Object Detection and Segmentation

Tran Giang Son, tran-giang.son@usth.edu.vn

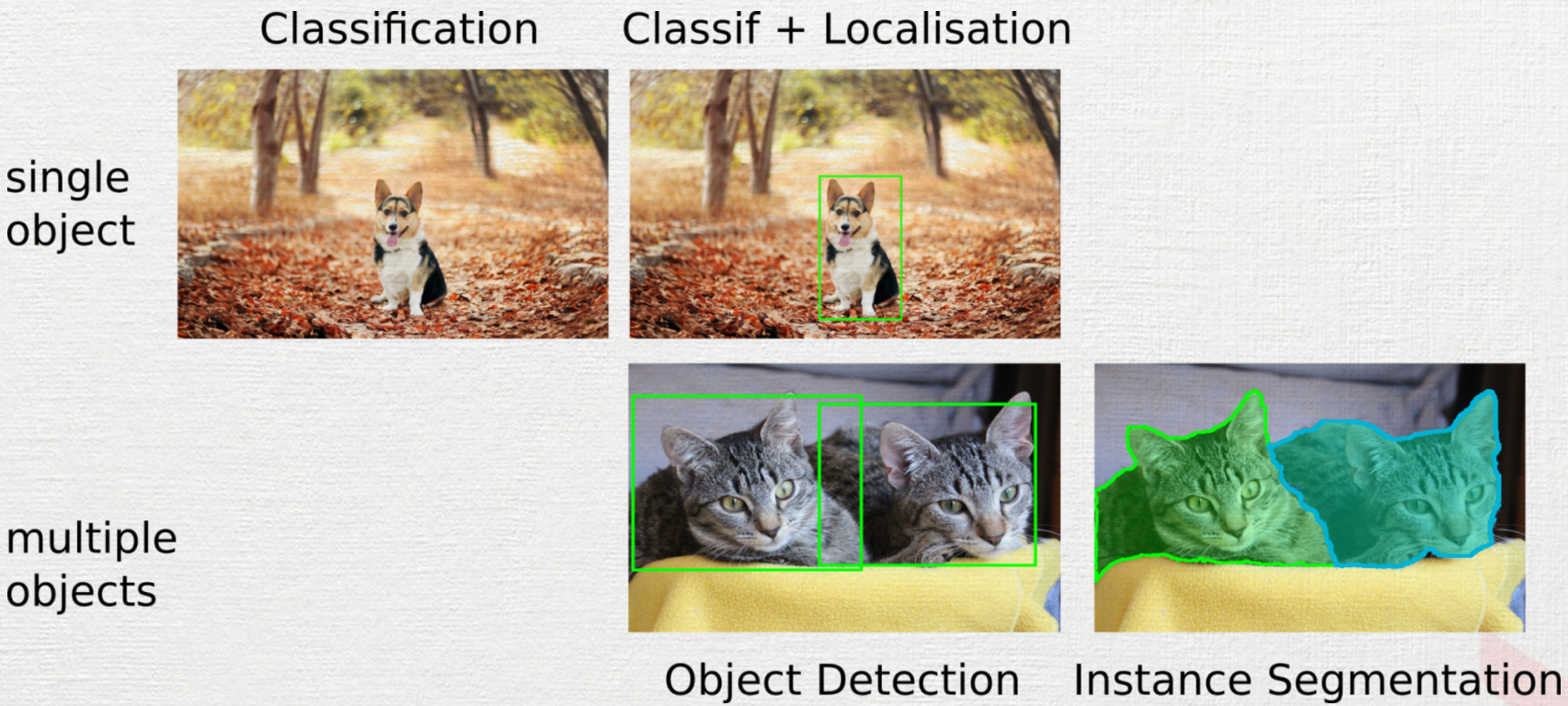
ICT Department, USTH

Introduction

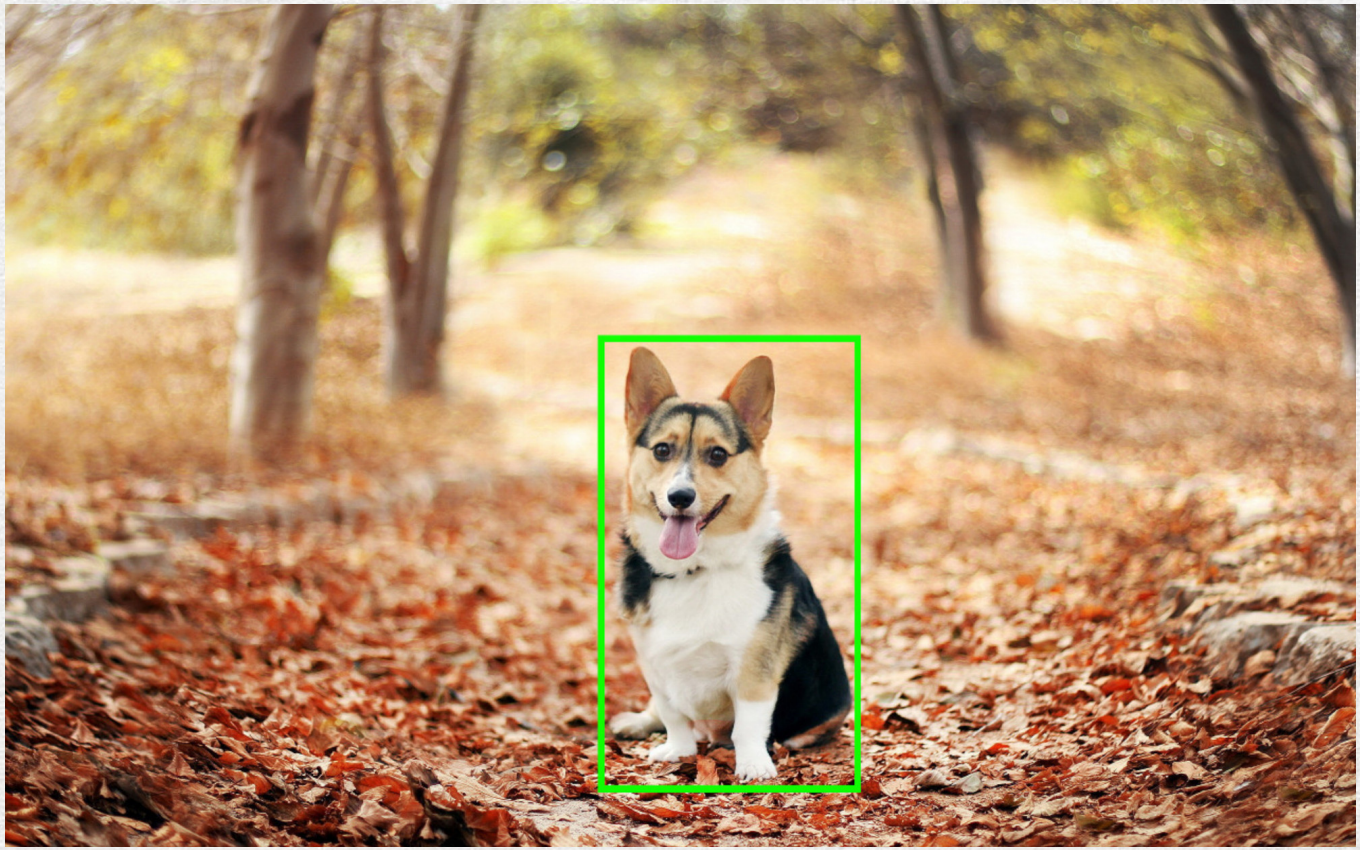
Problem of Image Classification

- Mostly on centered images
- Only a single object per image
- Cannot solve many real life vision tasks

