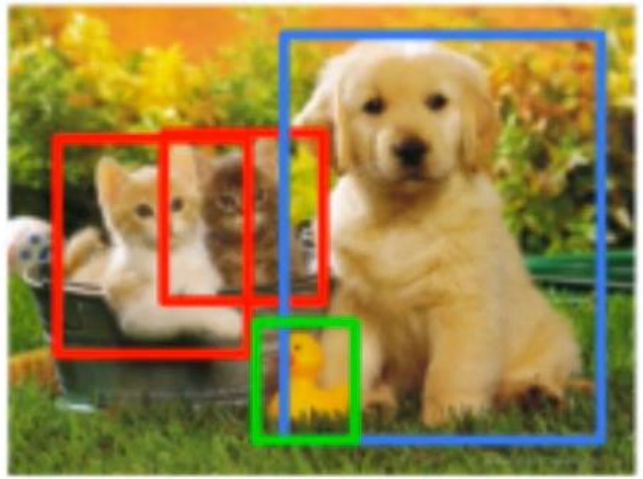
AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

Introduction to Deep Learning

ResNet (Residual Network)– How does it perform across tasks

CAT

CAT



CAT, DOG, DUCK

CAT, DOG, DUCK

Winner on the 5 main tasks:

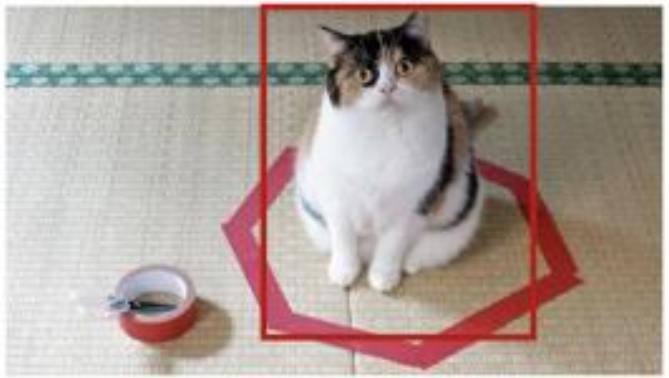
- ImageNet Classification
- ImageNet Localization*
- ImageNet Detection*
- Coco Detection*
- Coco Segmentation*

1

Is this image of Cat or not?

Image classification problem

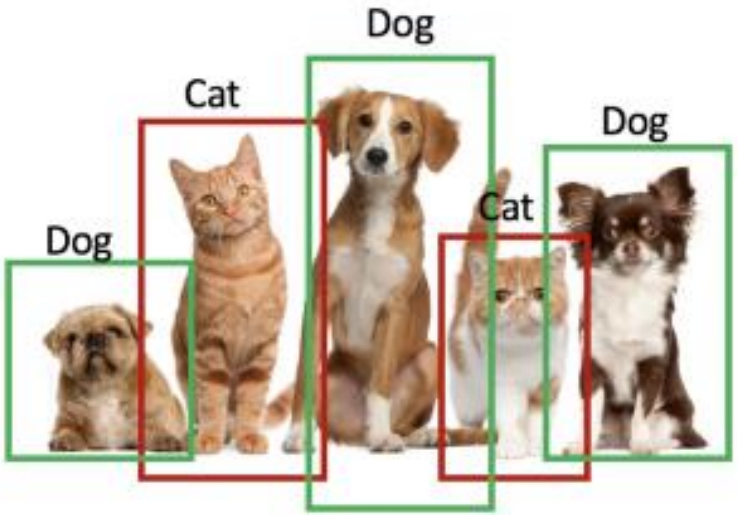
2



Where is Cat?

Classification with localization problem

3



Which animals are there in image and where?

Object detection problem

Image Classification

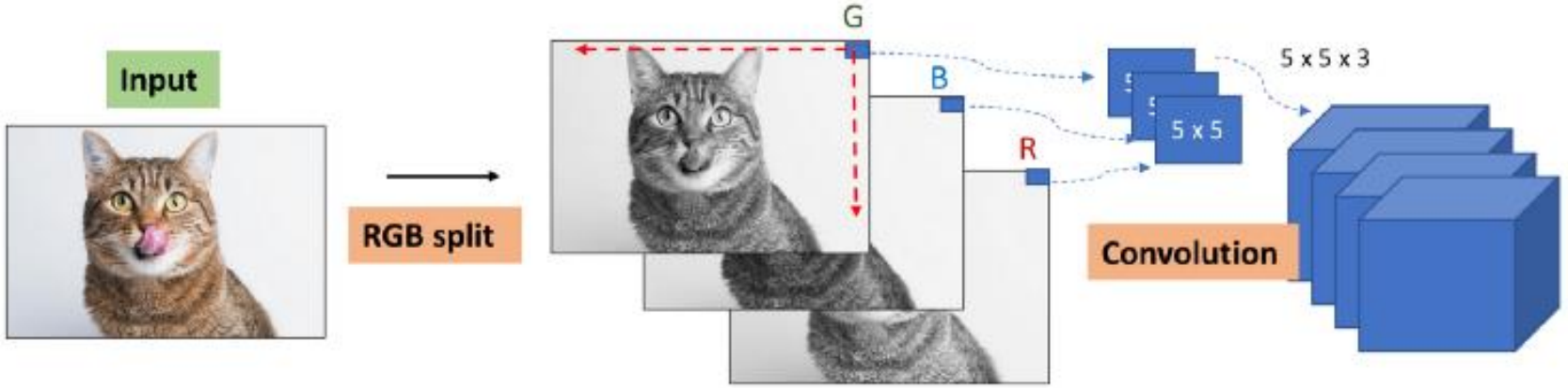
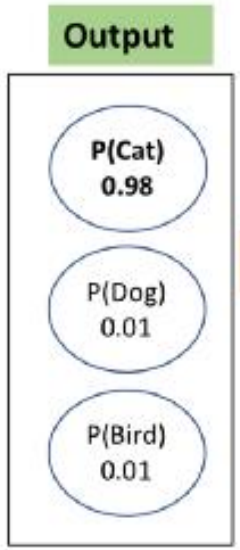


Image dimension: 600 x 400

Image split into 3 channels
Dimension: 600 x 400 x 3

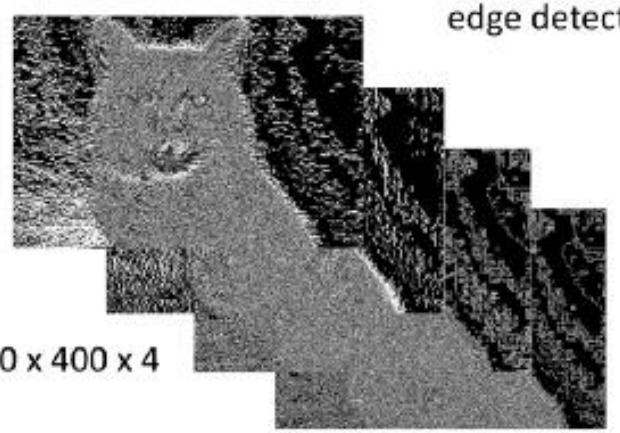
4 Convolution filters.
Each of dimension: 5 x 5 x 3

Convolution output
(if 4 filters are 4 different
edge detectors)



