Object Detection and Segmentation

What is Object Detection?

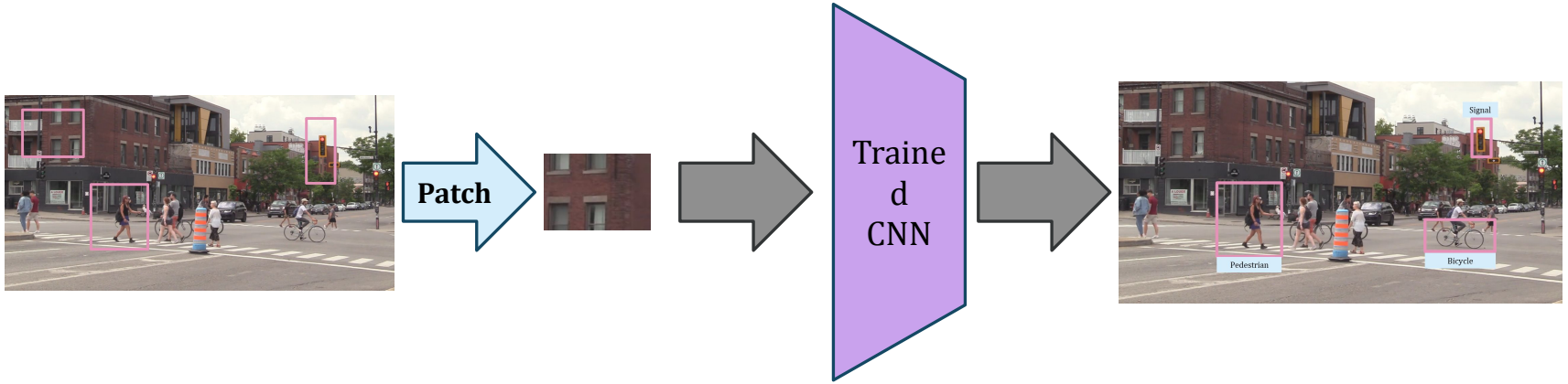
- Input: An Image

- Output: A set of detected objects, each with
  - A class label (from a set of pre-defined class labels)
  - A bounding box
    - May be of the form $[x_{left-top}, y_{left-top}, height, width]$

https://blog.jetbrains.com/kotlin/2022/01/object-detection-with-kotlindl-and-ktor/

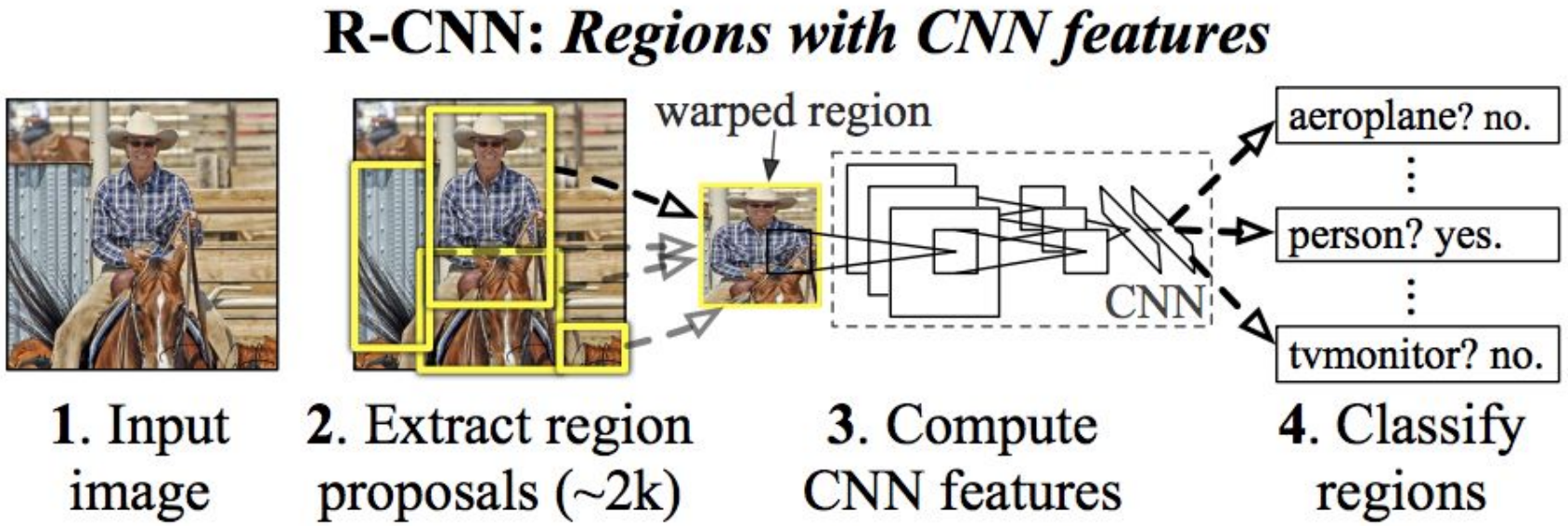
Object Detection using CNNs: A Simple Approach

- Pass the different sized patches through the trained CNN



https://www.flipkart.com/hero-sprint-knockout-26-t-road-cycle/p/itmfdd9f85e03106

R-CNN

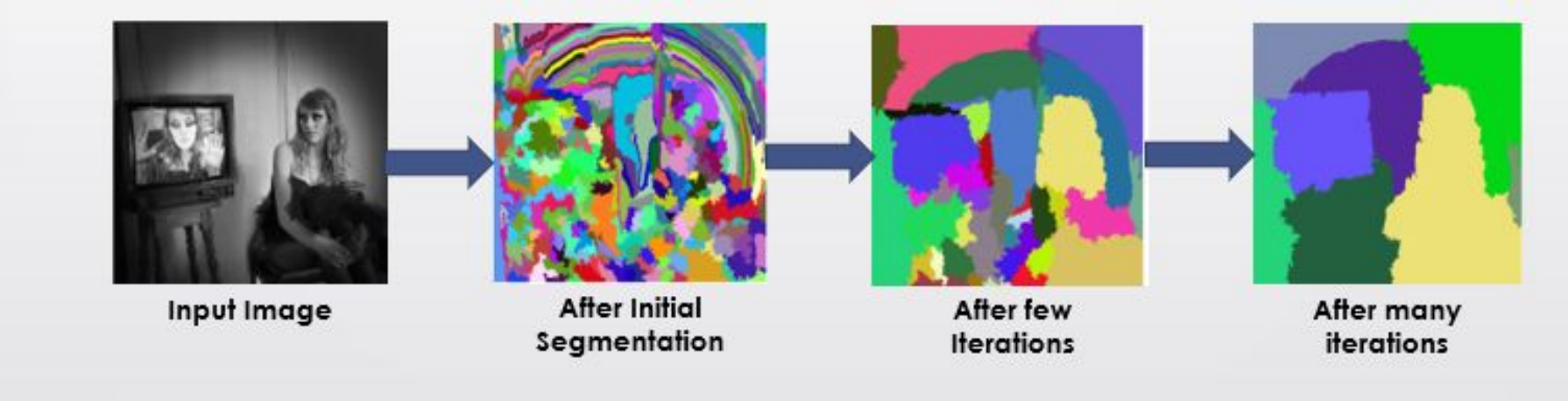

R-CNN: Regions with CNN features
warped region
CNN
aeroplane? no.
person? yes.
tvmonitor? no.
1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions

