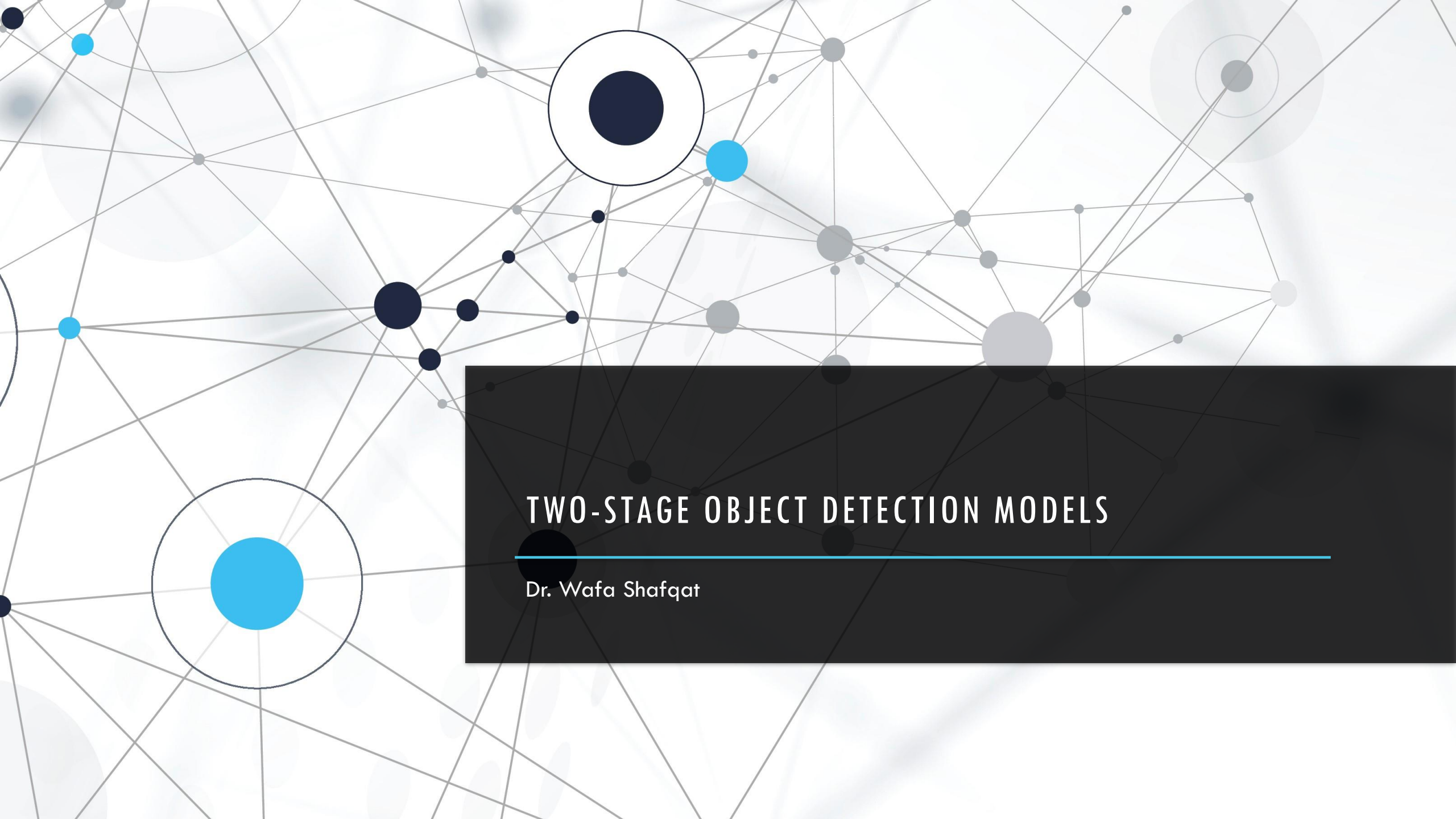
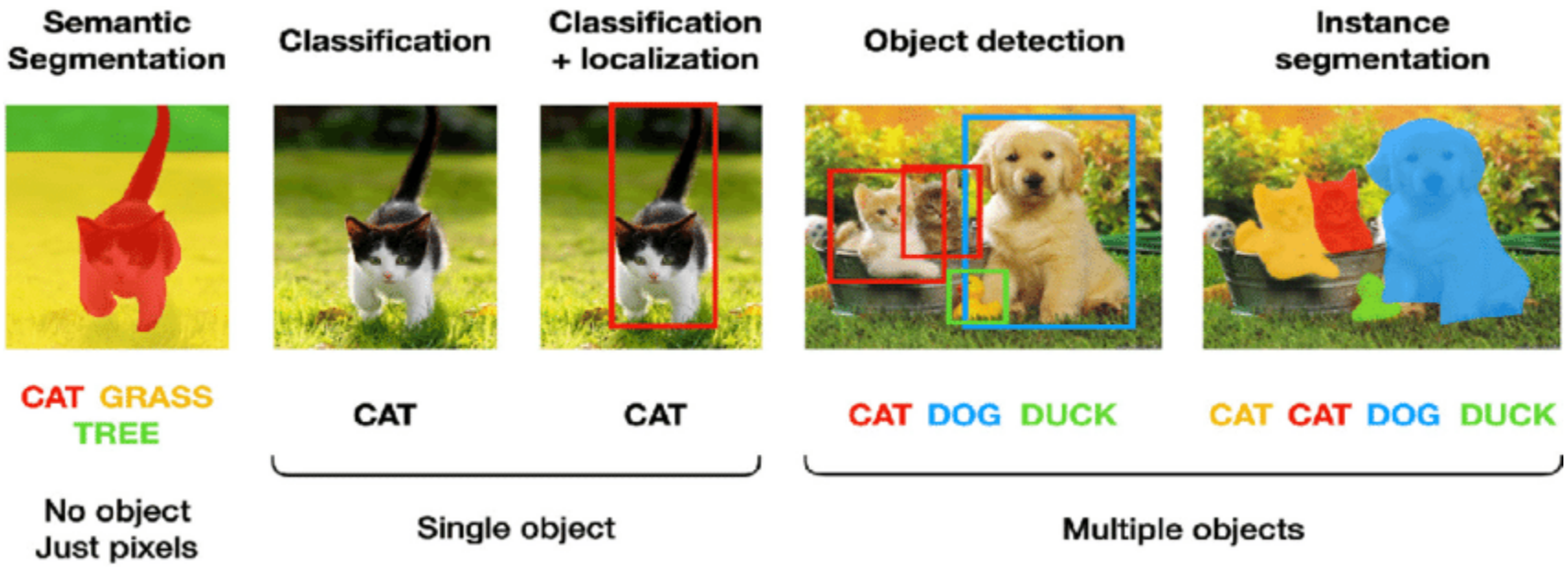