Problem of Image Classification

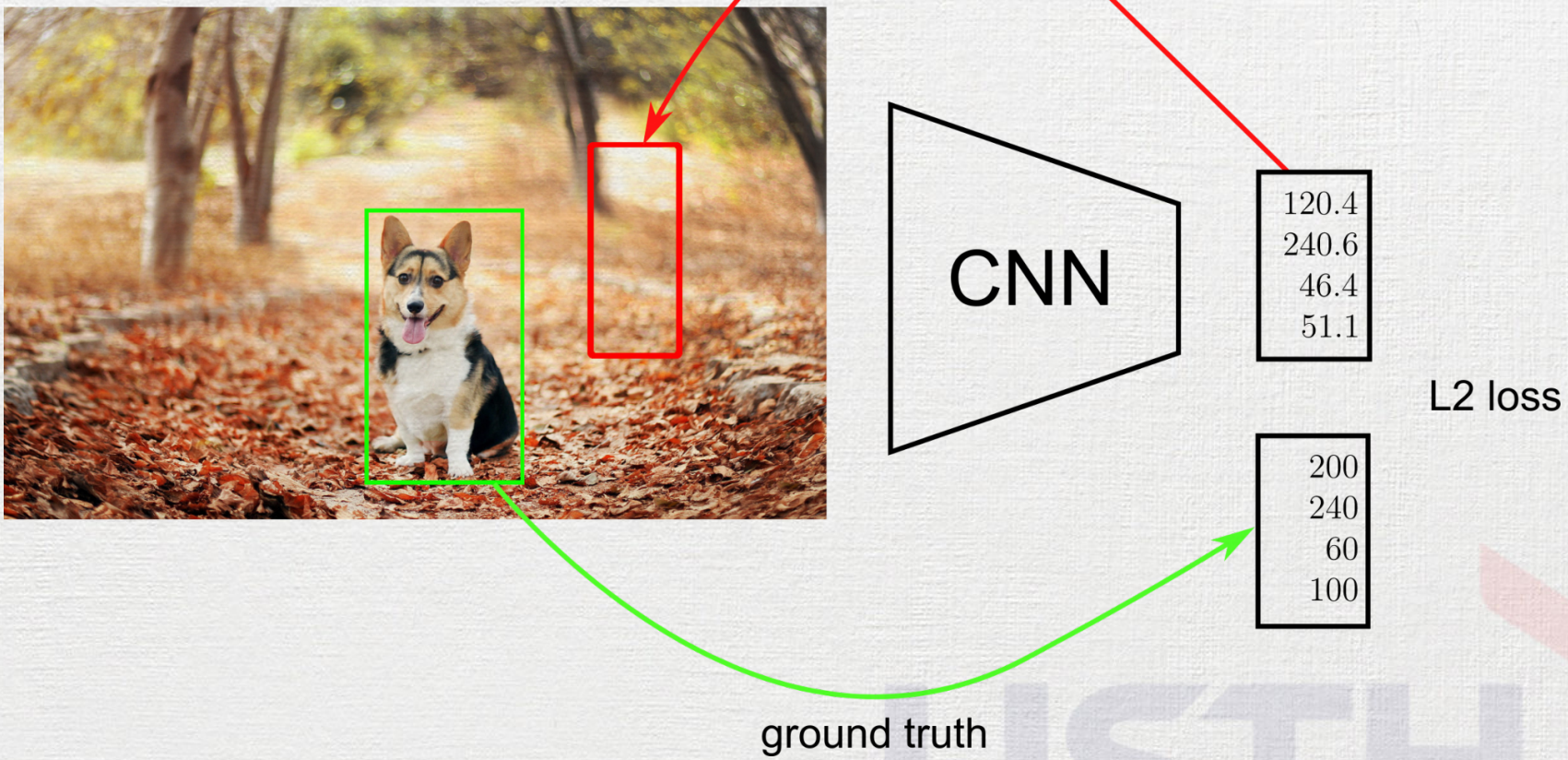


Localization

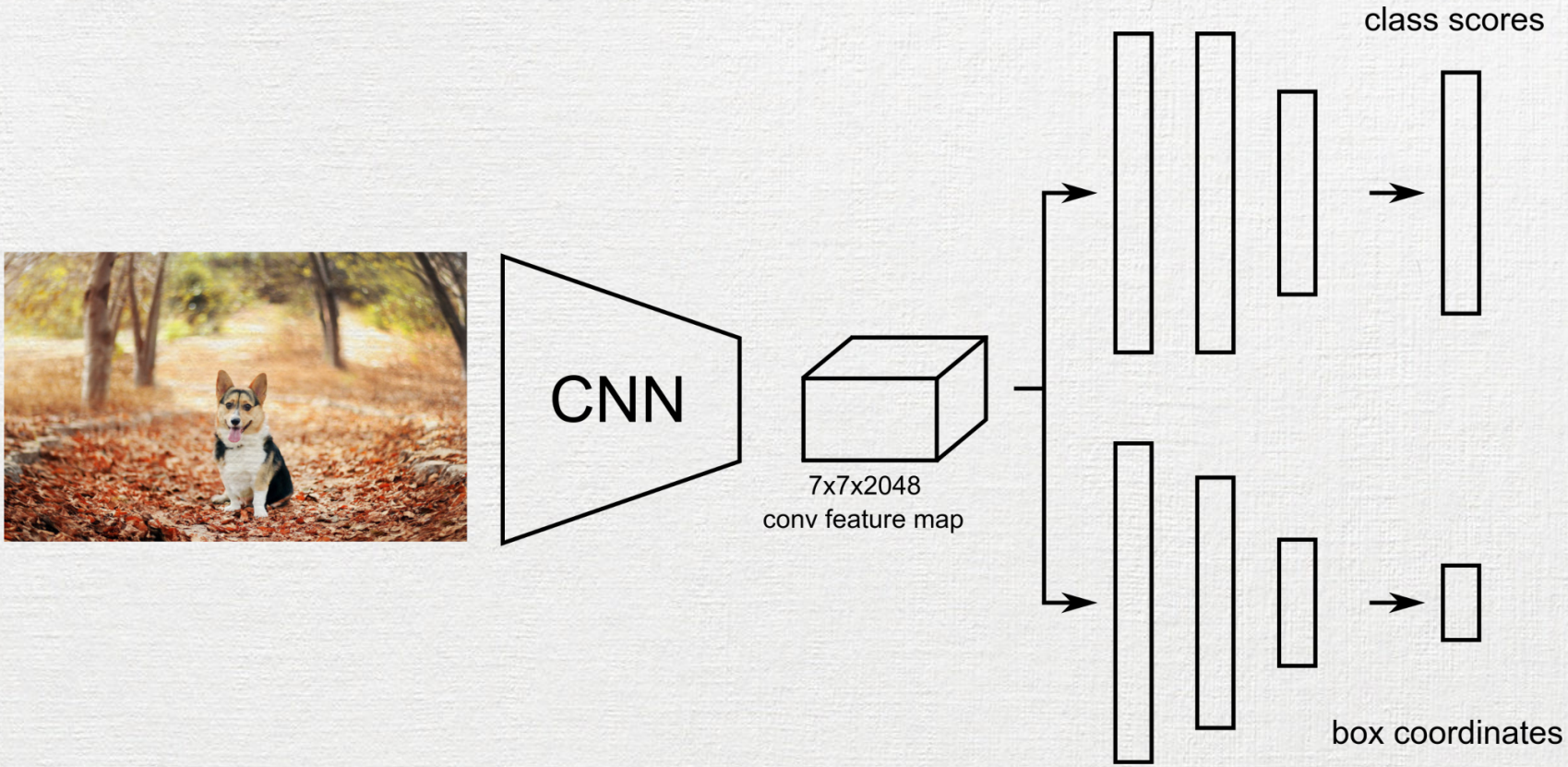


- Single object per image
- Predict coordinates of a bounding box (x, y, w, h)
- Evaluate via the metric Intersection over Union (IoU)

Localization as Regression

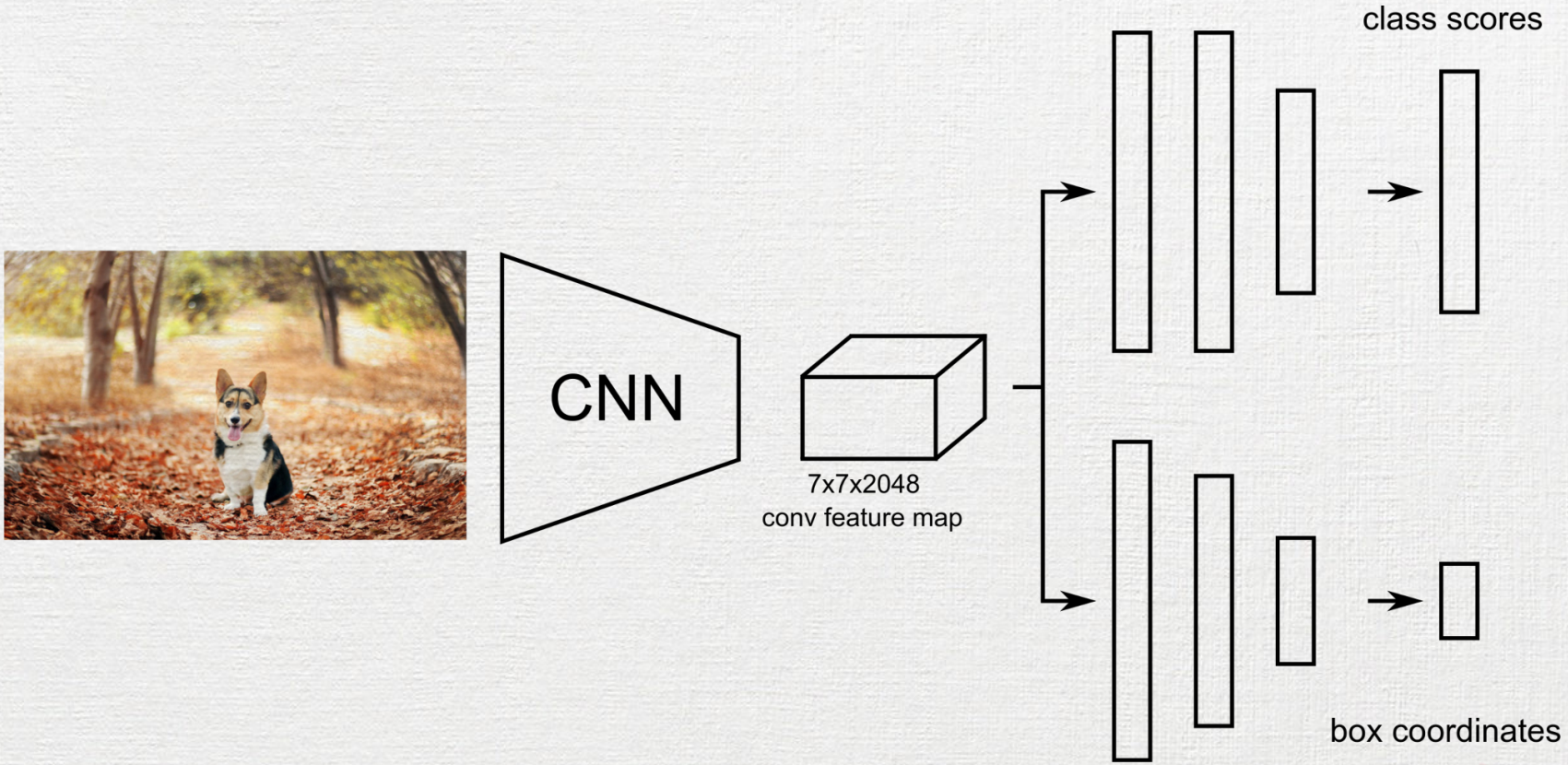


Localization and Classification



- Use a pre-trained CNN, e.g. ResNet
- The "localization head" is trained separately with regression
- Possible end-to-end fine-tuning of both tasks

Localization and Classification



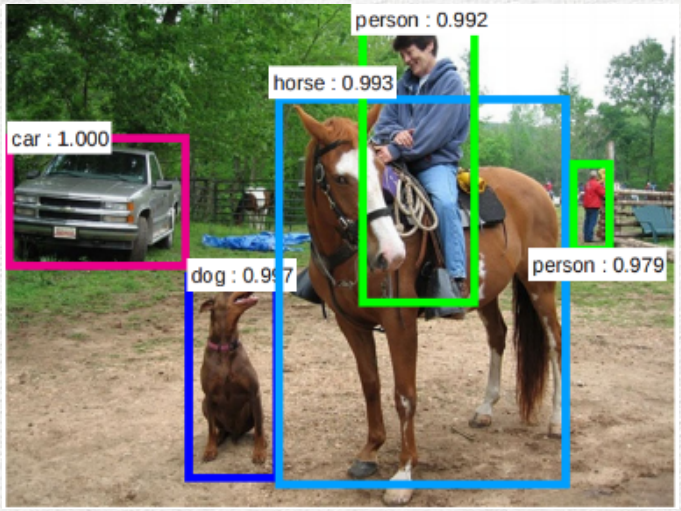
- C classes, 4 output dimensions (for 1 bounding box)
- Predict exactly N objects: predict $(N \times 4)$ coordinates and $(N \times K)$ class scores

Object Detection

Object Detection

- Problem: we don't know in advance the number of objects in the image
- Object detection performs two main tasks:
  - Object proposal: find regions of interest (ROIs) in the image
  - Object classification: classify the object in these regions

Object Detection



- Two main tasks in object detection can be expressed as follows:
- Define bounding boxes of the objects
- For each bounding box, classify the object inside it to some classes (e.g. dog, horse, person, car, etc. . .) with % of confidence

Object Detection

- Two main families to perform object detection:
  - Single-Stage: A grid in the image where each cell is a proposal
  - Two-Stage: Region proposal then classification (Faster R-CNN)
- Popular models
  - Faster R-CNN
  - YOLO
  - RetinaNet

Faster R-CNN

- Three main iteraions:
  - R-CNN
  - Fast R-CNN
  - Faster R-CNN

Faster R-CNN

- Instead of having a predefined set of box proposals, find them on the image by:
  - Selective Search - from pixels (not learnt, not used any more)
  - Faster R-CNN - Region Proposal Network (RPN)

Faster R-CNN

- Crop-and-resize operator (RoI-Pooling)
  - Input: Convolutional map + N regions of interest
  - Output: tensor of $N \times 7 \times 7 \times D$ (where D is depth) boxes
  - Allow to propagate gradient only on interesting regions, and efficient computation

R-CNN

- R-CNN (Region with CNN feature) algorithm:
  - Step 1: use Selective Search algorithm to get around 2000 bounding box in the input image which can contain the object
  - Step 2: with each bounding box, identify its class (e.g. person, car, etc. . .)