Identifying Potential Regions

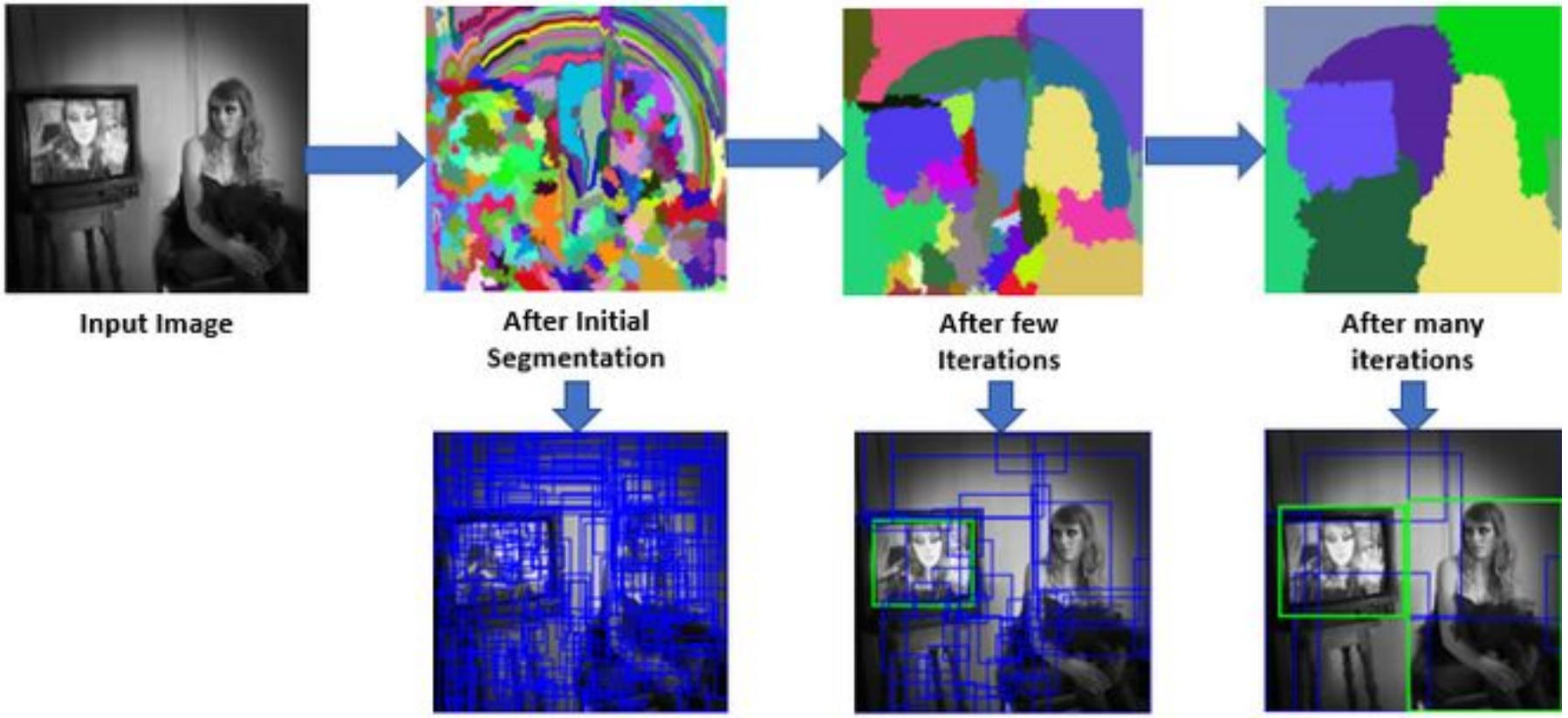
- R-CNN can use different methods to identify potential regions
  - Selective search
  - Objectness
  - Category-independent object proposals, etc.
- In the original paper, the authors use selective serach

Region Proposal - Selective Search

Region Proposal - Selective Search



Region Proposal - Selective Search



Region Proposal - Selective Search

- Color Similarity
- Texture Similarity
- Size Similarity
- Shape Compatibility

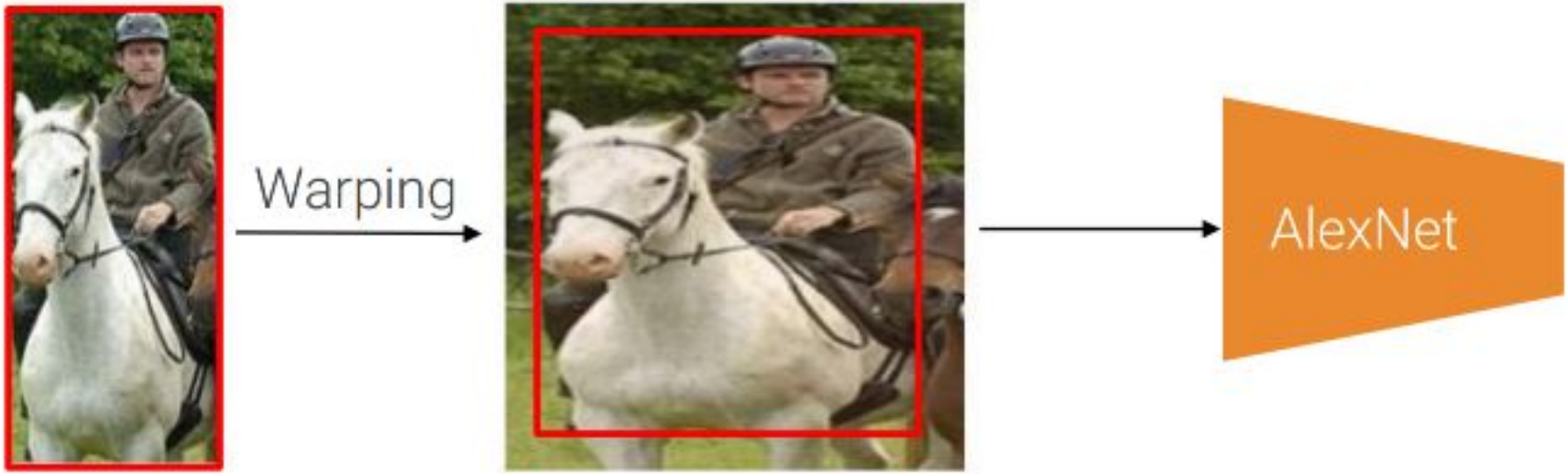
Selective Search

- Step 1: Use graph-based image segmentation to create initial regions
- Step 2: Extract features from each region that represents the characteristics of the region
- Step 3: Iteratively group the regions together based on the similarity of features
  - Step 3.1: Two most similar neighbour regions are grouped together
  - Step 3.2: The features of the grouped region is calculated
  - Step 3.3: The similarity of the grouped regions with its neighbours are calculated
  - Step 3.4: Go to step 3.1 if there is a pair of neighbouring regions having a feature similarity score more than a threshold. Stop, otherwise.

