EE655: Computer Vision & Deep Learning

Lecture 12

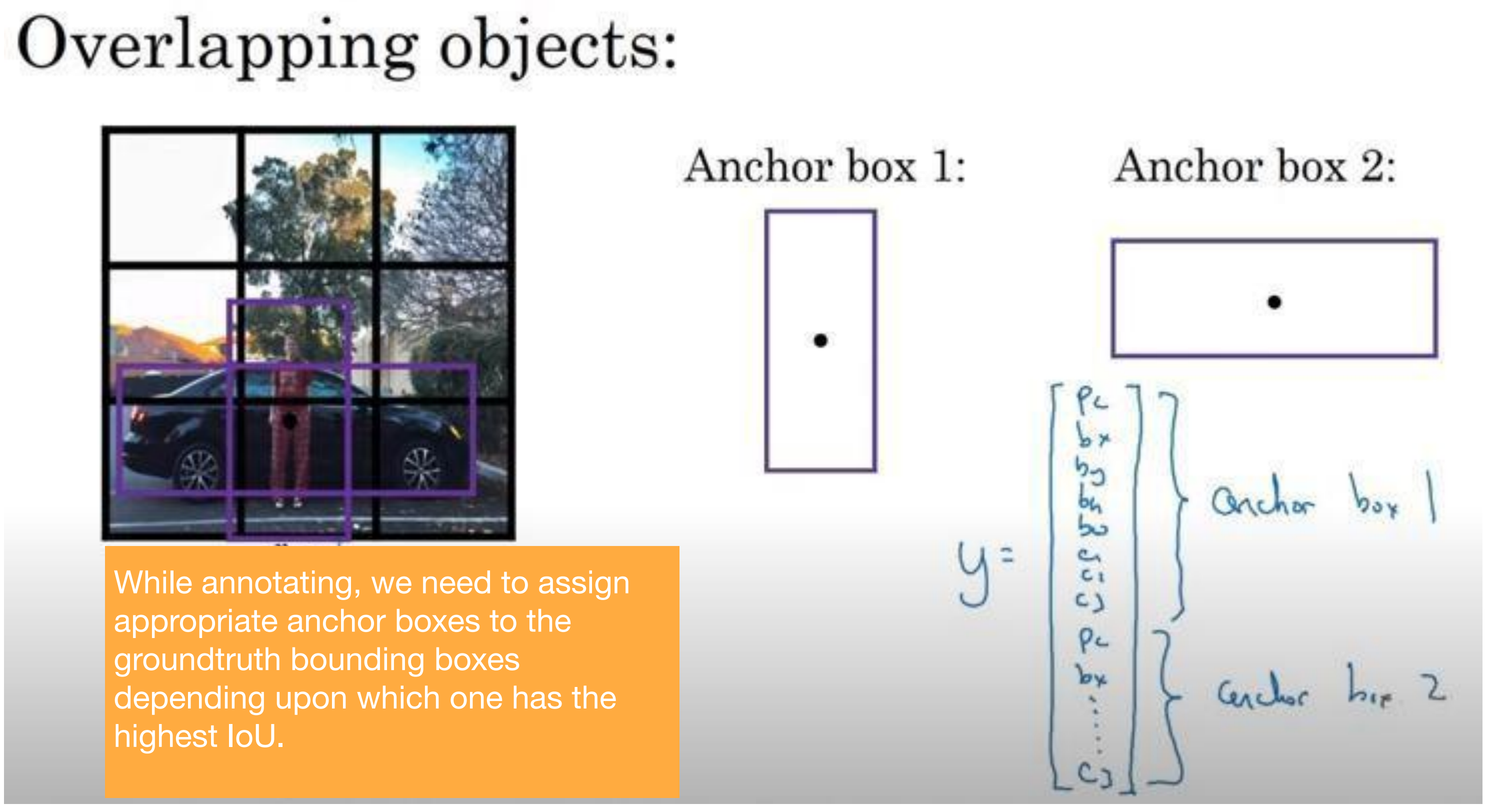
Koteswar Rao Jerripothula, PhD
Department of Electrical Engineering
IIT Kanpur

