Introduction to Deep Learning for Facial Understanding

Part III: Regional CNNs

Raymond Ptucha,
Rochester Institute of Technology, USA

www.nvidia.com/dli

Tutorial-9
May 19, 2018

The 13th IEEE Conference on Automatic Face and Gesture Recognition
FG 2018 Xi'an, China | 15-19 May 2018

Fair Use Agreement

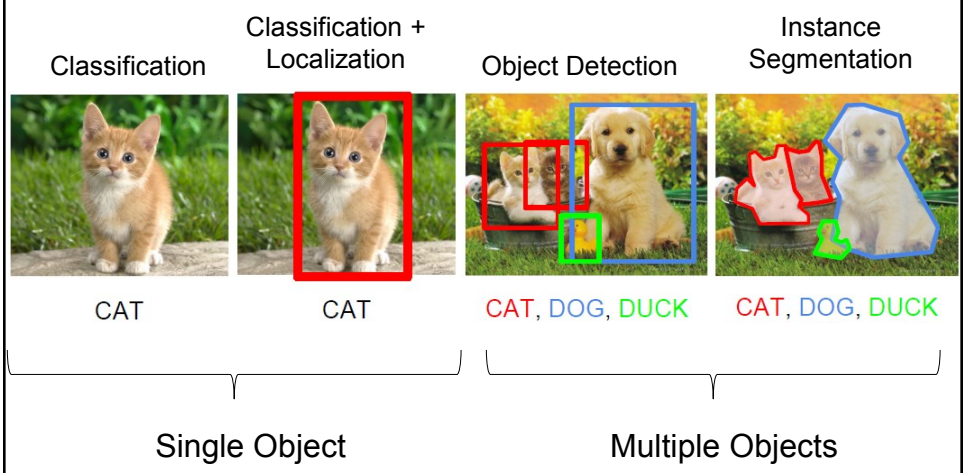
This agreement covers the use of all slides in this document, please read carefully.

- You may freely use these slides, if:
  - You send me an email telling me the conference/venue/company name in advance, and which slides you wish to use.
  - You receive a positive confirmation email back from me.
  - My name (R. Ptucha) appears on each slide you use.

Agenda

- 9-9:30am Part I: Introduction
- 9:30-10:00am Part II: Convolutional Neural Nets
- **10:00-10:30am Part III: Fully Convolutional Nets**
- 10:30-10:45am Break
- 10:45-11:15am Part IV: Facial Understanding
- 11:15-11:35am Part V: Recurrent Neural Nets
- 11:35-11:55pm Part VI: Generative Adversarial Nets
- 11:55-12:30am Hands-on with NVIDIA DIGITS

Vision Tasks



Classification vs. Classification + Localization

Classification
- **Input**: Image
- **Output**: Class label
- **Evaluation metric**: Accuracy

→ CAT

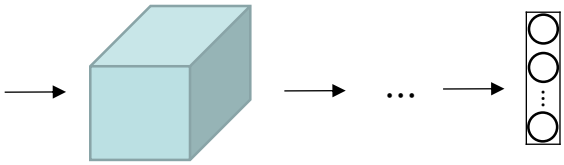
Classification + Localization
- **Input**: Image
- **Output**: Class label, Box coordinates
- **Evaluation metric**: Intersection over Union (IoU)

→ (CAT,x,y,w,h)

Classification with Localization

(0,0)

(1,1)



Lets allow a few classes:
1. Car
2. Truck
3. Pedestrian
4. Motorcycle

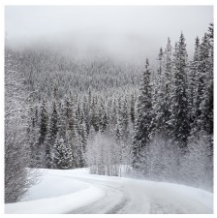
- For now, lets assume one object per image.
- Each object has {x, y, w, h}
- For this image, object location {x, y, w, h} = {0.4, 0.6, 0.4, 0.3}

Image from: deeplearning.ai, C4W3L01

Classification with Localization

Four classes:
1. Car
2. Truck
3. Pedestrian
4. Motorcycle

Localization {x, y, w, h}



Define *y* label: $y = \begin{bmatrix} P_c \\ b_x \\ b_y \\ b_w \\ b_h \\ C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix}$

Probability of an object

Bounding box location

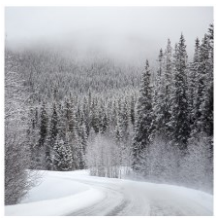
0/1 for each class

Image from: deeplearning.ai, C4W3L01

Classification with Localization

Four classes:
1. Car
2. Truck
3. Pedestrian
4. Motorcycle

Localization {x, y, w, h}



Cost function (squared error):

$$Loss = \sum_{i=1}^{9} (\hat{y}_i - y_i)^2$$ If y_1=1

$$Loss = (\hat{y}_1 - y_1)^2$$ If y_1=0

$$y = \begin{bmatrix} 1 \\ 0.4 \\ 0.6 \\ 0.4 \\ 0.3 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} P_c \\ b_x \\ b_y \\ b_w \\ b_h \\ C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix} \quad y = \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