R-CNN

- Step 1: Selective Search
  - Input: color image
  - Output: around 2000 region proposal (bounding box) which can contain the objects

R-CNN

Input Image

Output Image

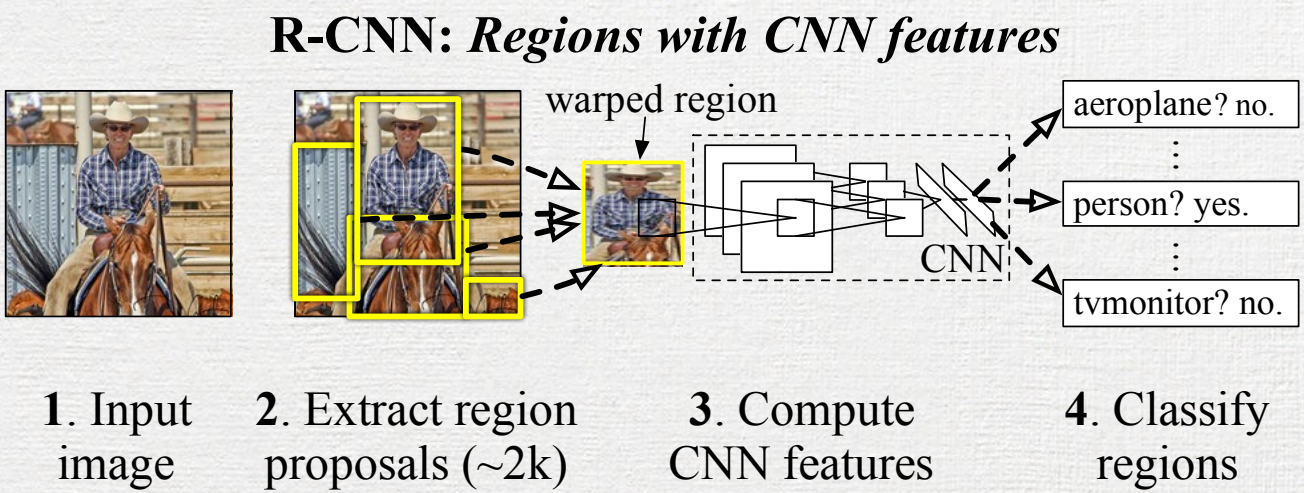
Image is segmented using the Graph Based Image Segmentation algorithm

- Cannot use 1 color to present 1 region proposal
  - Each object can contain several colors
  - Some parts of an object can be hidden by others

R-CNN

- Problem becomes Region Proposal Classification
- Issue: in 2000 region proposals output from selective search algorithm, there exists region proposal without any objects
- Solution: one background class is added to solve the issue

R-CNN



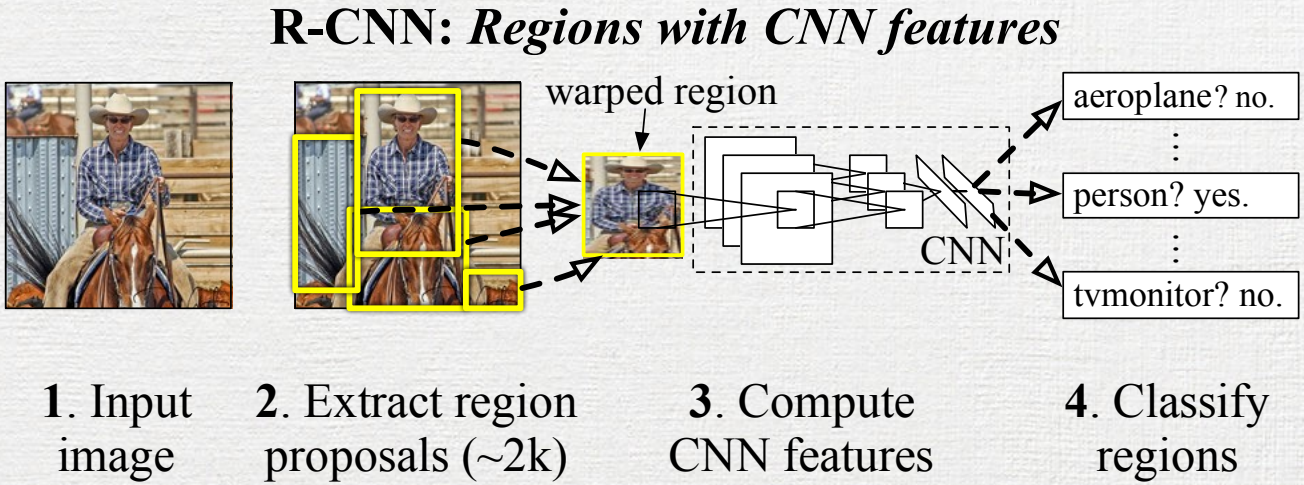
(1) Input image

(2) Extract region proposals, resize region proposals to same size

(3) Transfer learning with feature extractor

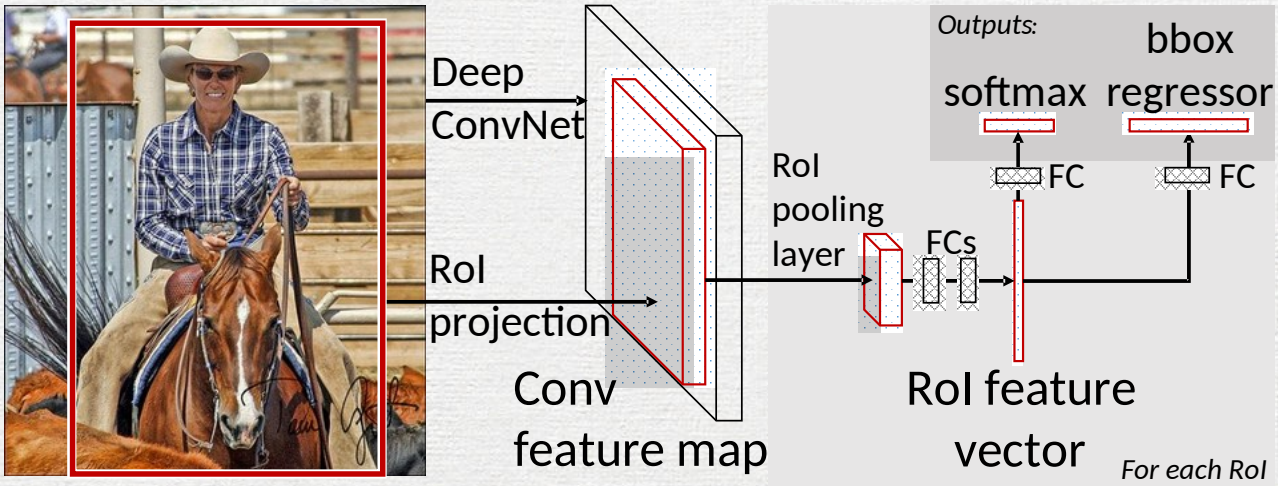
(4) Use SVM to classify the regions as person, or horse or background, etc.

R-CNN



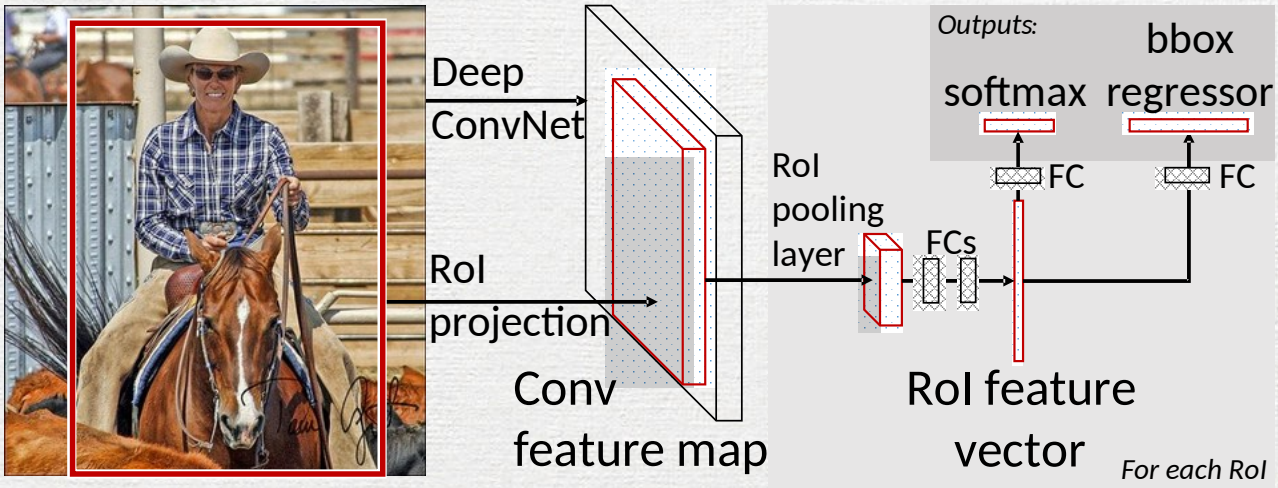
- Problem:
  - For each image, need to classify 2000 region proposals
  - Very long training time
  - Cannot apply real-time object detection

Fast R-CNN



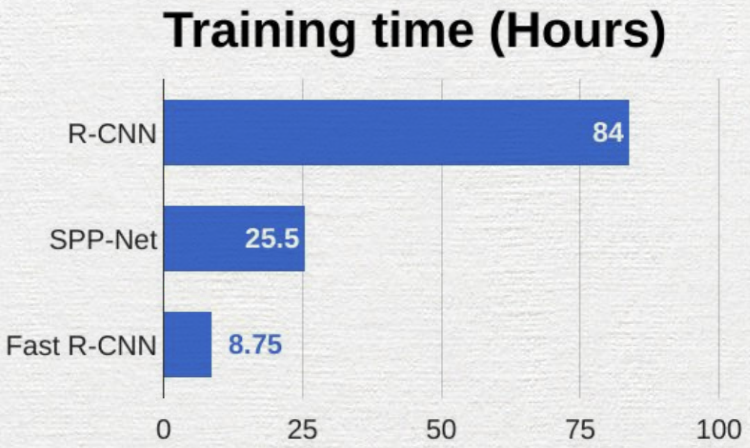
- The whole input image is fed into Deep ConvNet to produce Conv feature map
- RoI is projected into Deep ConvNet with the input image
  - Region proposals are obtained from Conv feature map

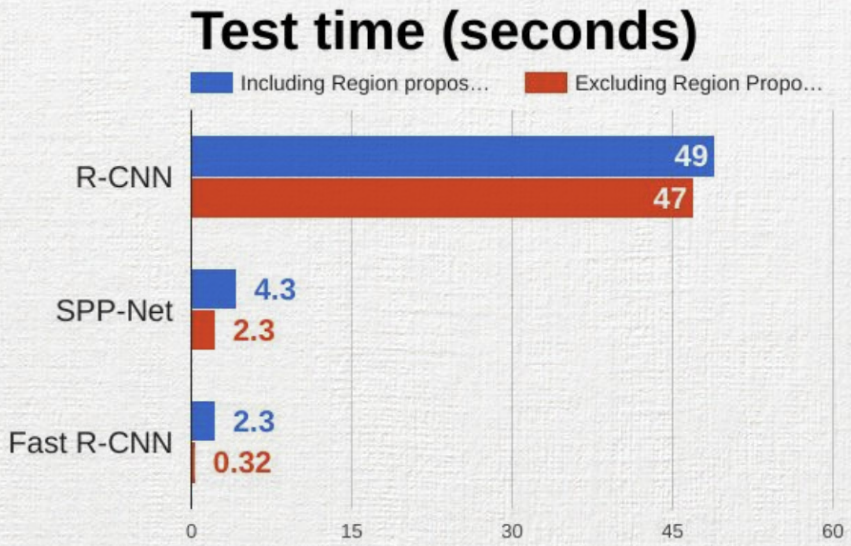
Fast R-CNN



- RoI pooling layer is used to transform region proposals in the Conv feature maps to the same size
- Region proposals are flattened and fed into 2 FCs layers to predict class and regress the offset values of bounding box

