Object Detection

April 1, 2024

Object Detection

- Object detection is a branch of computer vision that empowers computers to identify and locate objects within images and videos.
- Practical applications includes autonomous vehicles, surveillance systems, medical imaging, industrial automation, augmented reality etc.
- Object detection algorithms typically follow a two-step process:
  - **Localization**: Identifying the regions within an image that potentially contain objects of interest. These regions are often represented by bounding boxes.

    Techniques such as selective search, region proposal networks (RPNs), or sliding window approaches are commonly used for localization.
  - **Classification**: Extracting features from the regions and using machine learning or deep learning algorithms to classify them.

    Popular algorithms for classification include convolutional neural networks (CNNs),

Object Detection

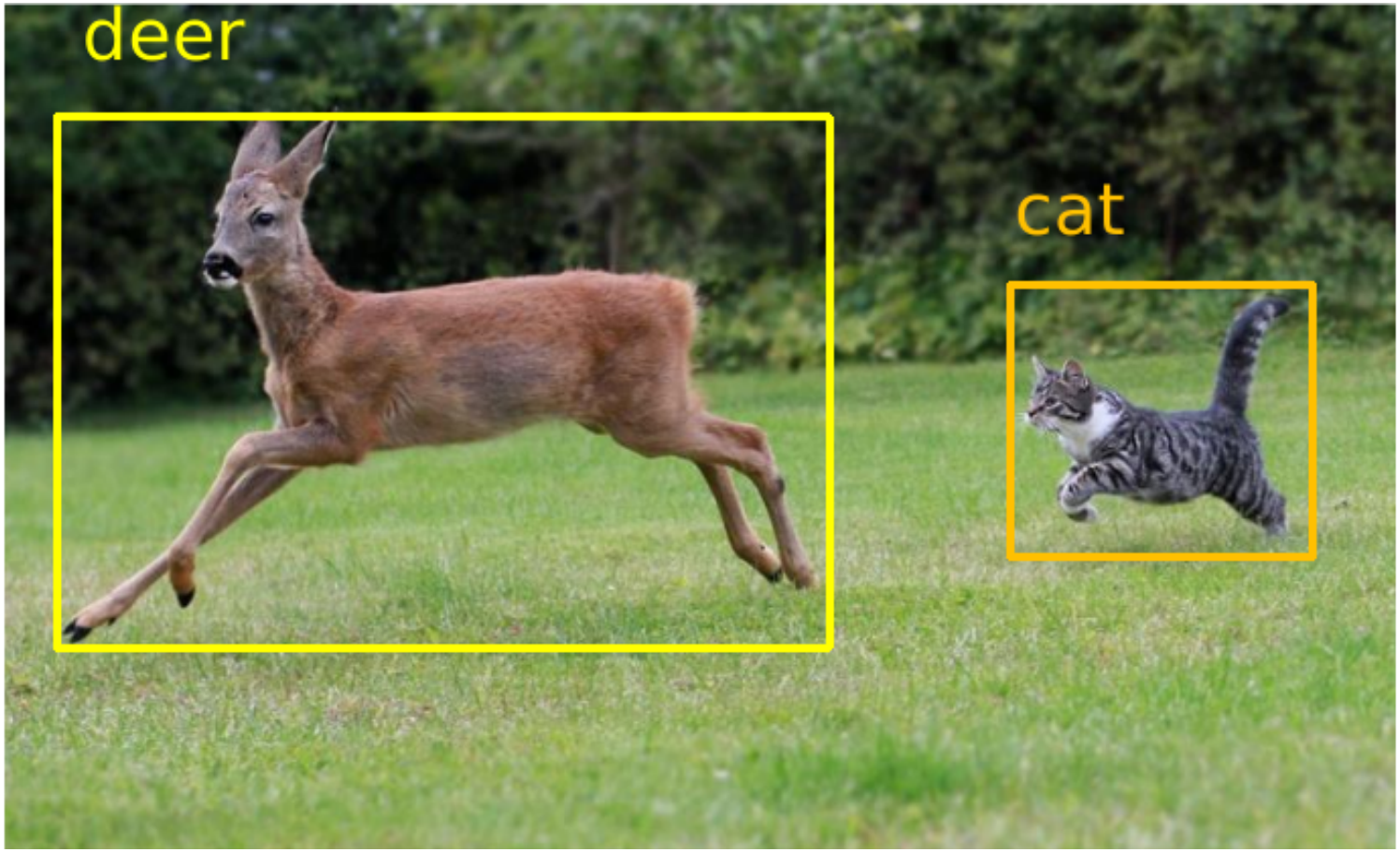


Figure: Object Detection.