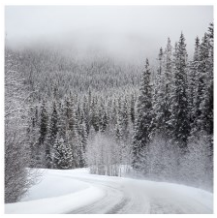
? = don't care

Image from: deeplearning.ai, C4W3L01

Classification with Localization

Four classes:
1. Car
2. Truck
3. Pedestrian
4. Motorcycle

Localization {x, y, w, h}



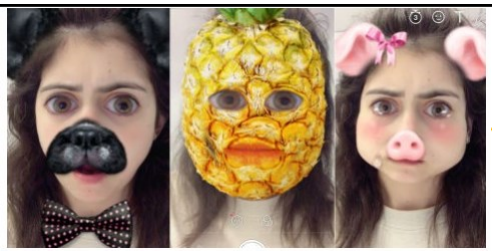
Alternate cost function:
- y_1 can be logistic loss
- $y_2 \rightarrow y_5$ can be squared error
- $y_6 \rightarrow y_9$ can be softmax cross entropy

$$y = \begin{bmatrix} 1 \\ 0.3 \\ 0.6 \\ 0.4 \\ 0.3 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad \begin{bmatrix} P_c \\ b_x \\ b_y \\ b_w \\ b_h \\ C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix} \qquad y = \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

? = don't care

Image from: deeplearning.ai, C4W3L01

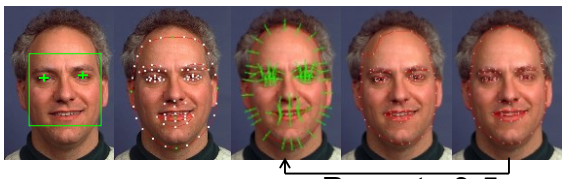
Snapchat Facewarp?



- Traditional approach:

Viola Jones Face Detection

Search for actual point locations using Mahalanobis distance



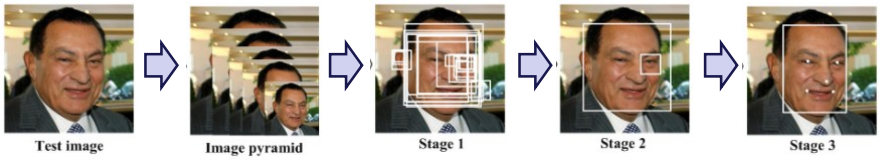
Repeat ~3-5×

Average eye and 82 facial feature points

Restrict based on PCA statistics

Snapchat Facewarp?

- Deep Learning approach:



Candidate faces

Refine face selection

Facial feature points

MT-CNN [Zhang et al. 2016]

Localization

- Facial feature

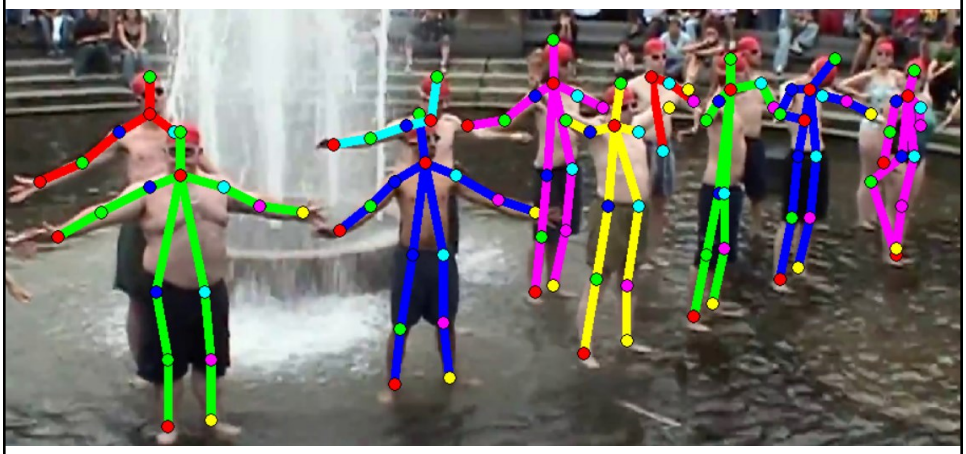
Each face has 68 points, so CNN would output:

Face?
pt1X
pt1Y
pt2X
pt2Y
.
.
.
pt68X
pt68Y

137 outputs

Of course, need GT for thousands of faces to train model.