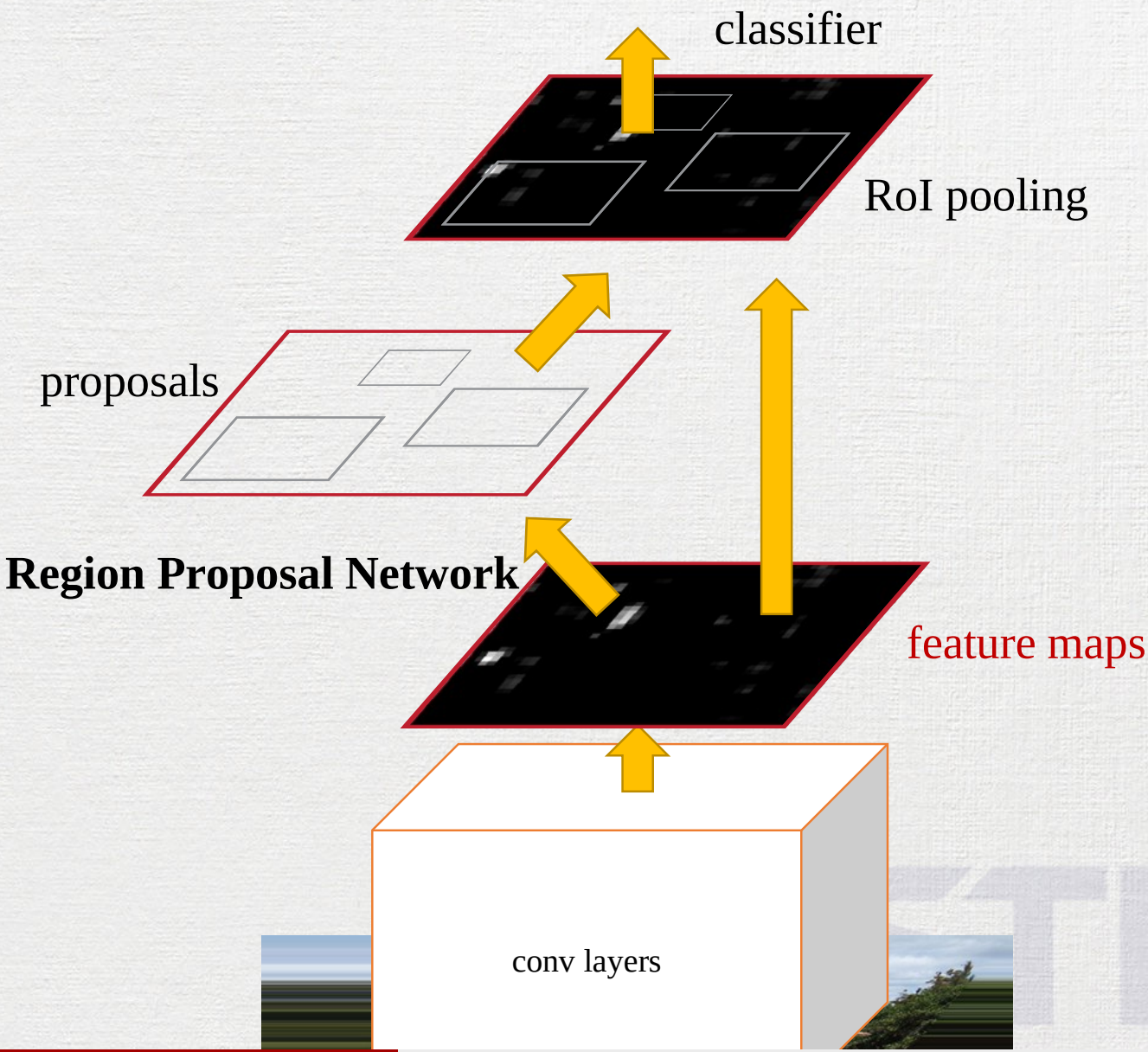
Fast R-CNN vs R-CNN





- Fast R-CNN is much faster than R-CNN
- Test time stills takes Selective search for region proposals
- Deep learning-based method for region proposal?

Faster R-CNN


classifier
RoI pooling
proposals
Region Proposal Network
feature maps
conv layers

Faster R-CNN

- Replace selective search algorithm to obtain region proposals from feature maps
- Input: feature map
- Output: region proposals
  - Anchor box is used to present region proposal

Faster R-CNN


- Each anchor box is defined by 4 paramaters (x_center, y_center, width, height)
- Number of of anchors are pre-defined
  - After passing through RPN, only anchor box containing objects are kept

Faster R-CNN

- Feature maps are fed into Conv layer 3×3, 512 kernels
- With each anchor, RPN calculates two steps:
  - Predict if anchor is foreground (contain object) or background (does not contain object)
  - Predict 4 offset values for x_center, y_center, width, height of anchor
- Non-maxima suppression is used to remove overlap anchor boxes
- Based on confidence score, RPN will get N (N can be 2000, 1000, etc.) anchor boxes to be the predicted region proposals

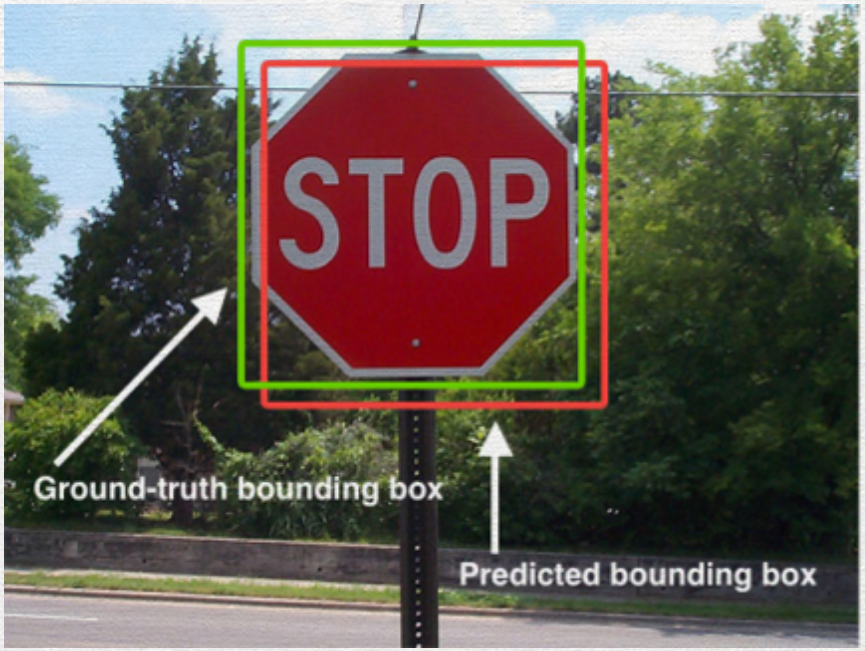
Faster R-CNN NMS

Non-maxima Suppression Algorithm:

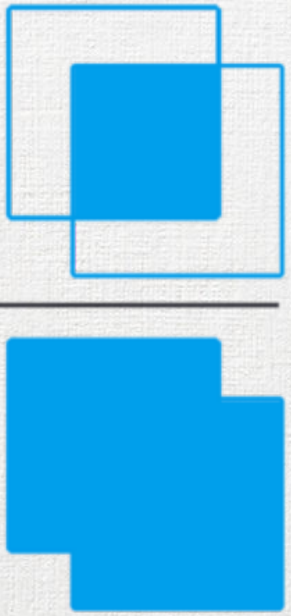
- Input: 9000 anchor boxes
- Output: 100 anchor boxes as region proposals
- Algorithm:
  (1) Choose anchor box (A) with a maximum value of foreground probability
  (2) Add anchor box A to the output set
  (3) Remove A and a set of anchor boxes in input set which has IoU value with A > 0.5
  (4) Check if input set is empty or output set is equal to 100, then stop, otherwise repeat step 1

Faster R-CNN

- Intersection over Union (IoU) Metric

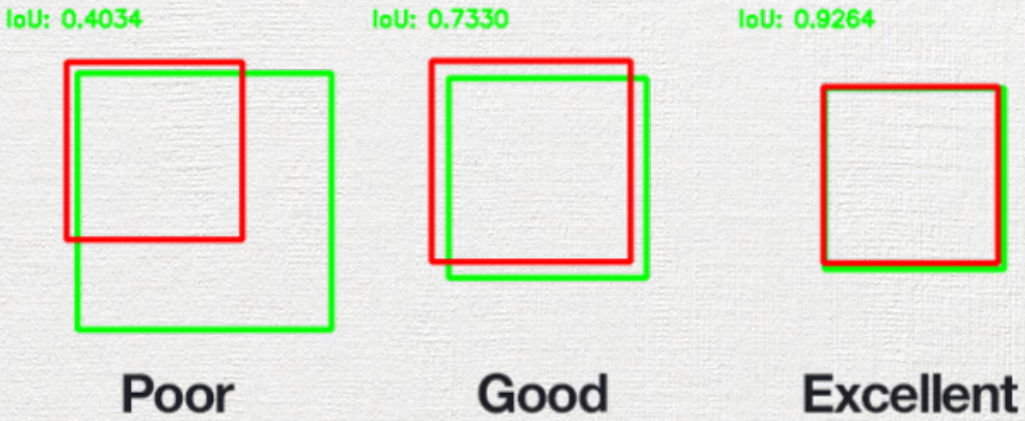


$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

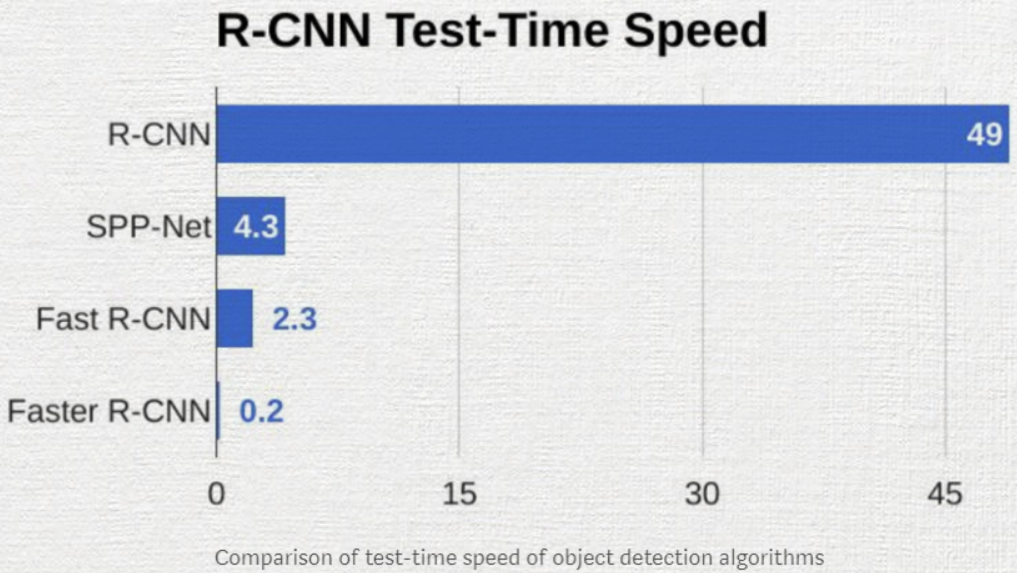


- $IoU \in [0, 1]$
- Higher IoU: predicted bounding box is closer to the ground truth

