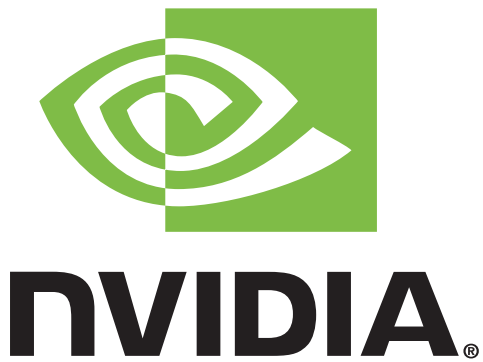
Introduction to Object Detection & Image Segmentation



Abel Brown

July 21, 2017

Outline

What is Object Detection?

Object detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings, or cars) in digital images and videos.. [1]

[1] Wikipedia

What is Image Segmentation?

In computer vision, image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as super-pixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. [2]

[2] Wikipedia

What is Image Segmentation?

More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain characteristics.[3]

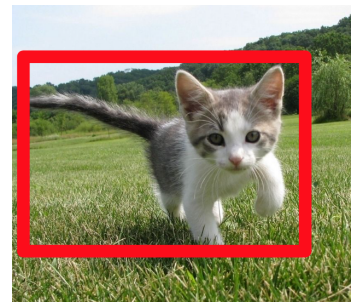
[3] Wikipedia

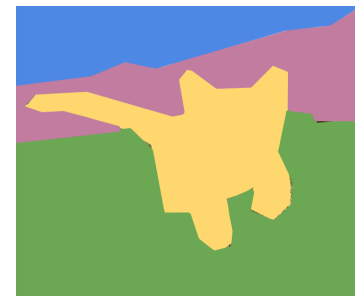
Generic Detection and Segmentation

Given an input tensor of size *CxHxW* constructed from pixel values of some image ...

- *identify* content of interest
- *locate* the interesting content
- *partition* input (i.e. pixels) corresponding to identified content

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...





4

- Workflow: object detection, localization, and segmentation

[4] Stanford cs231n (2017)

Examples: Binary Mask

Figure 1: SpaceNet sample data

Examples: Binary Mask

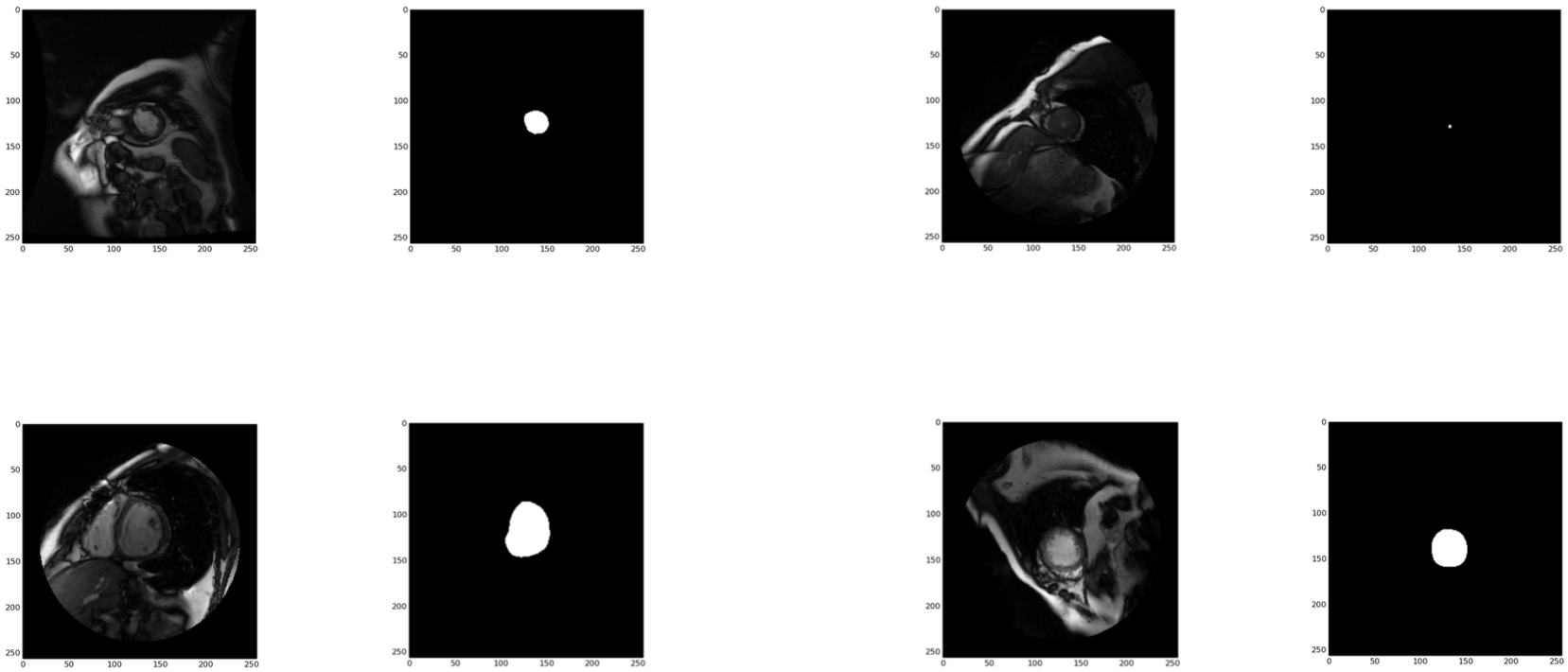


Figure 2: Sunnybrook - Left ventricle segmentation (fMRI)

Examples: Binary Mask

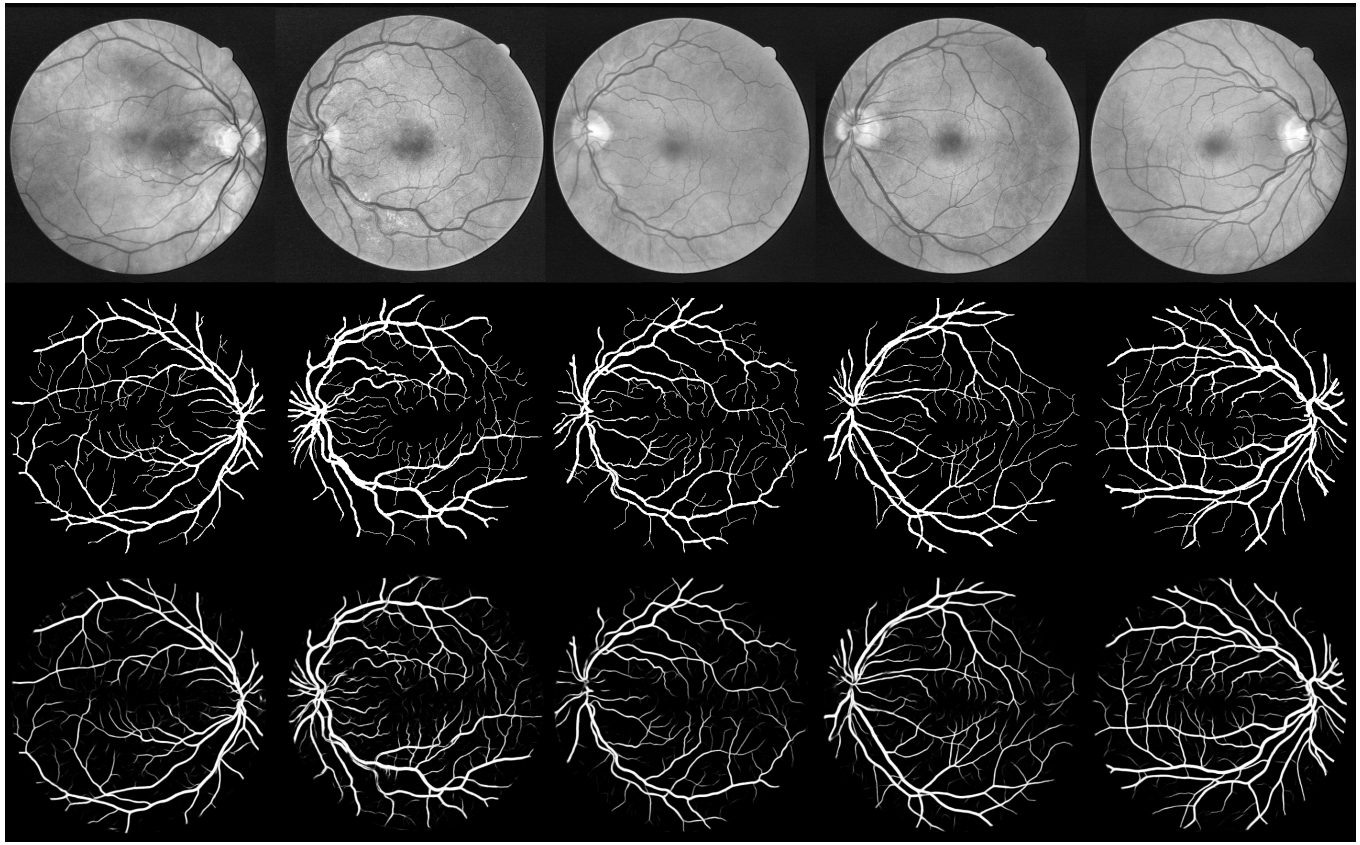


Figure 3: U-Net: CNNs for Biomedical Image Segmentation

Examples: Multiclass

Figure 4: FAIR: Learning to Segment

Examples (and More Lingo)