Can do same with Body Pose…



Pishchulin et al. CVPR'16

Object Detection

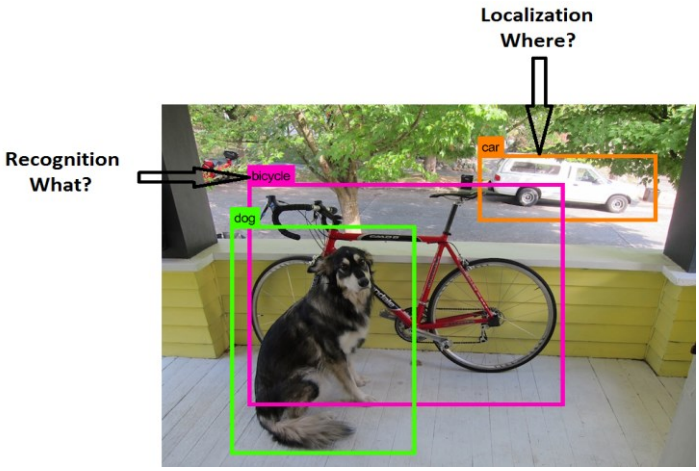


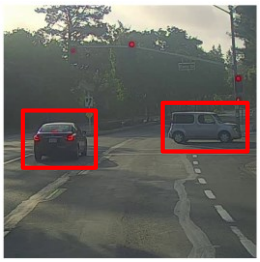
Image Credit: Redmon, Joseph, et al [4]

More than one object per image?

Training set:

x	y
	1
	1
	1
	0
	0

Car detection example

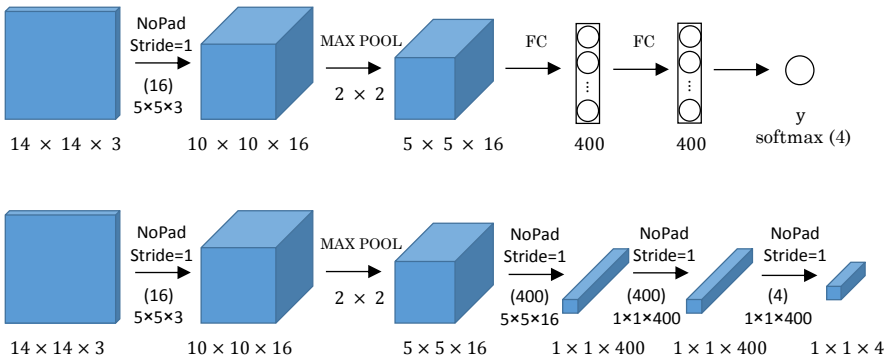


Images from: deeplearning.ai, C4W3L03

Sliding Window Detection

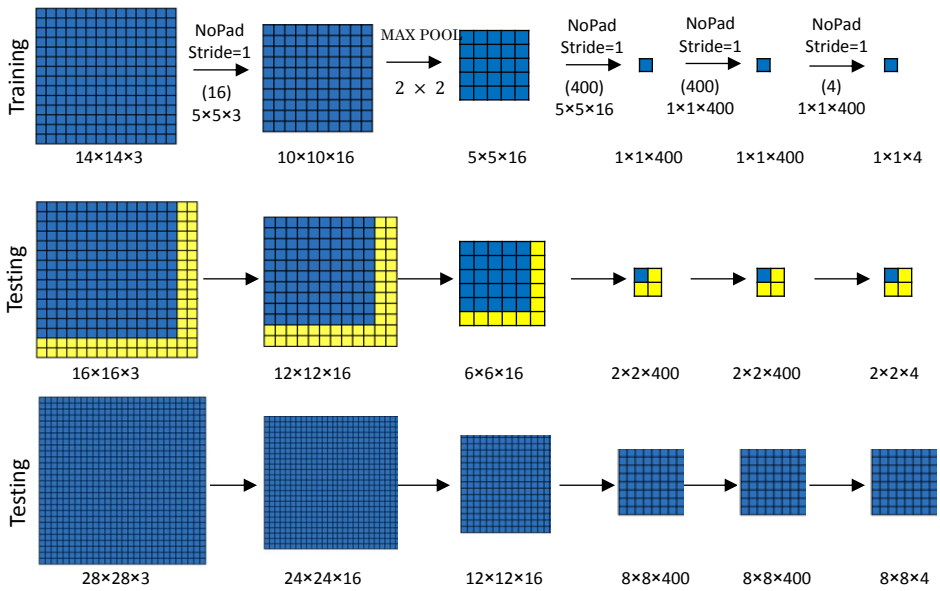
Images from: deeplearning.ai, C4W3L03

Computing FC layers with Convolution



Images from: deeplearning.ai, C4W3L04

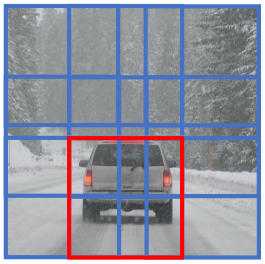
Replacing Sliding Windows w/Fully Convolutional CNNs