Object Detection

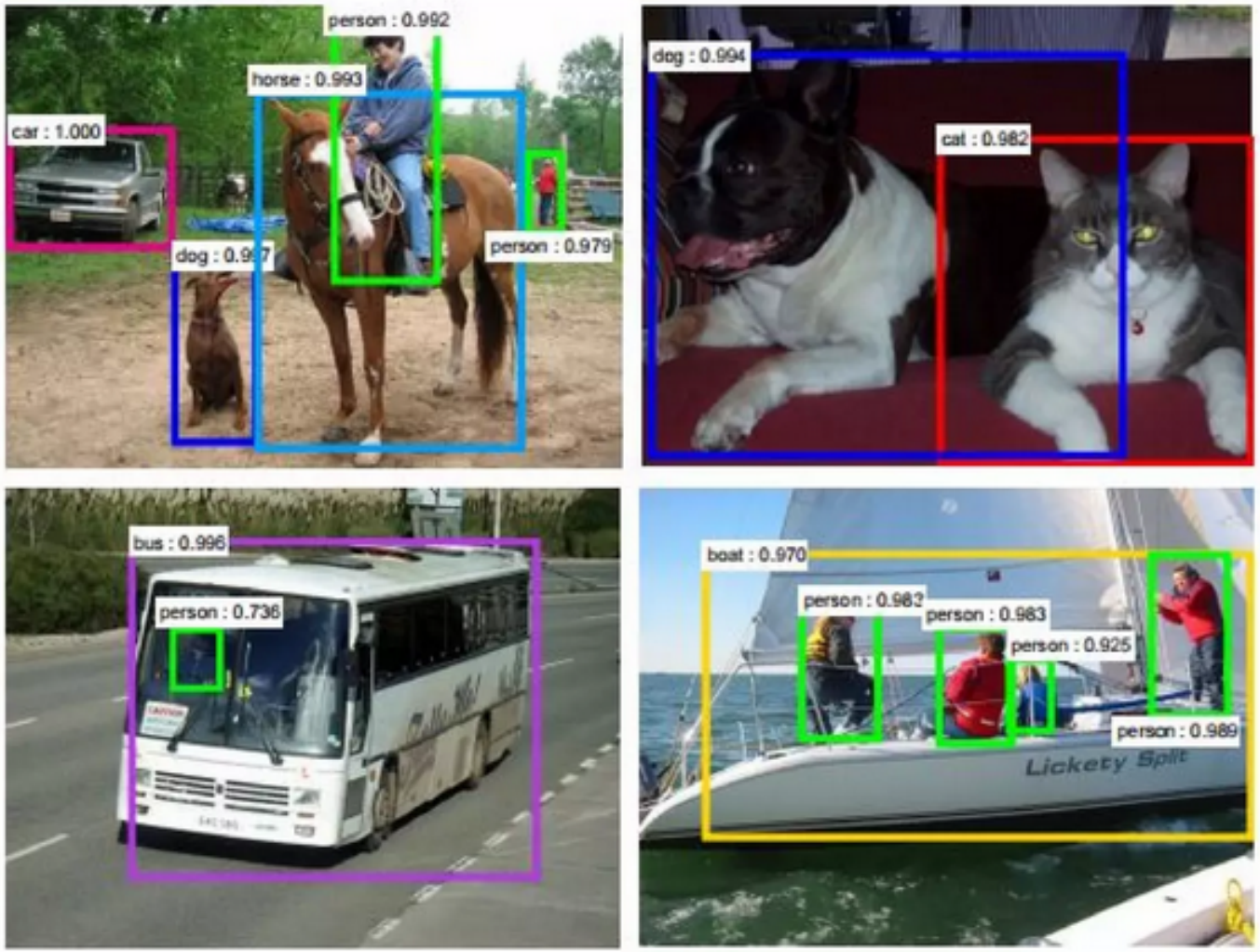


Figure: Object Detection.

CenterNet [1]

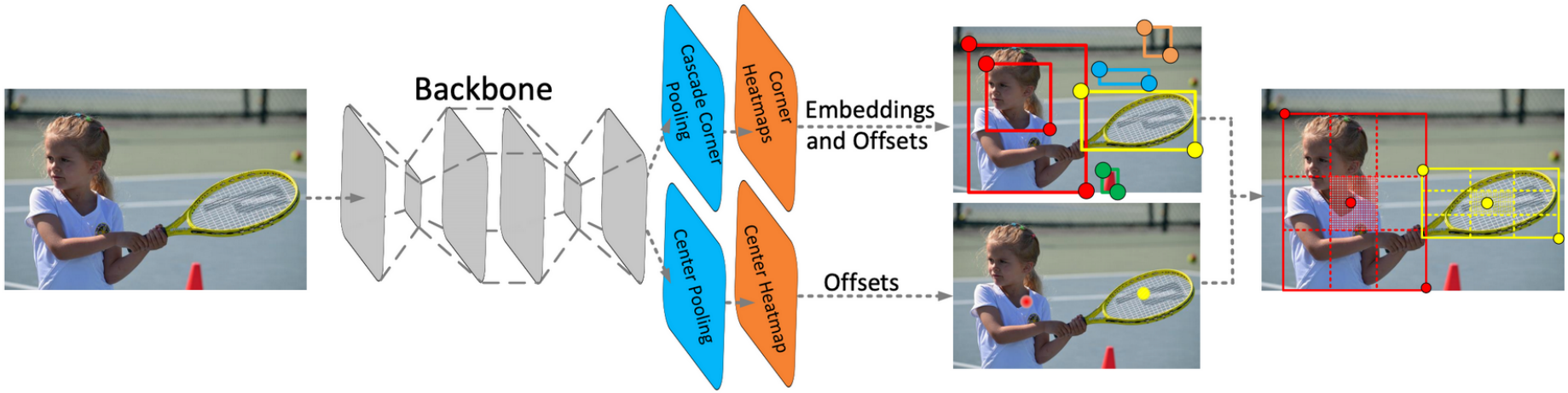


Figure: CenterNet Model

- CenterNet is a one-stage object detector that detects each object as a triplet, rather than a pair, of key points.
- Relies on key point estimation to find the center points and regress all other object properties.
- It utilizes two customized modules named cascade corner pooling and center pooling.

CenterNet

- **Cascade corner pooling**: Enriches information collected by both top-left and bottom-right corners.
- **Center pooling**: Provides more recognizable information at the central regions.
- The network outputs a heatmap of the same size as the input image.
- Where each pixel in the heatmap represents the probability that it corresponds to the center point of an object.
- CenterNet estimates the bounding box of each object by regressing from the center point to the four corners of the box.