TWO-STAGE OBJECT DETECTION MODELS

Dr. Wafa Shafqat

COMPUTER VISION

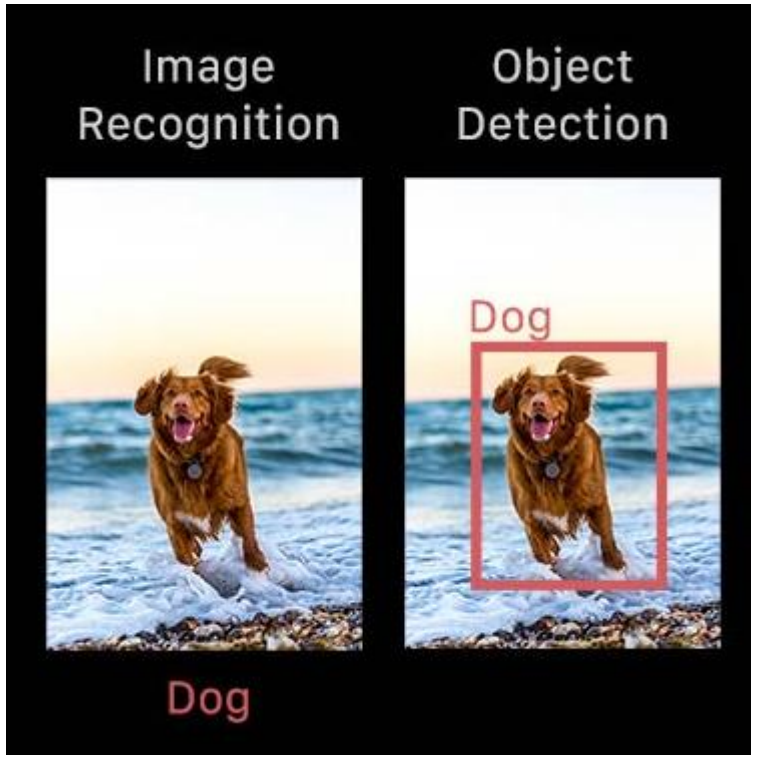


OBJECT DETECTION

Object detection is a computer vision technique that works to **identify and locate objects** within an image or video.

Image Recognition vs **Object Detection**

- **Image recognition** assigns a label to an image.
- **Object detection**, on the other hand, draws a box around each object and labels the box with the object name.



WHY IS OBJECT DETECTION IMPORTANT?

- **Image recognition** only outputs a class label for an identified object
- **Image segmentation** creates a pixel-level understanding of a scene's elements
- **Object Detection** can locate objects within image or video and then allow us to count and track those objects
  - Crowd counting
  - Self-driving cars
  - Video surveillance
  - Face detection
  - Anomaly detection

ONE-STAGE AND TWO-STAGE MODELS

Two-Stage Models

- The first stage is for **region proposal**
- **Image classification** on the proposed regions

Regions with CNN Features (R-CNN)
Fast R-CNN
Faster R-CNN
Mask R-CNN

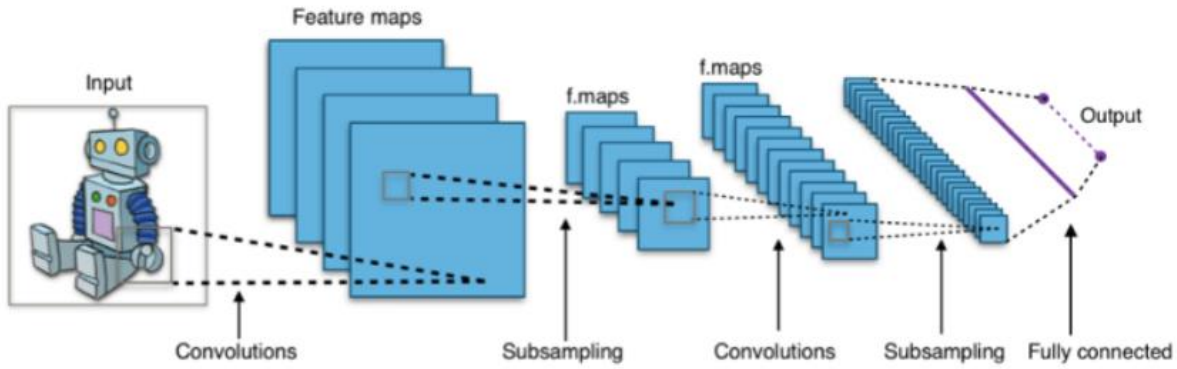
One-Stage Models

- Runs **detection directly** over a dense sampling of possible locations
- One-stage models usually divide the image into N ×N grids.
- Each grid cell is responsible for the object whose center falls into that grid.

You Only Look Once (YOLO)
YOLOv2 and YOLO9000
YOLOv3
Single Shot Detector (SSD)

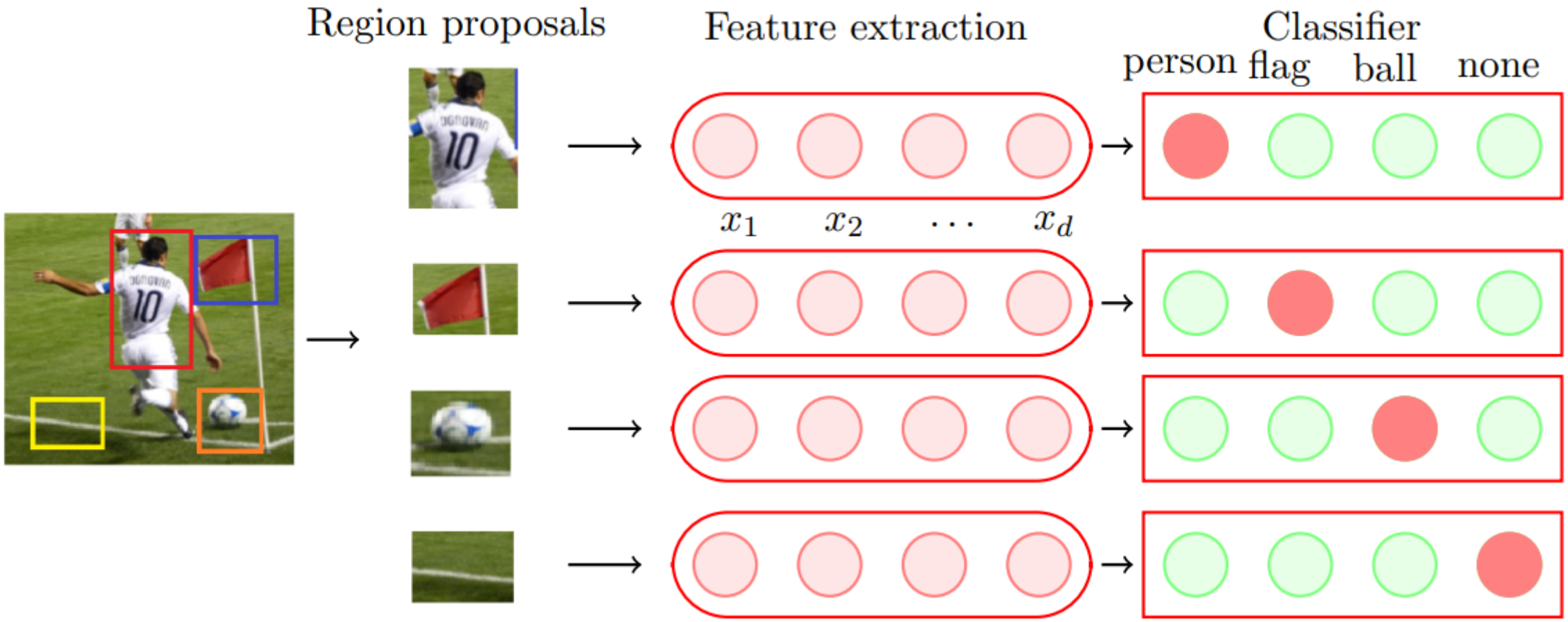
CONVOLUTIONAL NEURAL NETWORK (CNN)