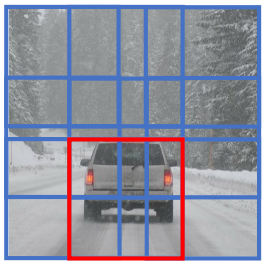
Images from: deeplearning.ai, C4W3L04

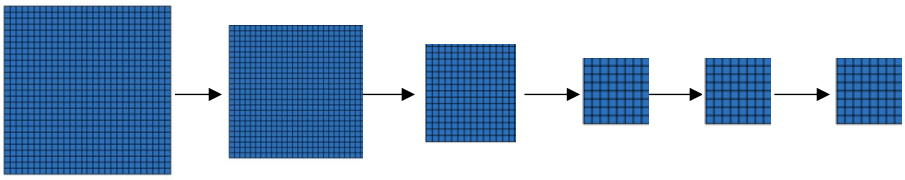
Replacing Sliding Windows w/Fully Convolutional CNNs

Sliding window approach:
Sequentially evaluate one window at a time



Fully convolutional approach:
Evaluate 64 regions at once





Images from: deeplearning.ai, C4W3L04

R. Ptucha '18

21

Replacing Sliding Windows w/Fully Convolutional CNNs

- Can think of this as evaluating 8×8 grid, where each of the 64 cells is independently checked for an object:

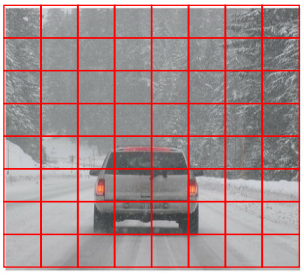
Each cell has a *y* label:

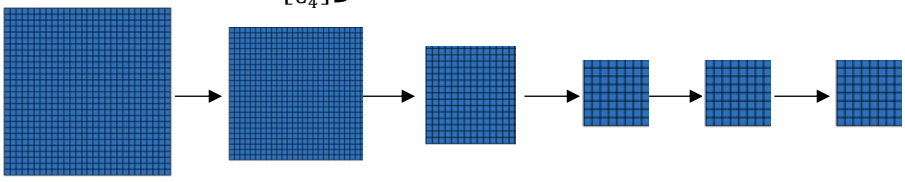
$$y = \begin{bmatrix} P_c \\ b_x \\ b_y \\ b_w \\ b_h \\ C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix}$$

Prob. of an object

Object location

0/1 for each class
(Four classes in this example)





R. Ptucha '18

22

Replacing Sliding Windows w/Fully Convolutional CNNs

- Overlay GT of object
- Cell where centroid lie is responsible.

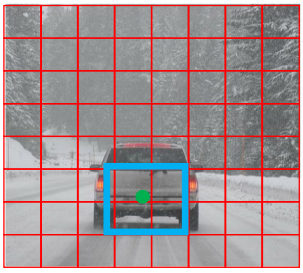
Each cell has a *y* label:

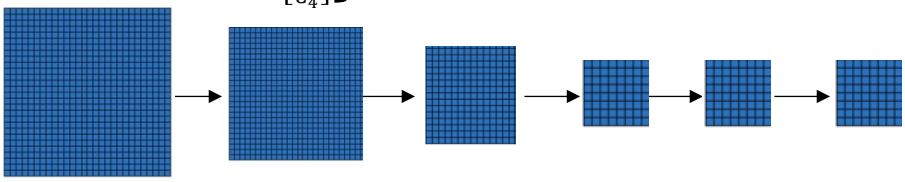
$$y = \begin{bmatrix} P_c \\ b_x \\ b_y \\ b_w \\ b_h \\ C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix}$$

Prob. of an object

Object location

0/1 for each class
(Four classes in this example)



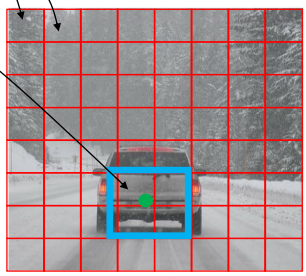


R. Ptucha '18

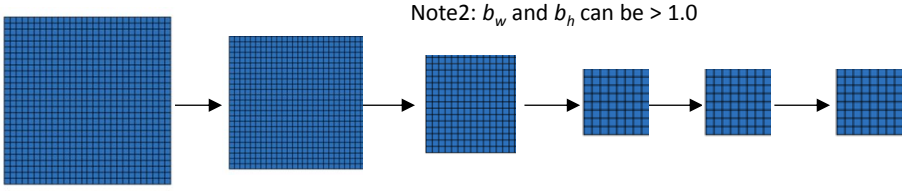
23

Replacing Sliding Windows w/Fully Convolutional CNNs

$$y = \begin{bmatrix} P_c \\ b_x \\ b_y \\ b_w \\ b_h \\ C_{pers} \\ C_{car} \\ C_{truck} \\ C_{motorc} \end{bmatrix} \quad y1 = \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix} \quad y2 = \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix} \quad \ldots \quad y44 = \begin{bmatrix} 1 \\ 0.8 \\ 0.9 \\ 2.0 \\ 1.8 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad \ldots$$



Note1: cell upper left (0,0); cell lower right (1,1)
Note2: b_w and b_h can be > 1.0

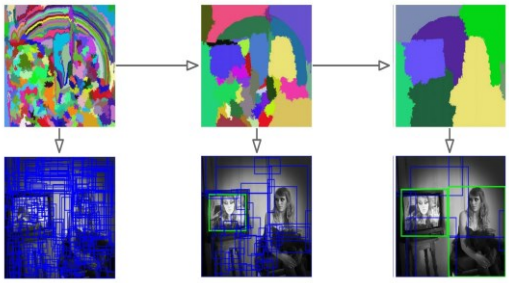


R. Ptucha '18

24

How is Done in Practice?