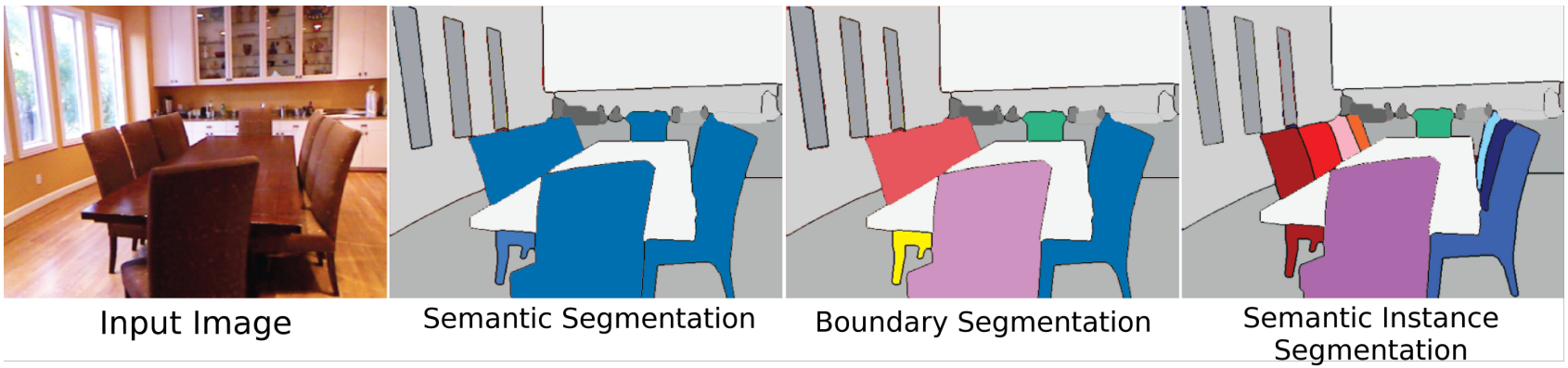


Figure 5: Silberman - Instance Segmentation

Boundary Segmentation Examples

Figure 6: Farabet - Scene Parsing

Boundary Segmentation Examples

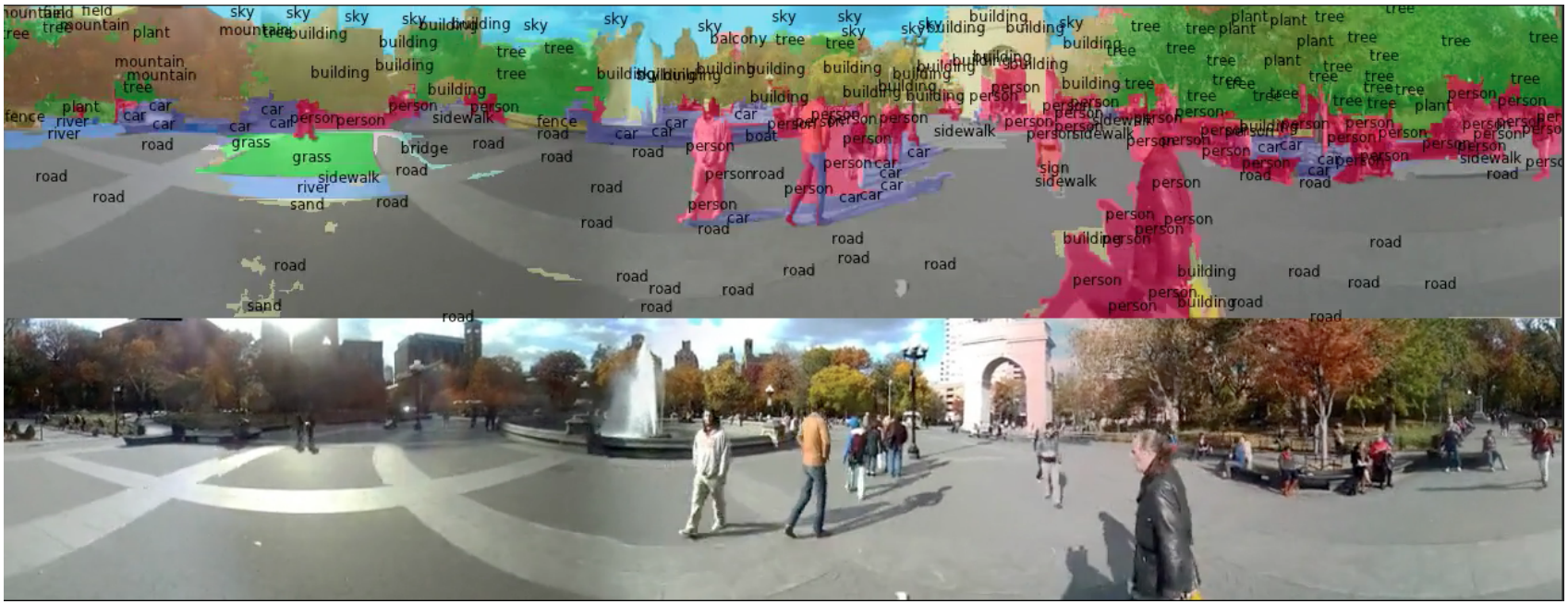
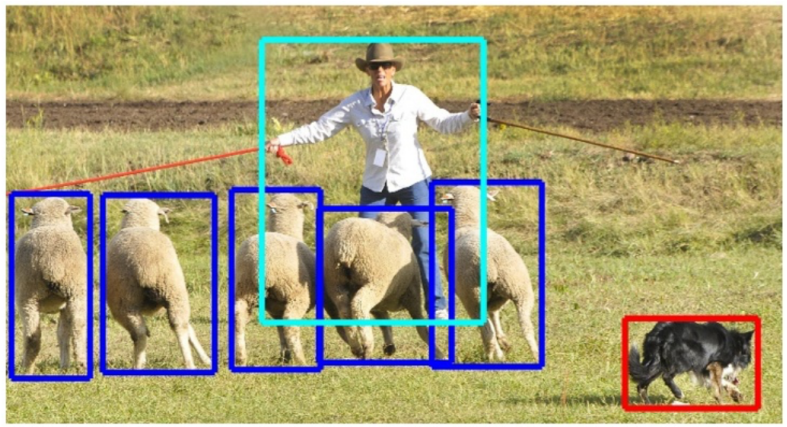


