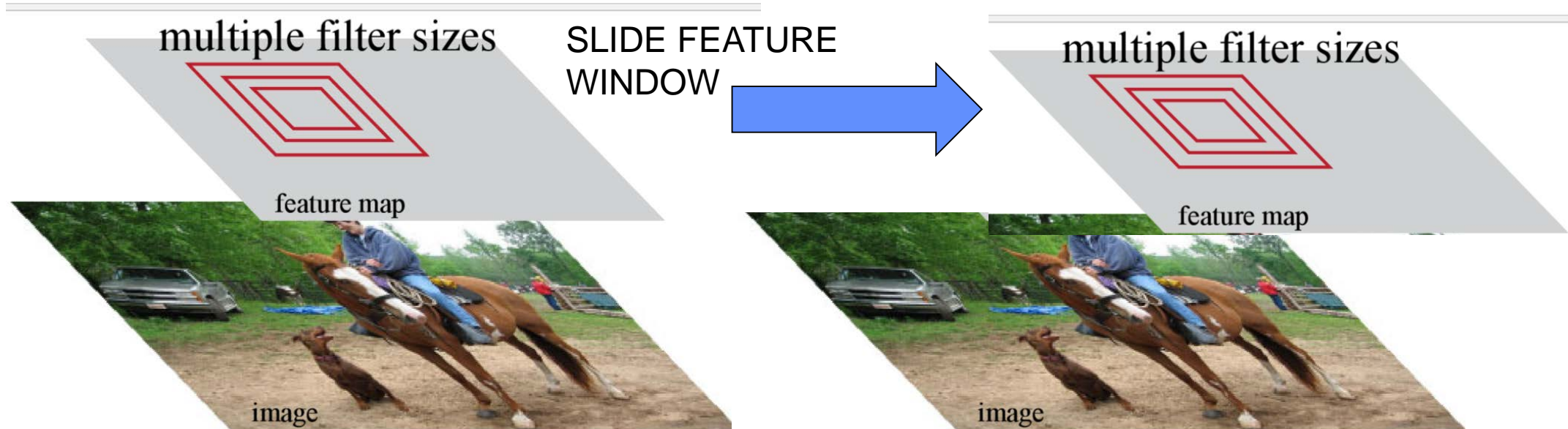## Paul Rodriguez SDSC

How to find object instances of different sizes across images?

Combination of image resizing, different filter sizes, and sliding windows helps find objects
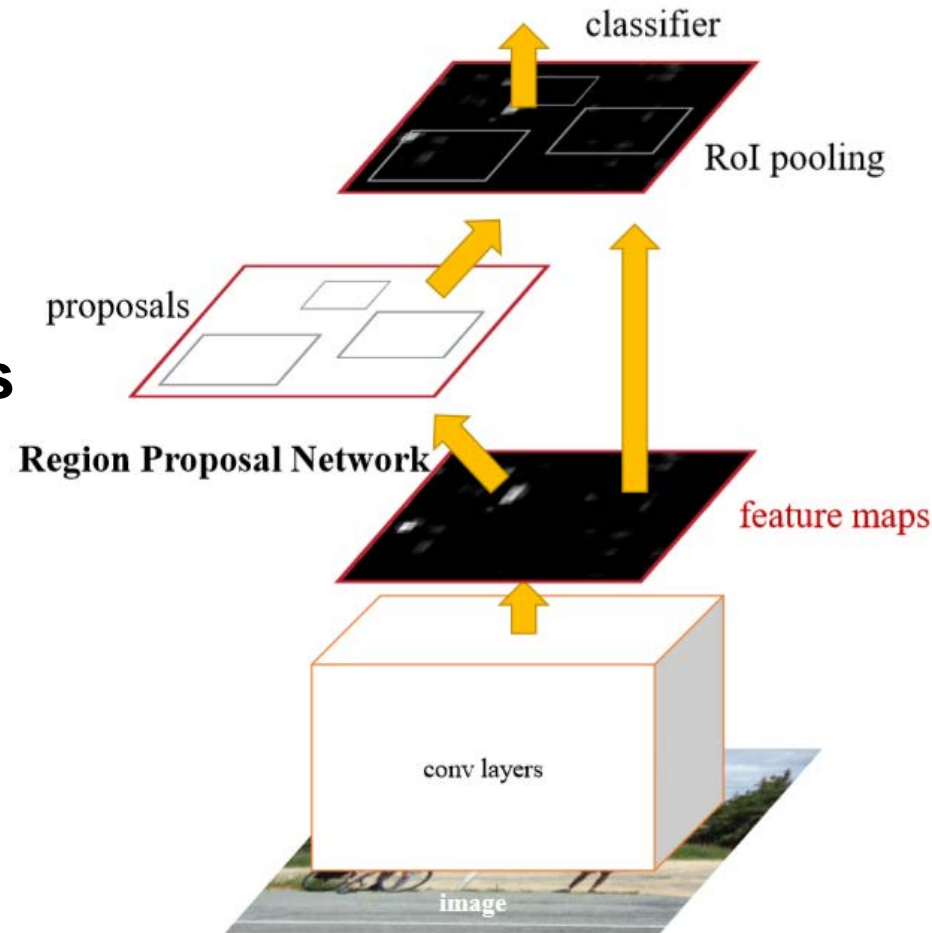
Combination of image resizing, different filter sizes, and sliding windows helps find objects