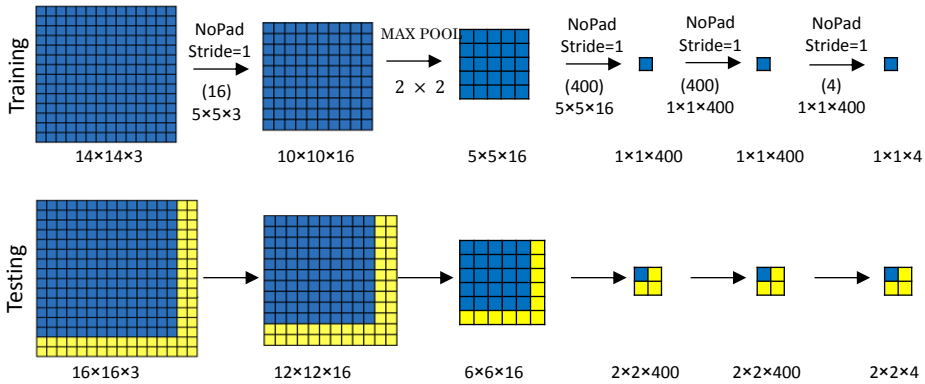
1. Region proposals- use algorithms such as selective search or edge boxes to suggest potential bounding box locations.



Uijlings et al., “Selective search for Object Recognition”, IJCV 2013.

How is Done in Practice?

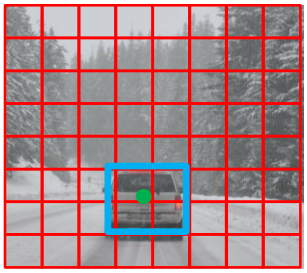
2. Fully convolutional methods- replace fully connected layers with convolution.

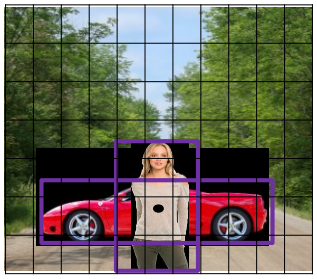


How is Done in Practice?

3. Anchor boxes. What if two object centroids land in same cell?

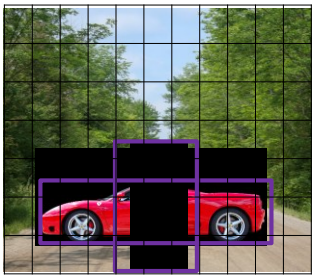
$$y = \begin{bmatrix} P_c \\ b_x \\ b_y \\ b_w \\ b_h \\ C_{pers} \\ C_{car} \\ C_{truck} \\ C_{motorc} \end{bmatrix}$$



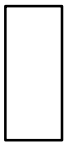


How is Done in Practice?

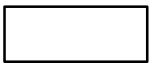
- Anchor boxes. Allow multiple detections per cell. Define multiple anchor boxes per cell.
- Assign each GT object to closest Anchor shape (use max IoU).



Anchor 1:



Anchor 2:



$$y = \begin{bmatrix} P_c \\ b_x \\ b_y \\ b_w \\ b_h \\ C_{pers} \\ C_{car} \\ C_{truck} \\ C_{motorc} \\ P_c \\ b_x \\ b_y \\ b_w \\ b_h \\ C_{pers} \\ C_{car} \\ C_{truck} \\ C_{motorc} \end{bmatrix} \begin{matrix} \text{Anchor 1} \\ \\ \text{Anchor 2} \end{matrix}$$

Faster R-CNN: Region Proposal Network

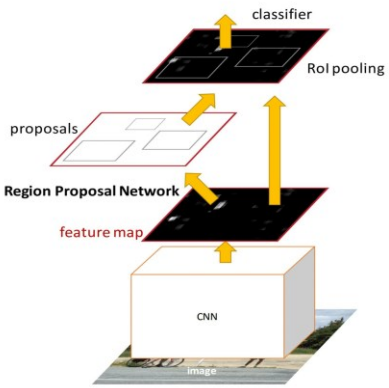


Image Credit: Ren et al. [3]

- Add Additional Region Proposal Network (RPN) after last convolution layer.
- Train RPN to extract region proposals
- After RPN, use ROI Pooling which feeds a classifier and bbox regressor.