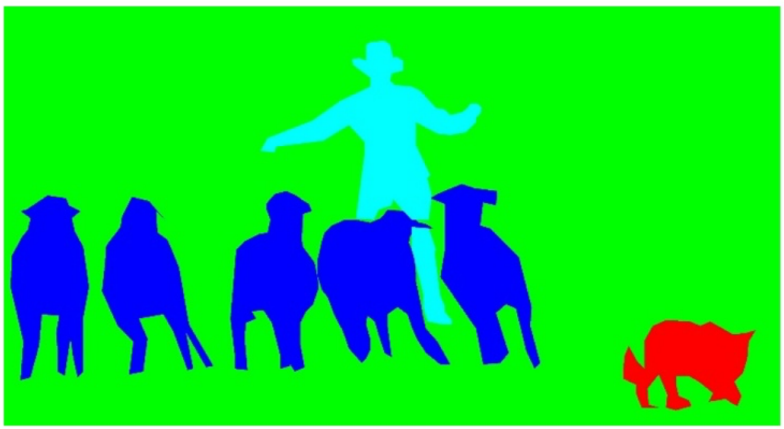
Figure 7: Farabet - Scene Parsing

Instance Segmentation Examples

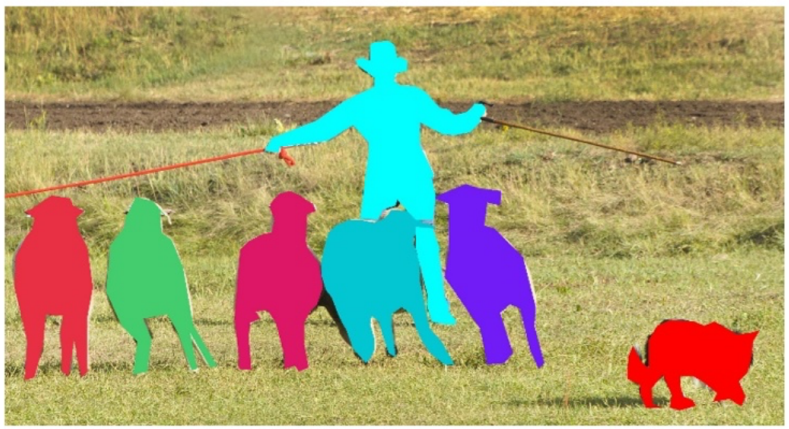
(a) Image classification



(b) Object localization



(c) Semantic segmentation



(d) This work

Figure 8: Microsoft COCO: Common Objects in Context

Instance Segmentation Examples

Figure 9: FAIR: A MultiPath Network for Object Detection

Image Segmentation Examples

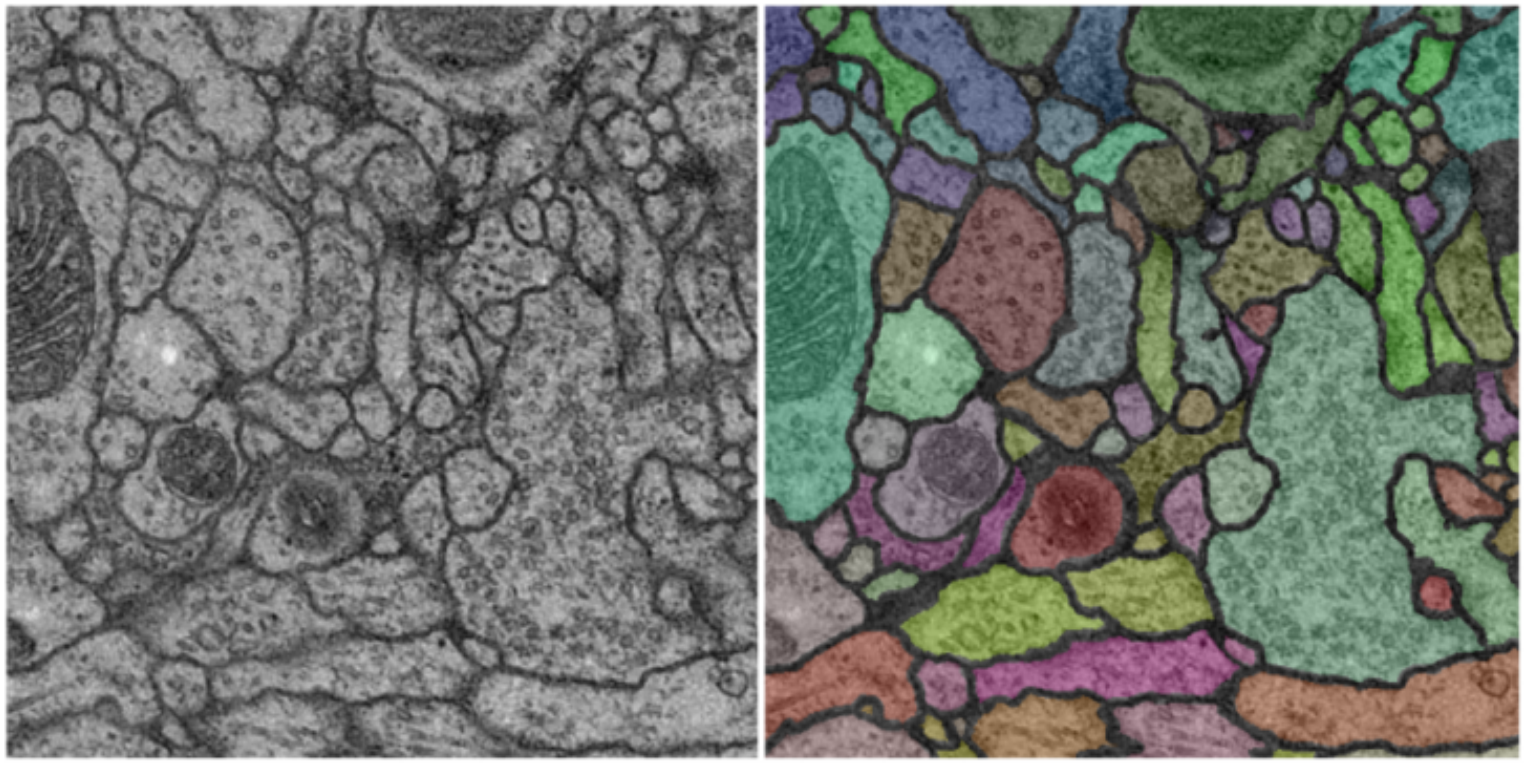


Figure 10: Ciresan - Neuronal membrane segmentation

Image Segmentation Examples

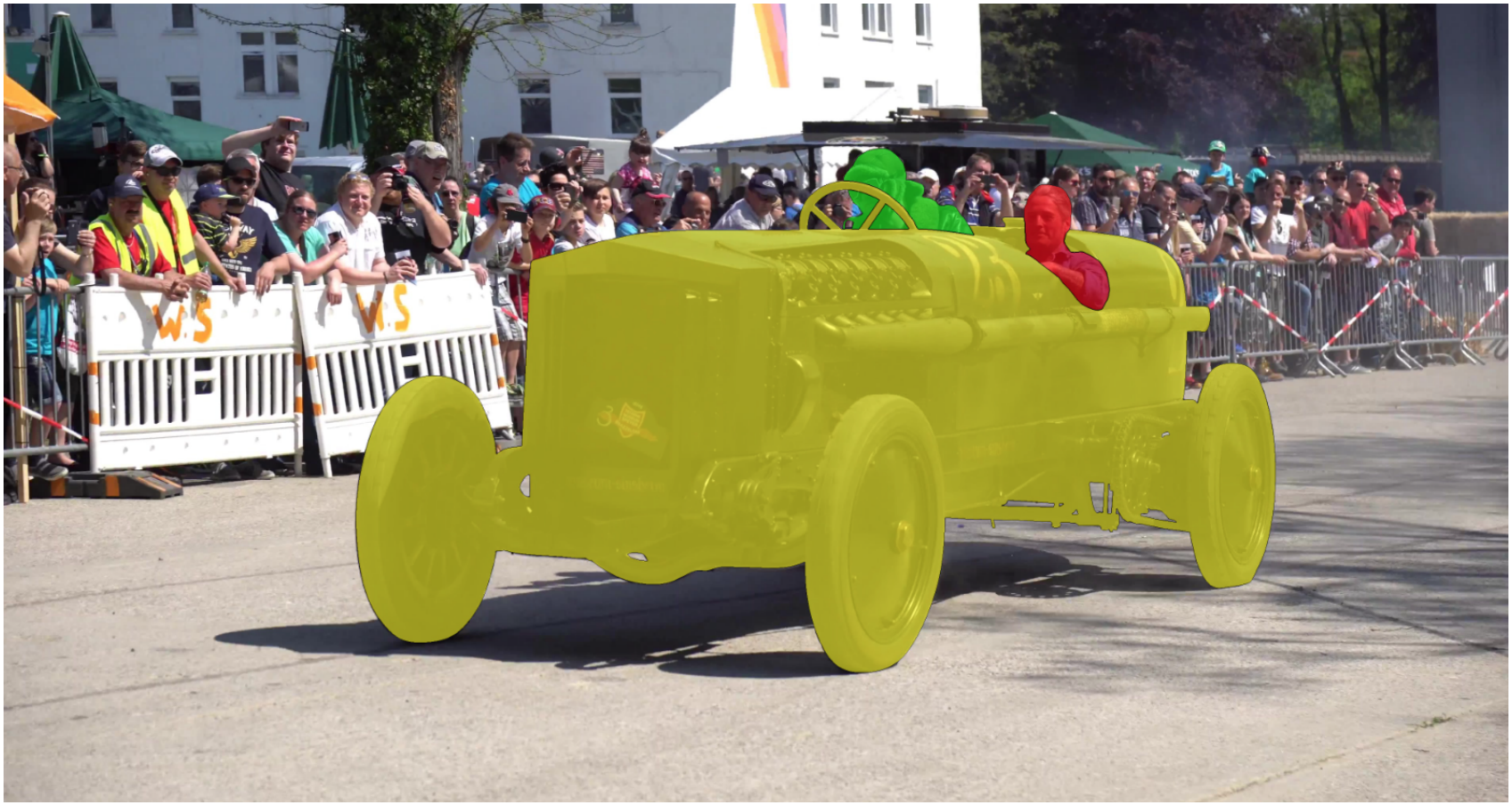


Figure 11: DAVIS: Densely Annotated VIdeo Segmentation

History: Pre Deep Learning

- Object detection, localization, and segmentation has a long history before deep learning became popular
- Years before ImageNet[5] and deep learning there was PASCAL[6,7] and custom computer vision techniques
- Many early algorithms shared similar structure:
  - identify potentially relevant content (region proposals)
  - for each proposed region, test/label region
  - aggregate results from all regions to form final answer/result/output for the image
- Even early DL based algorithms shared this structure (Overfeat, R-CNN, etc)
- Recently, some successful *single-stage* DL approaches

[5] ImageNet: A Large-Scale Hierarchical Image Database
[6] The PASCAL Visual Object Classes (VOC) Challenge
[7] The PASCAL Challenge: A Retrospective

Pre Deep Learning Methods

Example-Based Learning	1998	2435
Efficient Graph-Based Image Segmentation	2004	4787
Image Features from Scale-Invariant Keypoints	2004	42365
Histograms of Oriented Gradients	2005	19435
Category Independent Object Proposals	2010	367
Constrained Parametric Min-Cuts	2010	387
Discriminatively Trained Part Based Models	2010	5646
Measuring the objectness of image windows	2011	669
Selective Search for Object Recognition	2012	1212
Regionlets for Generic Object Detection	2013	218
Multiscale Combinatorial Grouping	2014	468

Common Issues with Algorithms

- Compute performance often poor
  - Too many region proposals to test and label
  - Difficult to scale to larger image size and/or frame rate
  - Cascading approaches help but not solve
  - Aggressive region proposal suppression leads to accuracy issues
- Accuracy problems
  - Huge number of candidate regions inflates false-positive rates
  - Illumination, occlusion, etc. can confuse test and label process
- Not really scale invariant
  - Early datasets not very large so limited feature variation
  - Now training datasets are many TB – helps but does not solve
  - Large variation of feature scale can inflate false-negative rates

Quality Assessment and Metrics

- Assessing the quality of a classification result is generally well defined
- Quality assessment of object localization and segmentation results is more complex
  - Object localization output is bounding box
  - How to assess overlap between ground truth and computed bounding boxes?
  - What about *sloppy* or *loose* ground truth bounding boxes?
  - Segmentation output is polygon-like pixel region
  - How to assess overlap of polygon-like ground truth and computed output region?
  - What about *sloppy* or *corse* ground truth regions?
- All this gets a bit more complicated when considering video (i.e. continuous stream of highly correlated images)

Quality Assessment and Metrics

Good, great, not bad, terrible?

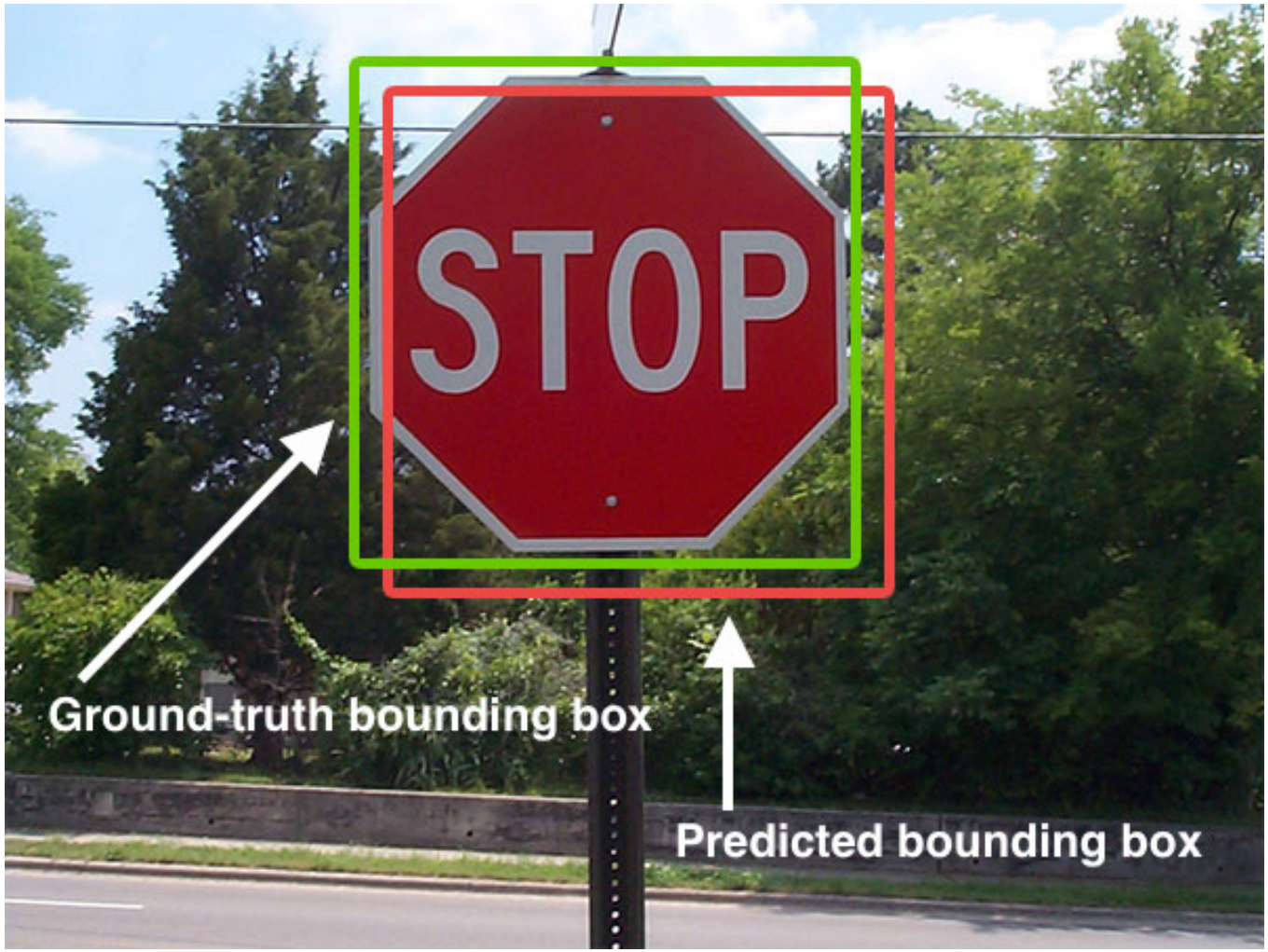


Figure 12: pyimagesearch.com

Quality Assessment and Metrics

Good, great, not bad, terrible?