Outline

Anchor Boxes

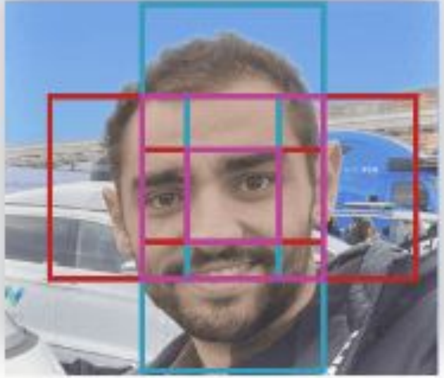
R-CNN

If we want to detect multiple objects per grid cell, we need to have multiple anchor boxes to accommodate prediction of multiple bounding boxes



While annotating, we need to assign appropriate anchor boxes to the groundtruth bounding boxes depending upon which one has the highest IoU.

ANCHOR BOXES



KEPT BOX

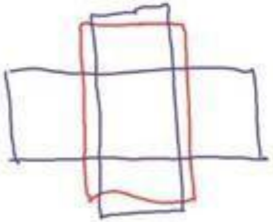
Anchor box algorithm

Previously:

Each object in training image is assigned to grid cell that contains that object's midpoint.

Output y:

$\underline{3 \times 3} \times \underline{8}$



With two anchor boxes:

Each object in training image is assigned to grid cell that contains object's midpoint and anchor box for the grid cell with highest IoU.

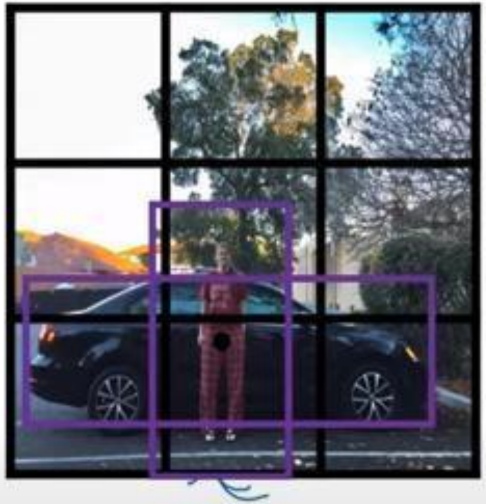
(grid cell, anchor box)

Output y:

$3 \times 3 \times \underline{16}$

$3 \times 3 \times 2 \times 8.$

Anchor box example



Ground truth

$$
y = \begin{array}{cccc}
 & \text{Both car and human} & \text{car only?} & \\
\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} &
\begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 1 \\ 0 \\ 0 \\ 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix} &
\begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix} &
\begin{array}{l} \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix}\right\} \text{anchor box 1} \\ \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix}\right\} \text{anchor box 2} \end{array}
\end{array}
$$

Anchor box 1: Anchor box 2:

Each grid cell produced 2 bounding boxes, because there were two anchor boxes in any grid cell

After pc<0.6 are discarded

After full NMS

Summary of ideas handling YOLO outputs

- NMS idea
- Purpose is to assign a single bounding box to any object after observing all the activations of the network
- Required at the time of post-processing

- Anchor box idea
- Purpose is to accommodate multiple bounding boxes in a grid cell
- Required for appropriate annotations

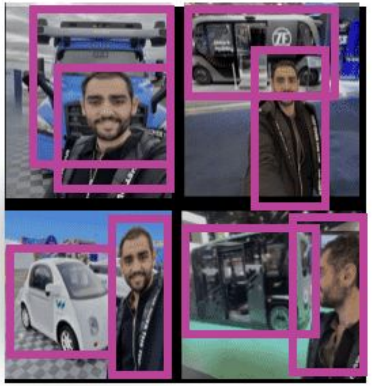
We can either increase the number of grid cells or increase the number of anchor boxes to get refined results. Both can be expensive.

Limitations

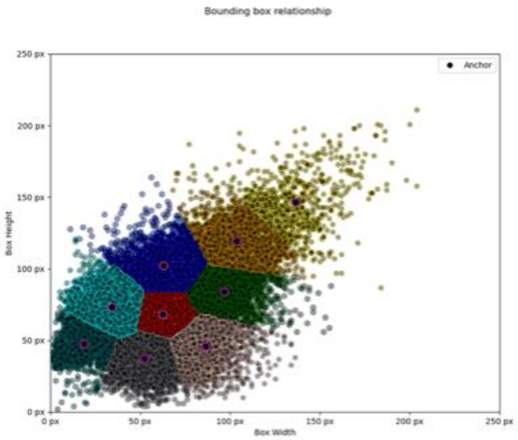
- More objects in a grid cell than number of anchor boxes chosen
- Two objects having same type of bounding boxes. Which bounding box will get the which anchor box?