Faster R-CNN: Region Proposal Network

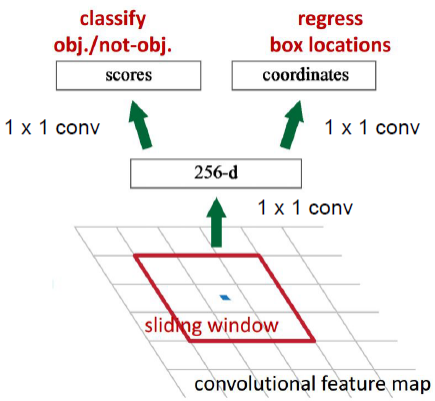
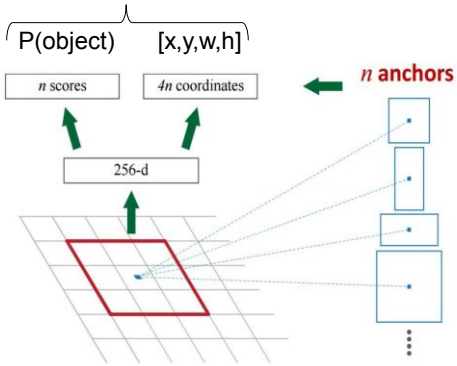


Image Credit: Kaiming He

- Use 3×3 sliding window over last conv layer (conv5) feature map
- Create small networks for object classification and bounding box coordinates.
- At each location,
  - we determine if object present, and
  - the bounding box [x,y,w,h] coordinates gives exact localization in input image.

Faster R-CNN: Region Proposal Network

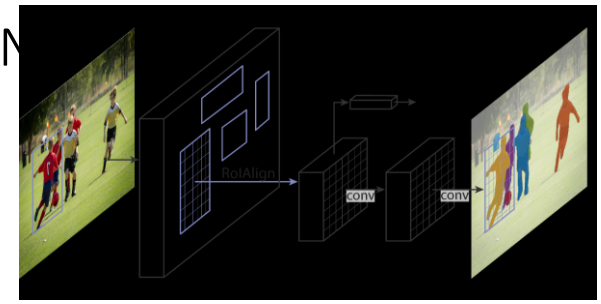
For each of the *n* anchor boxes:



- Use ***n*** anchor boxes at each location:
  - 3 aspect ratios 2:1,1:1,2:1
  - 3 scales, box areas of 128^2, 256^2, 512^2
- Use same anchor boxes at each location.
- Regression gives offset/scale of each anchor box.
- Classification gives probability of object at each anchor box.

Image Credit: Ren, Shaoqing, et al.[3]

Mask R-CNN

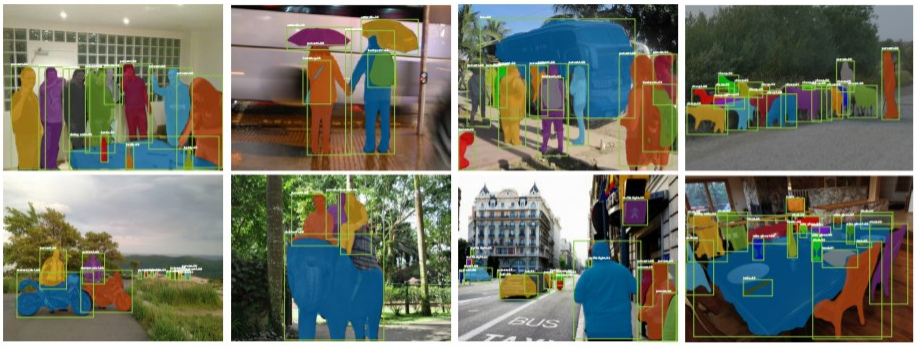


- Extension of Faster R-CNN
- Faster R-CNN used region proposal network (3×3 sliding window over conv5) predicts *n* anchor boxes at each location. Each anchor box has [P(object),x,y,w,h]
- Each anchor box then used for final network to predict [P(object),x,y,w,h,C_1,C_2,…,C_C]
- Extend to also predict 28×28 pixel mask of object

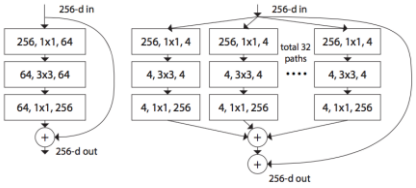
http://kaiminghe.com/iccv17tutorial/maskrcnn_iccv2017_tutorial_kaiminghe.pdf

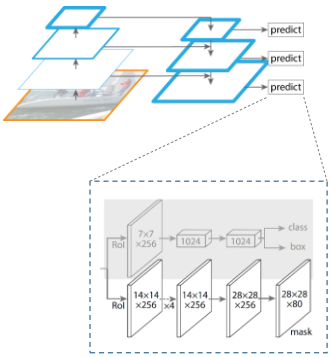
Mask R-CNN

- Extension of Faster R-CNN
- Simultaneously predict bounding box along with object mask.
- Shown to work well on instance segmentation, bounding-box object detection, and person keypoint (body joint) detection.