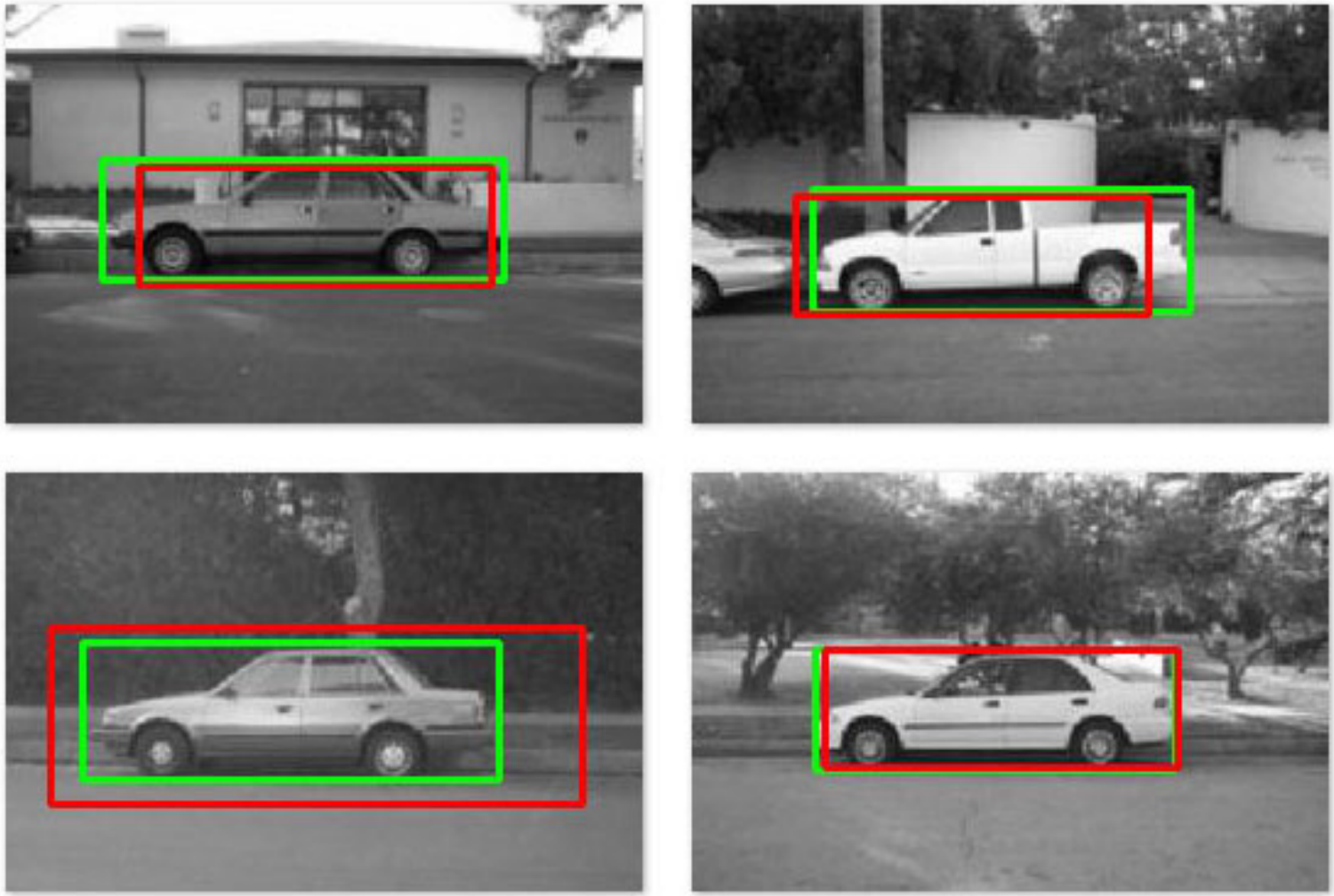


Figure 13: pyimagesearch.com

Quality Assessment and Metrics

Good, great, not bad, terrible?

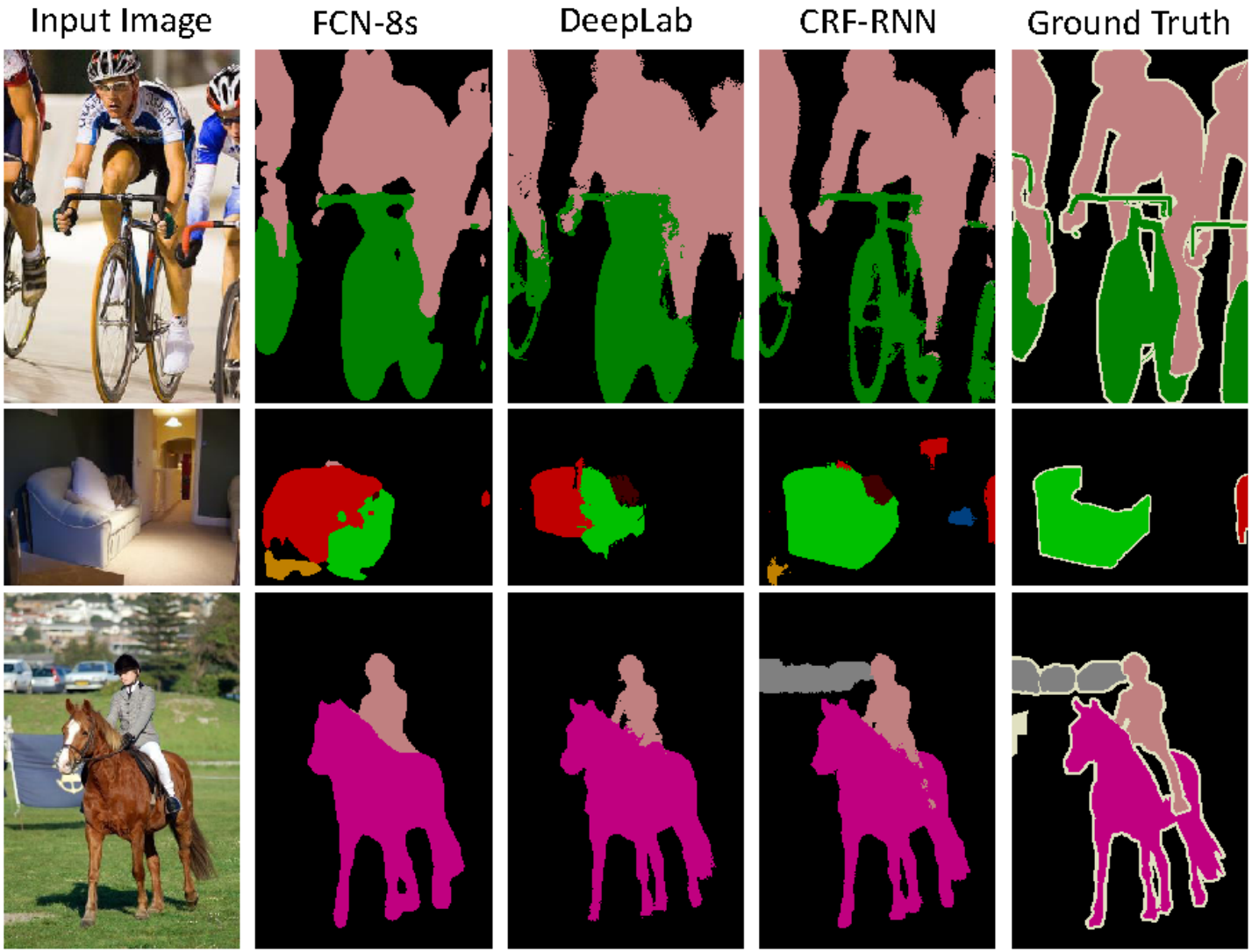


Figure 14: Zheng et al., CRF as RNN

Quality Assessment and Metrics

Good, great, not bad, terrible?