CenterNet

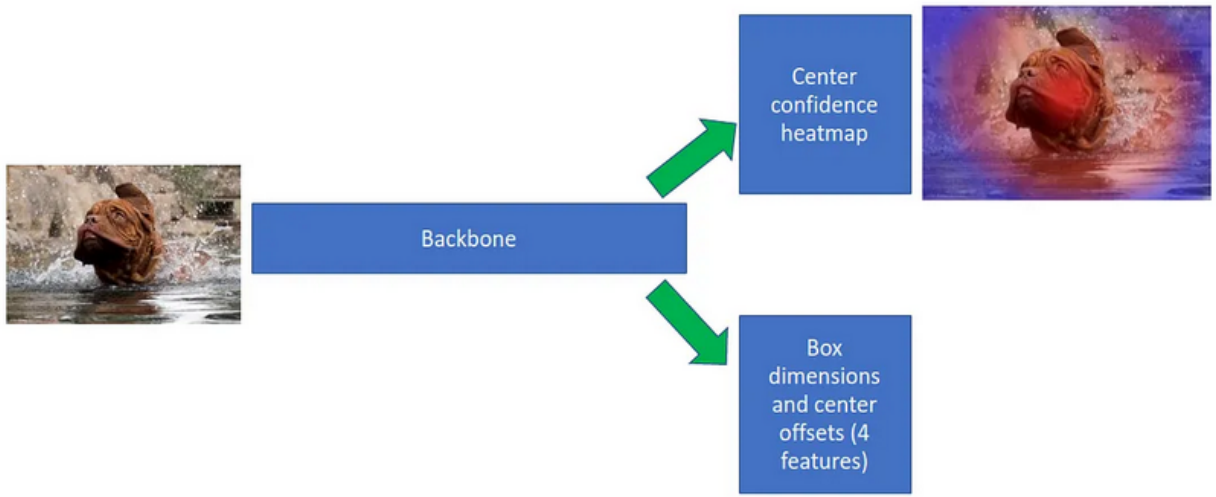


Figure: CenterNet heatmap prediction

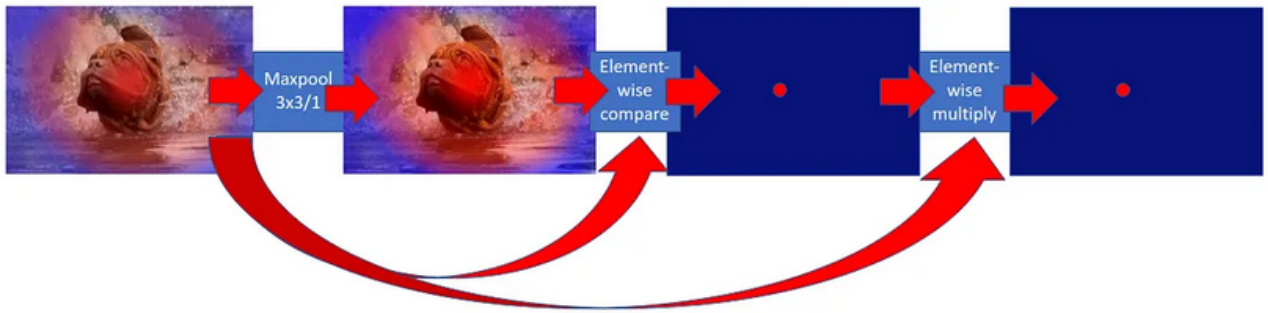


Figure: CenterNet NMS flow

CenterNet

- Anchor box structure:

$$[b_{x1}, b_{y_1}, b_{x2}, b_{y_2}]$$

Where:

b_{x1}, b_{y1}: correspond to the x , y coordinates of the top-left corner.

b_{x2}, b_{y2}: correspond to the x , y coordinates of the bottom-right corner.