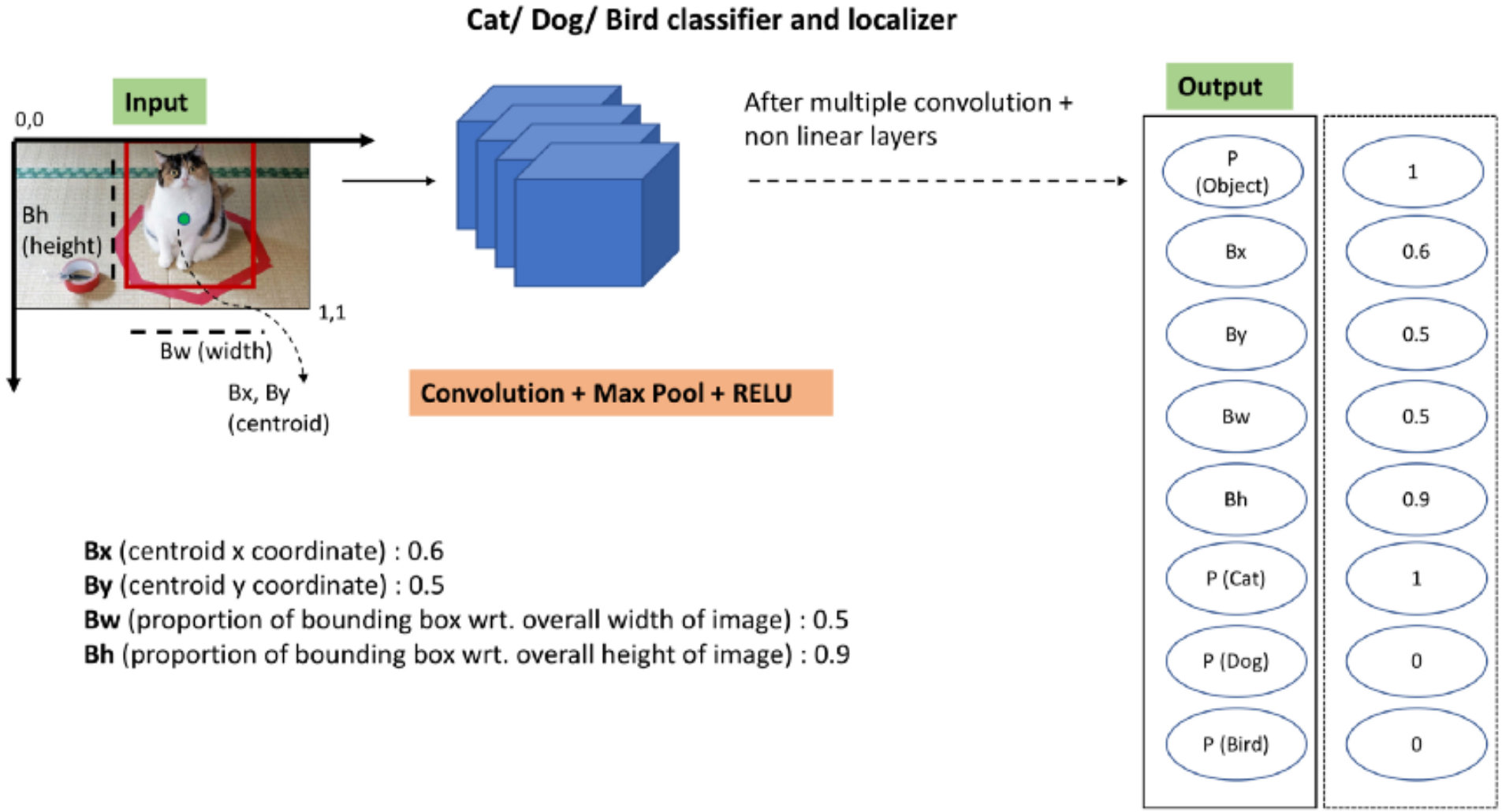
Softmax

After multiple convolution +
non linear layers



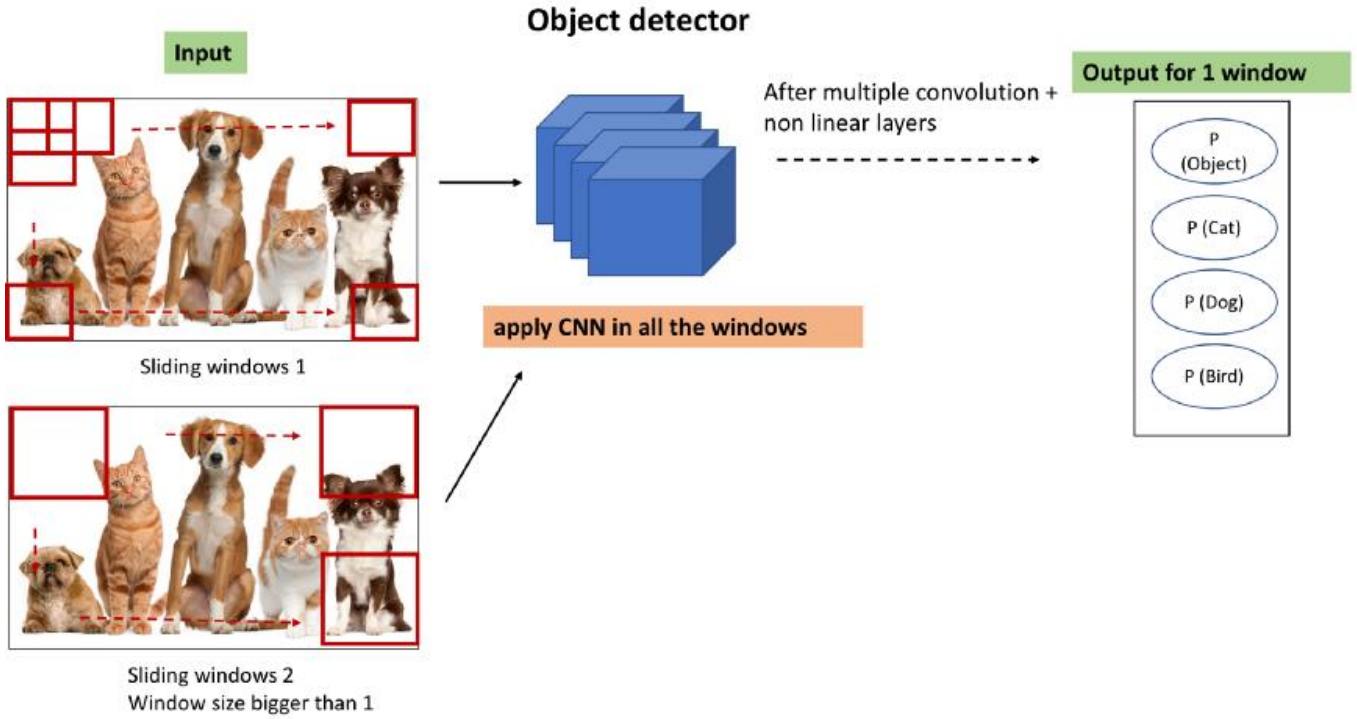
Output dimension : 600 x 400 x 4

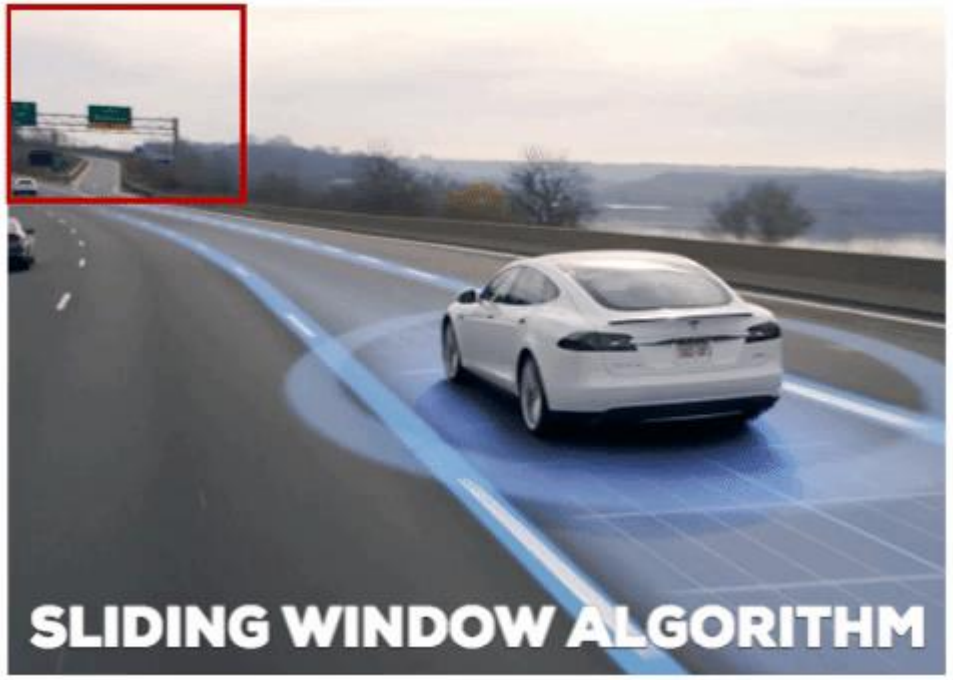
Image Classification with localization



Bx (centroid x coordinate) : 0.6
By (centroid y coordinate) : 0.5
Bw (proportion of bounding box wrt. overall width of image) : 0.5
Bh (proportion of bounding box wrt. overall height of image) : 0.9