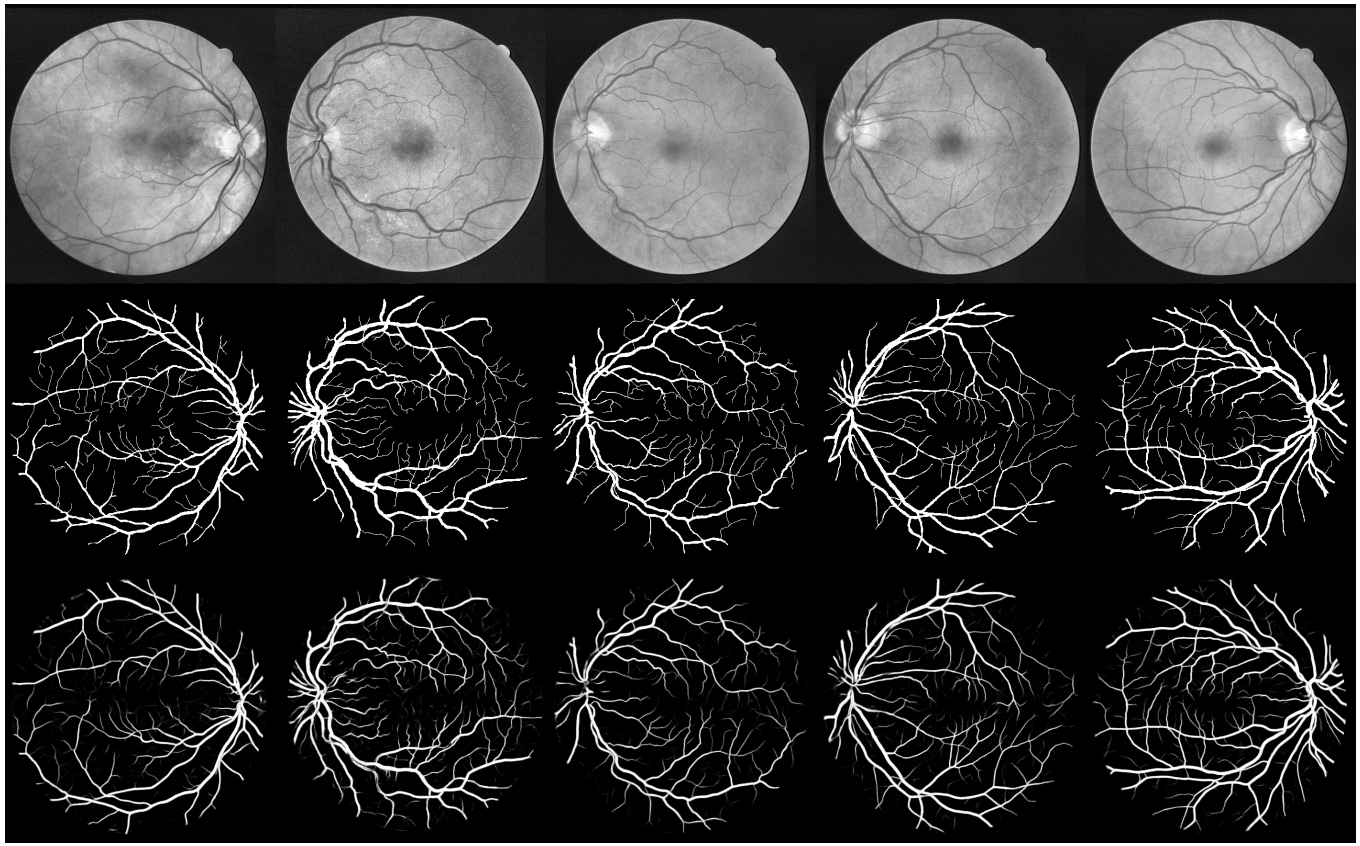


Figure 15: Ronneberger et al., U-Net: Biomedical Image Segmentation

Quality Assessment and Metrics

- A common metric is *mean average precision* (mAP)[8]
  - For each class c_i, calculate average precision $ap_i = AP(c_i)$
  - Compute the mean over all ap_i values calculated for each class

- Another common metric is *intersection over union* (IoU)
  - Each bounding box (i.e. detection) is associated with a confidence (sometimes called *rank*)
  - Detections are assigned to ground truth objects and judged to be true/false positives by measuring overlap
  - To be considered a correct detection (i.e. true positive), the area of overlap a_{ovl} between predicted bounding box BB_p and the ground truth bounding box BB_{gt} must exceed 0.5 according to

$$\text{area}_{ovl} = \frac{\text{area}(\text{BB}_p \cap \text{BB}_{gt})}{\text{area}(\text{BB}_p \cup \text{BB}_{gt})} \tag{1}$$

  - area_{ovl} is often called *intersection over union* (IoU)