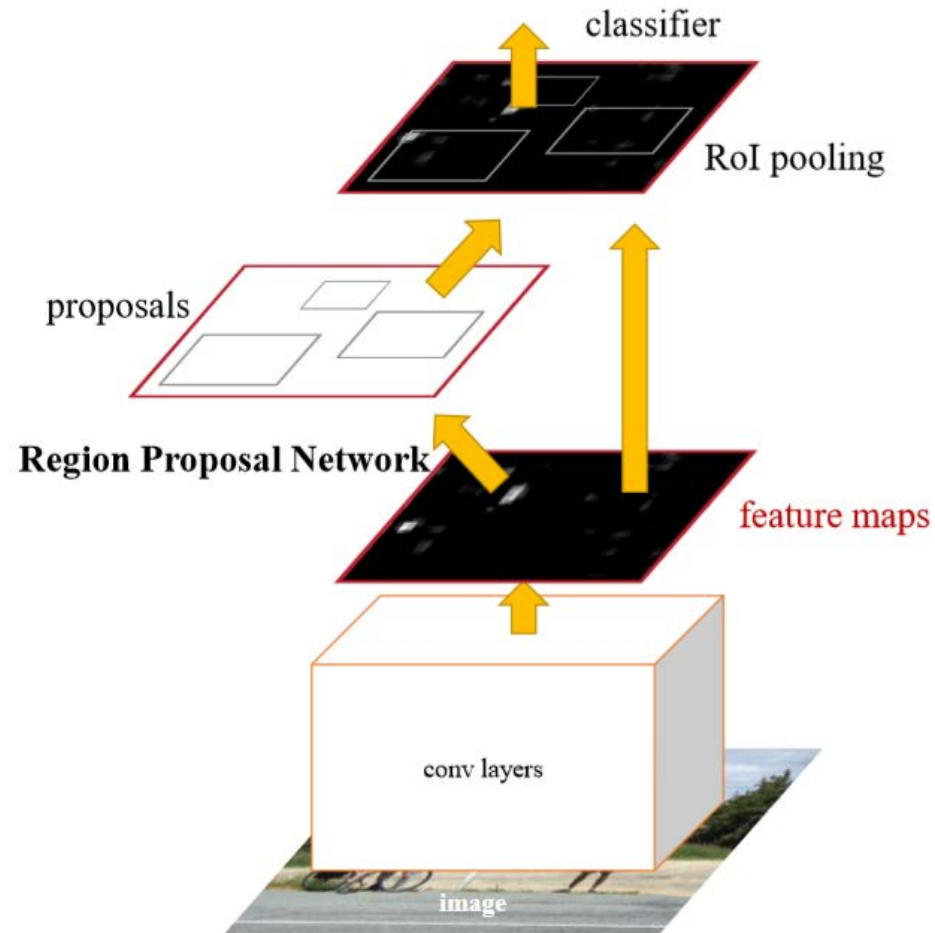# Or, use CNN to classify and detect: Faster RCNN
(Ren,He,Girschick,Sun)

A side path to select Regions

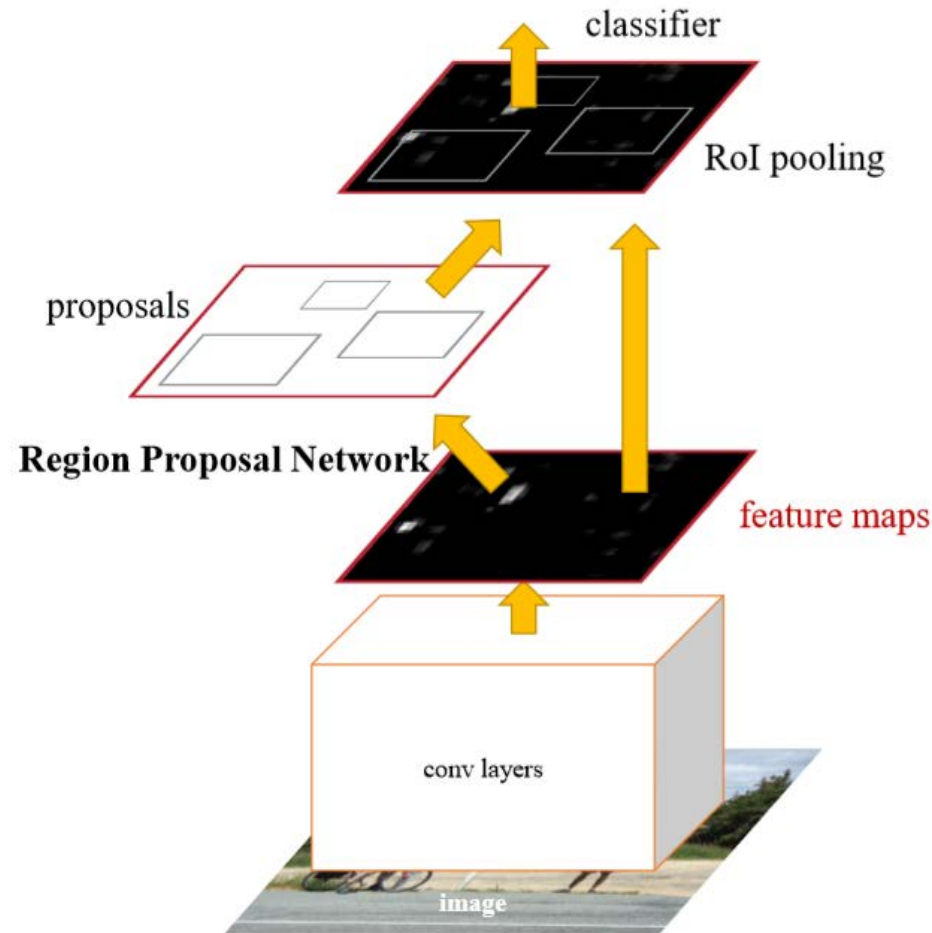# Or, use CNN to classify and detect: Faster RCNN
(Ren,He,Girschick,Sun)