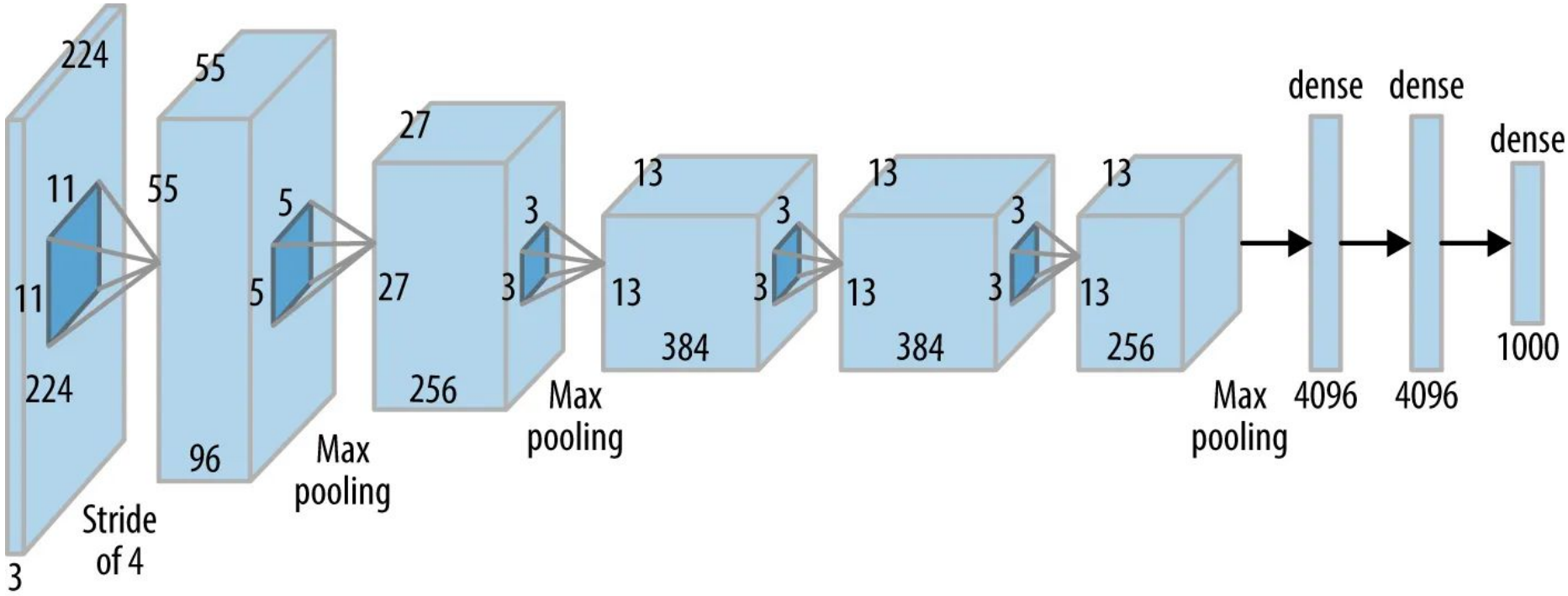
Feature of the regions could be colour distribution, texture, or other features, such as HOG

https://www.youtube.com/watch?v=VQXD6ZfYT6o&list=PLyqSpQzTE6M_PI-rIz4O1jEgffhJU9GgG&index=45

Feature Extraction



AlexNet

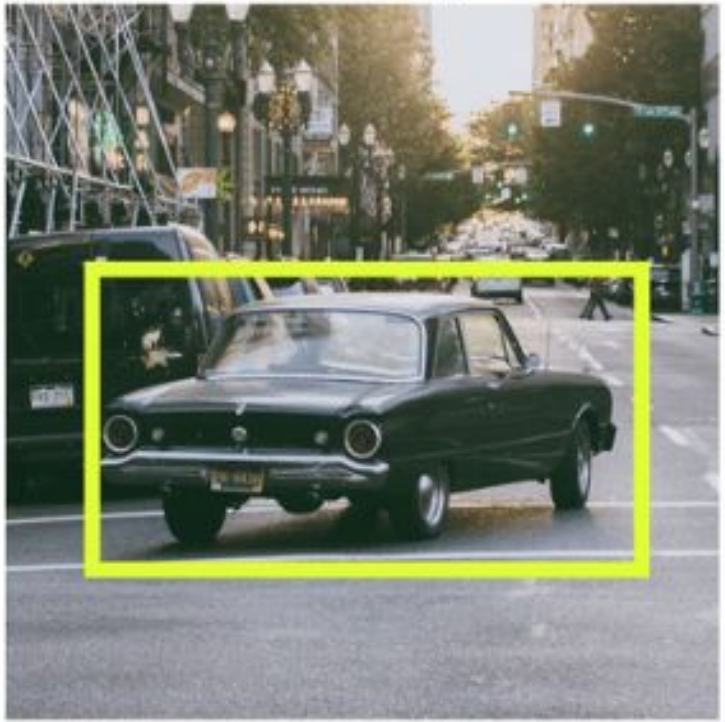

224
224
3
11
11
Stride
of 4
55
55
96
5
5
Max
pooling
27
27
256
3
3
Max
pooling
13
13
384
3
3
13
13
384
3
3
13
13
256
Max
pooling
dense
4096
dense
4096
dense
1000

Non-Maximal Suppression

Before non-max suppression

Non-Max Suppression

After non-max suppression



Improving the Performance: Non Maximal Suppression (NMS)

- Take the most confident bounding boxes among all the bounding boxes

Improving the Performance: Non Maximal Suppression (NMS)

- Take the most confident bounding boxes among all the bounding boxes

Improving the Performance: Non Maximal Suppression (NMS)

- Find out all the bounding boxes that overlap with the most confident box by a certain amount

Improving the Performance: Non Maximal Suppression (NMS)

- Find out all the bounding boxes that overlap with the most confident box by a certain amount
- Remove all such boxes