Faster R-CNN

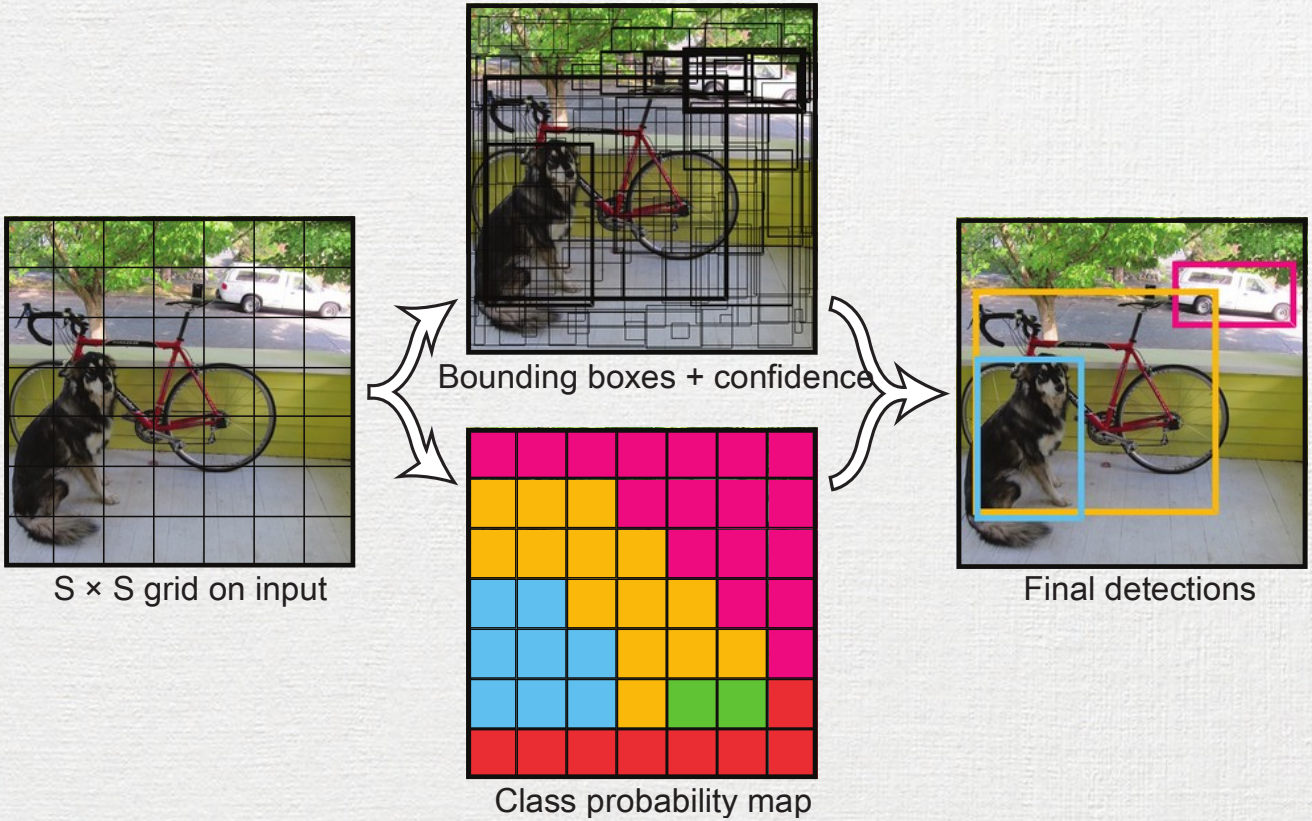


Faster R-CNN



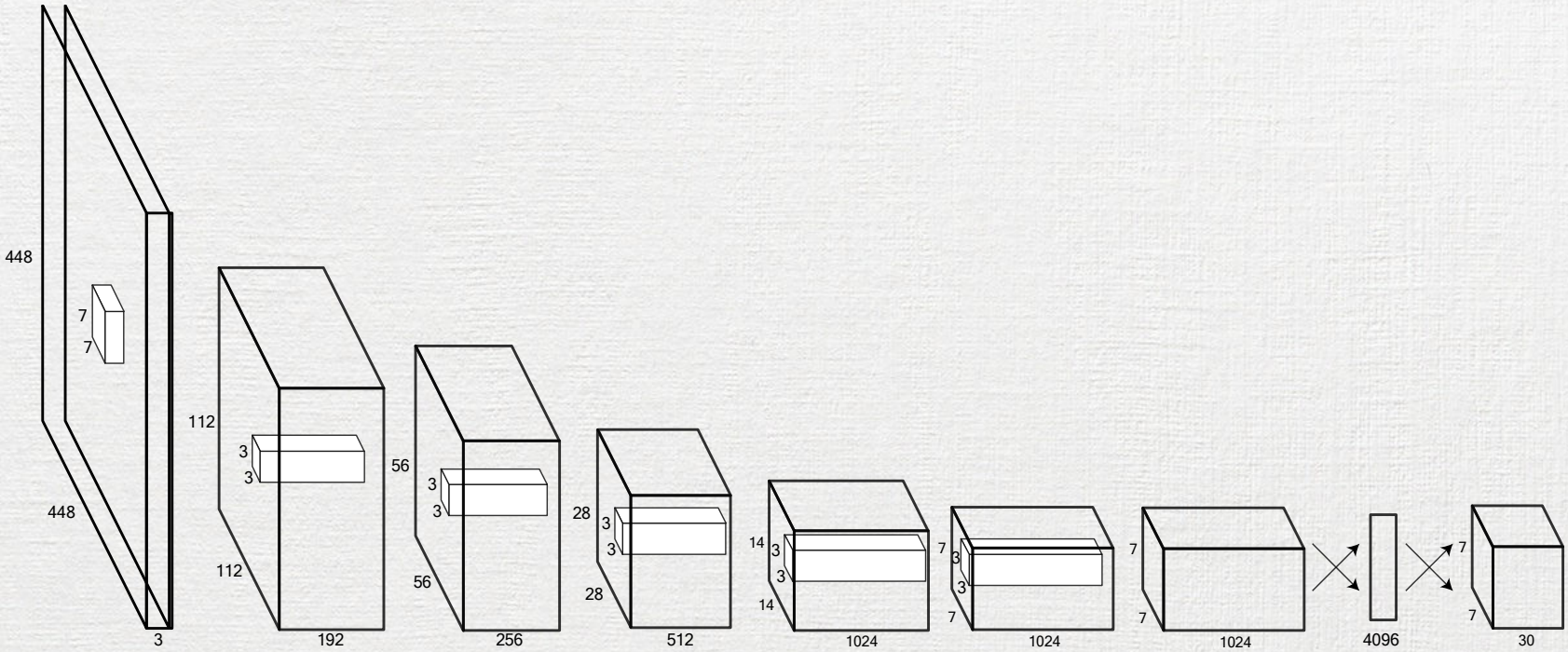
Comparison of test-time speed of object detection algorithms

YOLO



- For each cell of the S x S predict:
  - B boxes and confidence scores C (5 x B values) + classes c
- Final detections: $C_j * prob(c) > threshold$

YOLO



Conv. Layer
7x7x64-s-2
Maxpool Layer
2x2-s-2

Conv. Layer
3x3x192
Maxpool Layer
2x2-s-2

Conv. Layers
1x1x128
3x3x256
1x1x256
3x3x512
Maxpool Layer
2x2-s-2

Conv. Layers
1x1x256
3x3x512
} ×4
1x1x512
3x3x1024
Maxpool Layer
2x2-s-2

Conv. Layers
1x1x512
3x3x1024
} ×2
3x3x1024
3x3x1024-s-2

Conv. Layers
3x3x1024
3x3x1024

Conn. Layer

Conn. Layer

YOLO

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

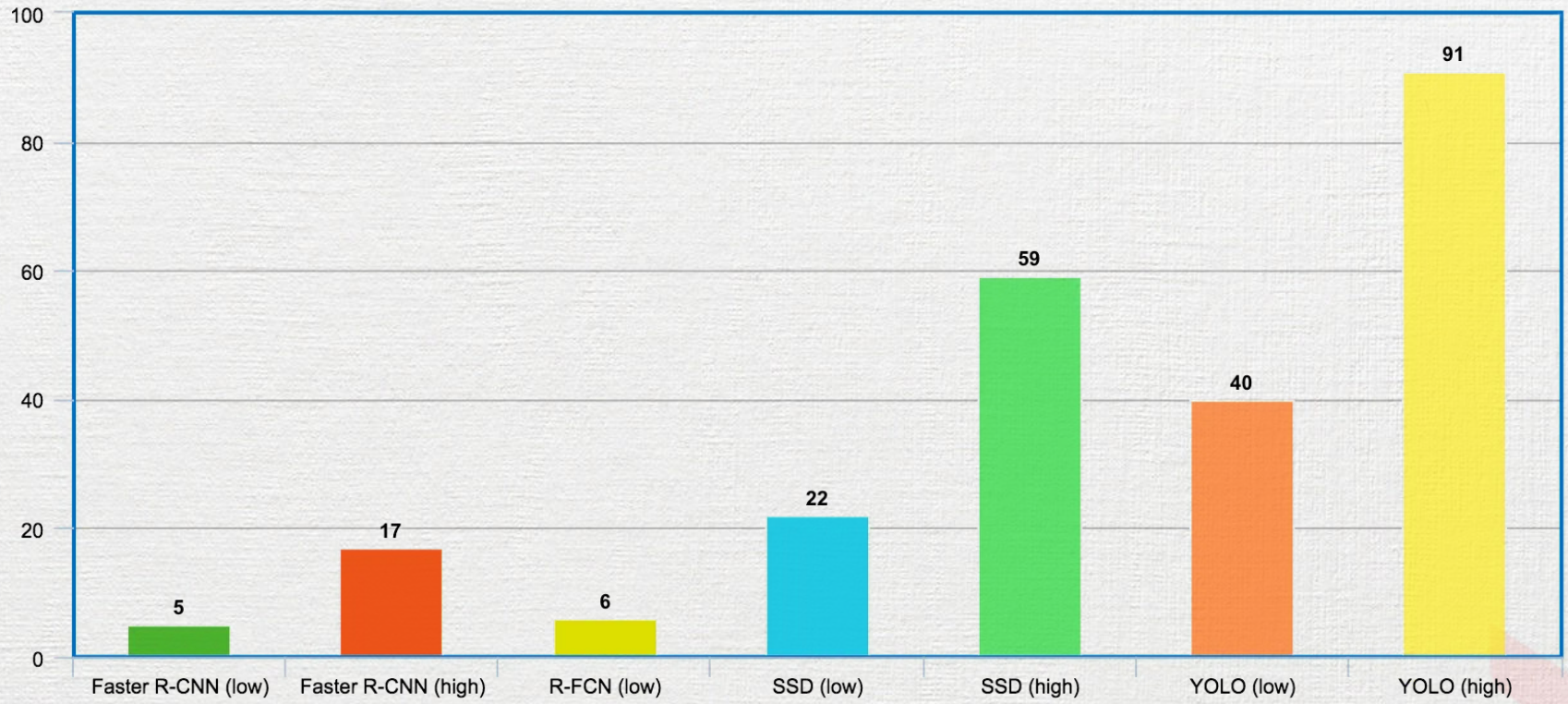
$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