Images are normalized and resized

Or, use CNN to classify and detect: Faster RCNN
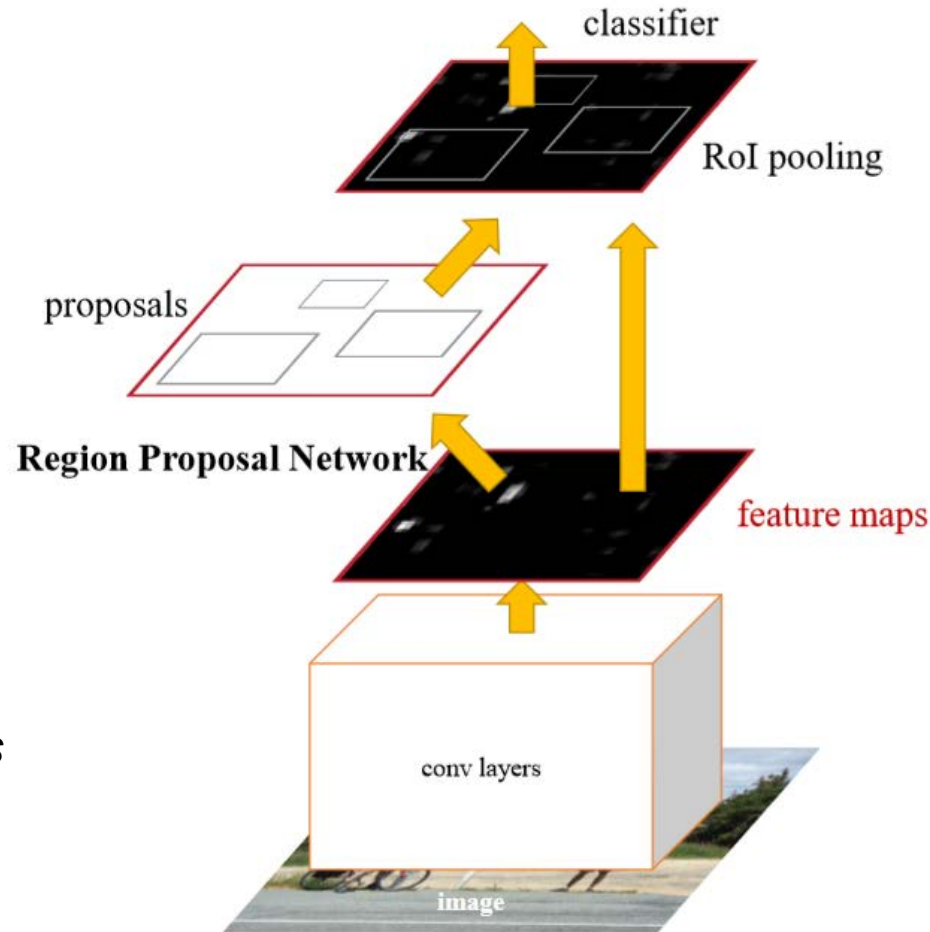## (Ren,He,Girschick,Sun)

Pool with adjustable windows to transform into one size vector for classification

Images are normalized and resized

Or, use CNN to classify and detect: Faster RCNN
(Ren,He,Girschick,Sun)