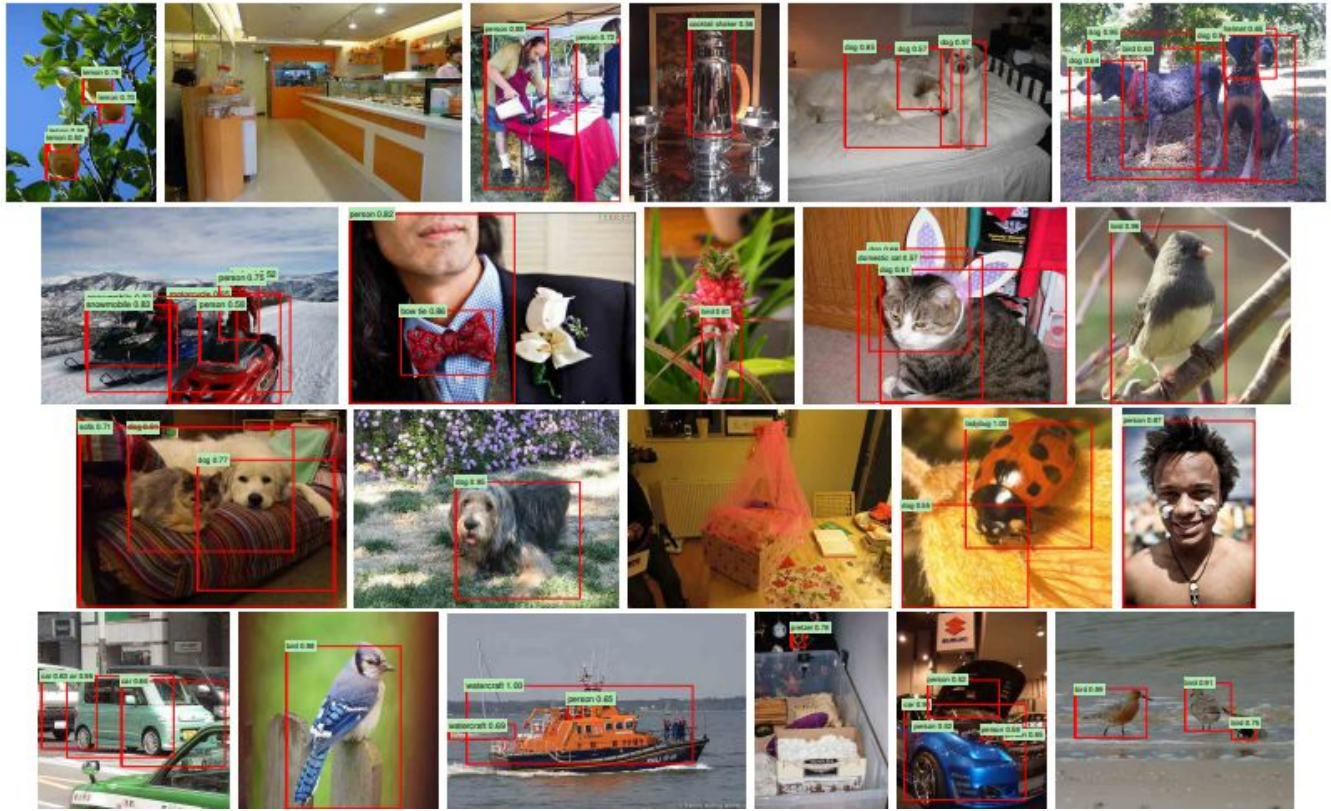
Improving the Performance: Non Maximal Suppression (NMS)

- Find the next most confident box

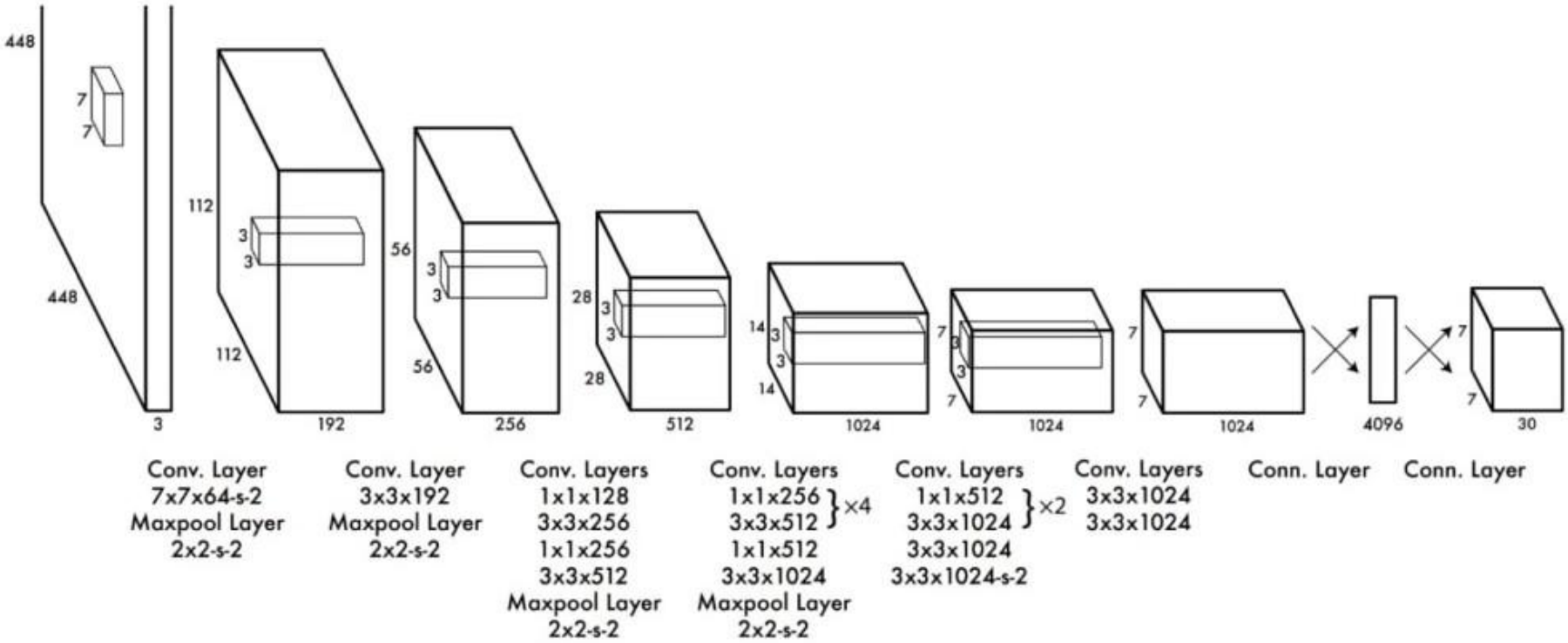
Improving the Performance: Non Maximal Suppression (NMS)