- Input Layer
- Hidden Layer
- Output Layer

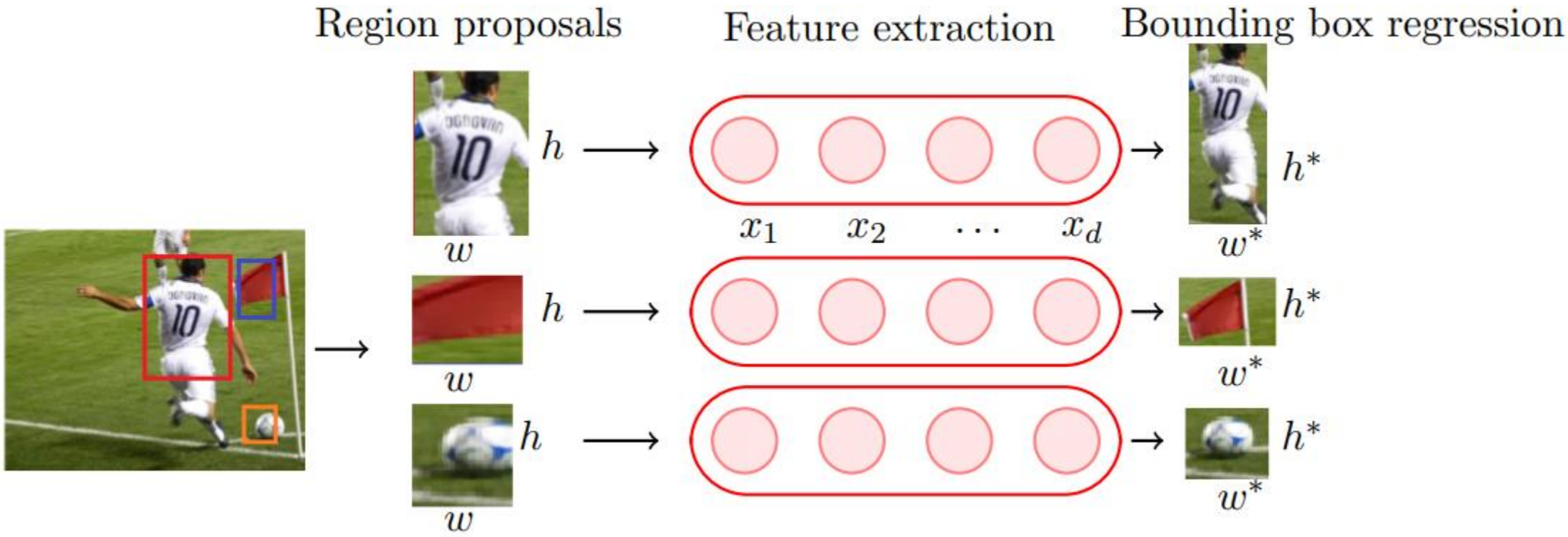


- The **hidden layers** are **convolutional layers** in this type of neural network which acts like a **filter** that first receives **input**, transforms it using a specific pattern/feature, and sends it to the next layer.
- As more and more layers the input goes through, the more sophisticated patterns the future ones can detect.

OBJECT DETECTION PIPELINE



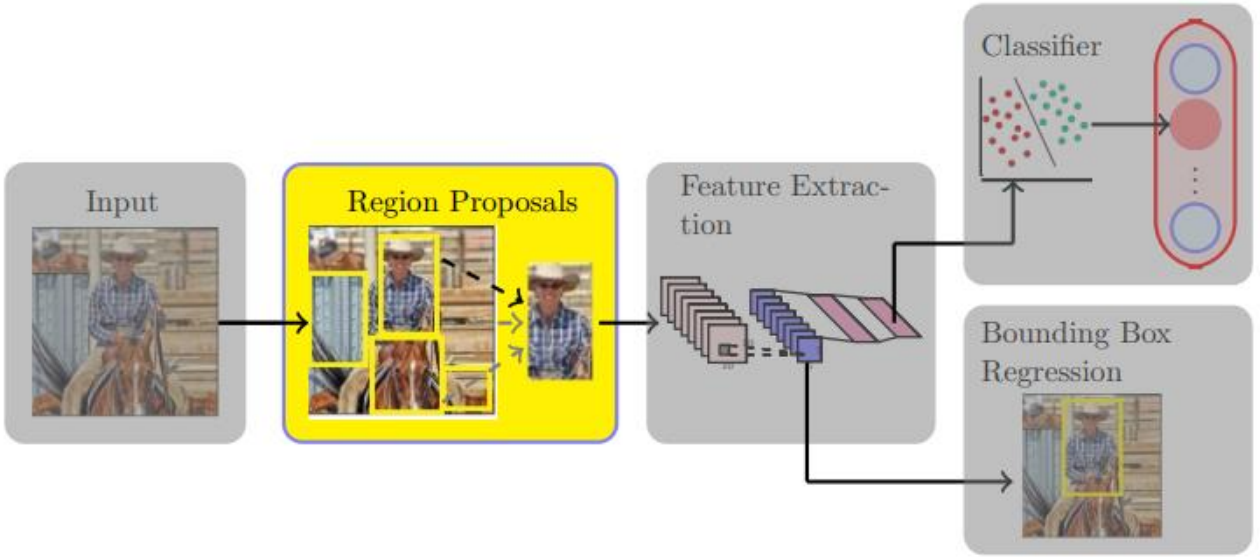
BOUNDING BOX REGRESSION



- We can correct the proposed bounding boxes
- This is posed as a regression problem (for example, we would like to predict $w*$, $h*$ from the proposed w and h)

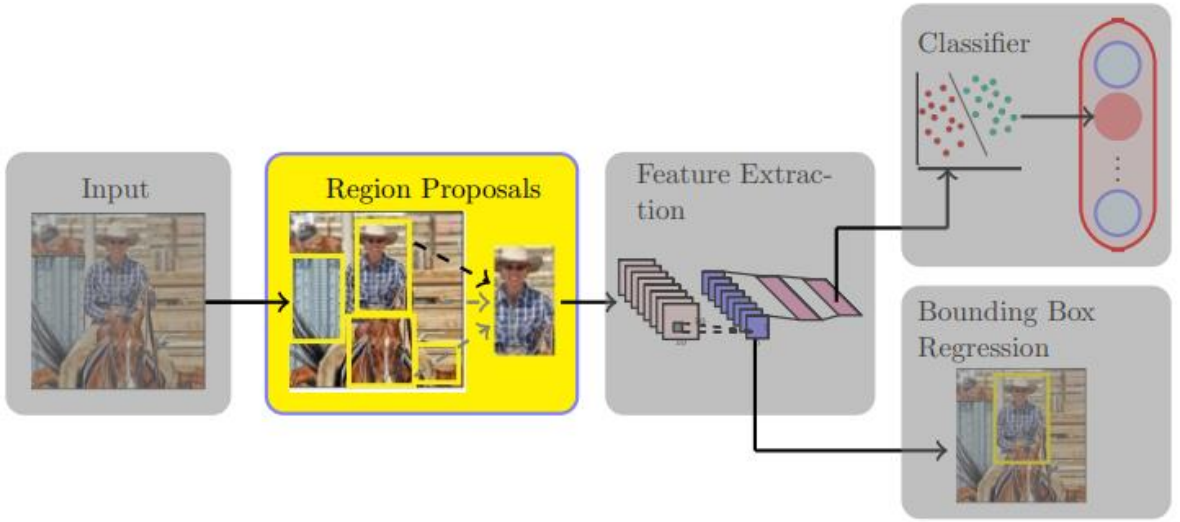
RCNN

- **Selective search** is used for region proposal.
- On average selective search gives **2K region proposal**
- It is based on computing **hierarchical clustering** of similar regions based on **color**, **texture**, **size** and **shape compatibility**.
- For example the figures from left to right show clusters of increasing sizes