[8] see Everingham et al. for more details

Quality Assessment and Metrics

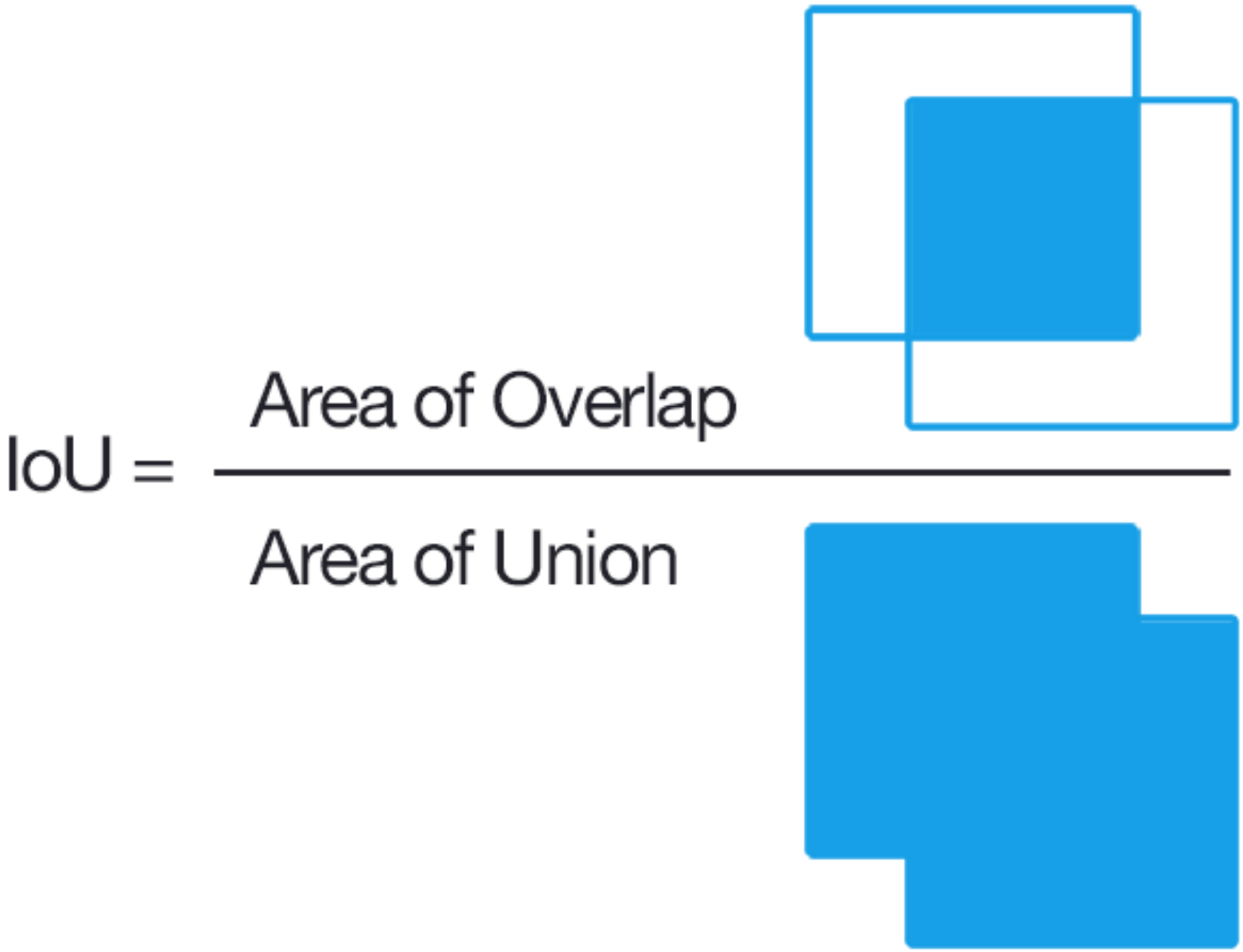


Figure 16: Pyimagesearch: IoU for object detection

Quality Assessment and Metrics

A few examples of IoU values and their associated configuration

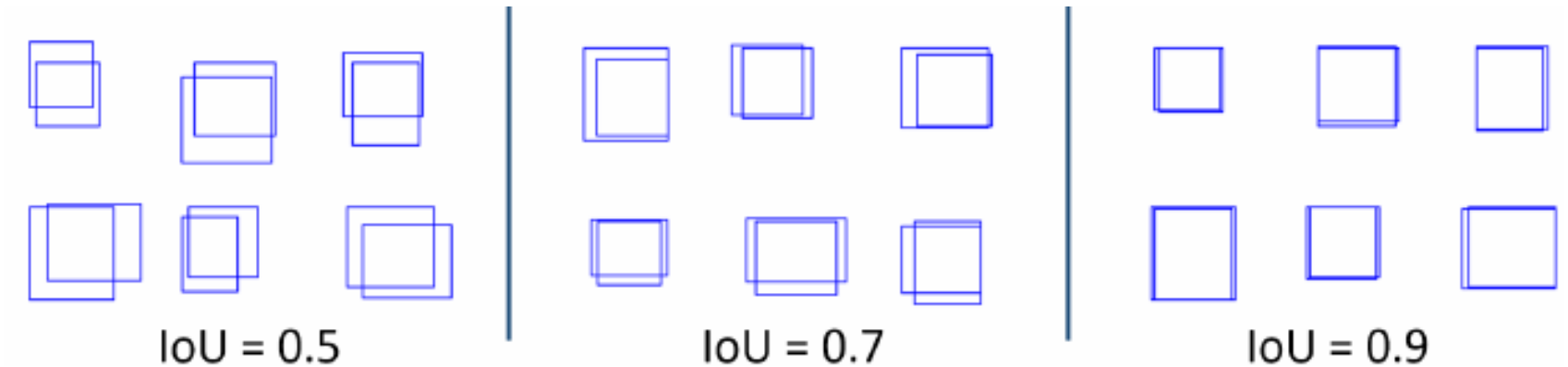


Figure 17: Leonardo Santos, Object Localization and Detection

Quality Assessment and Metrics

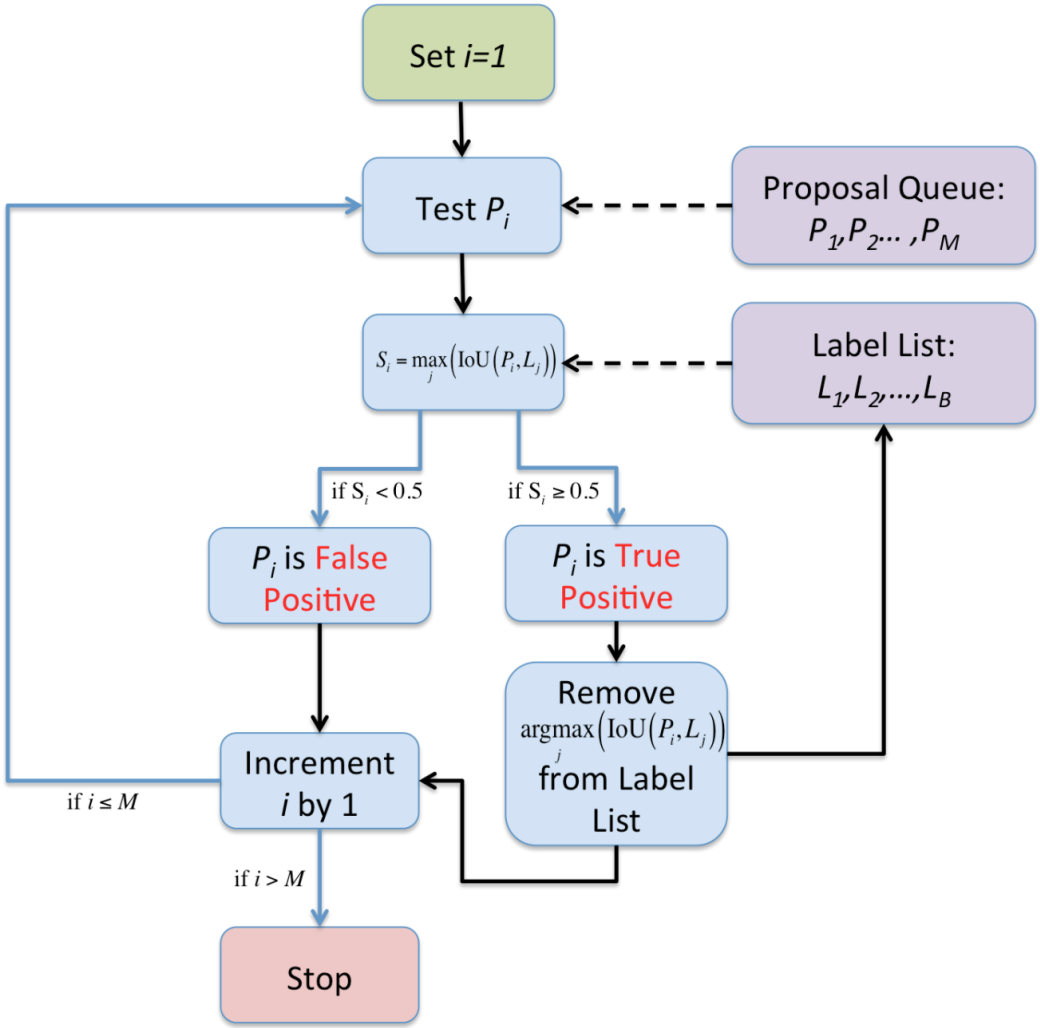


Figure 18: The SpaceNet Metric: A list of proposals is generated by the detection algorithm and compared to the ground truth in the list of labels

Quality Assessment and Metrics

- A common metric used to evaluate segmentation performance is the percentage of pixels correctly labeled.
- Although, percentage correctly labeled can lead to situations where label all pixels as "pedestrian" class to maximize score on pedestrian class.
- To rectify this, easy to modify assessment based on the intersection of the inferred segmentation and the ground truth divided by the union[9]. That is:

$$\text{seg.accuracy} = \frac{\text{true pos}}{\text{true pos} + \text{false neg} + \text{false pos}} \tag{2}$$

- Before machine learning, this was known as *Jaccard Index*

[9] Again, see Everingham et al. for additional discussion

PASCAL VOC2012 Leaderboard

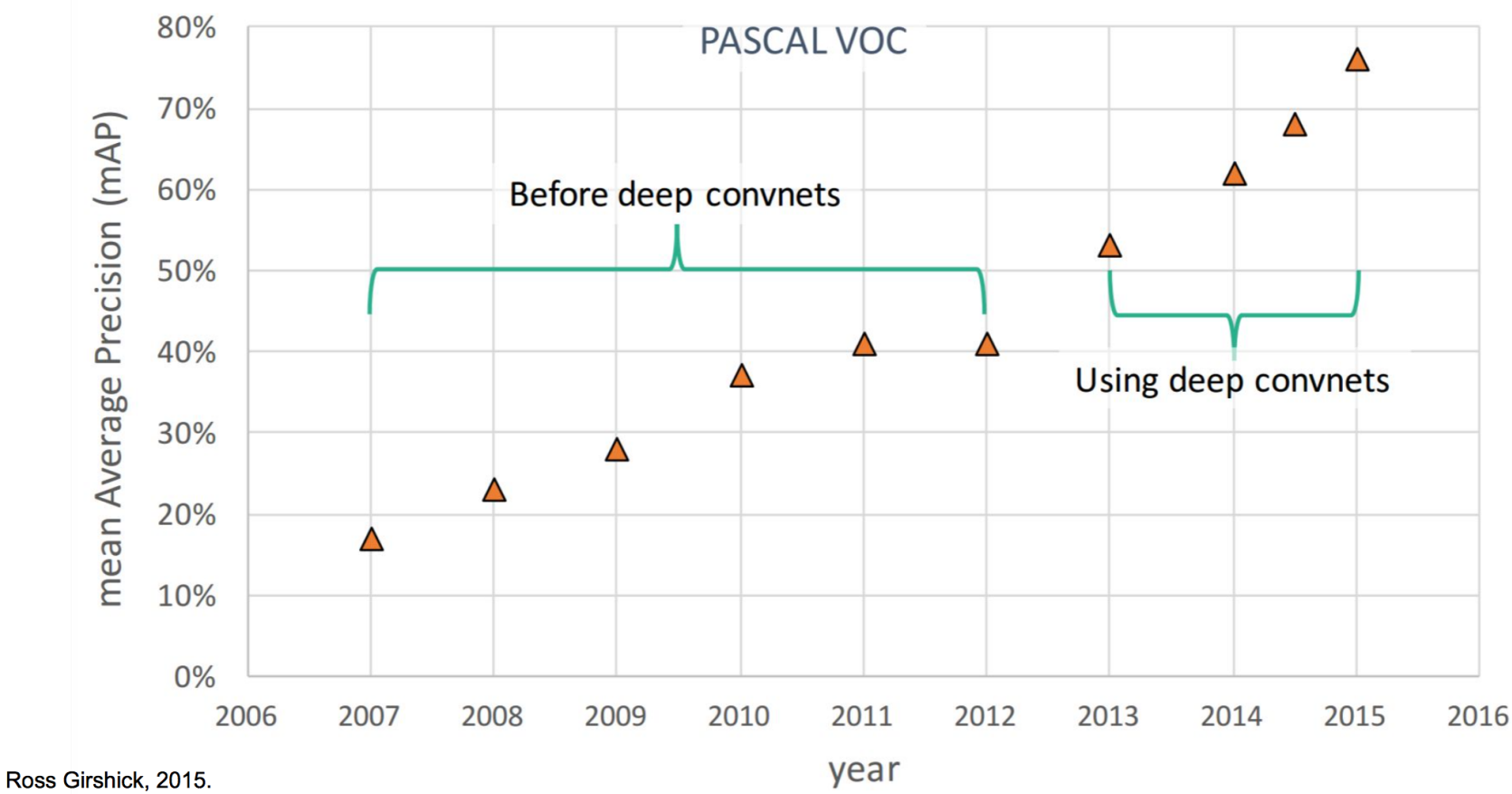


Figure 19: PASCAL VOC2012 segmentation leaderboard. As of 30-June-2017 top performance score of 86.3% mPA

Early DL Detection and Segmentation

- The early DL segmentation efforts looked a lot like traditional detection and segmentation workflows.
- Although convolution neural networks had been around since late 1990s[10], it was not until CNNs won the ImageNet competition in 2012 that *deep learning* really took off.
- The winning ImageNet solution in 2012 was called AlexNet and was largely based on the original network architecture defined in LeCun's original paper.
- The Overfeat (2013) solution was one of the first detection and localization strategies based on deep learning which leveraged the AlexNet success.
- The Overfeat solution "*explores the entire image by densely running the network at each location and at multiple scales*" via a sliding window approach.

[10] LeCun et al., Gradient-Based Learning Applied to Doc Recognition, 1998

Early DL Solutions: The R-CNN Family

- The original R-CNN approach combined aforementioned region proposal methods (i.e. selective search) with the AlexNet CNN in order to localize and segment objects
- Because region proposals are combined with CNNs, the method is referred to as "**R***egions with CNN features*" or **R**-CNN for short
- Additionally, R-CNN was one of the first to propose *transfer learning*: "*when labeled training data is scarce, supervised pre-training for an auxiliary task followed by domain-specific fine-tuning yields a significant performance boost*"

Early DL Solutions: The R-CNN Family

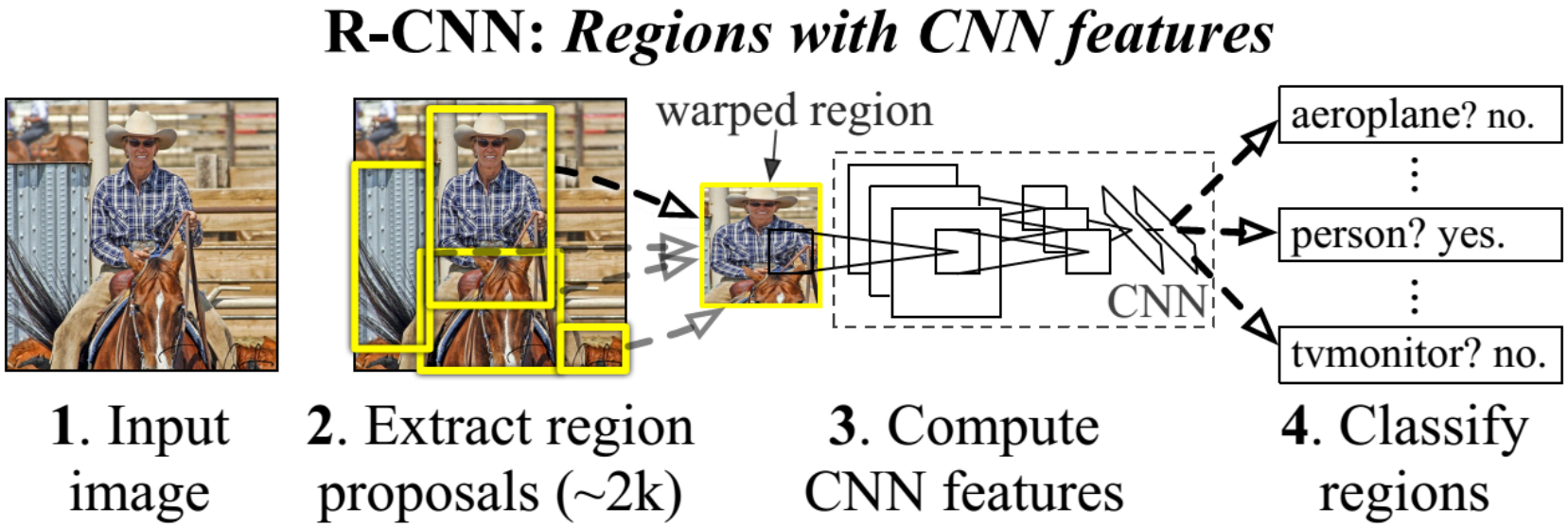


Figure 20: Girshick et al., Rich feature hierarchies, 2013

- identify potentially relevant content (≈2k proposals/img)
- for each proposed region: use CNN to generate feature vector
- Use SVMs to classify each feature vector
- Linear regression for bounding box offsets