1. Start with 2000 sampled regions; segment, group, get texture of possible foreground regions

2. Then use feature map values and a model to predict if an object (of any class) would be detected in each window.
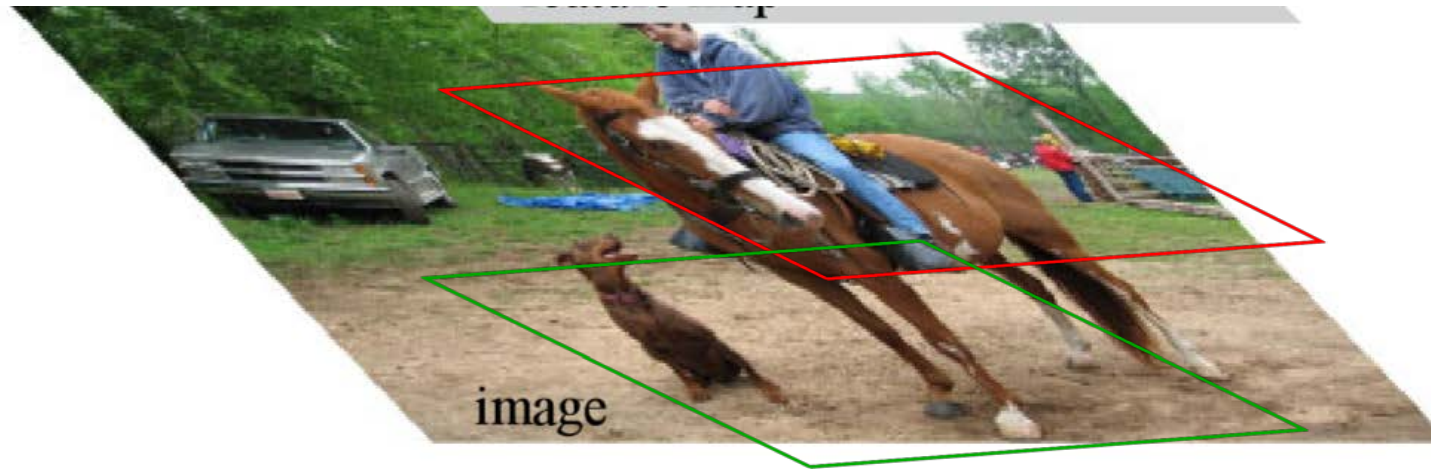
3. Pass maps for best regions to classifier

Pool with adjustable windows to transform into one size vector for classification

Images are normalized and resized

Training Data

- **Given positive sample, generate negative samples (and balance numbers)**

Sample and add boxes with < 40% overlap as 'negative', > 50% as positive

Training Data

- Given positive sample, generate negative samples

(and balance numbers)

Train region proposal and classifier alternatively in 4 phases
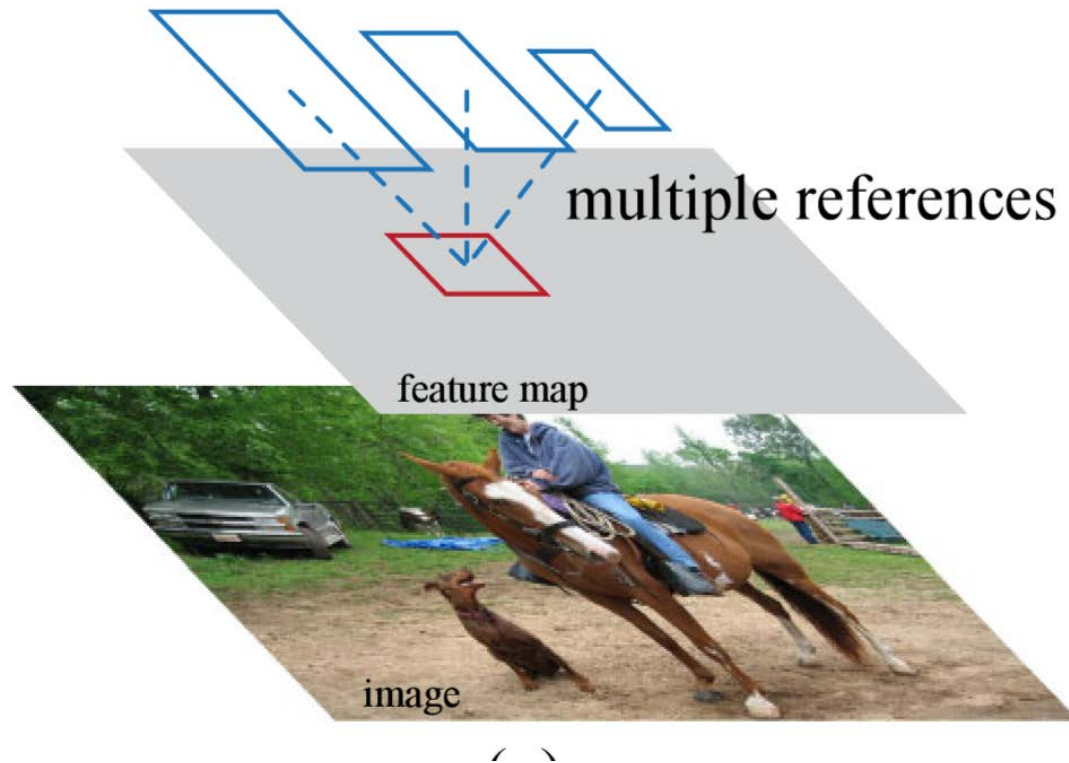
Phases:
      1,3 region proposal
      2,4 classification