Multiple Object detection





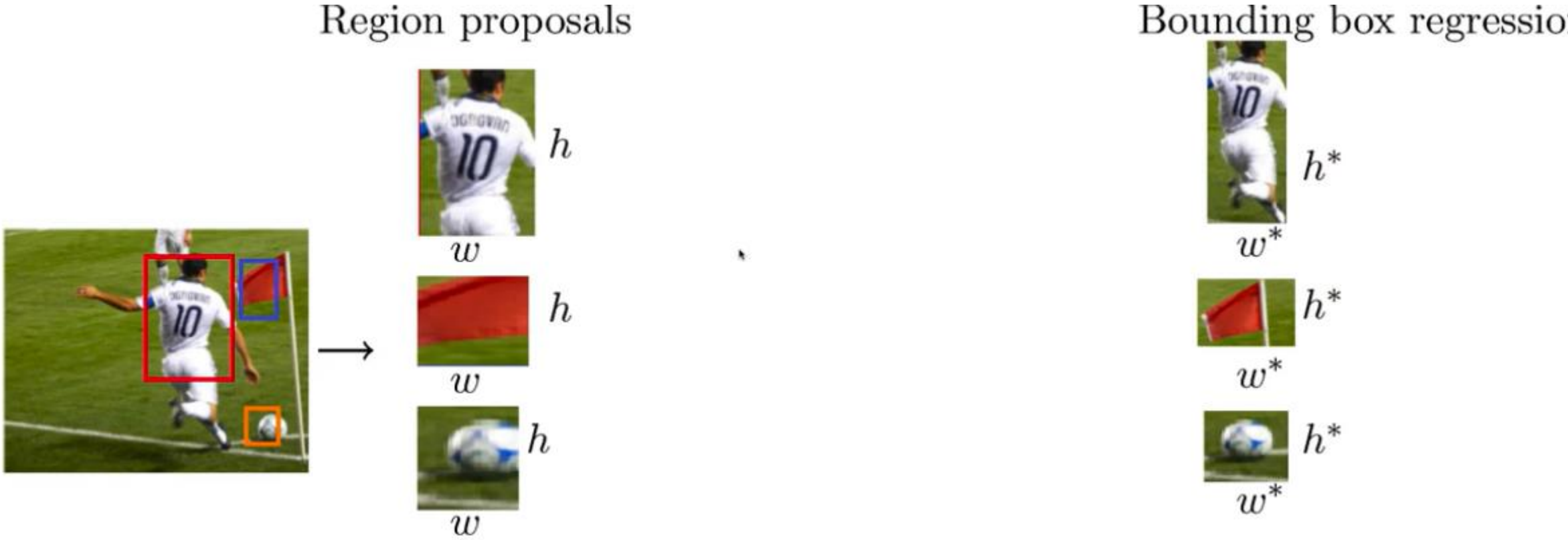
Now we have better solutions- pretrained models for object detection- YOLO, R-CNN Faster RCNN, MaskRCNN. We will see those later

Object Detection

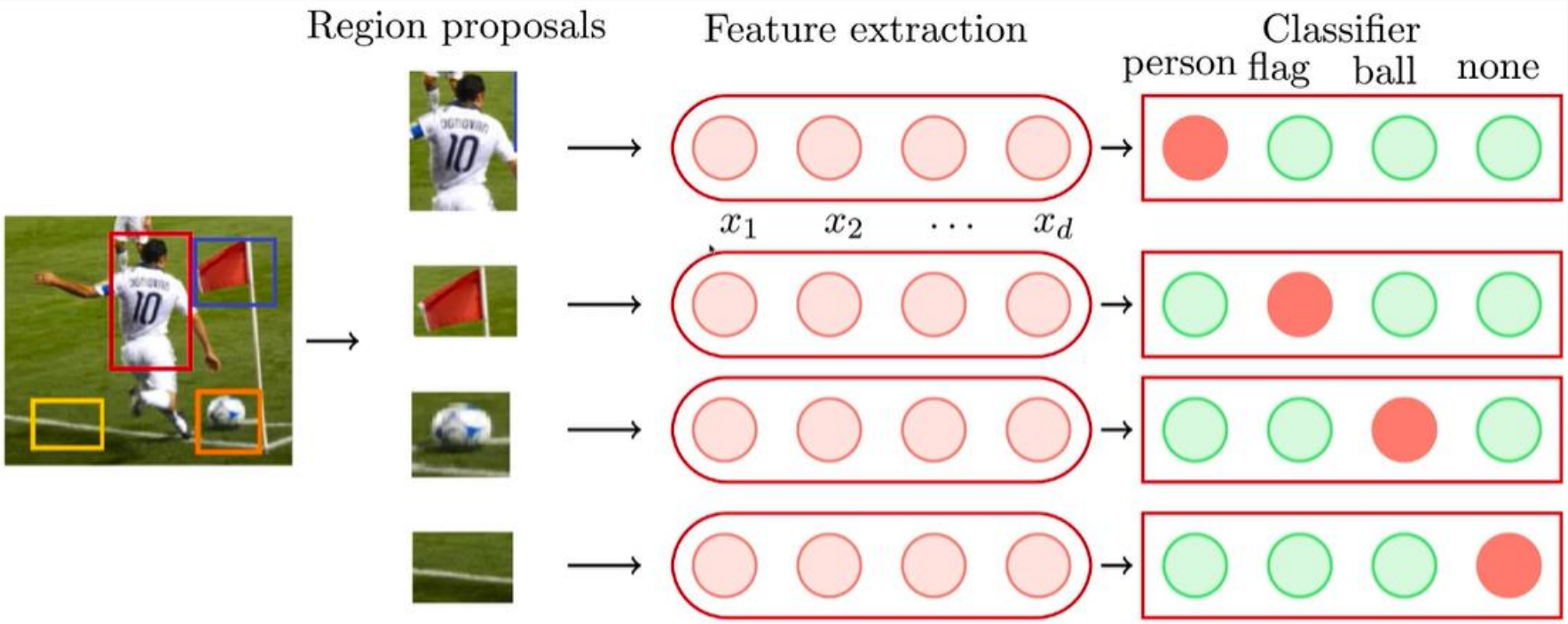
- RCNN
- Faster RCNN
- YOLO

Task	Image classification	Object Detection
Output	Car	Car, exact bounding box containing car