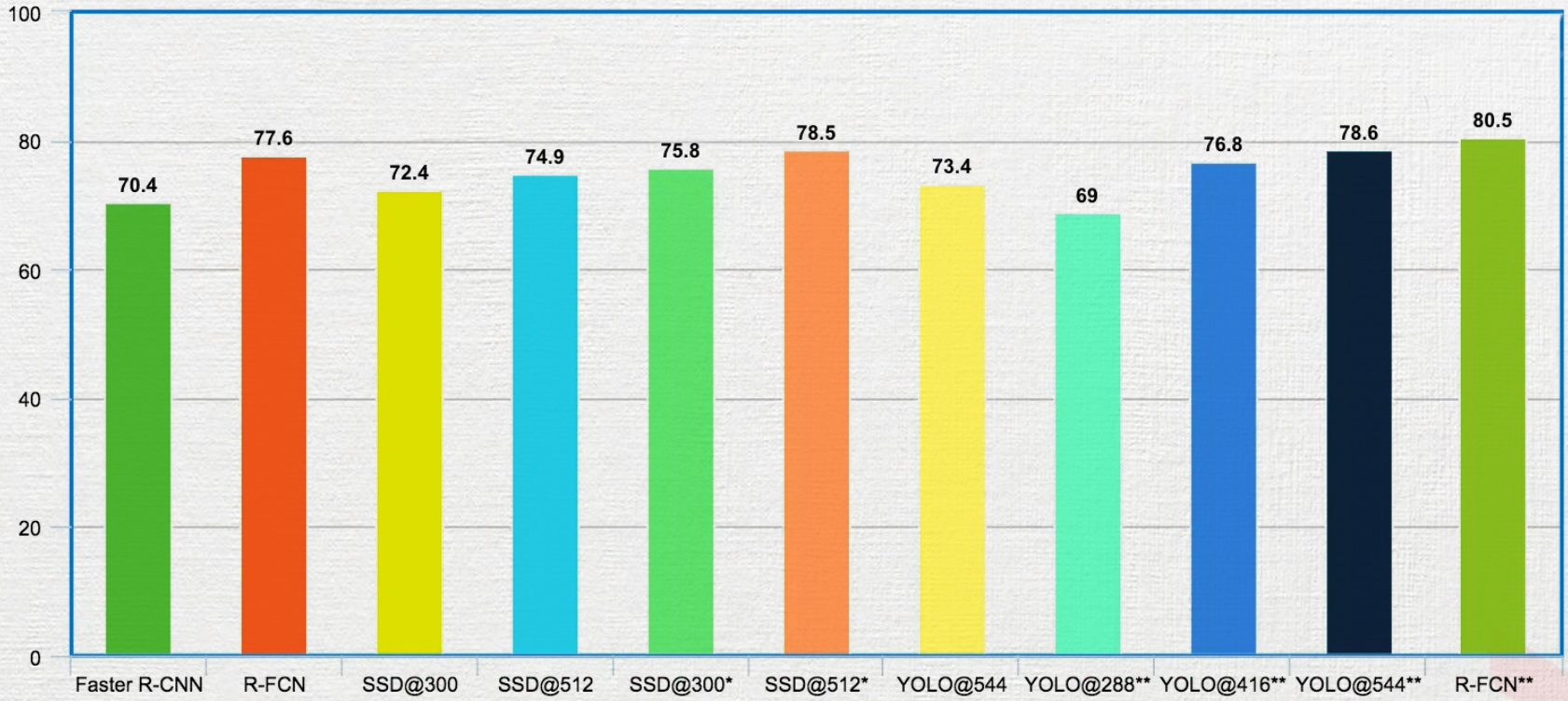
$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

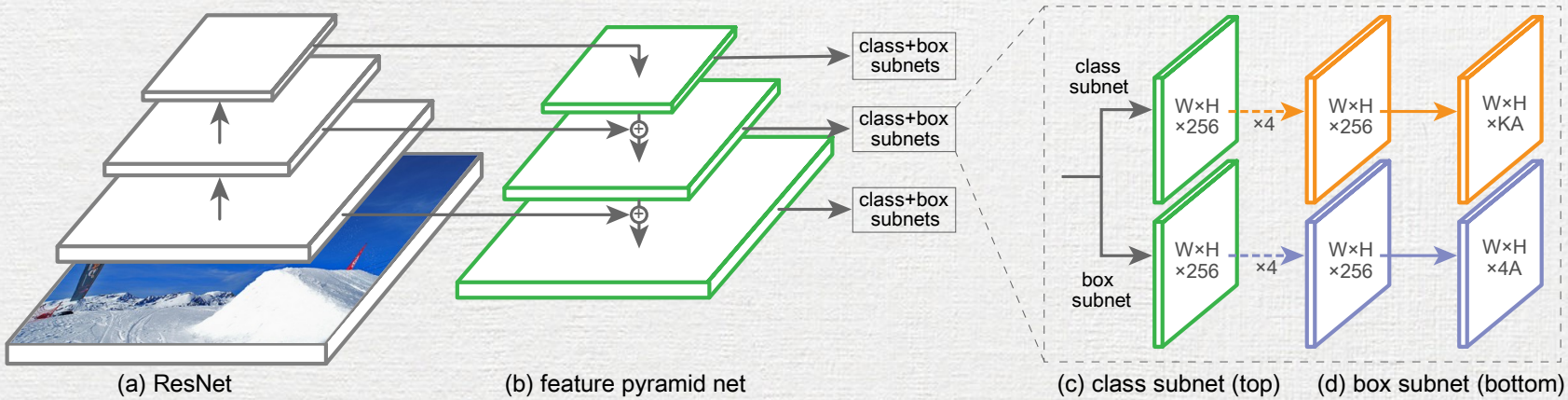
YOLO


100
80
60
40
20
0
5
17
6
22
59
40
91
Faster R-CNN (low)
Faster R-CNN (high)
R-FCN (low)
SSD (low)
SSD (high)
YOLO (low)
YOLO (high)

YOLO


100
80
60
40
20
0
70.4
77.6
72.4
74.9
75.8
78.5
73.4
69
76.8
78.6
80.5
Faster R-CNN
R-FCN
SSD@300
SSD@512
SSD@300*
SSD@512*
YOLO@544
YOLO@288**
YOLO@416**
YOLO@544**
R-FCN**

RetinaNet



- Single stage detector with:
- Multiple scales through a Feature Pyramid Network
  - Focal loss to manage imbalance between background and real objects

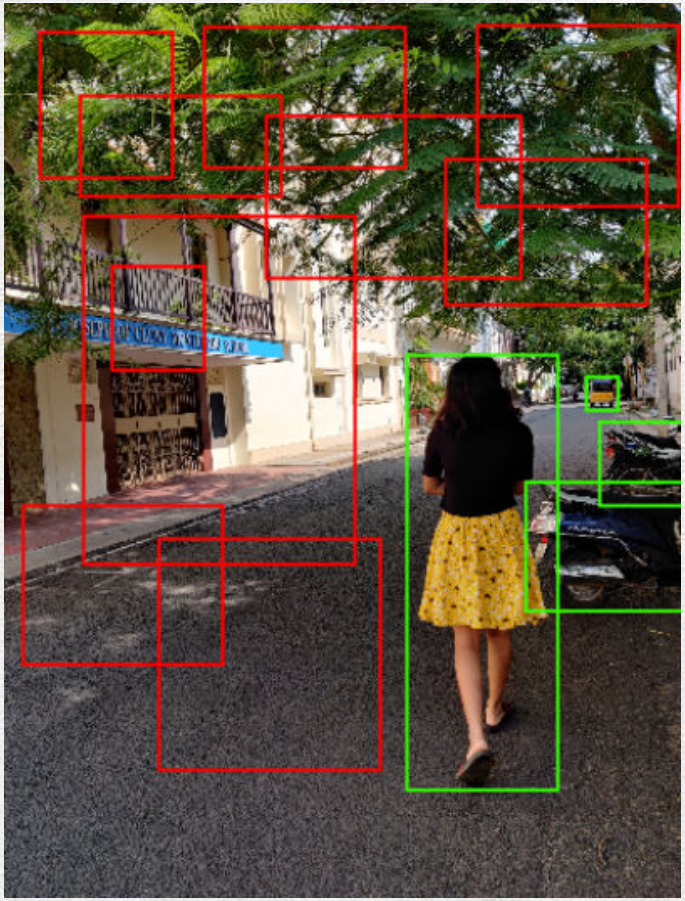
RetinaNet

Proposal comparison

- R-CNN/Fast R-CNN/Faster R-CNN: 1k-2k
- YOLOv1: 98 boxes
- YOLOv2: ~1k
- OverFeat: ~1–2k
- SSD: ~8–26k
- RetinaNet: ~100k

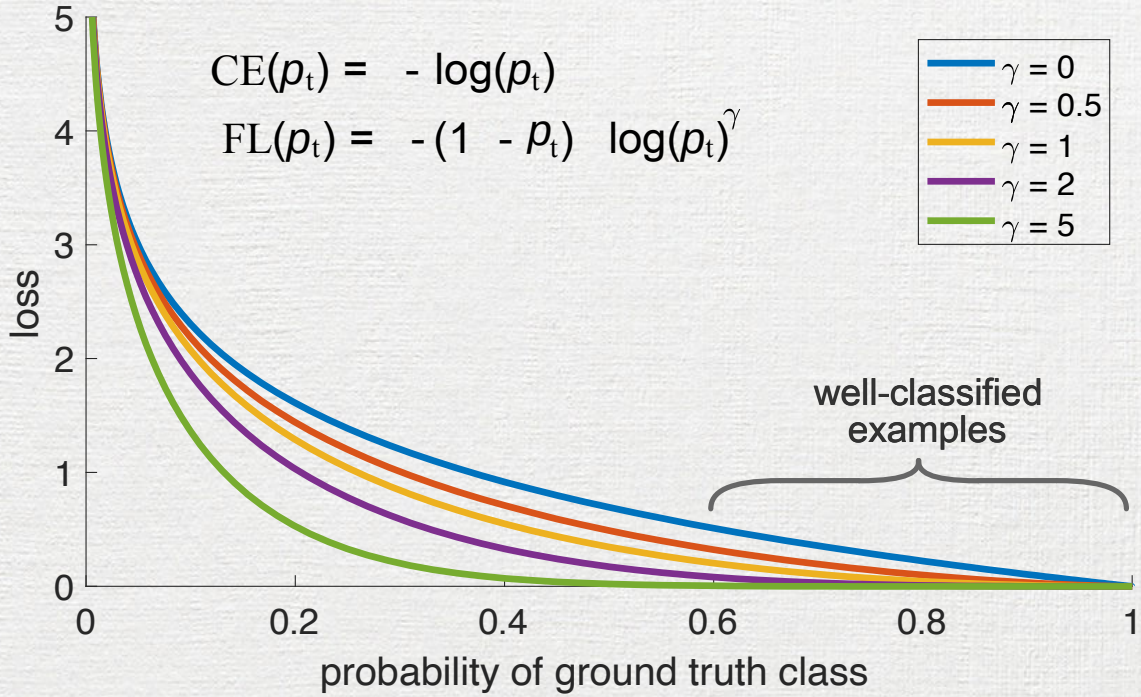
RetinaNet can have ~100k boxes with the resolve of class imbalance problem using focal loss.