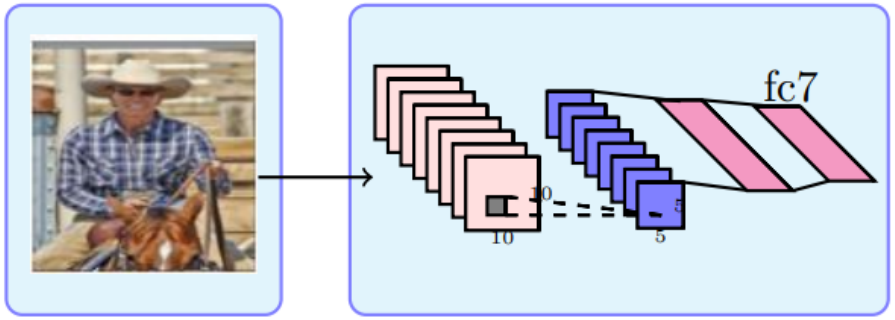
RCNN

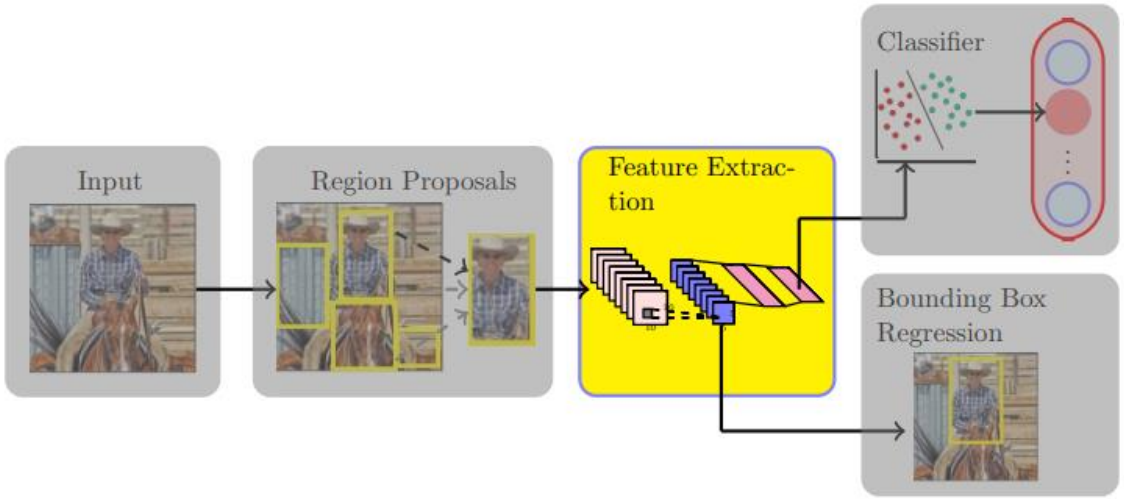
- **Proposed regions** are cropped to form **mini images**
- Each mini image is scaled to match the CNN's (**feature extractor**) **input size**



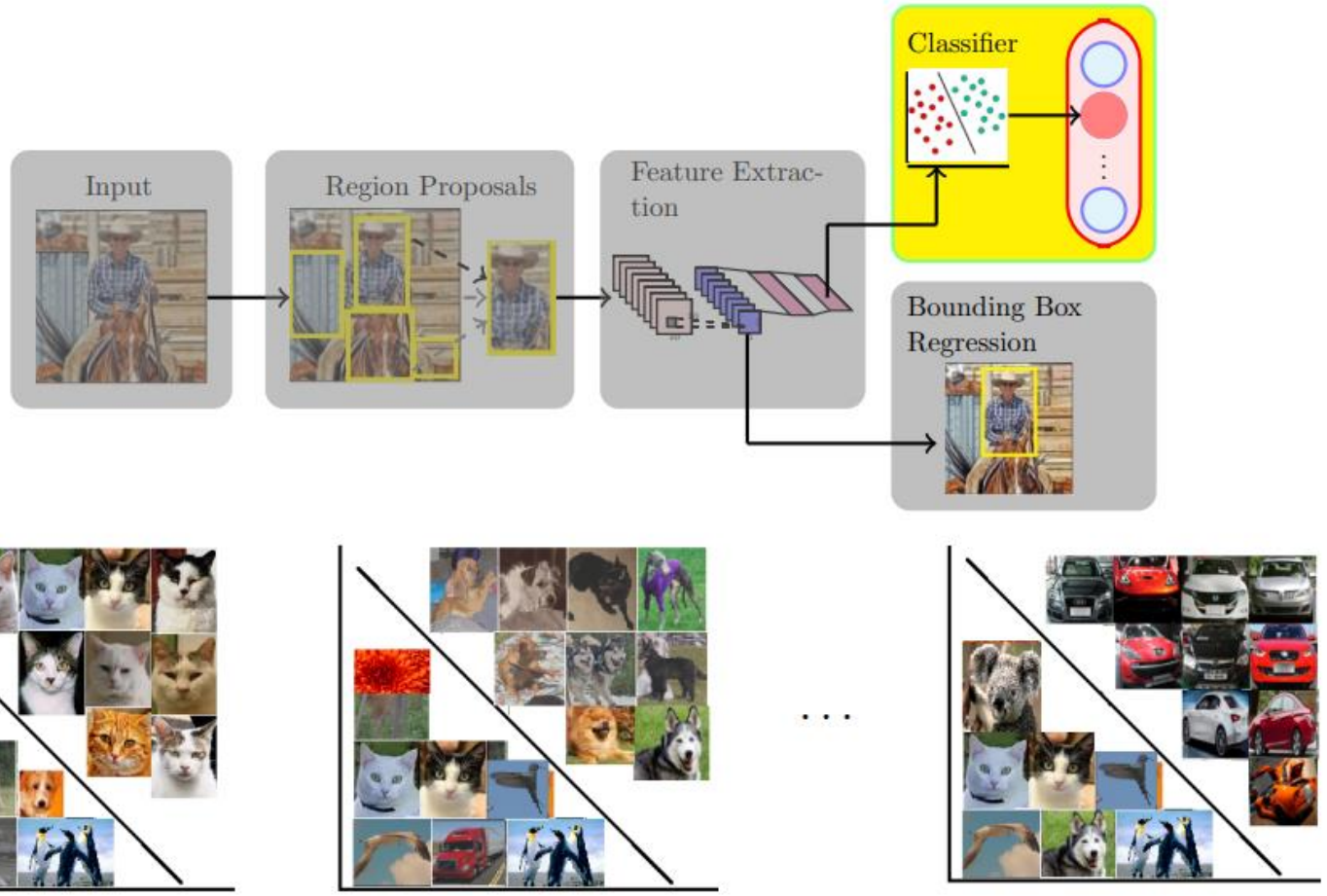
RCNN

- For **feature extraction** any CNN trained for Image Classification can be used (AlexNet/ VGGNet etc.)
- Outputs from fc7 layer are taken as features
- CNN is fine tuned using ground truth (cropped) object images