- Centernet proposal:

$$[c_x, c_y]$$

Where:

$$c_x = \left(\frac{b_{x1} + b_{x2}}{2}\right)$$

$$c_y = \left(\frac{b_{y1} + b_{y2}}{2}\right)$$

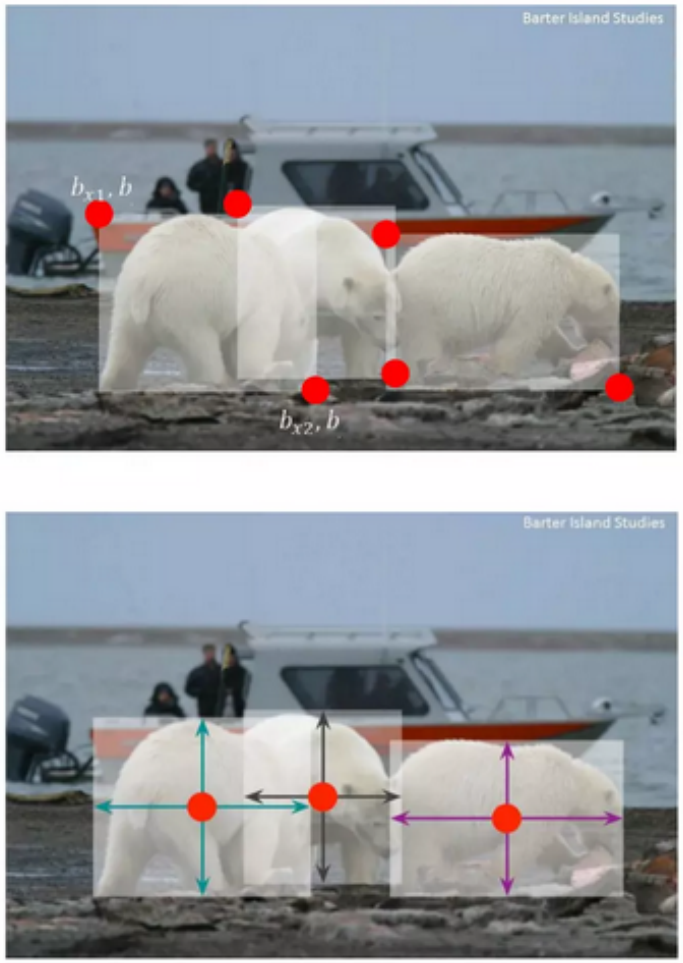


Figure: Object as points

CenterNet

- Let $I \in R^{W \times H \times 3}$ be an input image of width W and height H.
- The objective is to produce:
  - Keypoint heatmap $\hat{Y}$
  - Local offset $\hat{O}$
  - Size prediction $\hat{S}$

CenterNet

- A keypoint heatmap $\hat{Y} \in [0,1]^{\frac{w}{R} \times \frac{H}{R} \times C}$ is generated, where:
  C: number of keypoint types (classes)
  R: output stride
- The output stride downsamples the output prediction by a factor R. ($R = 4$ is default value)

$$\hat{Y}_{x,y,c} \begin{cases} 1, detected\ keypoint \\ 0, background \end{cases}$$

Loss calculation

- Pixel-wise logistic regression using focal loss:

$$L_k = -\frac{1}{N}\sum_{xyc}\begin{cases}(1-\hat{Y}_{xyc})^{\alpha}\log(\hat{Y}_{xyc}) & if\ Y_{xyc}=1\\ (1-Y_{xyc})^{\beta}(\hat{Y}_{xyc})^{\alpha}\log(1-\hat{Y}_{xyc}) & otherwise\end{cases}$$

where:

α, β: hyperparameters of focal loss.$(\alpha = 2, \beta = 4)$

N: number of keypoints

- The offset is trained with L1 loss.

$$L_{off} = \frac{1}{N}\sum_{\rho}\left|\hat{O}_{\rho} - \left(\frac{p}{R} - \tilde{p}\right)\right|$$

where:

N: number of keypoints

* Remember $\tilde{p} = \left\lfloor \frac{p}{R} \right\rfloor$

CenterNet

- The size prediction is trained with L1 loss.