Region Proposals, Bounding box regression

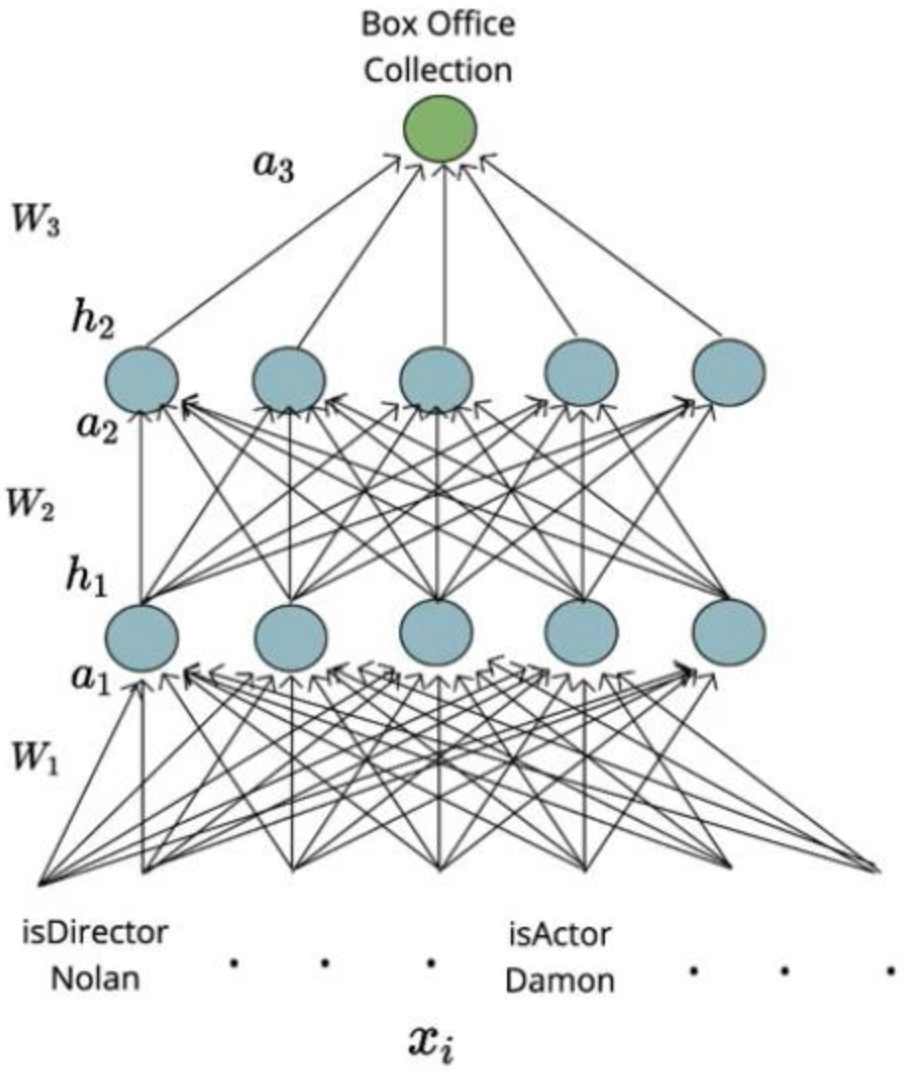


- In addition we would also like to correct the proposed bounding boxes
- This is posed as a regression problem



- Pass these through a standard image classifer to determine the class

Regression problem - FNN



$$W_3 = \begin{bmatrix} w_{3\,1} & w_{3\,2} & \cdot & \cdot & \cdot & w_{3\,5} \end{bmatrix} \qquad h_2 = \begin{bmatrix} h_{2\,1} \\ h_{2\,2} \\ \cdot \\ \cdot \\ h_{2\,5} \end{bmatrix}$$

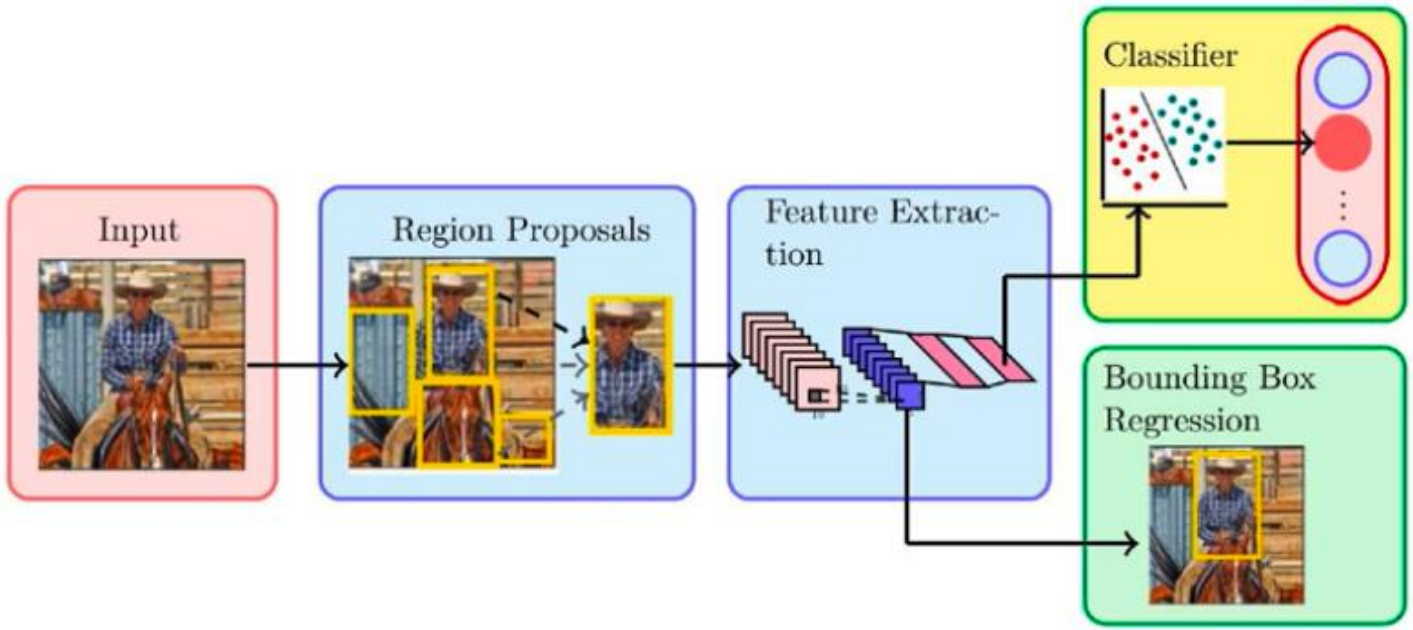
Squared Error Loss

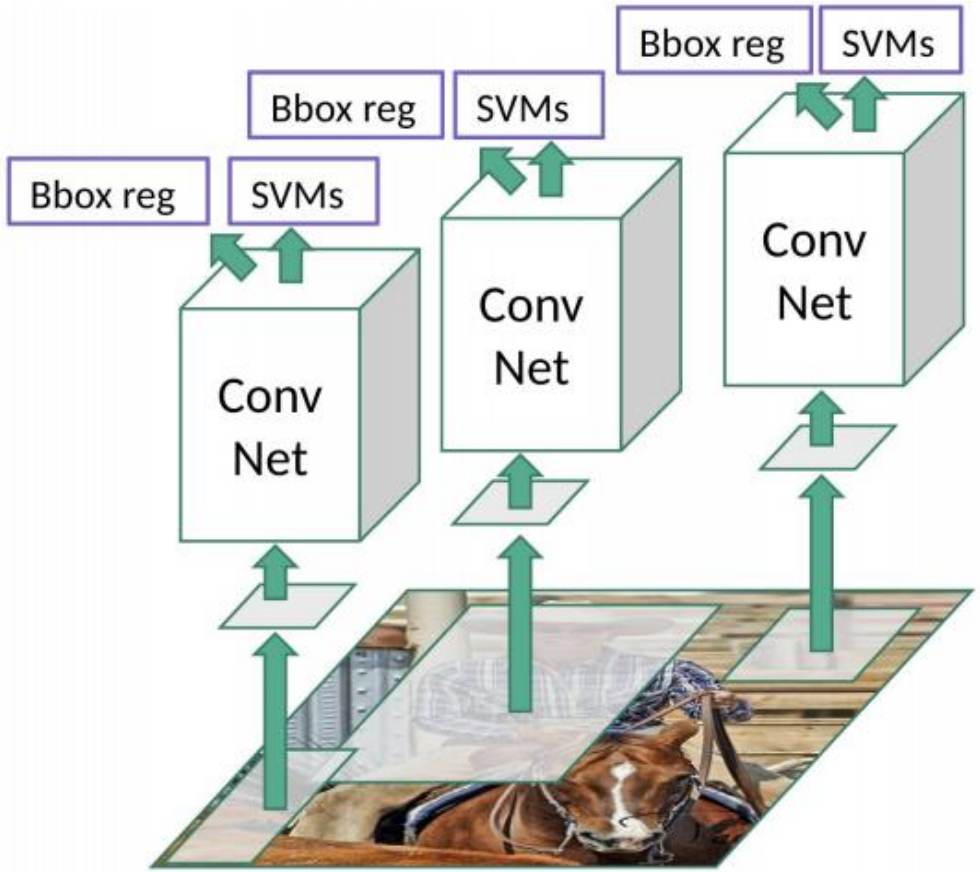
Squared Error loss for each training example, also known as **L2 Loss**, is the square of the difference between the actual and the predicted values:

The corresponding cost function is the **Mean** of these **Squared Errors (MSE)**.