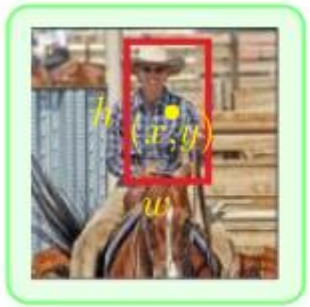
RCNN



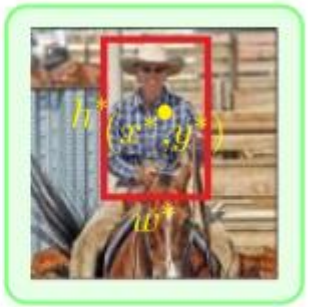
Linear models (SVMs) are used for classification (1 model per class)

RCNN

- The proposed regions may not be perfect
- We want to learn four regression models which will learn to predict x ∗ , y ∗ , w ∗ , h∗

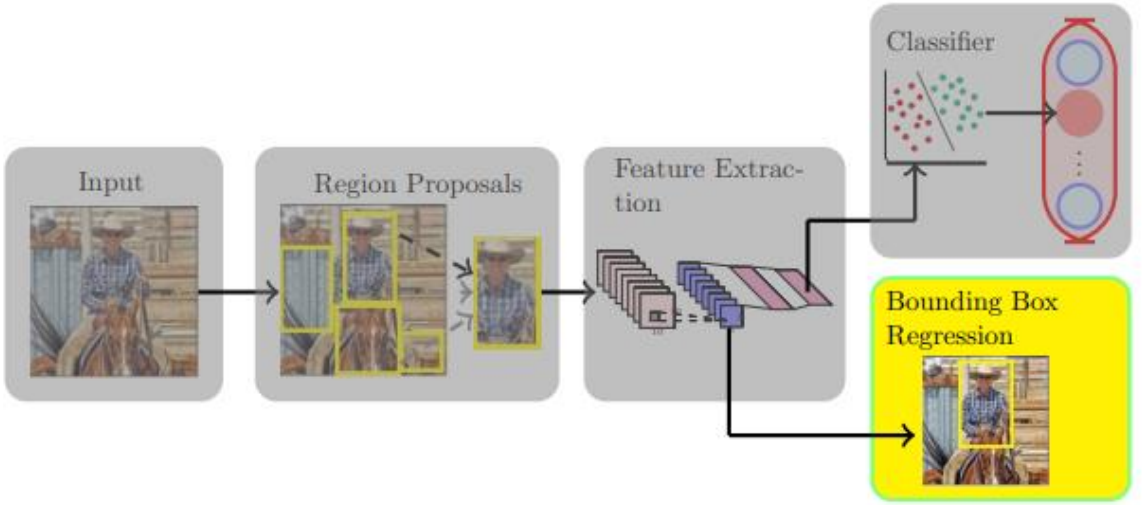


Proposed Box



True Box

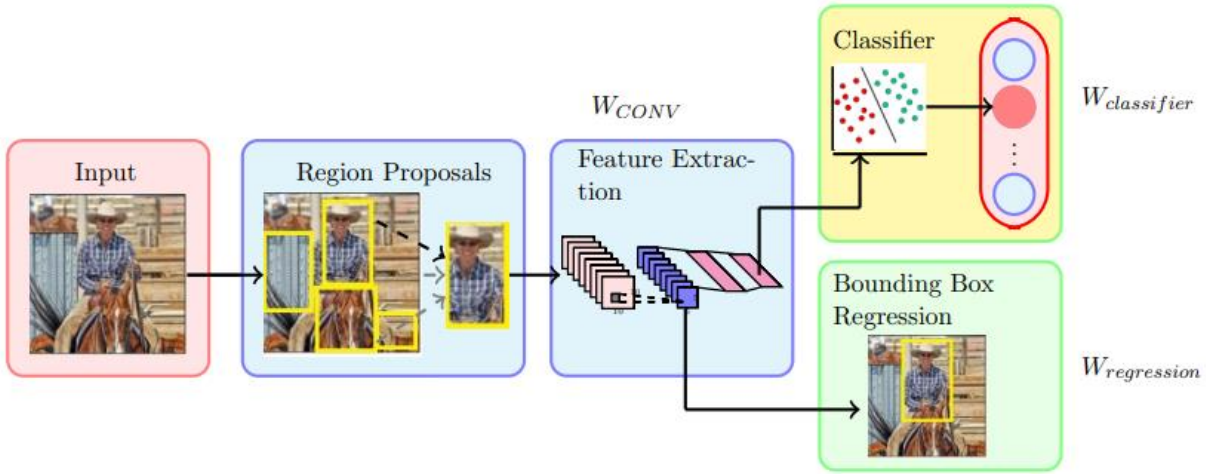
z : features from pool5 layer of the network



RCNN

Parameters of RCNN model

- **WCONV** is taken as it is from a CNN trained for Image classification (say on ImageN
- **WCONV** is then fine tuned using ground truth (cropped) object imageset)
- **Wclassifier** is learned using ground truth (cropped) object images
- **Wregression** is learned using ground truth bounding boxes



Computational Cost for processing one image at test time?

Inference Time = Proposal Time + # Proposals × Convolution Time + # Proposals × classification + # Proposals × regression

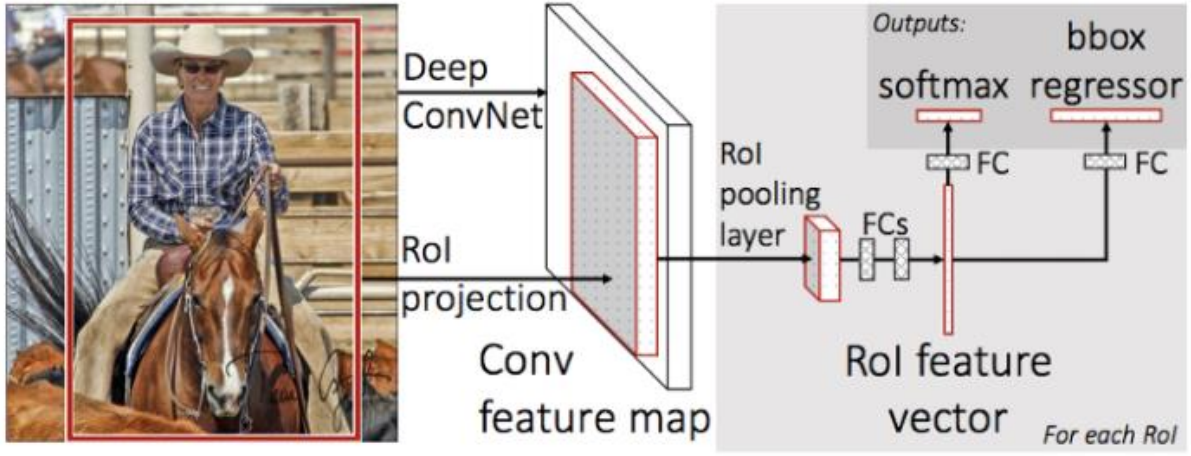
RCNN

Disadvantages of RCNN model

- 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.