Mask R-CNN





Built on:

- ResNext [Xie et al., CVPR'17]
- Feature Pyramid Net [Lin eta al., CVPR'17]

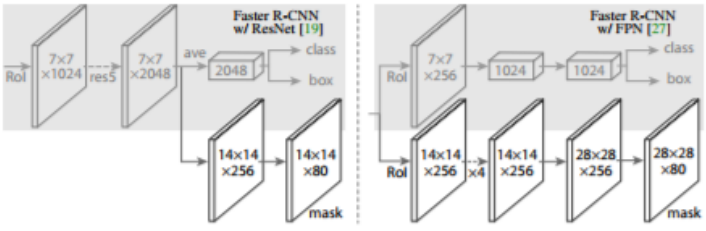
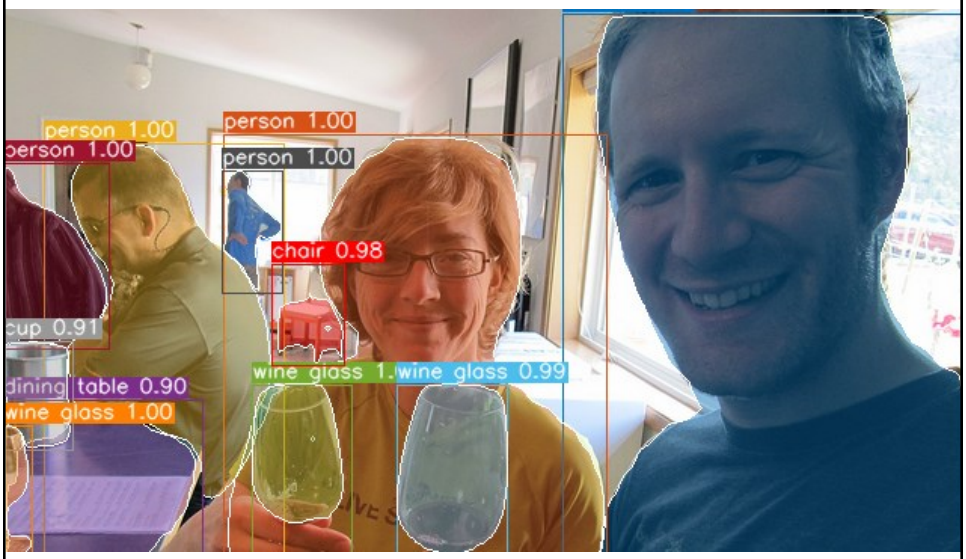


Figure 3. **Head Architecture**: We extend two existing Faster R-CNN heads [19, 27]. Left/Right panels show the heads for the ResNet C4 and FPN backbones, from [19] and [27], respectively, to which a mask branch is added. Numbers denote spatial resolution and channels. Arrows denote either conv, deconv, or *fc* layers as can be inferred from context (conv preserves spatial dimension while deconv increases it). All convs are 3×3, except the output conv which is 1×1, deconvs are 2×2 with stride 2, and we use ReLU [30] in hidden layers. *Left*: 'res5' denotes ResNet's fifth stage, which for simplicity we altered so that the first conv operates on a 7×7 RoI with stride 1 (instead of 14×14 / stride 2 as in [19]). *Right*: '×4' denotes a stack of four consecutive convs.

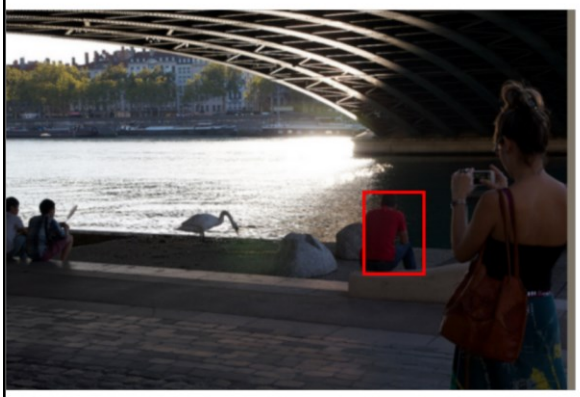
FAIR Mask R-CNN, COCO + Places Workshop, ICCV 2017



FAIR Mask R-CNN, COCO + Places Workshop, ICCV 2017

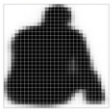
How get such nice masks with only 28×28 pix per mask?