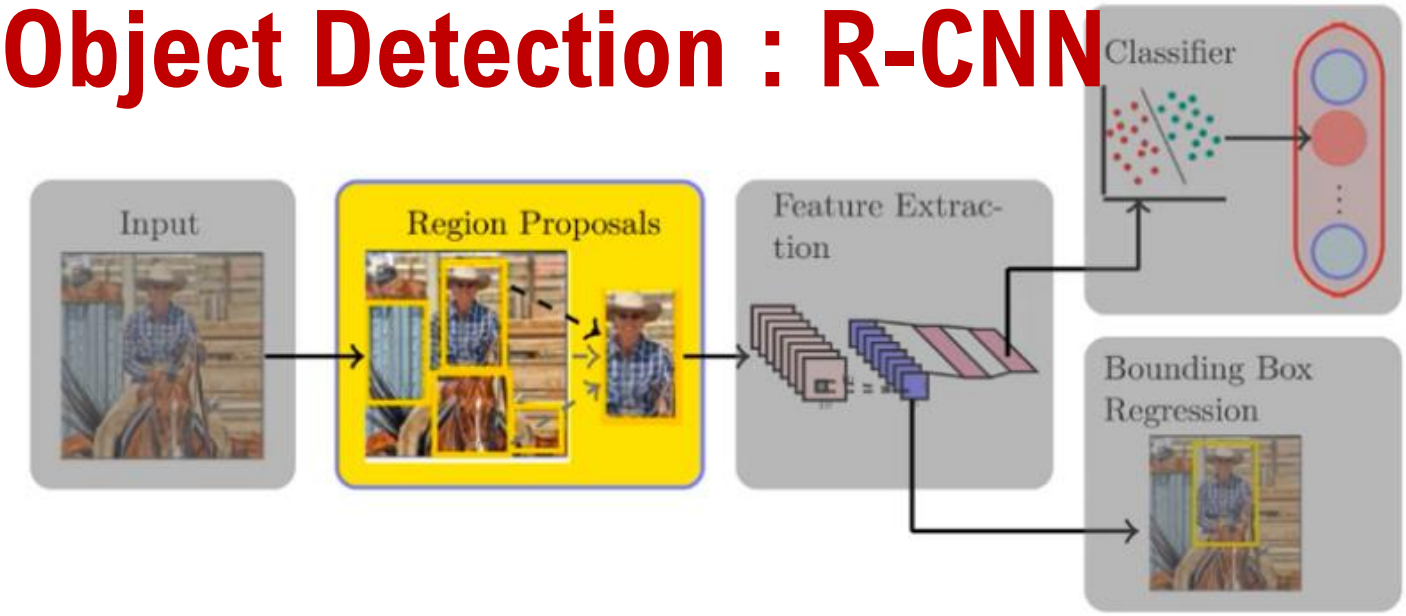
$$L = \left(y - f(x)\right)^2$$

Object Detection : R-CNN

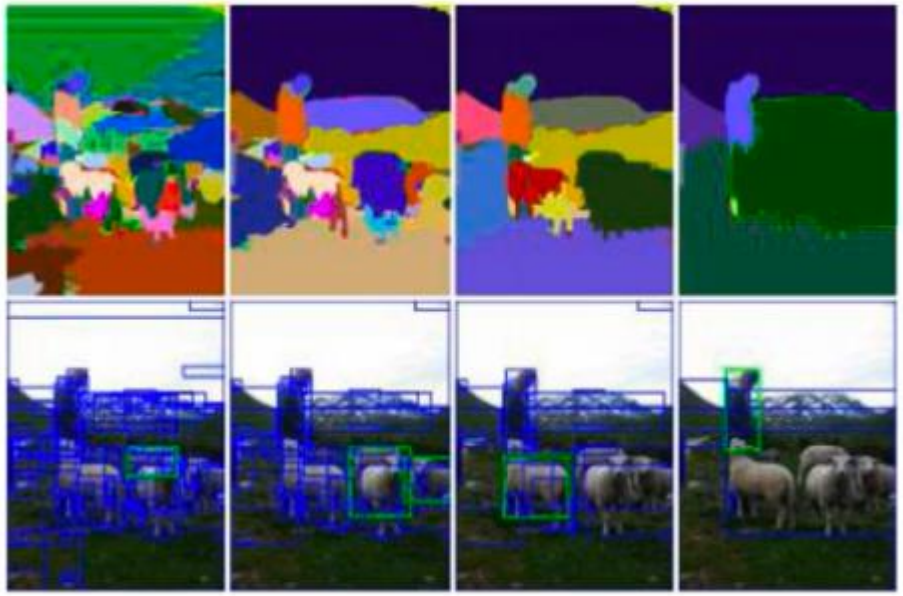




Object Detection : R-CNN

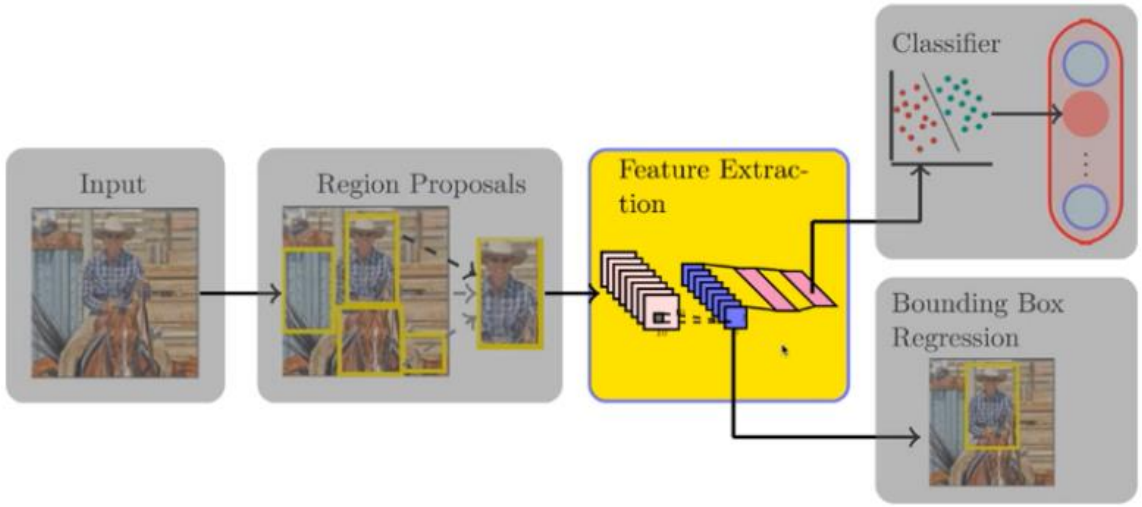


Selective Search:
1. Generate initial sub-segmentation, we generate many candidate regions
2. Use greedy algorithm to recursively combine similar regions into larger ones
3. Use the generated regions to produce the final candidate region proposals