Region output

- **Output bounding box information (box center, height, width)**

multiple references

feature map
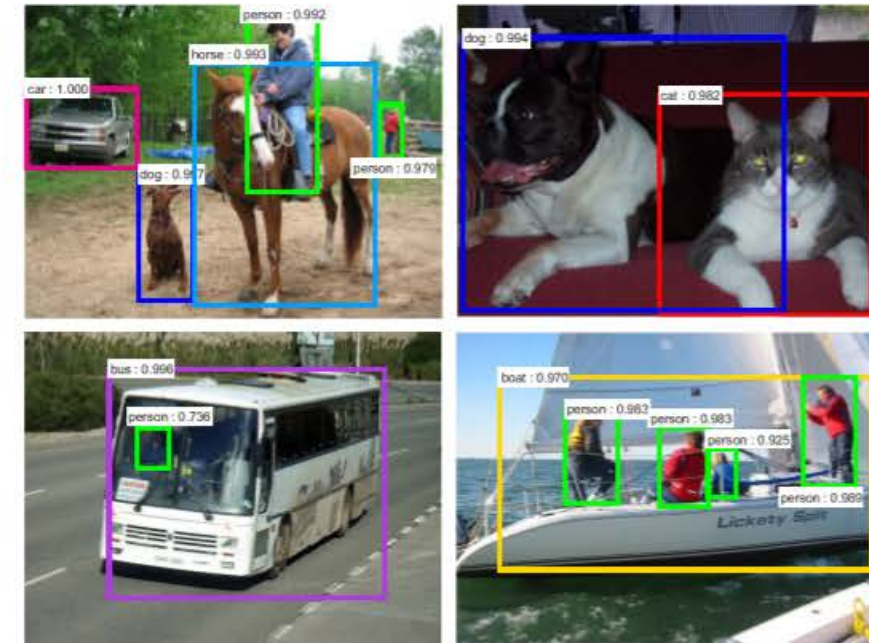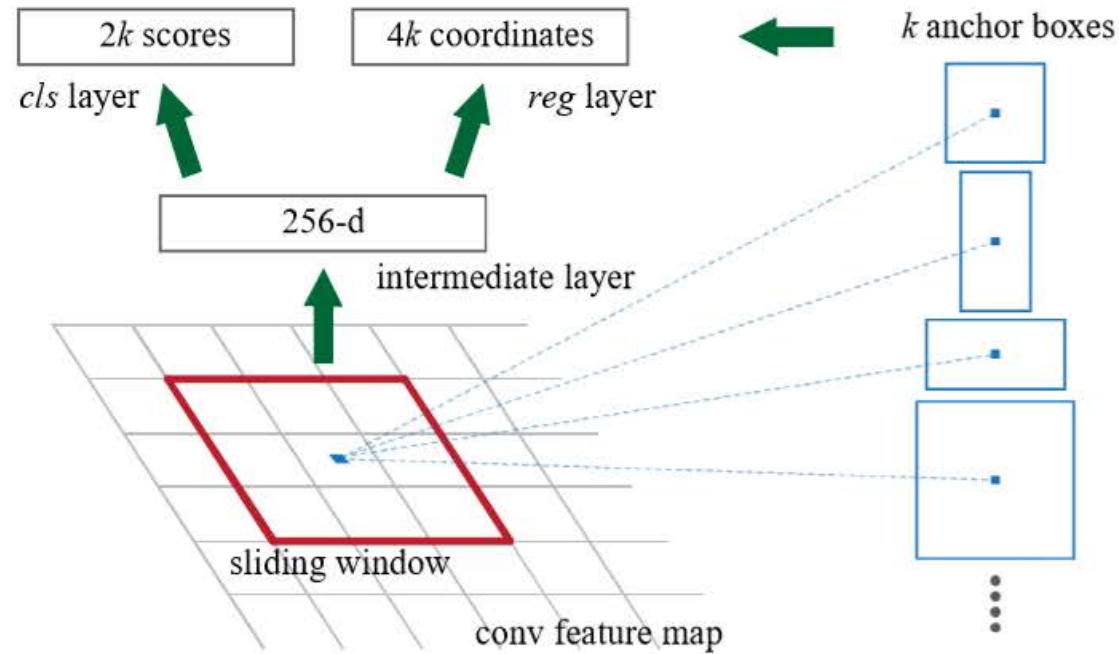
image

Region Box size ranges

Figure 3: **Left**: Region Proposal Network (RPN). **Right**: Example detections using RPN proposals on PASCAL VOC 2007 test. Our method detects objects in a wide range of scales and aspect ratios.

Samples of 'hand' training data

Person boxes (using YOLO in 'darknet' code)