Early DL Solutions: The R-CNN Family

Figure 21: Uijlings et al., Selective Search for Object Recognition, 2013

Early DL Solutions: The R-CNN Family

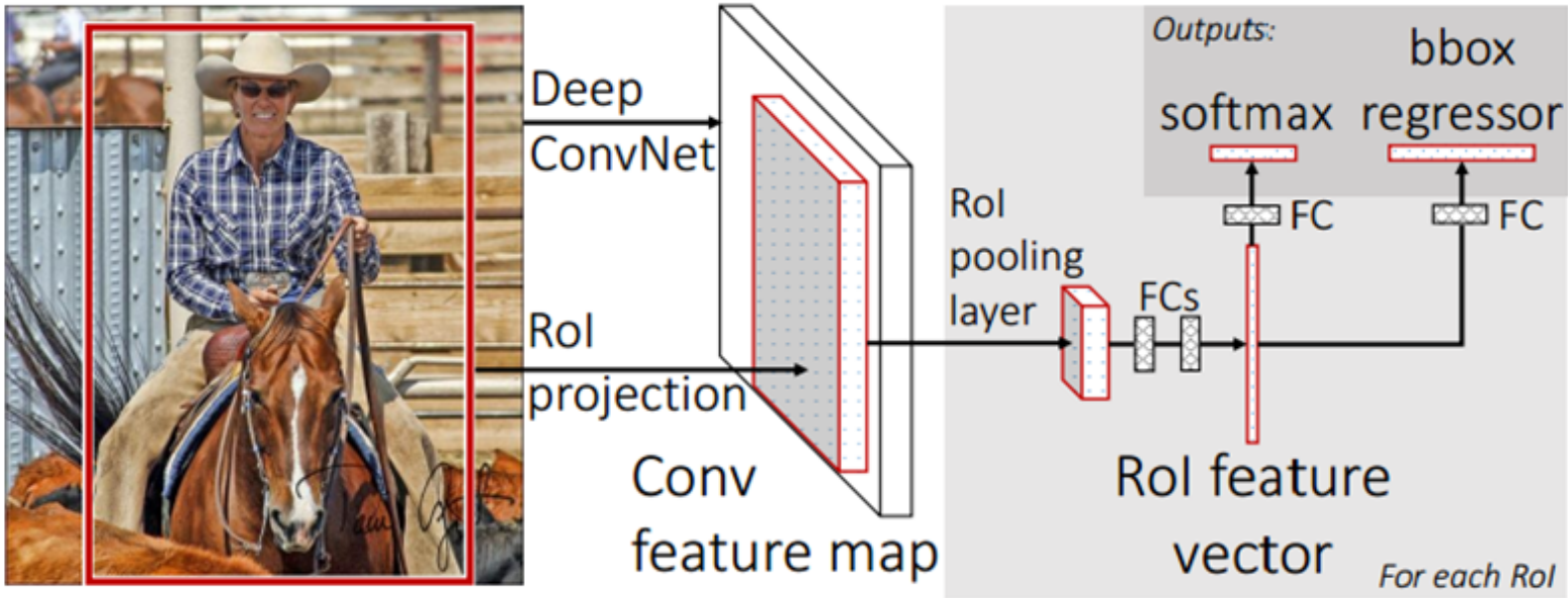
- Note that selective search produces *many* region proposals
- Multiple stages must trained independently
  - Fine-tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressions (least squares)
- Training is slow (84h), takes a lot of disk space
- R-CNN runtime roughly 47 seconds per image (!)
- Inference performance improved by spatial pyramid pooling networks SPPnets[11] which share convolutions between ROIs
- Difficult training and slow inference motivates an update . . .
- For more details check out Girshick's ICCV 2015 tutorial

[11] He et al., Spatial Pyramid Pooling in Deep Convolutional Networks, 2014

The R-CNN Family: Fast R-CNN (2015)

- Fast R-CNN is a combines stages 2 and 3 of R-CNN and is trained with multi-task loss (log loss and smooth L1 loss)
- A Fast R-CNN network takes as input an image and a set of object proposals
- The network first processes the whole image with conv and pooling layers to produce a conv feature map.
- For each object proposal an region of interest *ROI* pooling layer extracts associated features from the conv map.
- Each feature vector is fed into a fully connected (*fc*) layer that finally branches into two sibling output layers:
  - A softmax layer producing classification labels
  - A linear layer producing bounding box positions
- Higher detection quality (mAP) than R-CNN and SPPnet

The R-CNN Family: Fast R-CNN



Fast R-CNN workflow

Figure 22: Girshick, Fast R-CNN, 2015

Fast R-CNN achieved the top results on PASCAL VOC12 at the time with an mAP of 68.4% but still inference time is about 2.3 seconds per image (2 sec/image for region proposal generation)

The R-CNN Family: Fast R-CNN

Can we do better . . . ?

The R-CNN Family: Fast**er** R-CNN

- Fast R-CNN is still using independent region proposal stage.
- As you might guess, the Fast**er** R-CNN revision will introduce a Region Proposal Network (RPN) that shares full-image convolutional features with the detection network.
- The RPN simultaneously predicts object bounds and *objectness* scores at each position.
- The RPN is trained end-to-end to generate high-quality region proposals which are used by Fast R-CNN
- The RPN and Fast R-CNN are merged into a single network by sharing their convolutional feature. Using "*attention*" mechanisms, the RPN component tells unified network where to look.
- Multi-task training with 4 loss functions (obj/not obj, ROI bbox, classify, final obj bbox)

The R-CNN Family: Faster R-CNN

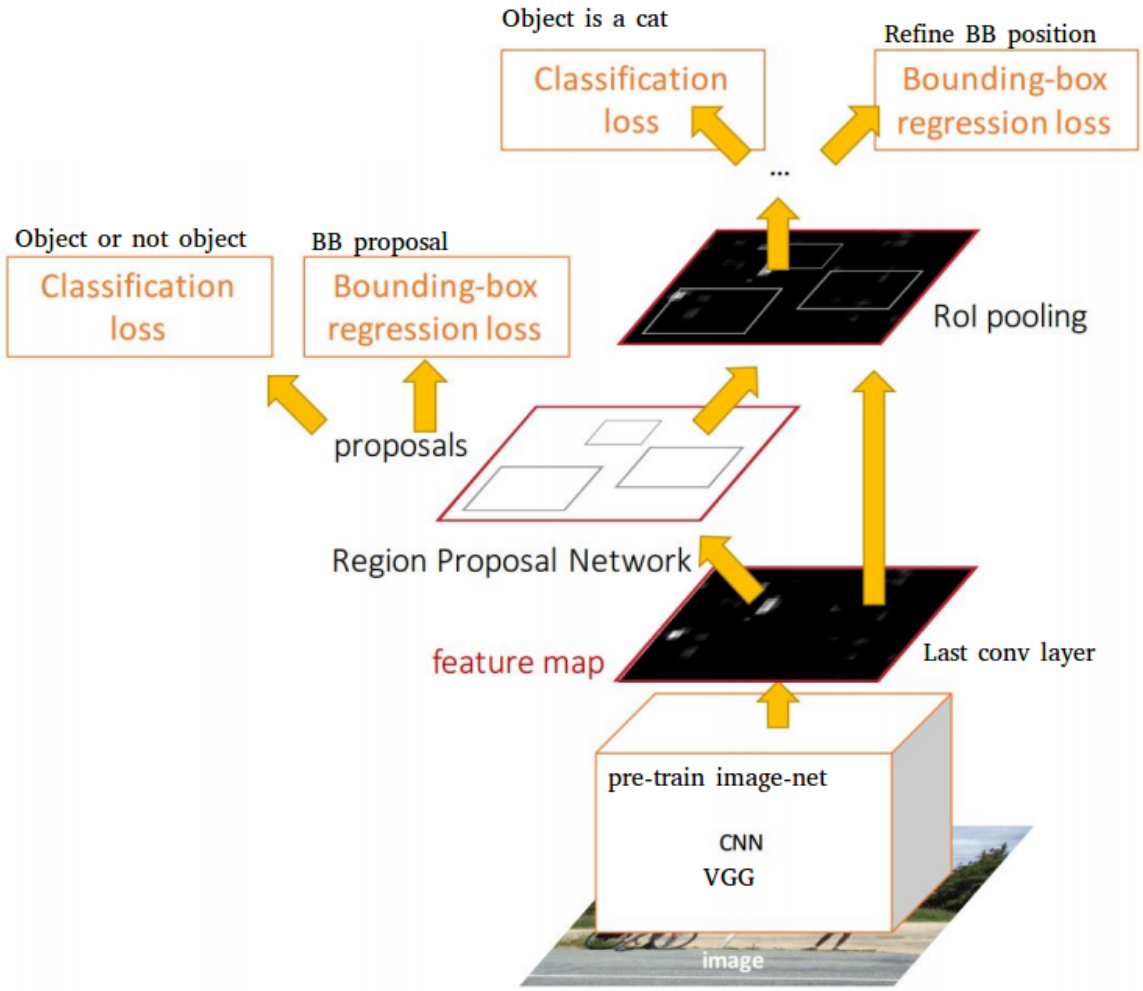


Figure 23: Leonardo Santos, Object Localization and Detection

The R-CNN Family: Fast**er** R-CNN

- Fast**er** R-CNN based on VGG-16 model, the detection system has a frame rate of 5 fps on a GPU (i.e. 200 ms)
- Fast**er** R-CNN generates about 300 proposals per image
- Fast**er** R-CNN, at the time, achieved state-of-the-art object detection accuracy on PASCAL VOC12 and MS COCO datasets
- All codes for Fast**er** R-CNN are available on GitHub here

The R-CNN Family: Faster R-CNN

Can we do *even* better . . . ?