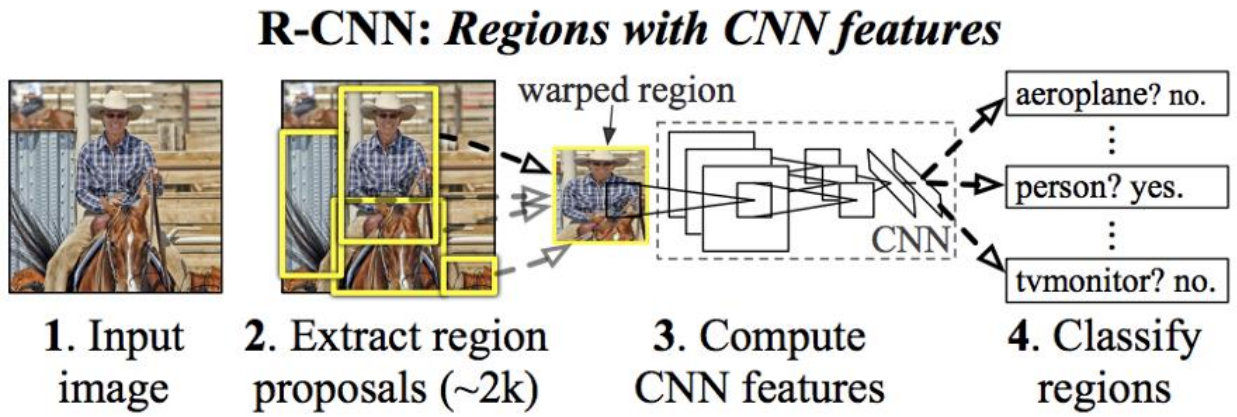


- **Selective Search** for region proposals
- Does hierarchical clustering at different scales
- For example the figures from left to right show clusters of increasing sizes
- Such a hierarchical clustering is important as we may find different objects at different scales

Object Detection : R-CNN

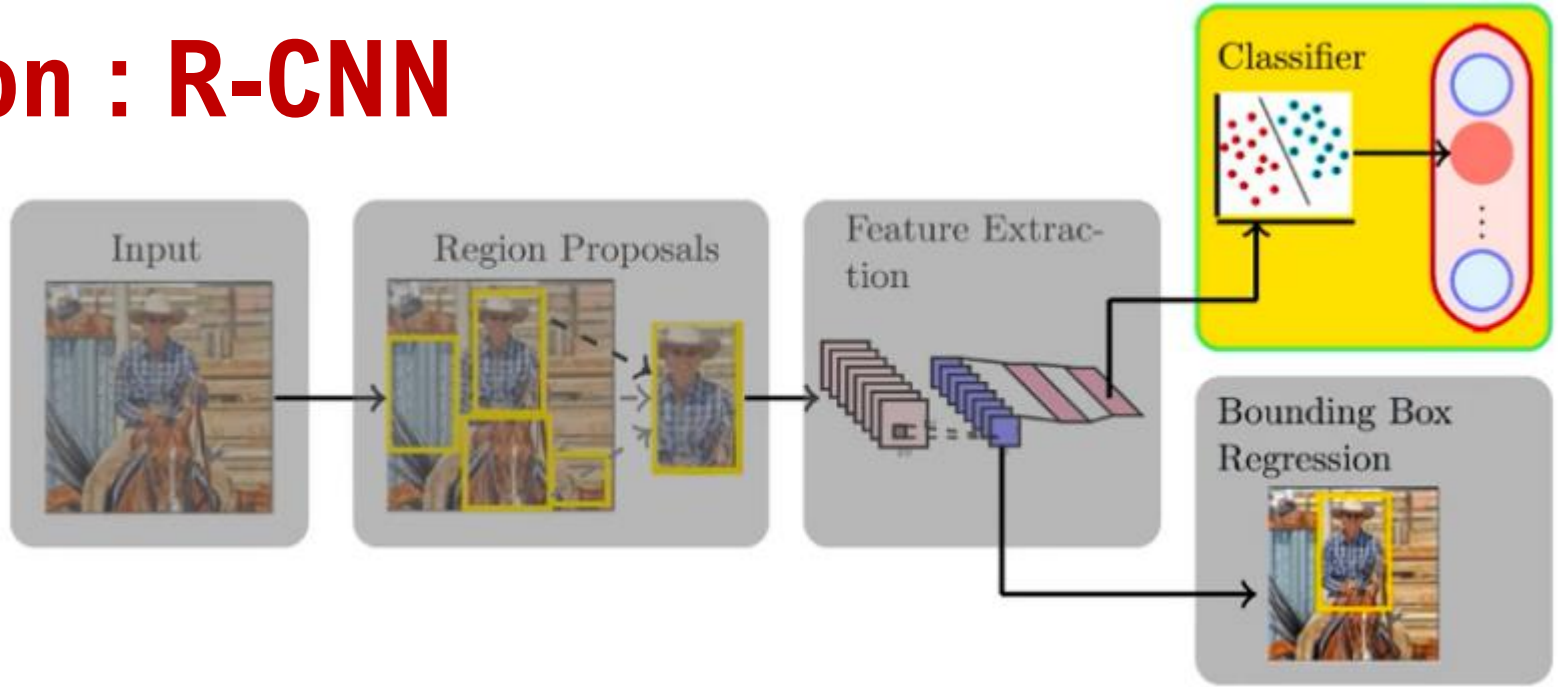


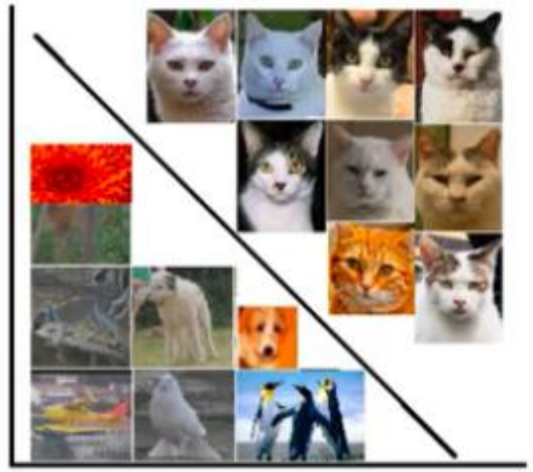
- Proposed regions are cropped to form mini images
- Each mini image is scaled to match the CNN's (feature extractor) input size



- For feature extraction any CNN trained for Image Classification can be used (AlexNet/ VGGNet etc.)