Mask Prediction



Validation image with box detection shown in red

28x28 soft prediction from Mask R-CNN
(enlarged)

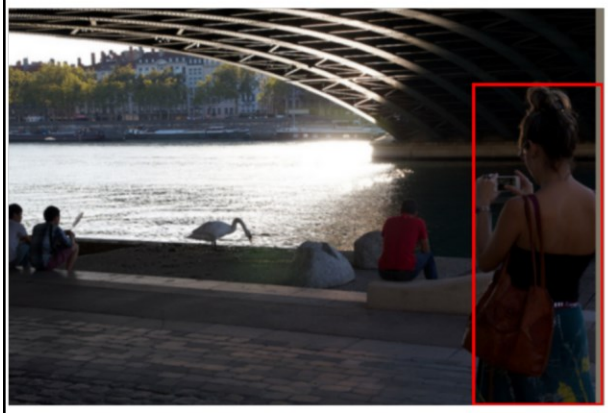


Soft prediction resampled to image coordinates
(bilinear and bicubic interpolation work equally well)

Final prediction (threshold at 0.5)

http://deeplearning.csail.mit.edu/instance_ross.pdf

Mask Prediction



Validation image with box detection shown in red

28x28 soft prediction

Resized soft prediction

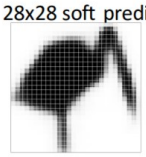
Final mask

http://deeplearning.csail.mit.edu/instance_ross.pdf

Mask Prediction

Validation image with box detection shown in red

28x28 soft prediction

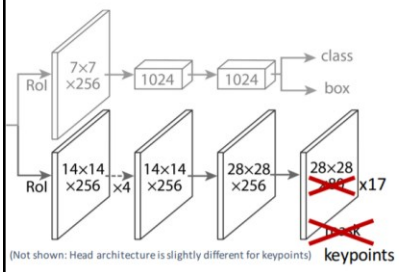


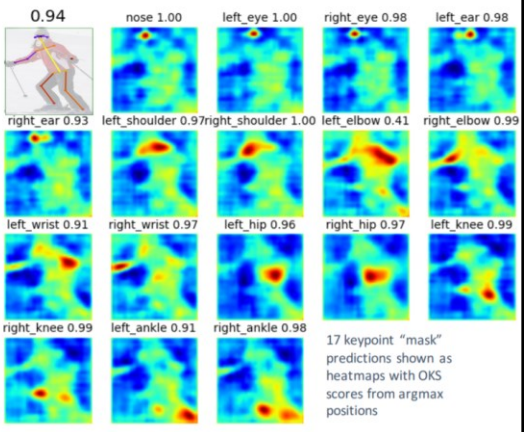
Resized Soft prediction

Final mask

http://deeplearning.csail.mit.edu/instance_ross.pdf

Human Pose

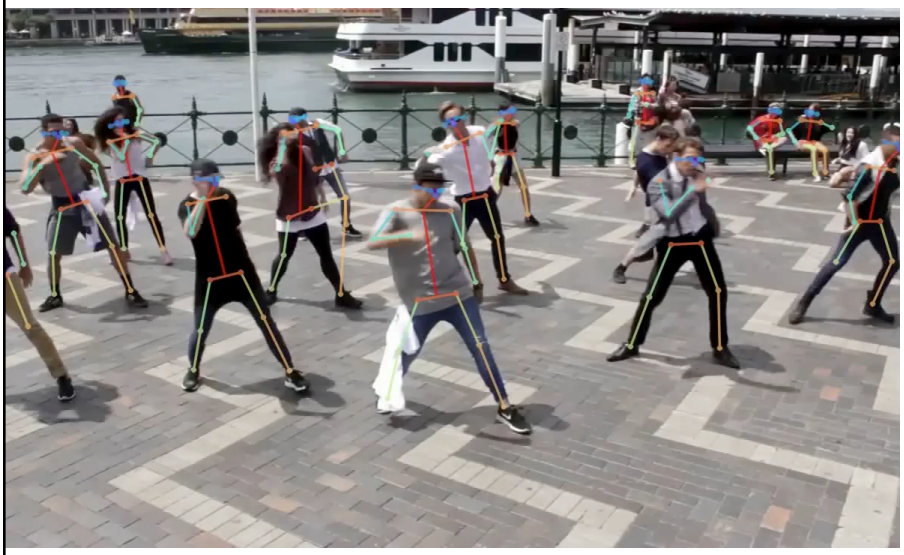




- Add keypoint head (28x28x17)
- Predict one "mask" for each keypoint (keypoint==bodyjoint)
- Softmax over spatial locations (encodes one keypoint per mask "prior")

http://deeplearning.csail.mit.edu/instance_ross.pdf

FAIR Mask R-CNN, COCO + Places Workshop, ICCV 2017



FAIR Mask R-CNN, COCO + Places Workshop, ICCV 2017

Good Object Detection Libraries to Explore

- Tensorflow library for object detection (good for Tensorflow users, lots of algorithms)
  - https://github.com/tensorflow/models/tree/master/research/object_detection
- Facebook library for object detection (includes Mask R-CNN,based on Caffe2)
  - https://github.com/facebookresearch/Detectron
- YOLO
  - https://pjreddie.com/darknet/yolo/
  - https://pjreddie.com/media/files/papers/YOLOv3.pdf

Thank you!!

Ray Ptucha
rwpeec@rit.edu

https://www.rit.edu/mil