- Find the next most confident box
- Remove the over lapping boxes

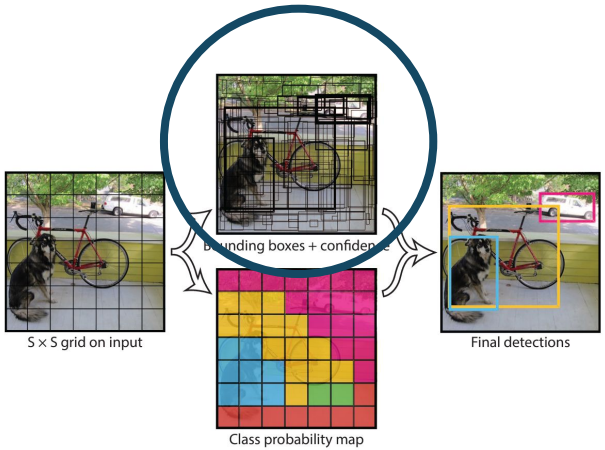
Examples



YOLO - You Only Look Once

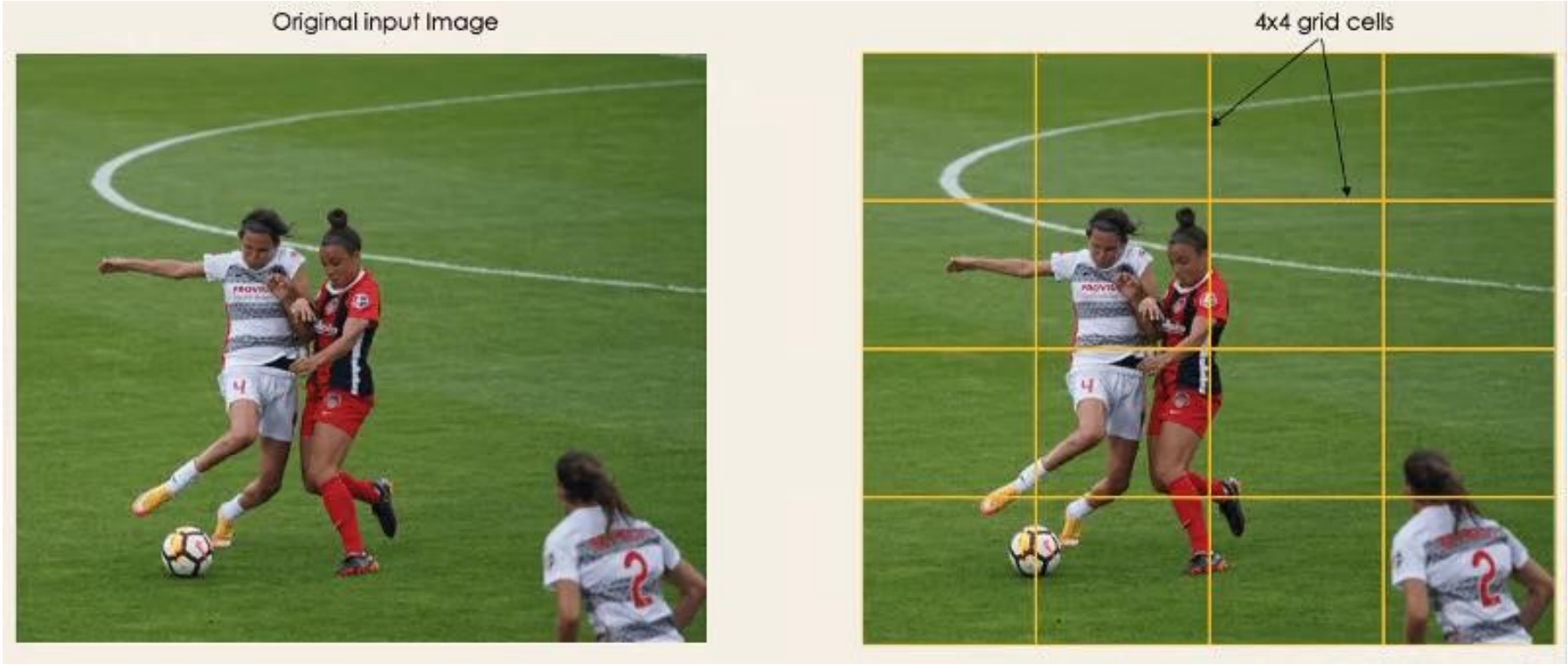


YOLO: You Only Look Once

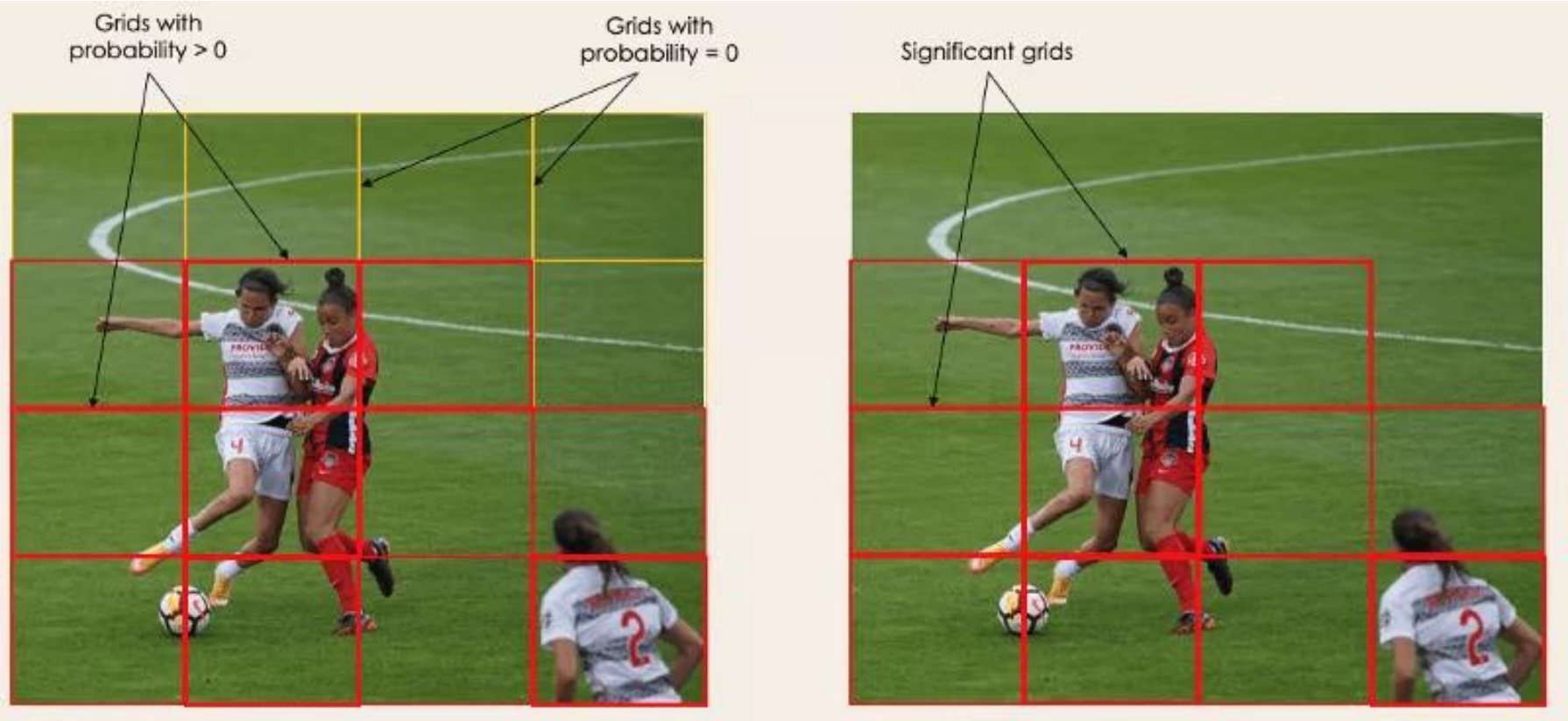


- Input image divided into $S \times S$ grid
- For each grid cell, the model predicts B bounding boxes
- Each bounding box has 5 predictions
  - x, y, w, h
    - x, y: represent the center of the box relative to the bounds of the grid cell
    - w, h: predicted relative to the whole image
  - Confidence score: reflect how confident the model is that the box contains an object and also how accurate it thinks the box is that it predicts
    - $P(object) \times IOU_{pred}^{gt}$
    - If no object exists, confidence score should be zero