Object Detection : R-CNN





- Linear models (SVMs) are used for classification

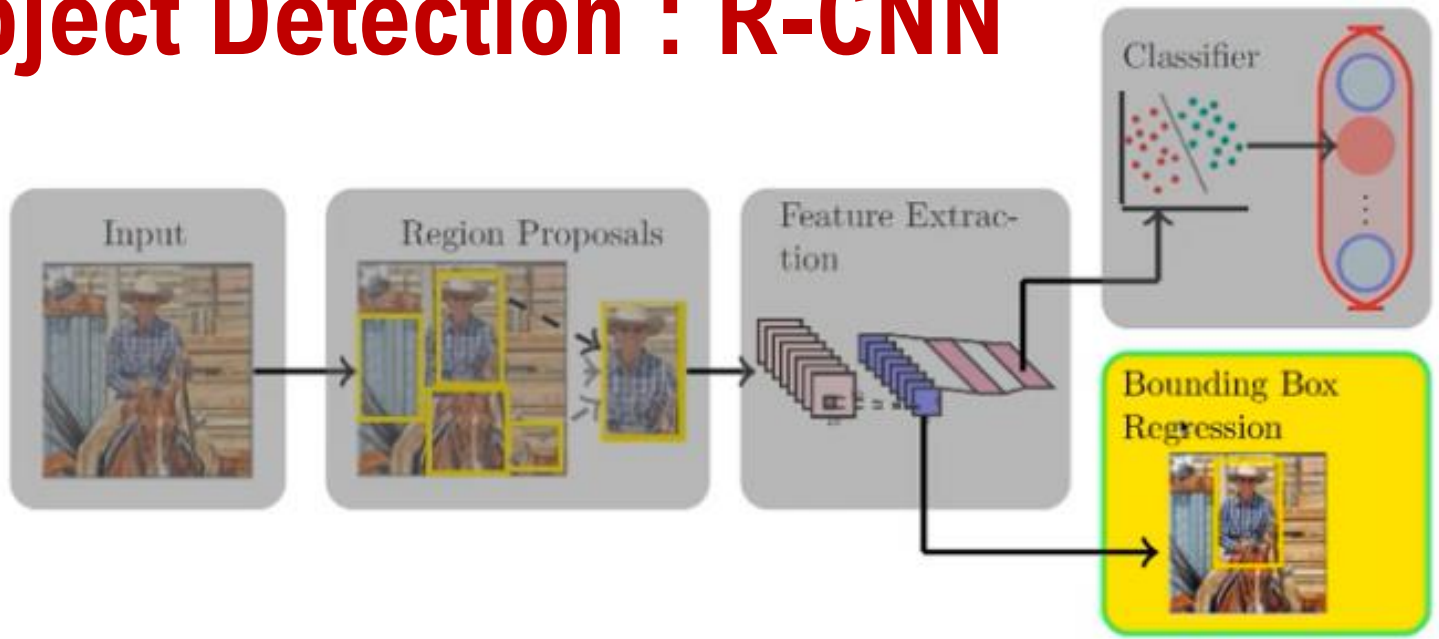
Object Detection : R-CNN

Find out the correct bounding box

$\boldsymbol{x \rightarrow image(x, y, w, h)}$

$\boldsymbol{y\_hat \rightarrow (x^{|}, y^{1} w^{1} h^{1})}$

$\boldsymbol{y \rightarrow (x^{*}, y^{*} w^{*} h^{*})}$ **ground truth**



$$\min \sum_{i=1}^{N} \left(\frac{x^* - x}{w} - w_1^T z \right)^2$$

$i=1$

Proposed Box

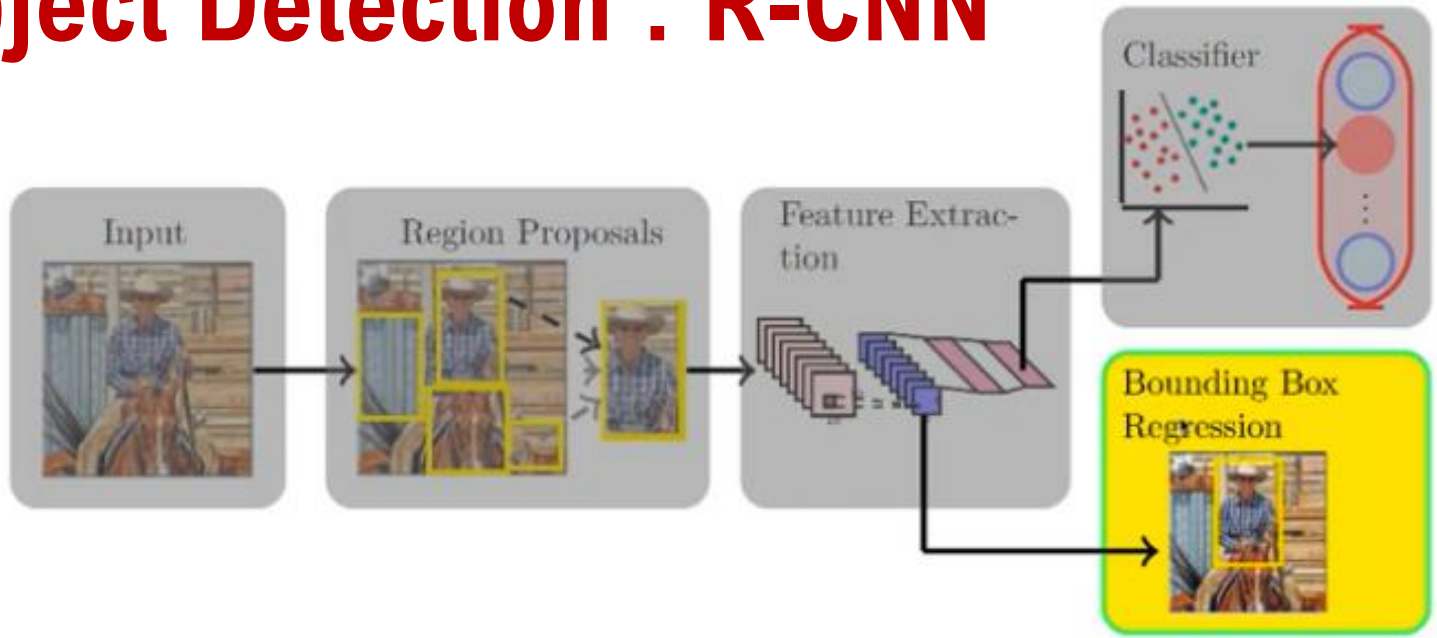
True Box

z : features from pool5 layer of the network

- $\frac{x^* - x}{w}$ is the normalized difference between proposed x and true x^*
- If we can predict this difference we can compute x^*
- The model predicts $w_1^T z$ as this difference

In case of overlapping boxes, the ones which has more than 50% overlap with ground truth box is considered

Object Detection : R-CNN



$$\min \sum_{i=1}^{N} \left(\frac{y^* - y}{h} - w_2^T z \right)^2$$

Proposed Box

True Box

z : features from pool5 layer of the network

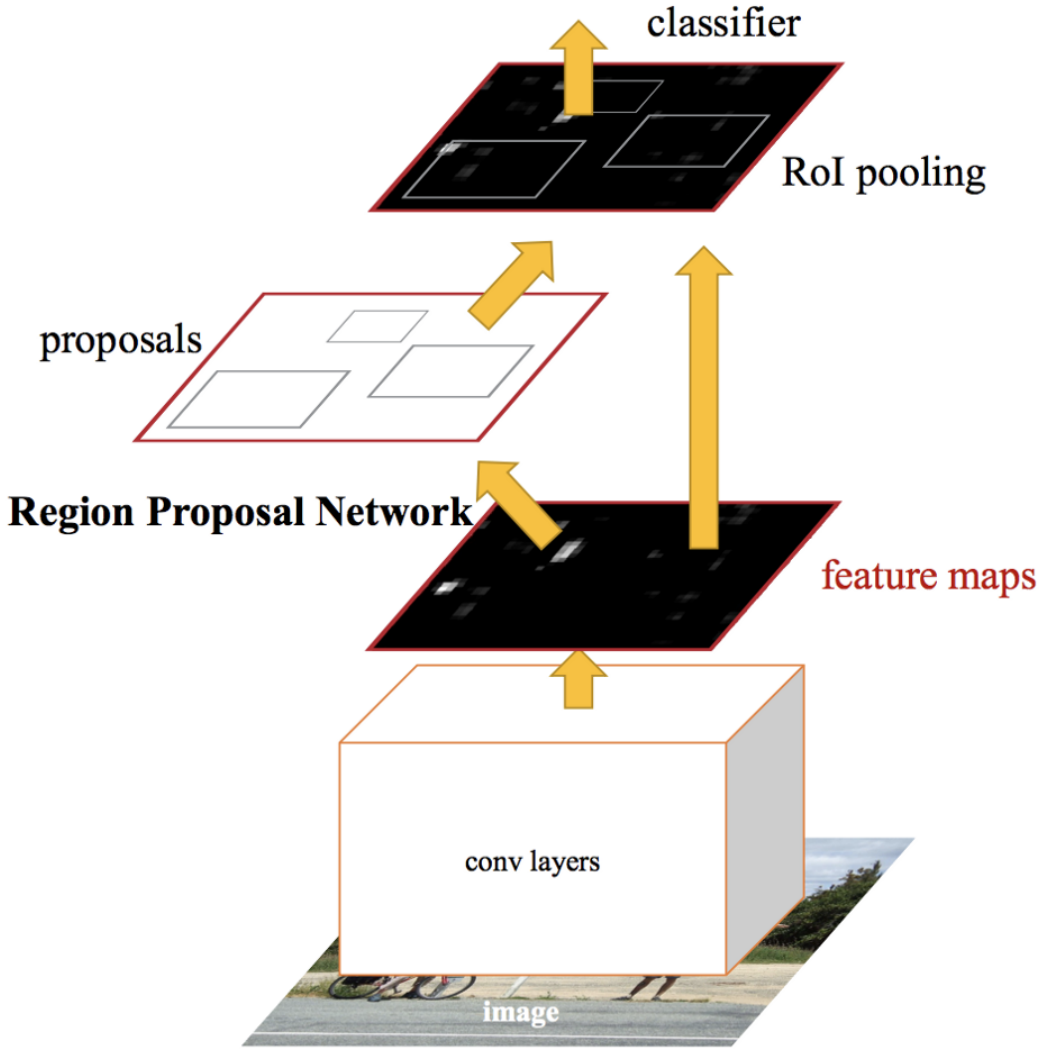
- Similarly for y

Problems with R-CNN

- It still takes a huge amount of time to train the network as you would have to classify 2000 region proposals per image.
- It cannot be implemented real time as it takes around 47 seconds for each test image.
- The selective search algorithm is a fixed algorithm. Therefore, no learning is happening at that stage. This could lead to the generation of bad candidate region proposals.