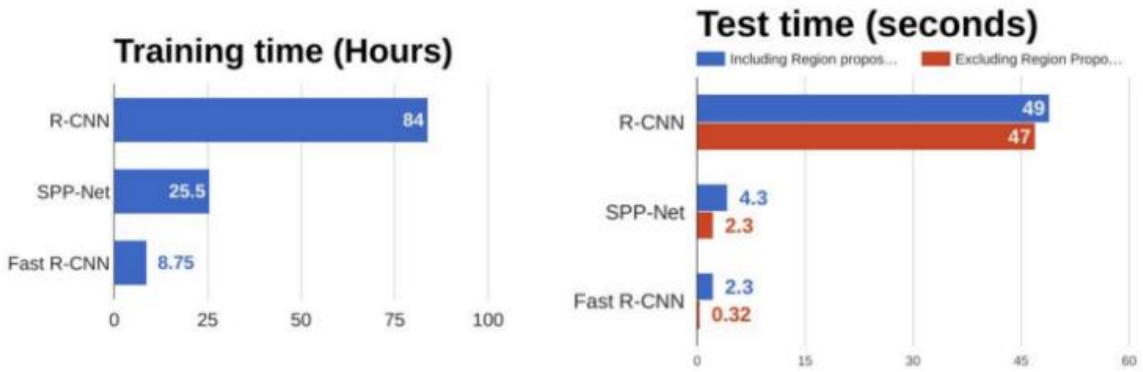
FAST RCNN

- The approach is similar to the R-CNN algorithm
- Instead of feeding the region proposals to the CNN, the **input image is fed to the CNN** to generate a convolutional feature map.
- From the convolutional feature map, **the region of proposals are identified and are warped into squares**
- Using a **RoI pooling layer** it is reshaped into a fixed size so that it can be fed into a **fully connected layer**.
- From the RoI feature vector, a **softmax layer** is used to predict the class of the proposed region and also the offset values for the bounding box.



FAST RCNN

- 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**.



Fast RCNN

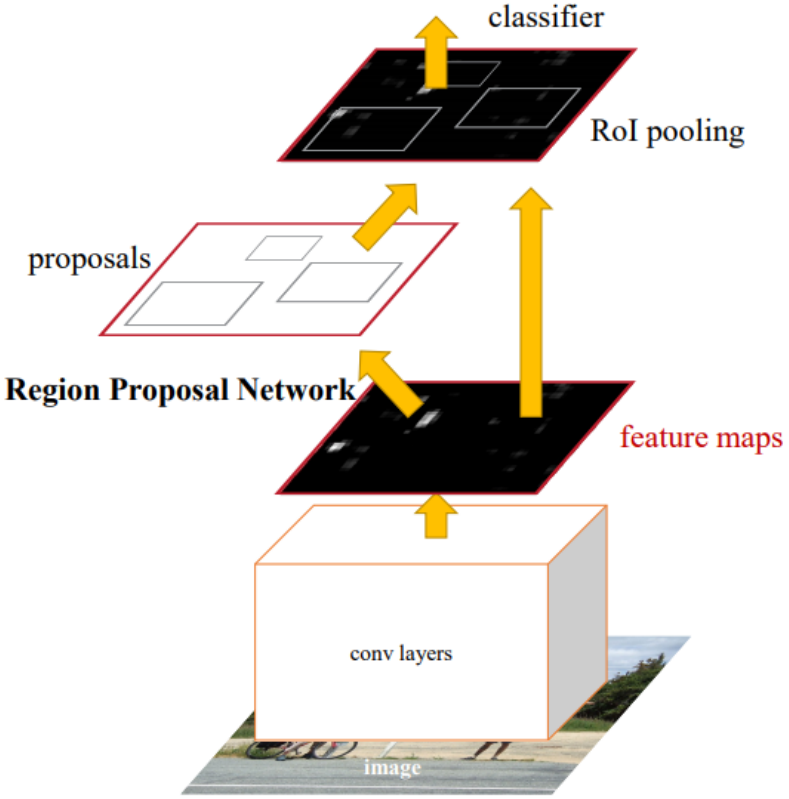
Region Proposals: Selective Search

Feature Extraction: CNN

Classifier: CNN

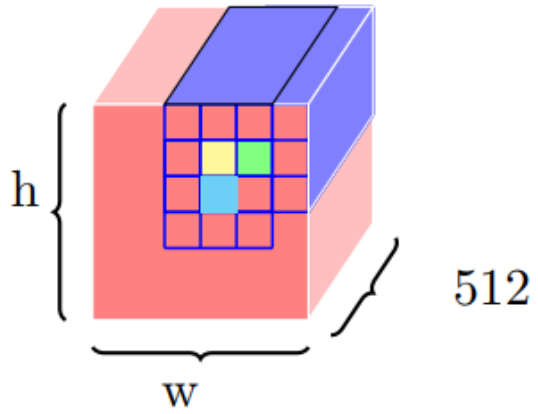
FASTER RCNN

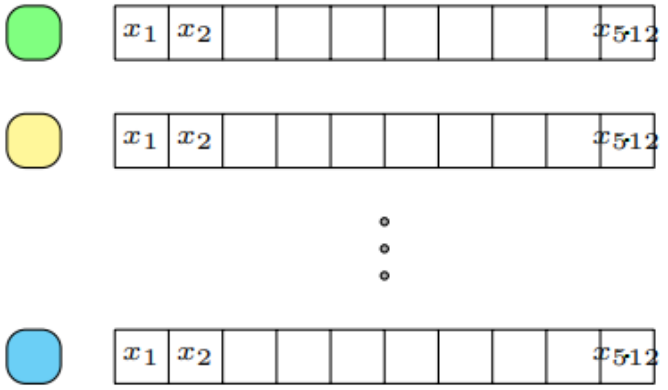
- So far the region proposals were being made using **Selective Search algorithm**
- Idea: Can we use a **CNN** for making region proposals also?
- The image is provided as an **input to a convolutional network** which provides a convolutional feature map
- Instead of using selective search algorithm on the feature map to identify the region proposals, a **separate network is used to predict the region proposals**
- The predicted region proposals are then reshaped using a RoI pooling layer
- Which is then used to classify the image within the proposed region and predict the offset values for the bounding boxes.