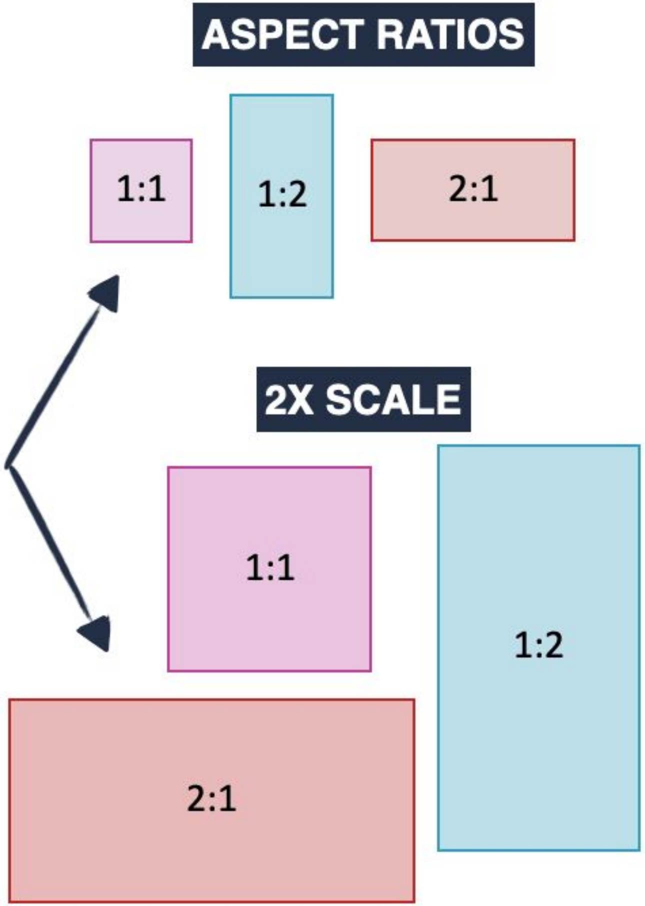
How to decide which anchor boxes to use?