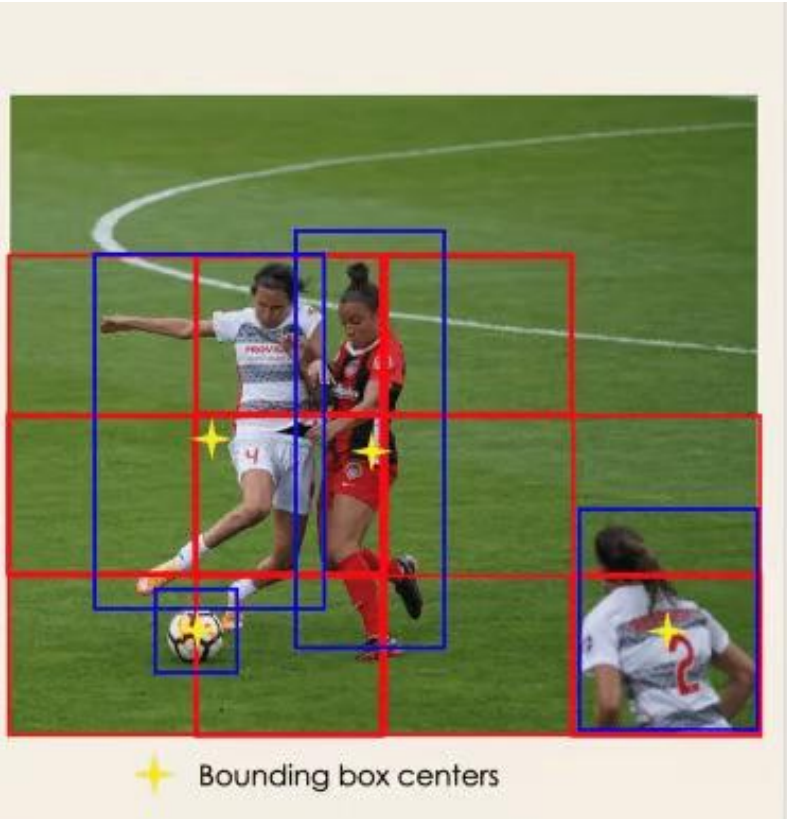
Residual Blocks

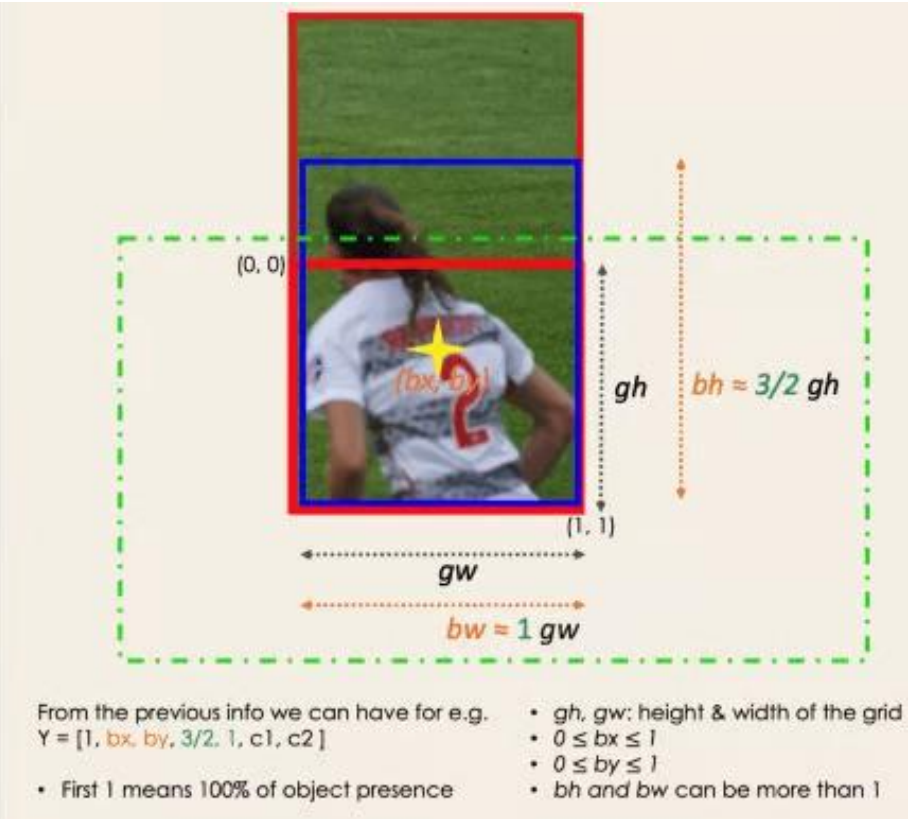


Bounding box regression

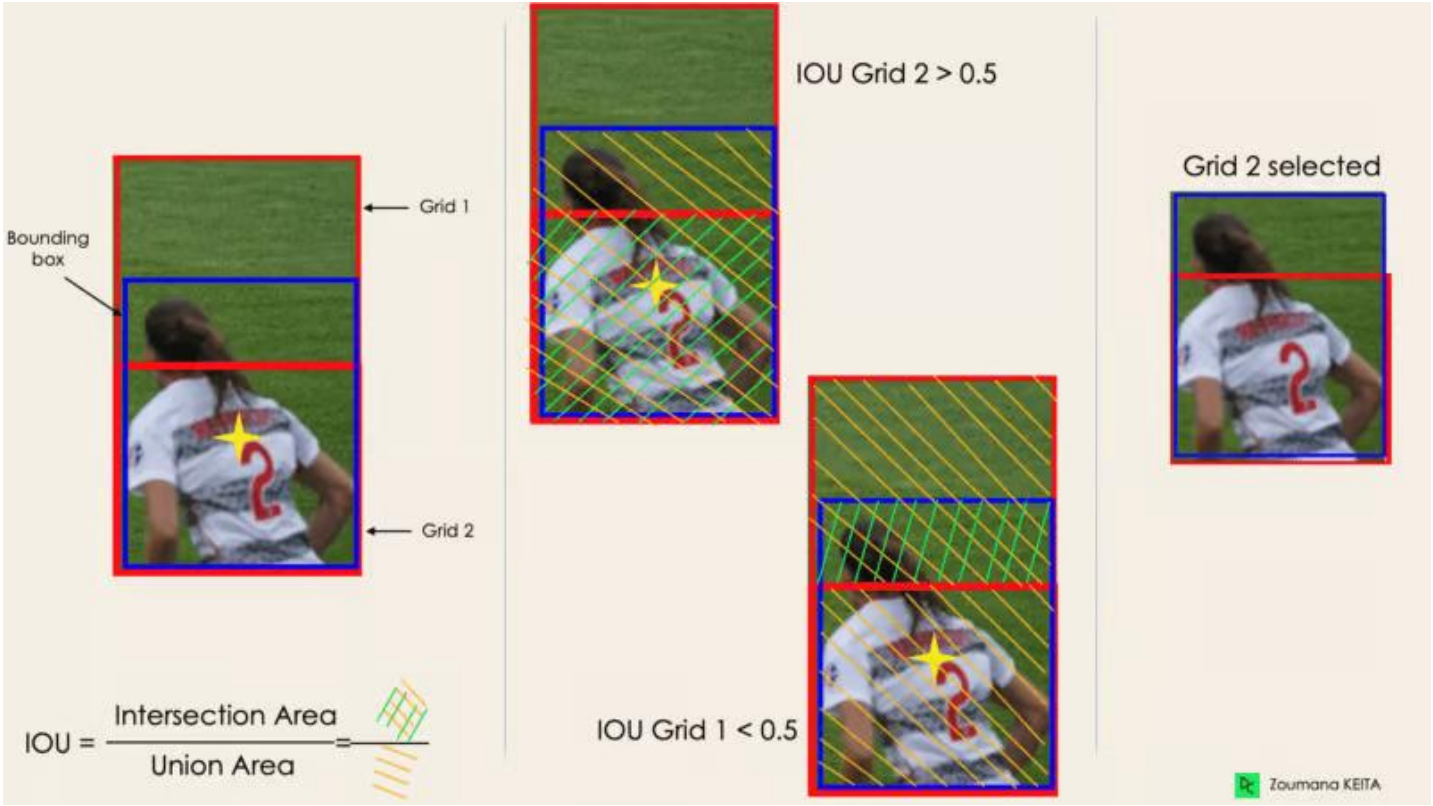


Bounding box regression

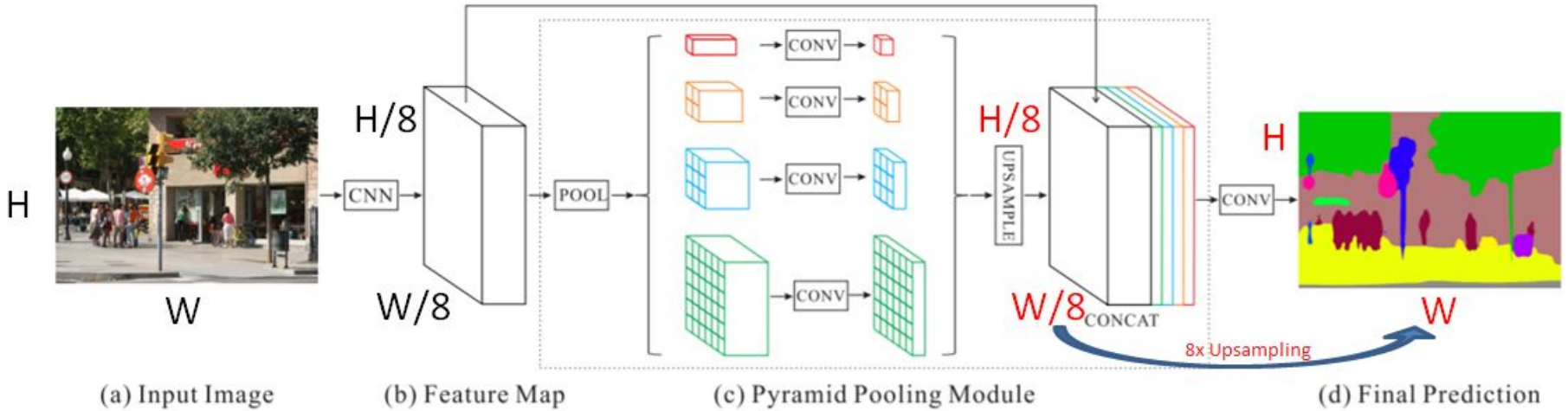