Face detection using 'DLIB' library

Hand boxes were hand drawn(Mittal etal.)
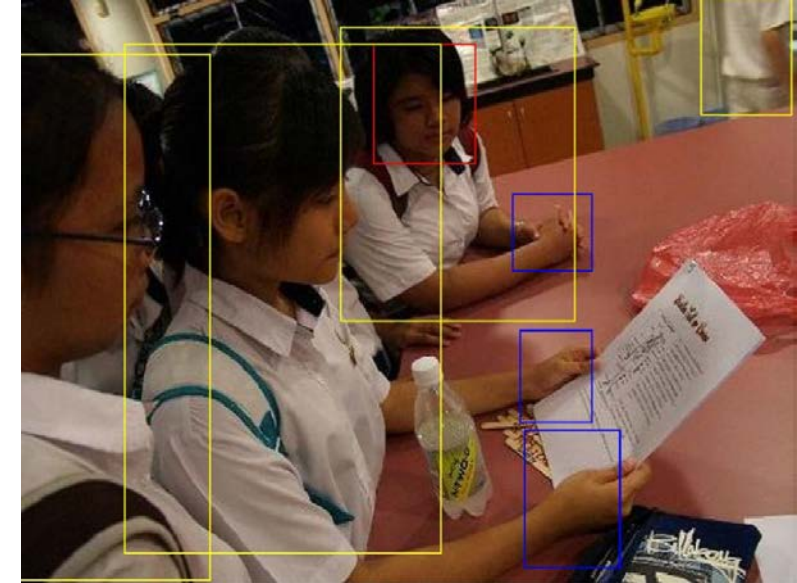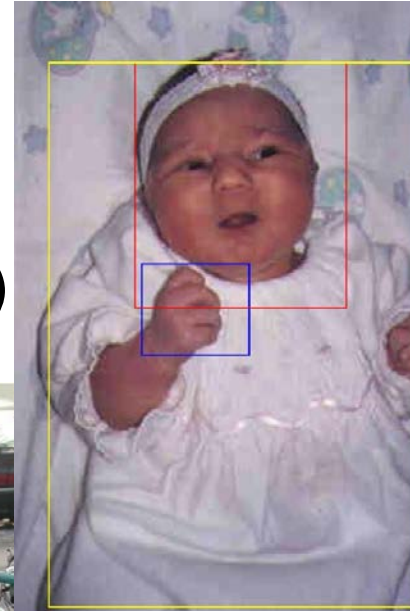
Matlab Implementation

detector = trainFasterRCNNObjectDetector(

trainingData, layers, options, ...

Table of ~4K image file names and ~8K boxes

A defined set of layers

learning rate, epochs
.0001 phase1,2
.00001 phase 3,4
15 epochs

Matlab Implementation

```
detector = trainFasterRCNNObjectDetector(
                    trainingData, layers, options, ...
        'NegativeOverlapRange', [0.1 0.4], ...
        'PositiveOverlapRange', [0.6 1], ...
        'NumStrongestRegions',200,…
        'BoxPyramidScale',1.2,...
        'NumBoxPyramidLevels',5);
```

For making pos/neg samples

For selecting object (anchor) boxes

Matlab Faster RCNN

- **My parameters:**

stride=2

128x128 input (resized)

5x5 convolution

Matlab Faster RCNN

- **My parameters:**

classification output:
hand , 'background'