DATASET



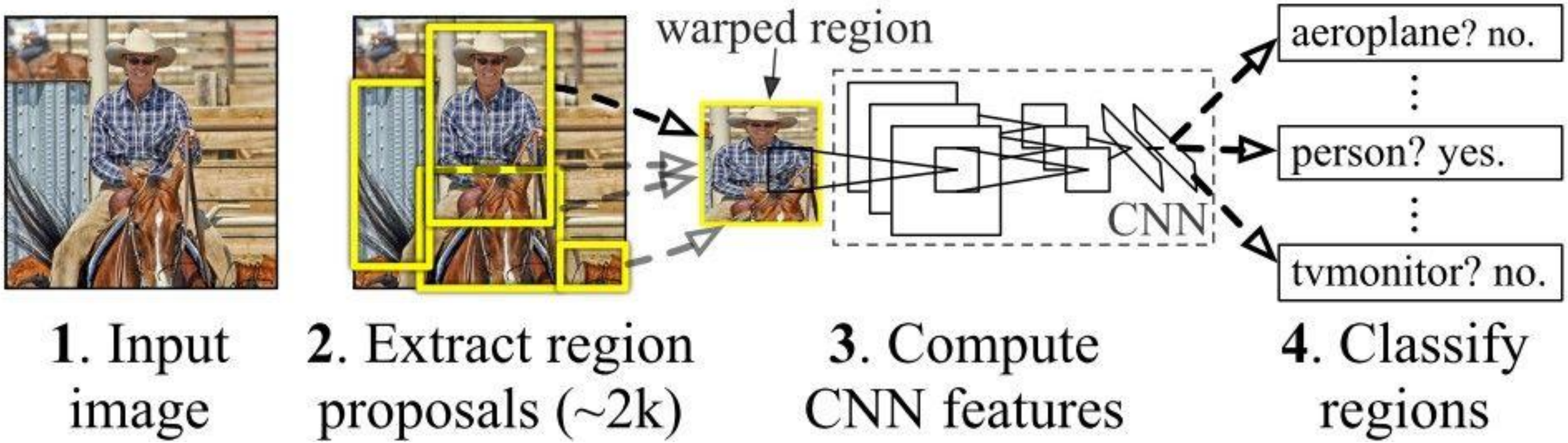
CLUSTERING



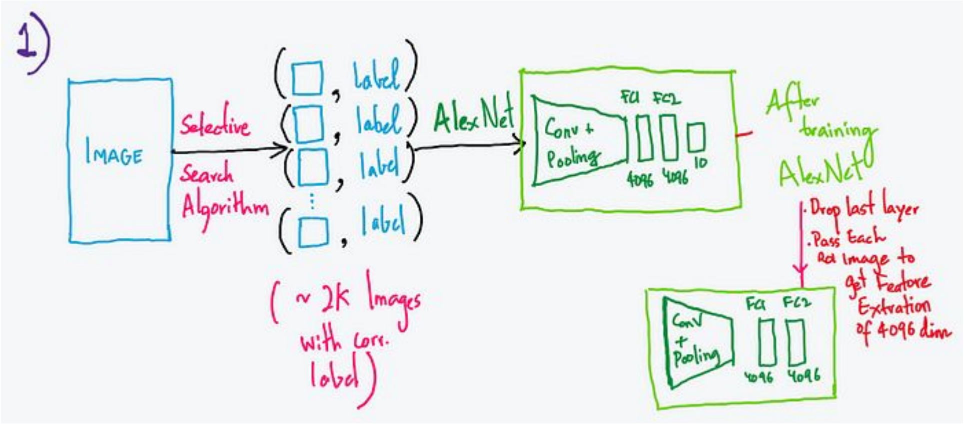


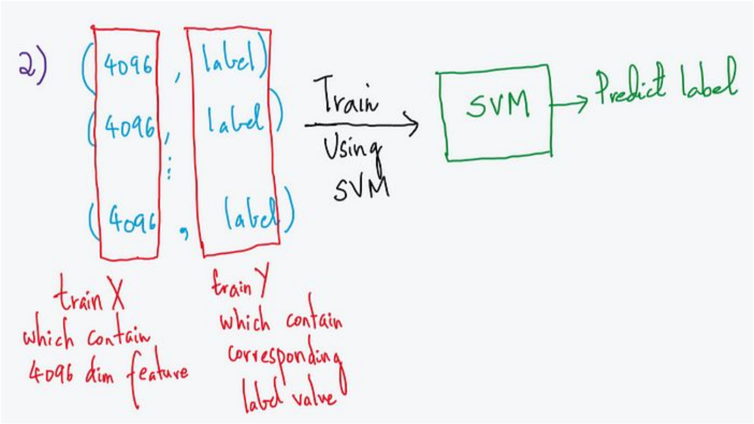
Once you have these, you'll generate multiple anchor boxes of a certain size (16,32), (32,32), (32, 16), and then have variations of these both in aspect ratio and scale (2x, 3x, 4x, ...).

R-CNN

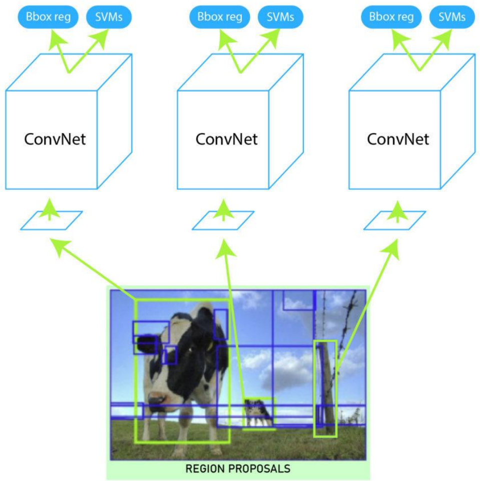


R-CNN Paper:
https://arxiv.org/pdf/1311.2524v5





3) Bounding-box Regression



$$t_x = (G_x - P_x)/P_w$$
$$t_y = (G_y - P_y)/P_h$$
$$t_w = \log(G_w/P_w)$$
$$t_h = \log(G_h/P_h).$$