FASTER RCNN

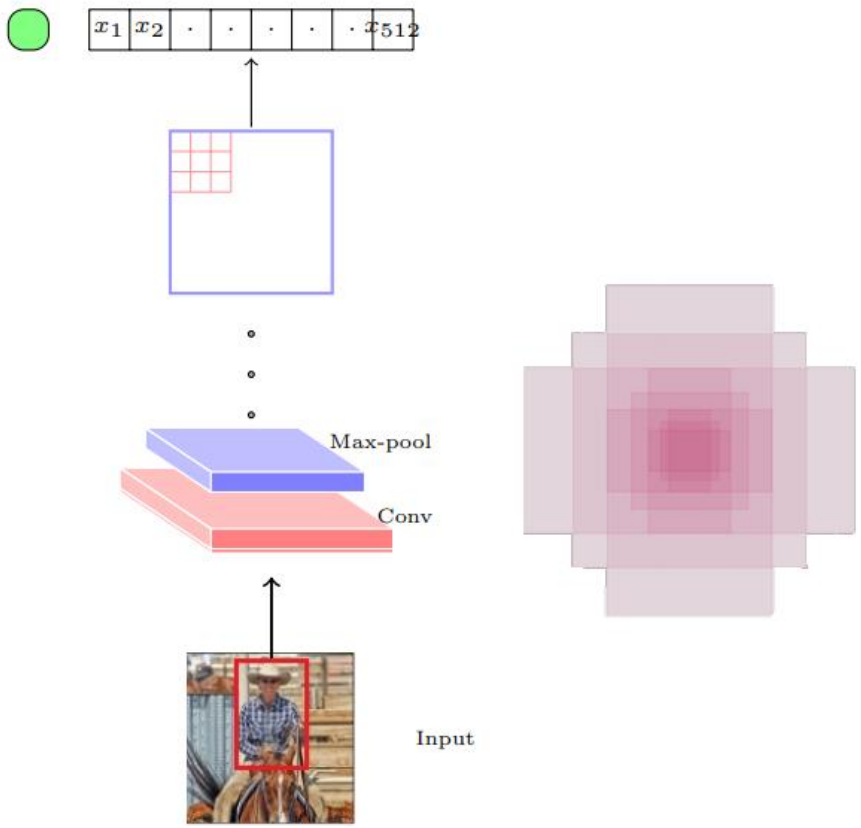
- Consider the output of the **last convolutional layer of VGGNet**
- Now consider **one cell in one of the 512 feature maps**
- If we apply a **3 ×3 kernel around this cell** then we will get a **1D representation for this cell**
- If we repeat this for all the **512 feature maps** then we will get a **512 dimensional representation** for this position
- We use this process to get a **512 dimensional representation for each of the w × h positions**





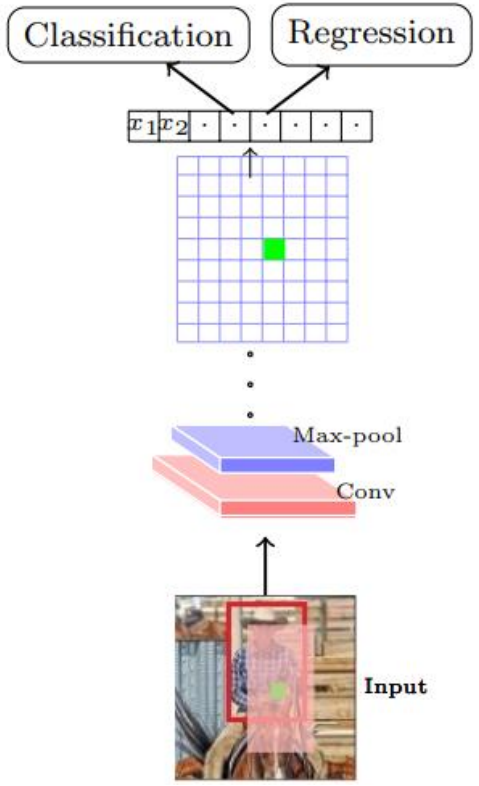
FASTER RCNN

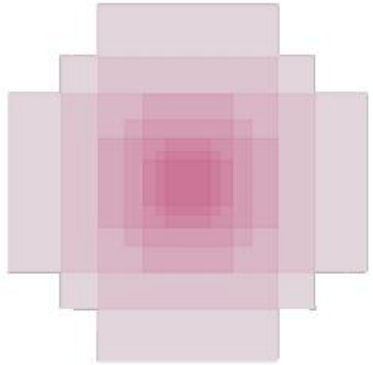
- We now consider **k bounding boxes (called anchor boxes) of different sizes & aspect ratio**
- We are interested in the following two questions:
- Given the 512d representation of a position, **what is the probability that a given anchor box centered at this position contains an object**? (Classification)
- **How do you predict the true bounding box from this anchor box**? (Regression)



FASTER RCNN

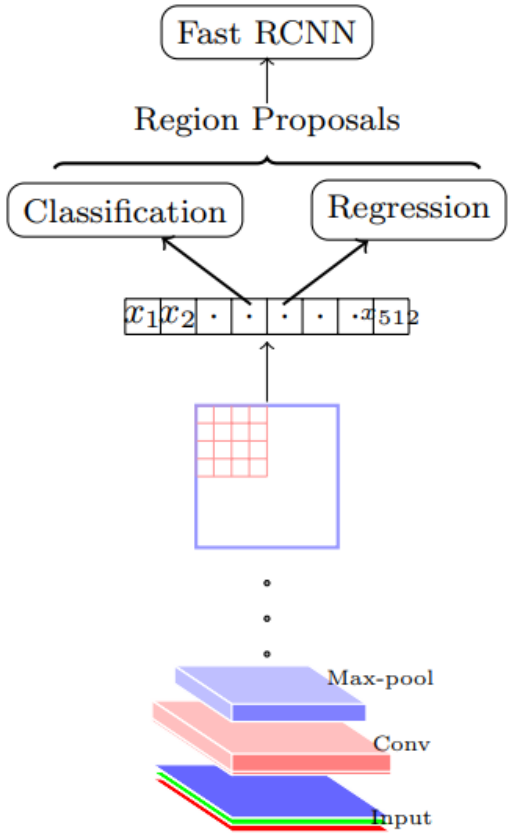
- We train a classification model and a regression model to address these two questions
- How do we get the ground truth data?
- Consider a ground truth object and its corresponding bounding box
- Consider the projection of this image onto the conv5 layer
- Consider one such cell in the output This cell corresponds to a patch in the original image
- Consider the center of this patch
- We consider anchor boxes of different sizes
- For each of these anchor boxes, we would want the classifier to predict 1 if this anchor box has a reasonable overlap (IoU > 0.7) with the true grounding box