Max pooling

2 convolution
layers

32 filters
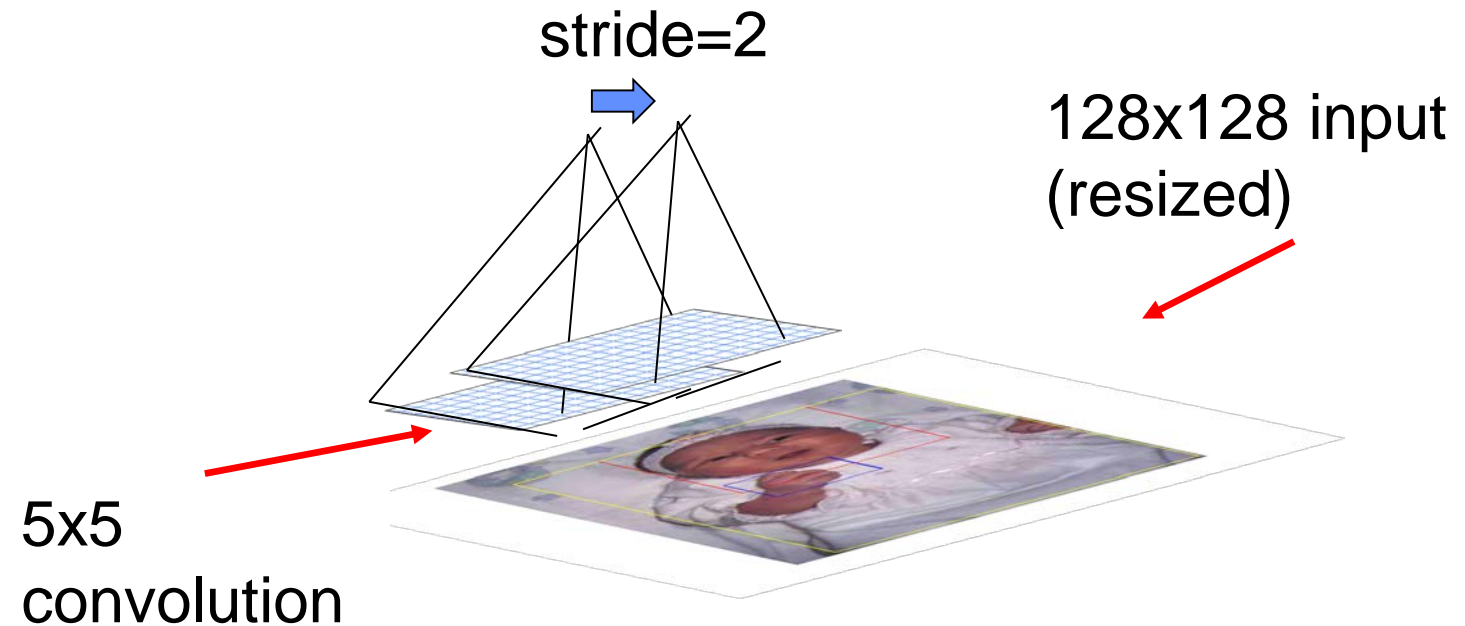
32 filters

Network detail

>> detector.Network.Layers

11x1 Layer array with layers:

```
 1   'imageinput'        Image Input          128x128x3 images with 'zerocenter' normalization
 2   'conv_1'            Convolution          32 5x5x3 convolutions with stride [2  2] and padding [1  1  1  1]
 3   'relu_1'            ReLU                 ReLU
 4   'conv_2'            Convolution          32 5x5x32 convolutions with stride [2  2] and padding [1  1  1  1]
 5   'relu_2'            ReLU                 ReLU
 6   'roi pooling layer' ROI Max Pooling      ROI Max Pooling with grid size [14 14]
 7   'fc_1'              Fully Connected      32 fully connected layer
 8   'relu_3'            ReLU                 ReLU
 9   'fc_2'              Fully Connected      2 fully connected layer
10   'softmax'           Softmax              softmax
11   'classoutput'       Classification Output   crossentropyex with classes 'hand' and 'Background'
```

Results (so far)

- **Using 50% overlap with true box as correct ~25% TP rate**
- **4-8 hours on 1 compute node (CPU) 15 epochs ~4K images**
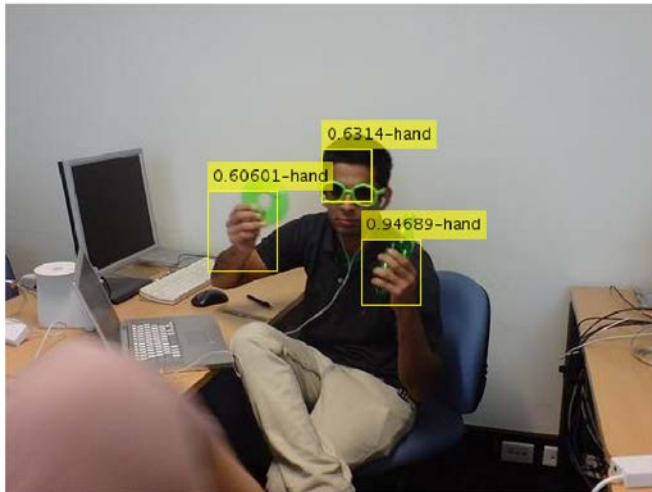
Matlab Implementation

detector = trainFasterRCNNObjectDetector(

trainingData, vgg19, options, ...