We try to predict these transformation parameters via ridge regression using CNN features

Selective Search

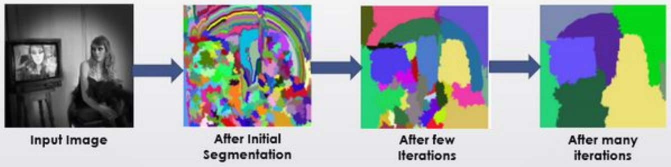
Algorithm Of Selective Search :

1. Generate initial sub-segmentation of input image using the method describe by *Felzenszwalb et al* in his paper "Efficient Graph-Based Image Segmentation ".

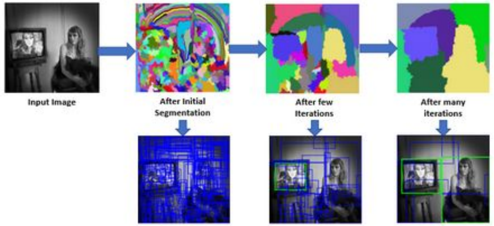
2. Recursively combine the smaller similar regions into larger ones. We use Greedy algorithm to combine similar regions to make larger regions. The algorithm is written below.

```
Greedy Algorithm :

1. From set of regions, choose two that are most similar.
2. Combine them into a single, larger region.
3. Repeat the above steps for multiple iterations.
```



3. Use the segmented region proposals to generate candidate object locations.



Similarity in Segmentation:

The selective search paper considers four types of similarity when combining the initial small segmentation into larger ones. These similarities are:

- **Color Similarity :** Specifically for each region we generate the histogram of each channels of colors present in image .In this paper 25 bins are taken in histogram of each color channel. This gives us 75 bins (25 for each R, G and B) and all channels are combined into a vector (n = 75) for each region. Then we find similarity using equation below:

$$S_{color}(r_i, r_j) = \sum_{k=1}^{n} \min(c_i^k, c_j^k)$$
$$C_i^k, c_j^k = k^{th}\ value\ of\ histogram\ bin\ of\ region\ r_i\ and\ r_j\ respectively$$

@ 8 different orientations

- **Texture Similarity :** Texture similarity are calculated using generated 8 Gaussian derivatives of image and extracts histogram with 10 bins for each color channels. This gives us 10 x 8 x 3 = 240 dimensional vector for each region. We derive similarity using this equation.

$$S_{texture}(r_i, r_j) = \sum_{k=1}^{n} \min(t_i^k, t_j^k)$$
$$t_i^k, t_j^k = k^{th}\ value\ of\ texture\ histogram\ bin\ of\ region\ r_i\ and\ r_j\ respectively$$

- **Size Similarity :** The basic idea of size similarity is to make smaller region merge easily. If this similarity is not taken into consideration then larger region keep merging with larger region and region proposals at multiple scales will be generated at this location only.

$$S_{size}(r_i, r_j) = 1 - (size\,(r_i) + size\,(r_j)) \div size\,(img)$$
$$where\ size\,(r_i)\ ,\ size\,(r_j)\ and\ size\,(img)\ are\ the\ sizes\ of\ regions\ r_i\ ,\ r_j\ and\ image\ respectively\ in\ pixels$$

- **Fill Similarity :** Fill Similarity measures how well two regions fit with each other. If two region fit well into one another (For Example one region is present in another) then they should be merged, if two region does not even touch each other then they should not be merged.

$$S_{fill}(r_i, r_j) = 1 - (size\,(BB_{ij}) - size\,(r_i) - size\,(r_j)) \div size\,(img)$$
$$size\,(BB_{ij})\ is\ the\ size\ of\ bounding\ box\ around\ i\ and\ j$$

Now, Above four similarities combined to form a final similarity.

$$S_{(r_i, r_j)} = a_1 * S_{color}(r_i, r_j) + a_2 * S_{texture}(r_i, r_j) + a_3 * S_{size}(r_i, r_j) + a_4 * S_{fill}(r_i, r_j)$$
$$where\ a_i\ is\ either\ 0\ or\ 1\ depending\ upon\ we\ consider\ this\ similarity\ or\ not\ .$$

Reference:
https://www.geeksforgeeks.org/selective-search-for-object-detection-r-cnn/