The R-CNN Family: Mask R-CNN

- At the time of writing, Mask R-CNN (2017) is gaining significant popularity.
- As the name implies, Mask R-CNN is an R-CNN derivative combining the best of Feature Pyramid Pooling (FPN), Fully Convolutional Networks (FCNs), and Residual networks all together under the Fast**er** R-CNN architecture.
- Furthermore, Mask R-CNN extends Fast**er** R-CNN by adding an additional branch for predicting an object mask (i.e. pixel segmentation) in parallel with the existing branch for bounding box recognition.
- Mask R-CNN achieves top marks in all three tracks of the COCO suite of challenges, including instance segmentation, bounding-box object detection, and person keypoint detection.

The R-CNN Family: Mask R-CNN

Results on COCO test images using ResNet-101-FPN at 5 fps.

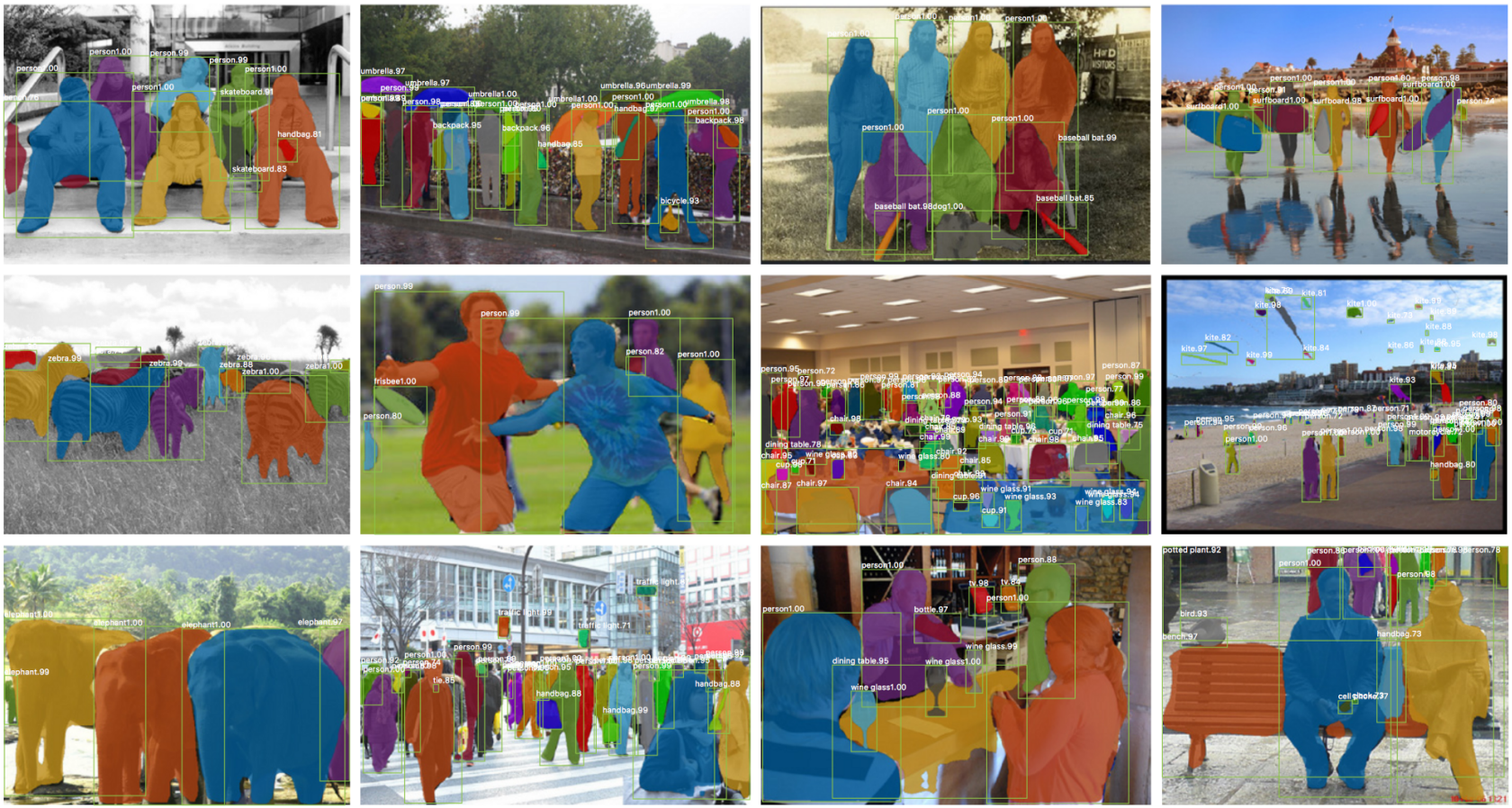


Figure 24: He et al., Mask R-CNN, 2017

Wrapping up: A Thriving Ecosystem

- The R-CNN family history is a classic example of traditional computer vision approaches incrementally adopting convolution neural networks to iteratively improve performance and expand algorithm capabilities
- However, the R-CNN variety architecture is only one of *many* different approaches for object detection and segmentation
- Other key CNN based contributions include:
  - Fully Convolutional Networks (2014)
  - Learning Deconvolution Networks (2015)
  - Single-Shot Deep MultiBox (2015)
  - SegNet: an Encoder-Decoder Architecture (2015)
  - DeepLab: Atrous Convolutions and CRFs (2016)
  - The FB suite: DeepMask, SharpMask, & MultiPath (2016)
  - TensorFlow Object Detection API (2017)

Detection and Segmentation Atlas

There are too many outstanding contributions to cover in a single deck. Here is a brief high-level overview intended to provide broader context and help guide additional exploration.

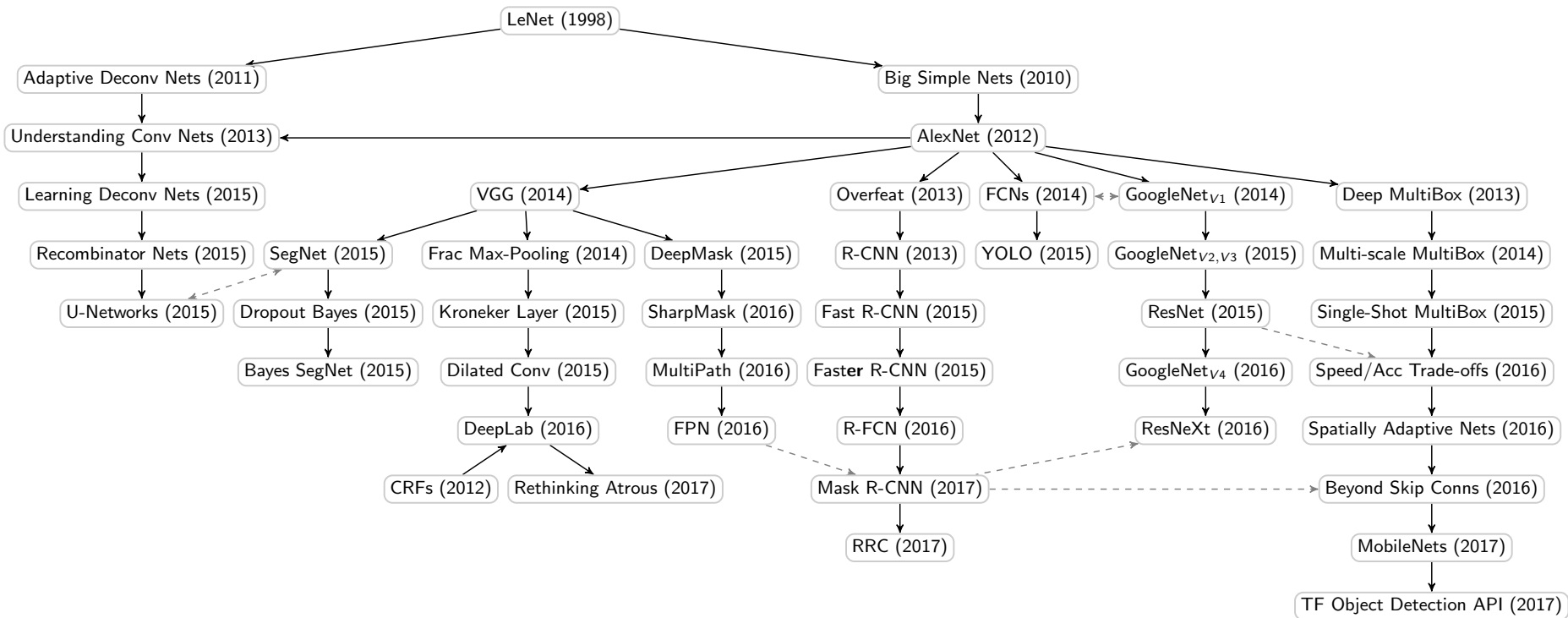


Figure 25: Nodes are hyperlinks to the associated arXiv.org papers.

Public Datasets

Inria Aerial Image Labeling	2017	paper	data
DAVIS Challenge	2017	paper	data
Mapillary Vistas	2017	paper	data
ADE20K	2016	paper	data
SYNTHIA	2016	paper	data
SpaceNet	2016	N/A	data
Playing for Data	2016	paper	data
SUN RGB-D	2015	paper	data
Cityscapes	2015	paper	data
Common Objects in Context (COCO)	2014	paper	data
Oxford RoboCar Dataset	2014	paper	data
KITTI Vision Benchmark Suite	2012	paper	data
Visual Object Classes 2012 (VOC12)	2012	[1][2]	data
NYU Depth Dataset V2	2012	paper	data
Caltech Pedestrian Detection Benchmark	2009	[1][2]	data
CamVid: Motion-based Segmentation	2008	paper	data