RetinaNet



- Too many background proposals, too few object proposals
- Imbalance problem negatively impacts prediction results

RetinaNet



- Greatly penaltize negative classes

Segmentation

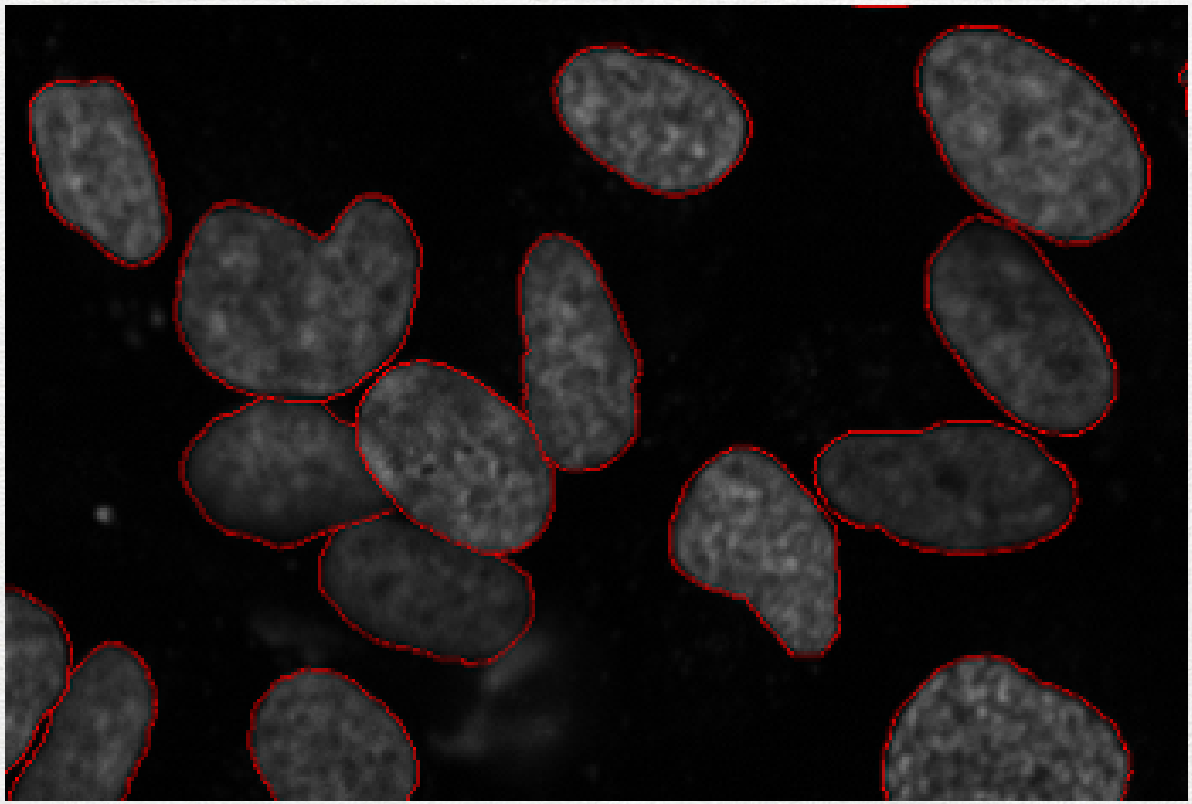
Segmentation

- Input: image
- Output: a class map for each pixel (here: dog vs background)

Instance Segmentation

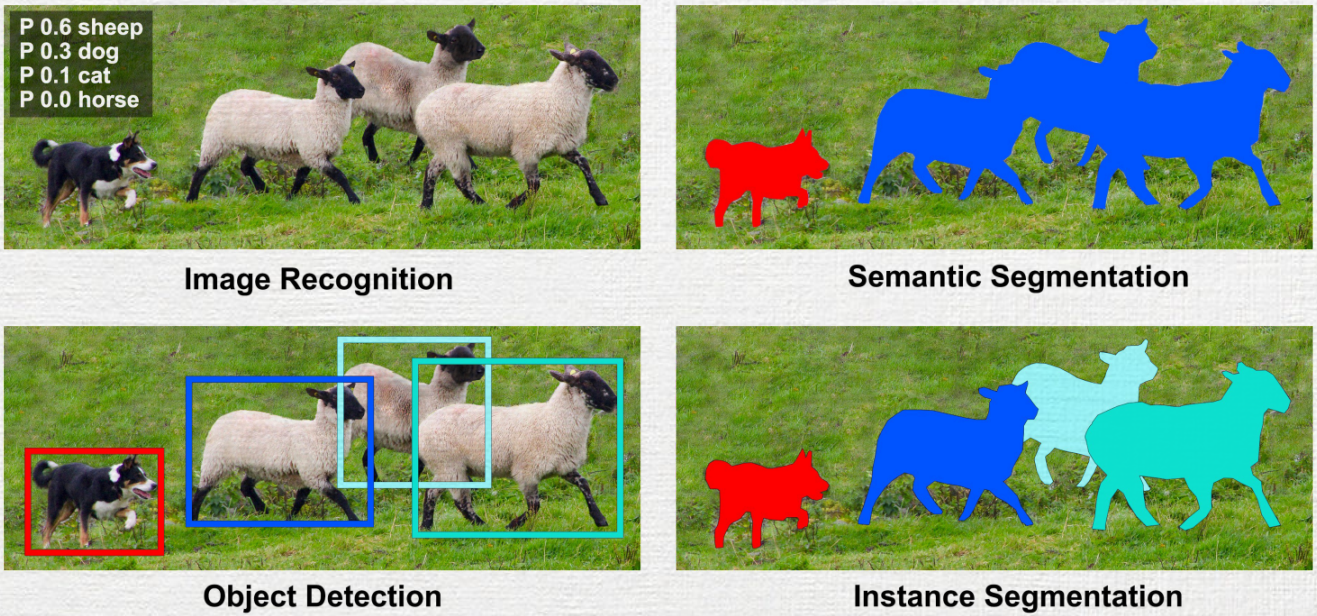
- Instance segmentation: specify each object instance as well
  - two dogs have different instances
- Simple method: object detection + segmentation

Object detection vs. Image segmentation



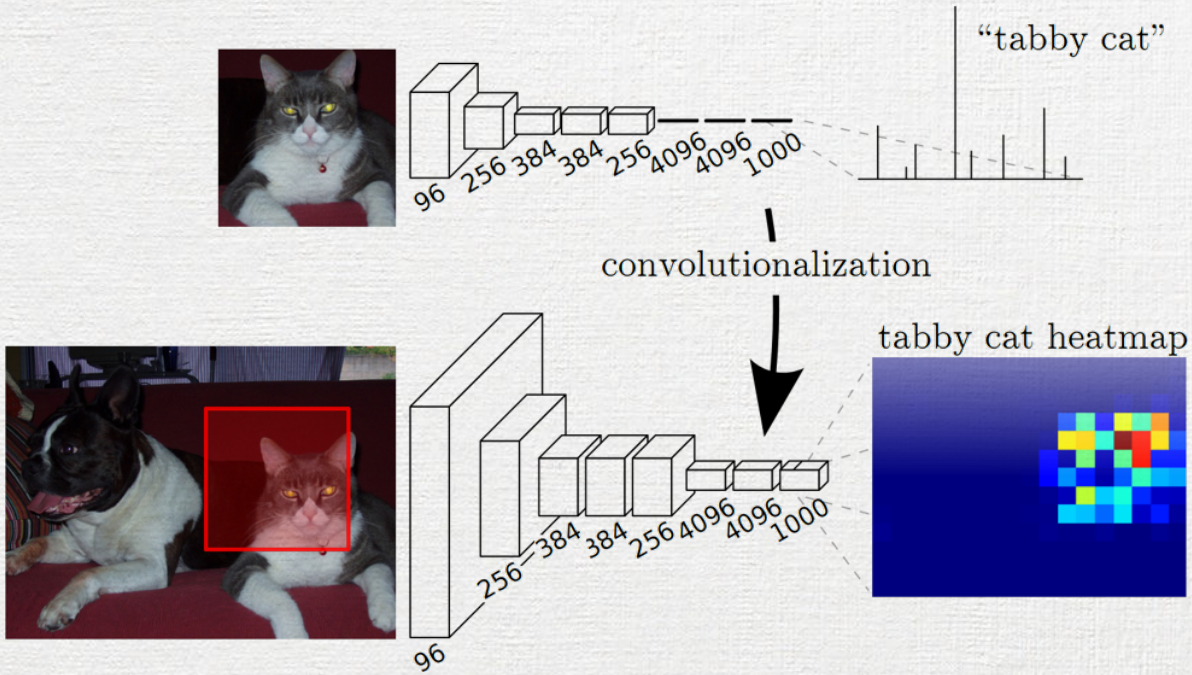
- Shape of cancer cell is very helpful in supporting doctors in cancer diagnosis

Object detection vs. Image segmentation



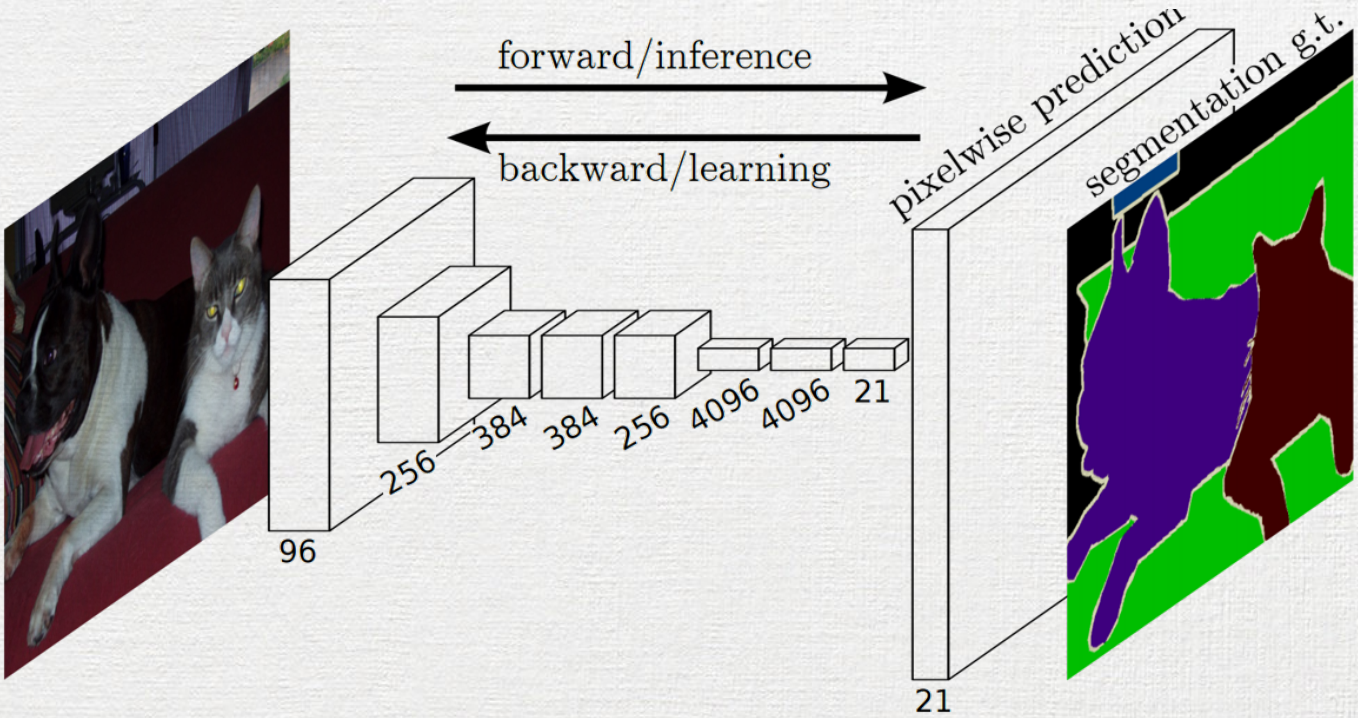
- Semantic segmentation: perform segmentation on each class
- Instance segmentation: perform segmentation on each object of the class
- Depend on the problem to apply semantic or instance segmentation