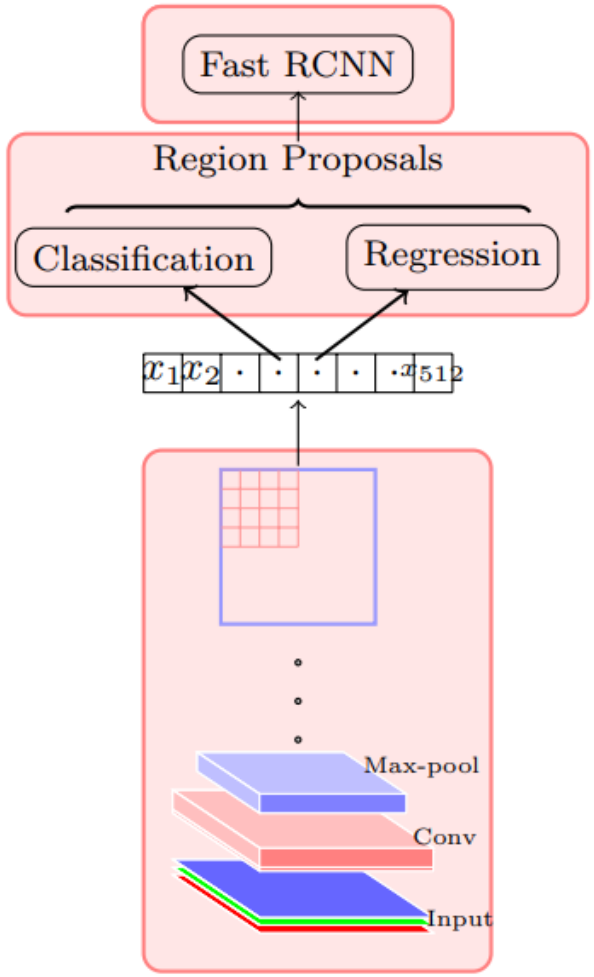
FASTER RCNN

- So far we have seen a CNN based approach for region proposals instead of using selective search
- We can now take these **region proposals and then add fast RCNN on top of it to predict the class of the object and regress the proposed bounding box**
- But the fast RCNN would again use a VGG Net
- **Can't we use a single VGG Net and share the parameters of RPN and RCNN** ?
- Yes, we use a 4 step alternating training process



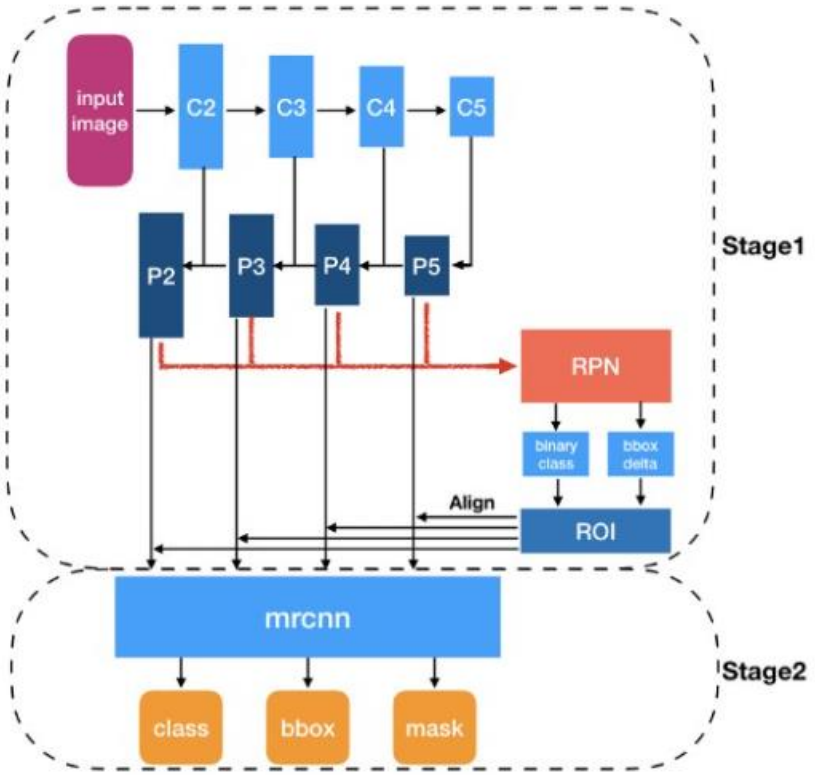
FASTER RCNN

- Faster RCNN: Training Fine-tune RPN using a pre-trained ImageNet network
- Fine-tune fast RCNN from a pretrained ImageNet network using bounding boxes from step 1
- Keeping common convolutional layer parameters fixed from step 2, finetune RPN (post conv5 layers)
- Keeping common convolution layer parameters fixed from step 3, finetune fc layers of fast RCNN



MASK RCNN

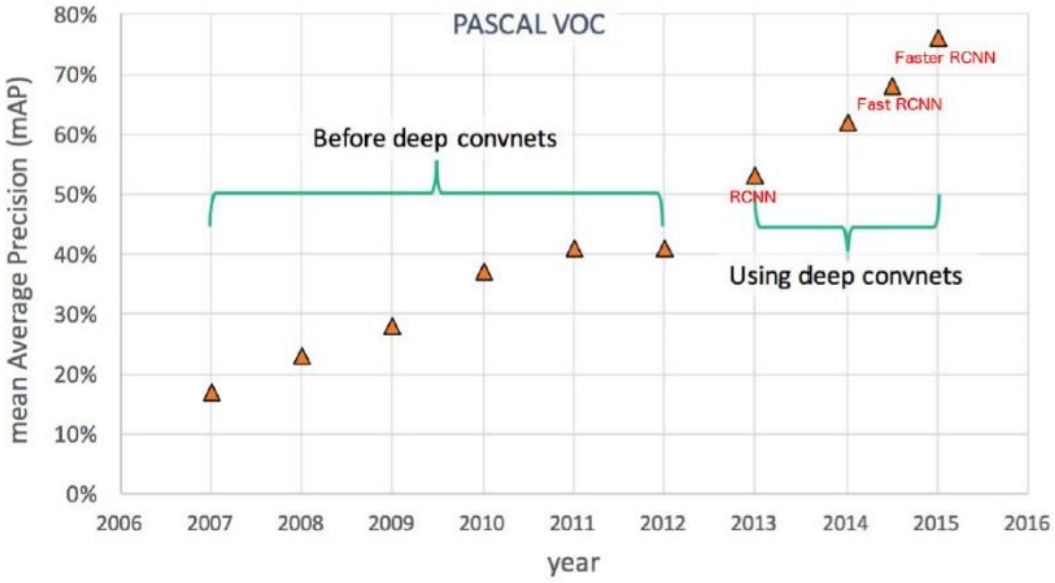
- Mask R-CNN has **three outputs**
- For each candidate object, **a class label** and a **bounding-box offset**
- Third output is the **object mask**
- **Region proposal network (RPN)** to proposes candidate object bounding boxes.
- **Binary mask classifier** to generate mask for every class



MASK RCNN

- Image is run through the **CNN** to generate the **feature maps**.
- Region Proposal Network(RPN) uses a CNN to generate the **multiple Region of Interest(RoI)** using a lightweight **binary classifier**. It does this using **9 anchors boxes** over the image. The classifier returns object/no-object scores. **Non Max suppression** is applied to Anchors with high objectness score
- The RoI Align network **outputs multiple bounding boxes** rather than a single definite one and warp them into a fixed dimension.
- Warped features are then fed into fully connected layers to **make classification using softmax** and boundary box prediction is further refined using the regression model
- Warped features are also fed into **Mask classifier**, which consists of **two CNN's** to **output a binary mask for each RoI**. Mask Classifier allows the network to generate masks for every class without competition among classes

CONCLUSION



Method	Pascal 2007 mAP	Speed
DPM v5	33.7	0.07 FPS — 14 sec/ image
RCNN	66.0	0.05 FPS — 20 sec/ image
Fast RCNN	70.0	0.5 FPS — 2 sec/ image
Faster RCNN	73.2	7 FPS — 140 msec/ image
YOLO	69.0	45 FPS — 22 msec/ image

THANK YOU !!!

Any Questions ?