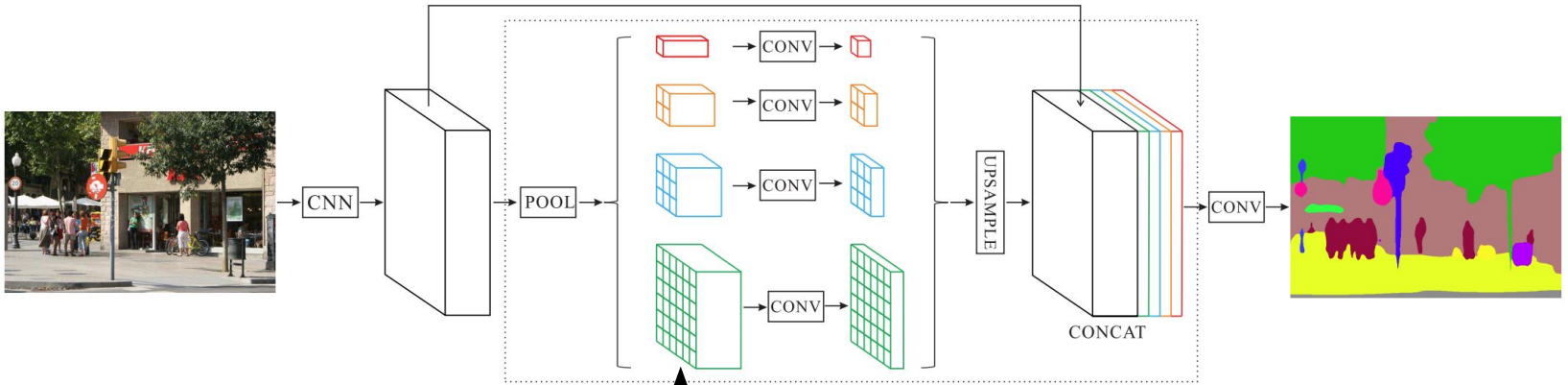
Intersection Over Unions or IOU



PSPNet


H
W
CNN
H/8
W/8
POOL
CONV
CONV
CONV
CONV
H/8
UPSAMPLE
W/8
CONCAT
CONV
H
W
8x Upsampling
(a) Input Image
(b) Feature Map
(c) Pyramid Pooling Module
(d) Final Prediction