Table of ~1K image file names and ~2K boxes

A pretrained network from Vis. Geom. Group at Oxford

learning rate, epochs
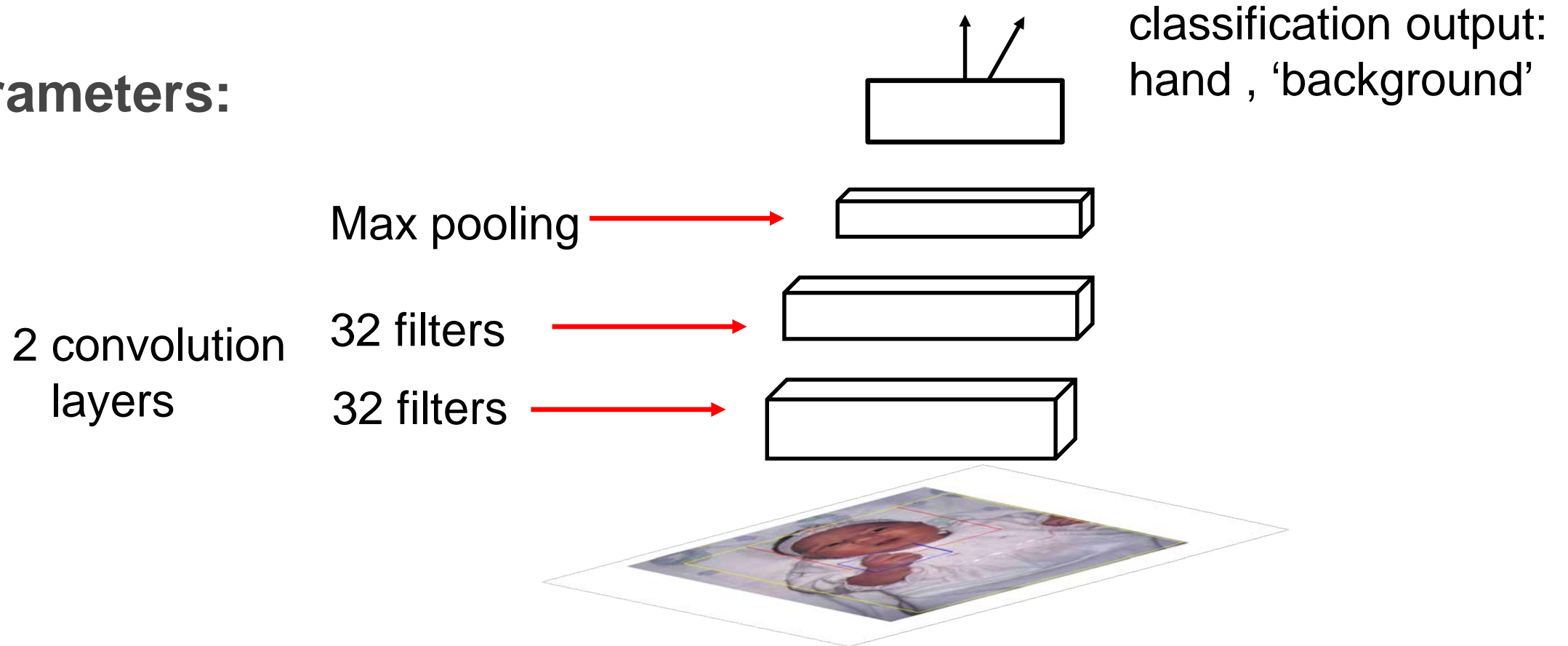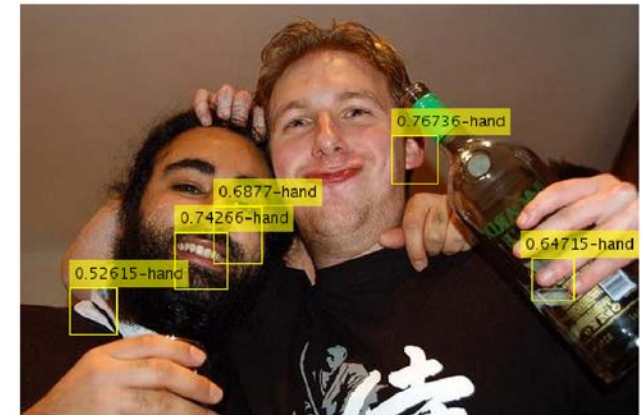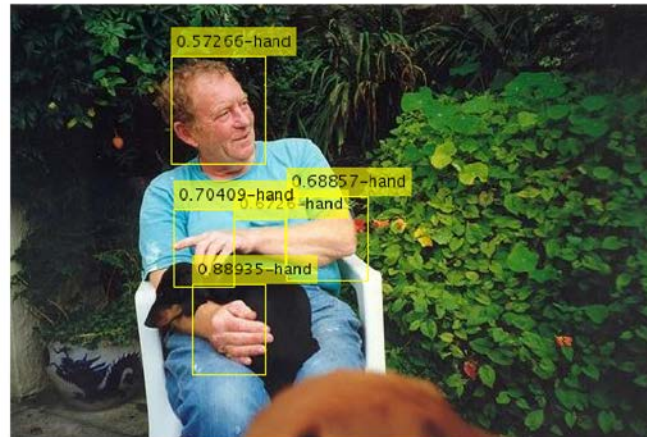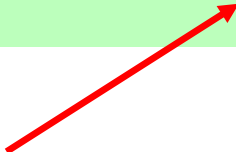*.00001   phase1,2*
.000001 phase 3,4
5 epochs

detector.Network.Layers(1:7)

1 'input'   Image Input   224x224x3 images with 'zerocenter' normalization
2 'conv1_1' Convolution   64 3x3x3 convolutions with stride [1 1] and padding [1 1 1 1]
3 'relu1_1' ReLU   ReLU
4 'conv1_2' Convolution   64 3x3x64 convolutions with stride [1 1] and padding [1 1 1 1]
5 'relu1_2' ReLU   ReLU
6 'pool1'   Max Pooling   2x2 max pooling with stride [2 2] and padding [0 0 0 0]
7 'conv2_1' Convolution   128 3x3x64 convolutions with stride [1 1] and padding [1 1 1 1]

……

detector.Network.Layers(35:47)

1 'relu5_3'   ReLU   ReLU
2 'conv5_4'   Convolution   512 3x3x512 convolutions with stride [1 1] and padding [1 1 1 1]
3 'relu5_4'   ReLU   ReLU
4 'roi pooling layer' ROI Max Pooling   ROI Max Pooling with grid size [7 7]
5 'fc6'   Fully Connected   4096 fully connected layer
6 'relu6'   ReLU   ReLU
7 'drop6'   Dropout   50% dropout
8 'fc7'   Fully Connected   4096 fully connected layer
9 'relu7'   ReLU   ReLU
10 'drop7'   Dropout   50% dropout
11 'fc_detection'   Fully Connected   2 fully connected layer
12 'softmax'   Softmax   softmax
13 'classoutput'   Classification Output   crossentropyex with classes 'hand' and 'Background'

Using VGG19 network

Results so far

- **Using 50% overlap with true box as correct ~50% TP rate**
- **4-8 hours on 1 compute node (CPU) 5 epochs on ~1K images**

In Summary

faster RCNN get near 60% TP rate

(**https://pjreddie.com/media/files/papers/YOLOv3.pdf**)

proposing regions takes much time

Matlab fasterRCNN easy to use but could use more options

(like turning off weight changes for transfer RCNN learning, reading in images faster, etc..)