Convolutionize



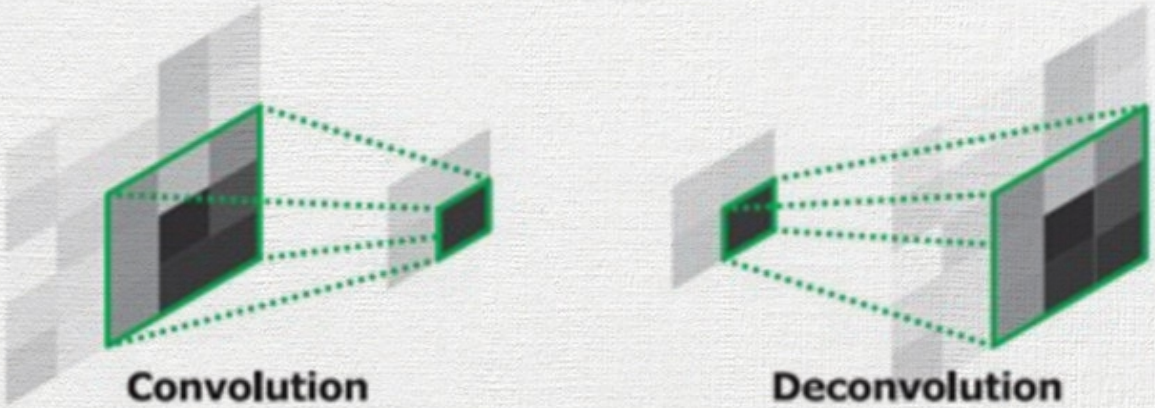
- Slide the network with an input of (224, 224) over a large image. Output of varying spatial size
- Convolutionize: change Dense (4096, 1000) to 1 x 1 convolution, with 4096 input and 1000 output channels
- Give a coarse segmentation (no extra supervision)

Fully Convolutional Network



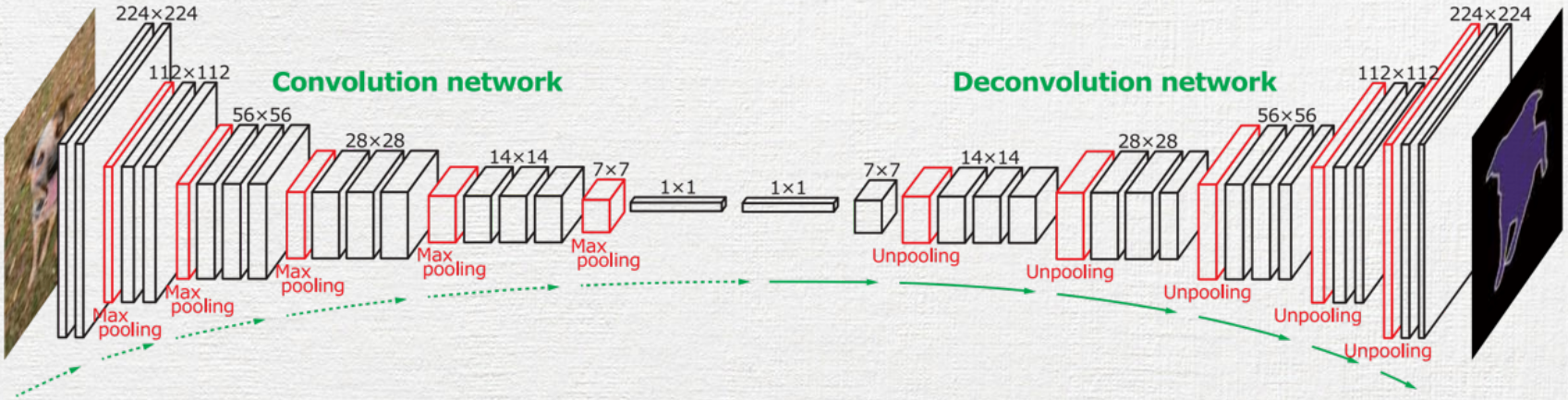
- Predict / backpropagate for every output pixel
- Aggregate maps from several convolutions at different scales for more robust results

Convolutional - Deconvolutional



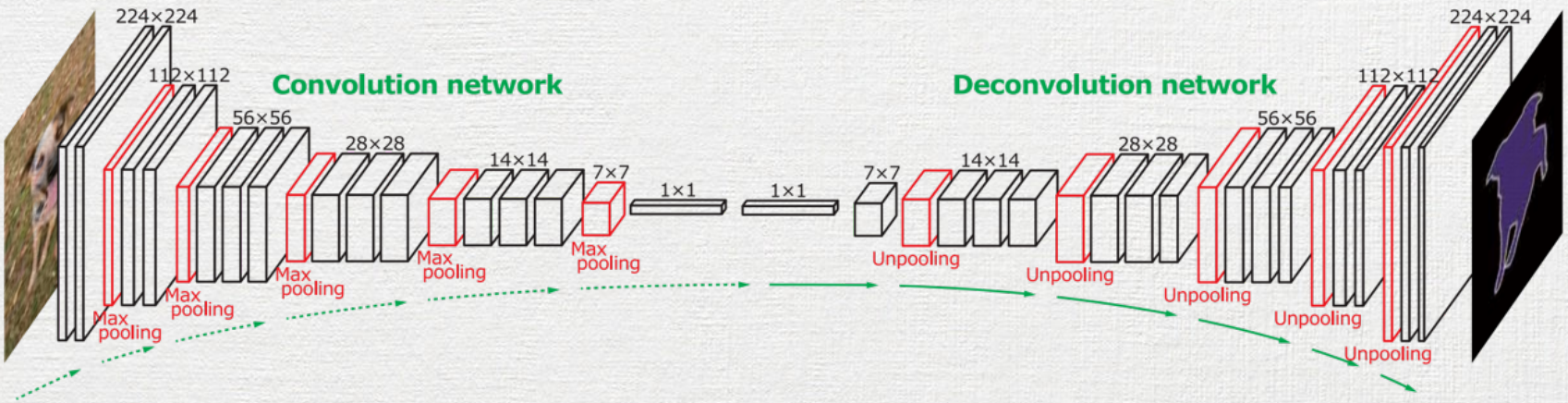
- Deconvolution: transposed convolutions

Convolutional - Deconvolutional

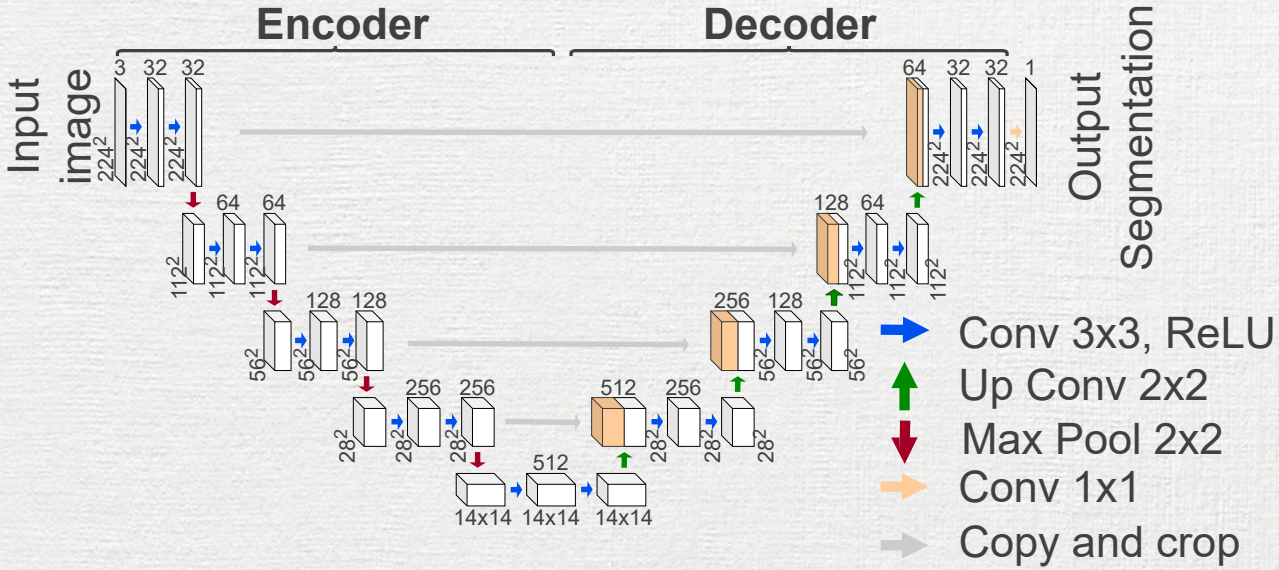


- Skip connections between corresponding convolution and deconvolution layers
- Sharper masks by using precise spatial information (early layers)
- Better object detection by using semantic information (late layers)

Convolutional - Deconvolutional

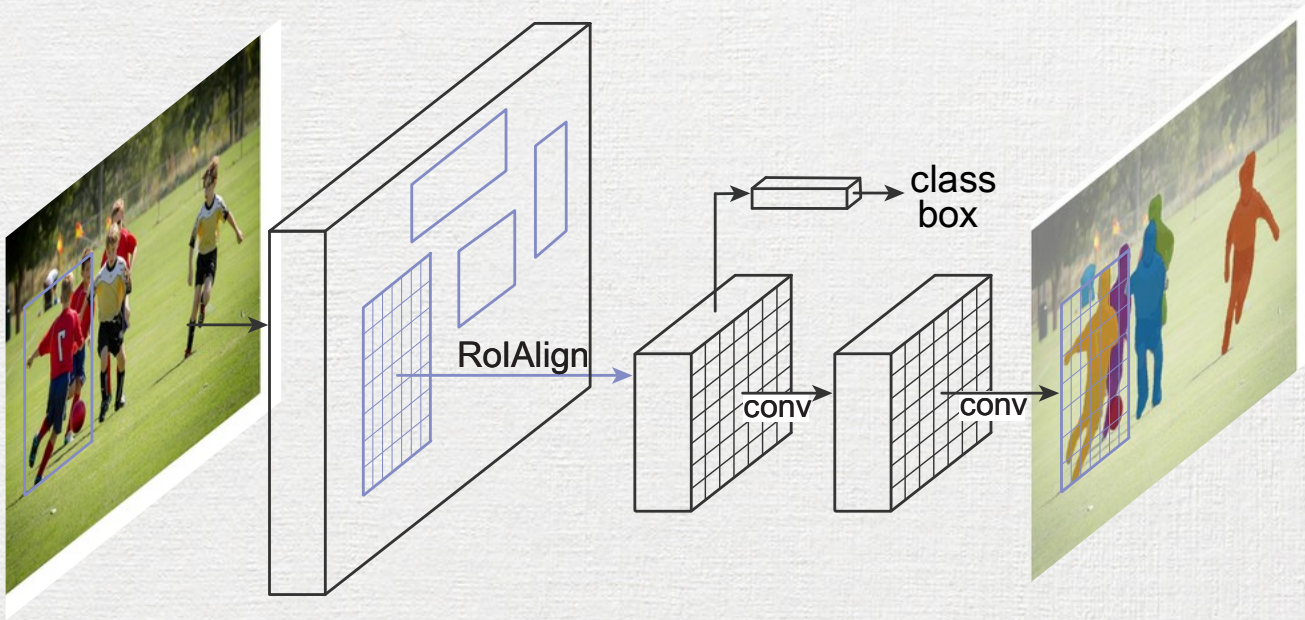


U-Net for semantic segmentation



- The left part of U-Net is called encoder part
- The right part of U-Net is called decoder part

Mask R-CNN for semantic segmentation



- Faster-RCNN architecture with a third, binary mask head

Practice!

Labwork 7: License Plate Detection and Digit Segmentation

- Implement a License Plate Detection and Digit Segmentation model
  - Input: image of cars/motorbike
  - Output: properly segmented digits in the license plate
  - Method: two stage (detection then segmentation)
- Train and test the implemented network on a dataset of your choice
- Note: **don't** load a pretrained model!

Labwork 7: Neural Network

- Write a report (in LaTeX):
  - Name it « Report.7.Detection.Segmentation.tex »
  - How you design and implement the network architecture
  - Evaluation of the network using detection segmentation metrics
  - Extra: comparison with ResNet19 if you are fast enough :)
- Push your code and report to your forked repository