Faster RCNN

- The approach is similar to the R-CNN algorithm.
- But, instead of feeding the region proposals to the CNN, we feed the input image to the CNN to generate a convolutional feature map.
- From the convolutional feature map, we identify the region of proposals and warp them into squares and by using a RoI pooling layer we reshape them into a fixed size so that it can be fed into a fully connected layer.
- From the RoI feature vector, we use a softmax layer to predict the class of the proposed region and also the offset values for the bounding box.



The reason “Fast R-CNN” is faster than R-CNN is because you don’t have to feed 2000 region proposals to the convolutional neural network every time. Instead, the convolution operation is done only once per image and a feature map is generated from it.

What is YOLO and Why is it Useful?

YOLO is an algorithm to provide real-time object detection. This algorithm is popular because of its speed and accuracy. It has been used in various applications to detect traffic signals, people, parking meters, and animals.

The R-CNN family of techniques primarily use regions to localize the objects within the image. The network does not look at the entire image, only at the parts of the images which have a higher chance of containing an object.

The YOLO framework (You Only Look Once) on the other hand, deals with object detection in a different way. It takes the entire image in a single instance and predicts the bounding box coordinates and class probabilities for these boxes. **The biggest advantage of using YOLO is its superb speed** – it's incredibly fast and can process 45 frames per second. YOLO also understands generalized object representation.
This is one of the best algorithms for object detection and has shown a comparatively similar performance to the R-CNN algorithms

- YOLO algorithm works using the following three techniques:
- Residual blocks
- Bounding box regression
- Intersection Over Union (IOU)

Residual blocks

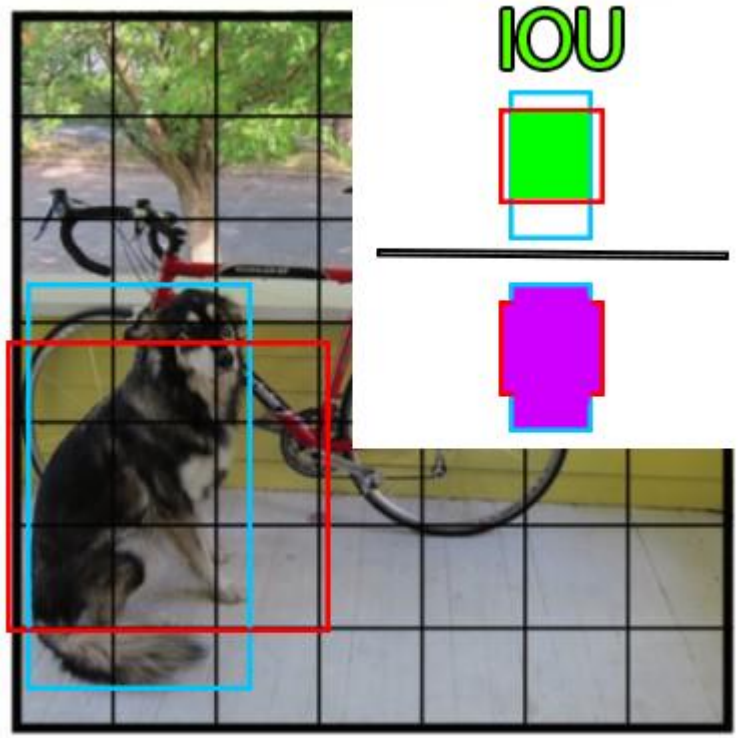
First, the image is divided into various grids. Each grid has a dimension of S x S. The following image shows how an input image is divided into grids.

In the image above, there are many grid cells of equal dimension. Every grid cell will detect objects that appear within them. For example, if an object center appears within a certain grid cell, then this cell will be responsible for detecting it.

The IOU stands for Intersection Over Union and is the area of the intersection of the predicted and ground truth boxes divided by the area of the union of the same predicted and ground truth boxes.

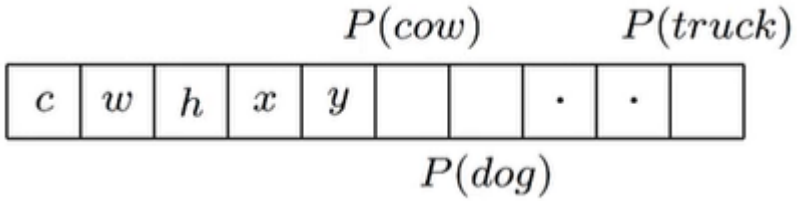
Each of these bounding boxes is made up of 5 numbers: the x position, the y position, the width, the height, and the confidence. The coordinates `(x, y)` represent the location of the center of the predicted bounding box, and the width and height are fractions relative to the entire image size.

The confidence represents the IOU between the predicted bounding box and the actual bounding box, referred to as the ground truth box.



Here is an example of an IOU: the area of intersection of the ground truth and predicted box in green divided by the area of the union of the two boxes, in purple. This will be between 0 and 1, 0 if they don't overlap at all, and 1 if they are the same box. Therefore, a higher IOU is better as it is a more accurate prediction

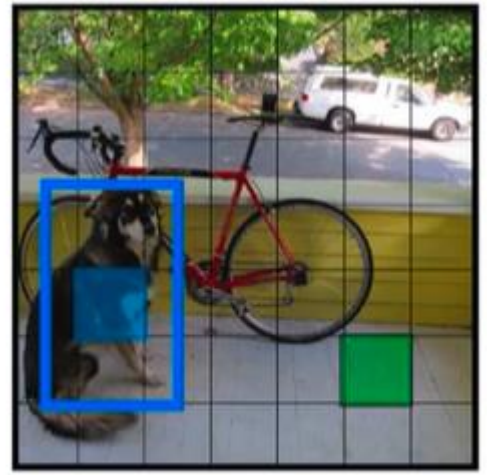
YOLO



S × S grid on input

- Divide an image into $S \times S$ grids (S=7)

- How do we interpret this $S \times S \times (5+k)$ dimensional output?
- For each cell, we are computing a bounding box, its confidence and the object in it
- We then retain the most confident bounding boxes and the corresponding object label



S × S grid on input

YOLO Training

$\hat{c}$	$\hat{w}$	$\hat{h}$	$\hat{x}$	$\hat{y}$	$\hat{\ell_1}$	$\hat{\ell_2}$	·	·	$\hat{\ell_k}$

S × S grid on input

- $\sum_{i=1}^{k}(\ell_i - \hat{\ell_i})^2$
- And train the network to minimize the sum of these losses

- How do we train this network ?
- Consider a cell such that the center of the true bonding box lies in it
- The network is initialized randomly and it will predict some values for c, w, h, x, y & ℓ
- We can then compute the following losses
- $(1 - \hat{c})^2$
- $(\sqrt{w} - \sqrt{\hat{w}})^2$
- $(\sqrt{h} - \sqrt{\hat{h}})^2$
- $(x - \hat{x})^2$
- $(y - \hat{y})^2$

Namah Shivaya