PSP Net: Pyramid Scene Parsing Network



(a) Input Image (b) Feature Map (c) Pyramid Pooling Module (d) Final Prediction

***c* channels**

Pooling at different scales

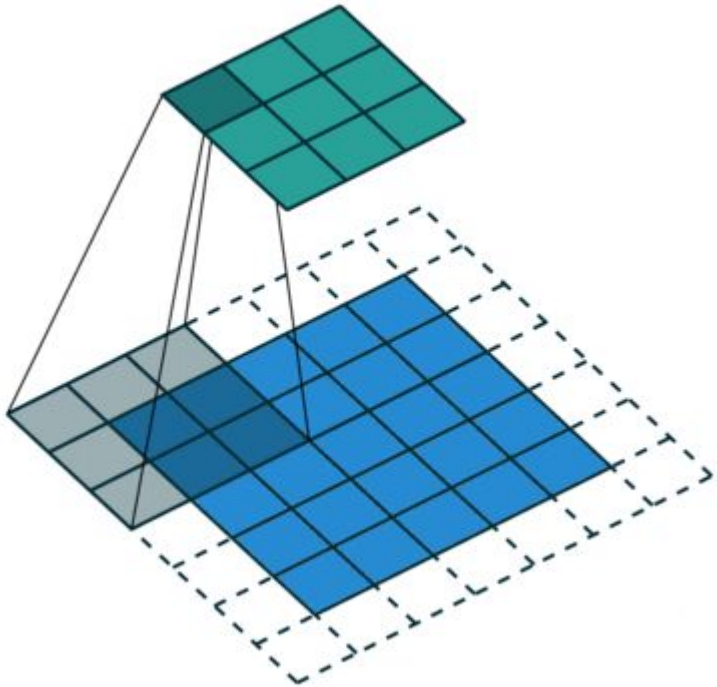
Global average pooling ($1 \times 1 \times c$ output maps)

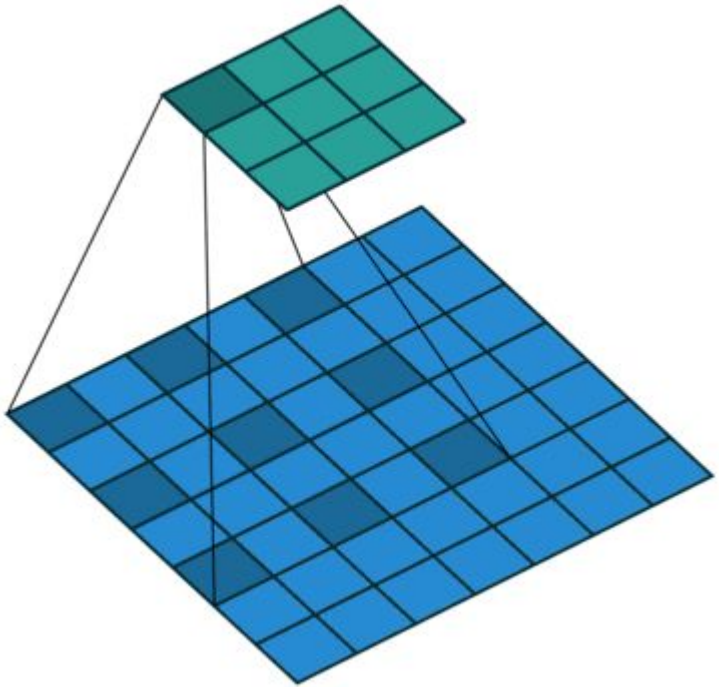
Pooling that results in $2 \times 2 \times c$ output maps

Pooling that results in $3 \times 3 \times c$ output maps

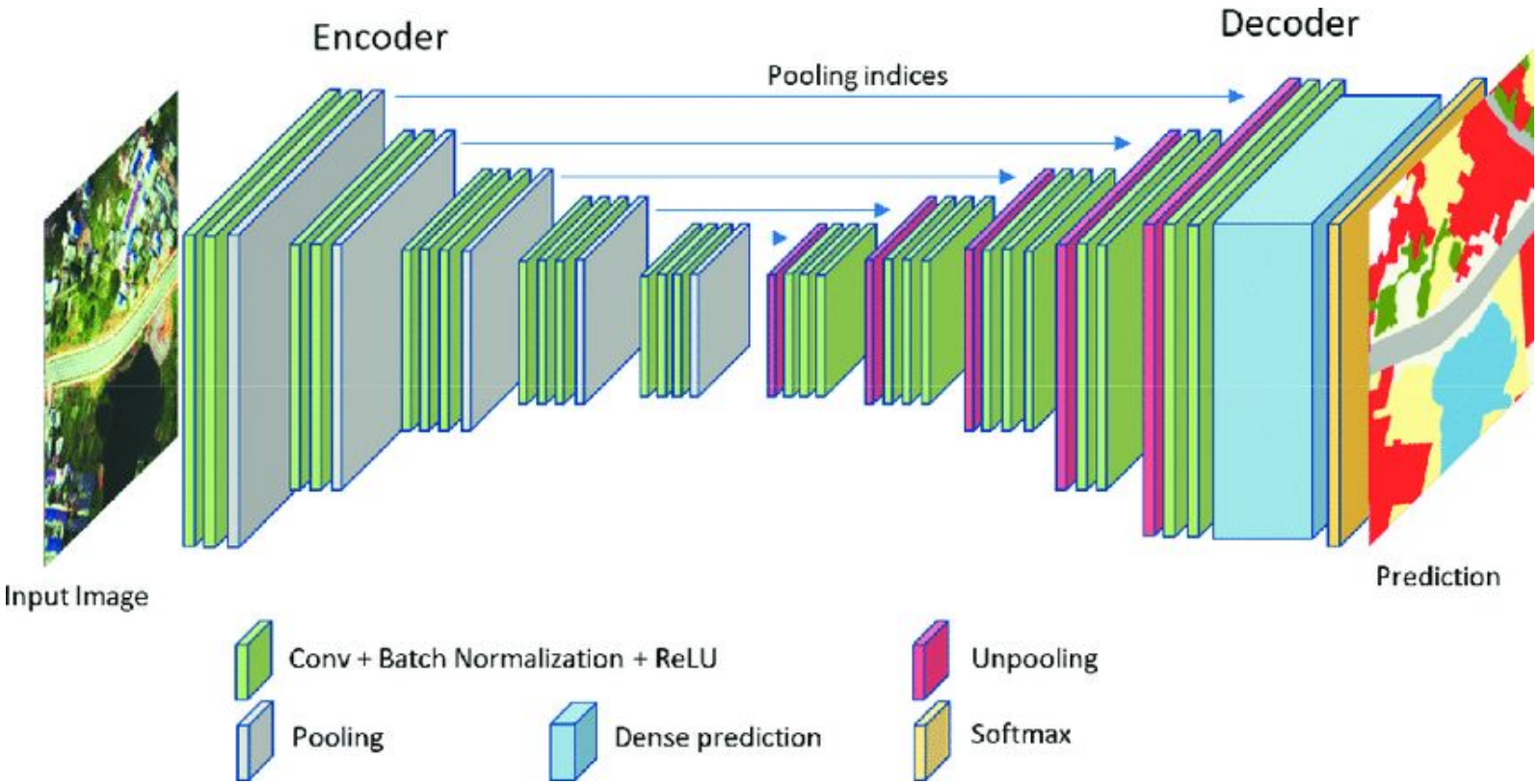
Pooling that results in $6 \times 6 \times c$ output maps

Dilated Convolutions





SegNet


Encoder
Decoder
Pooling indices
Input Image
Prediction
Conv + Batch Normalization + ReLU
Unpooling
Pooling
Dense prediction
Softmax