$$L_{size} = \frac{1}{N}\sum_{k=1}^{N}\left|\hat{s}_{P_k} - S_k\right|$$

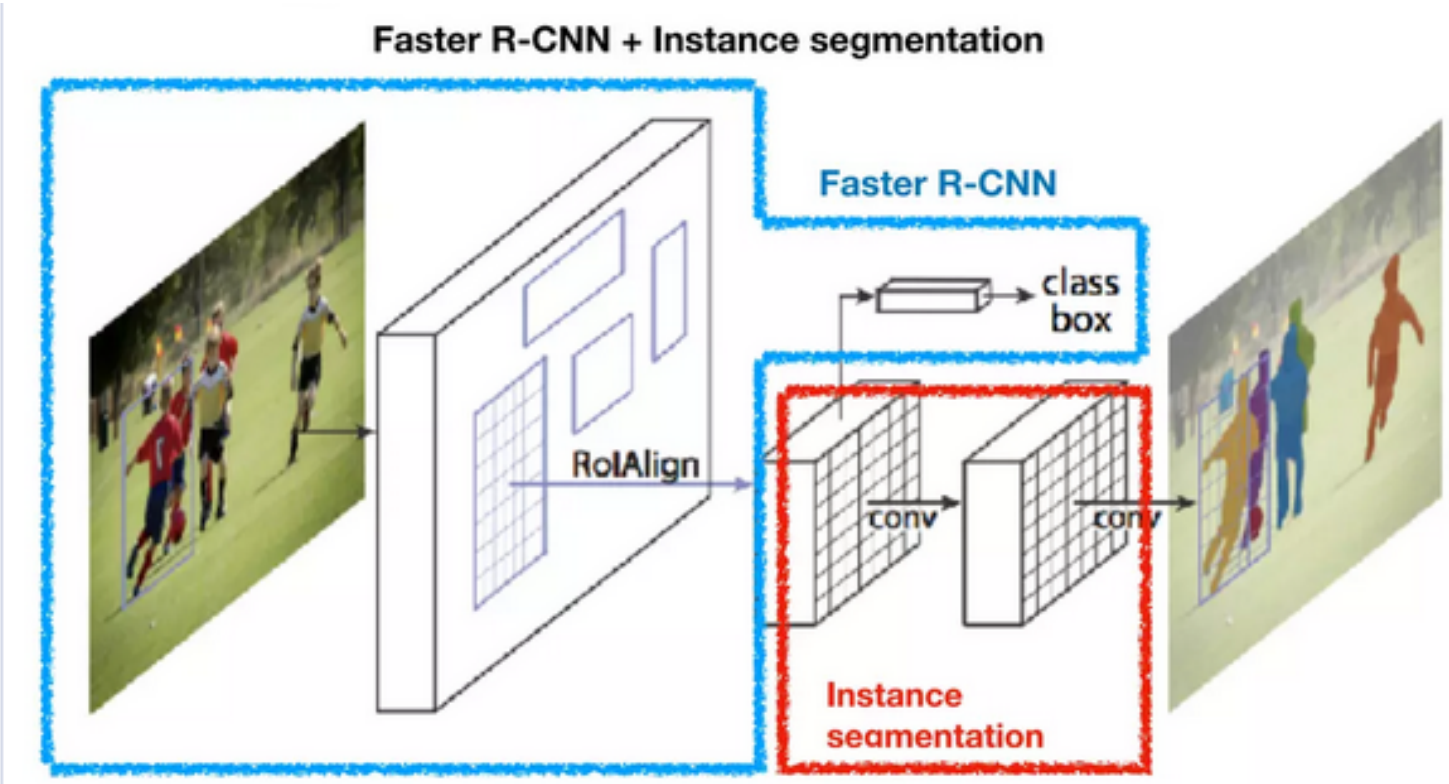
where:

N: number of keypoints

- At calculation, the scale is not normalized and directly used raw pixel coordinates.

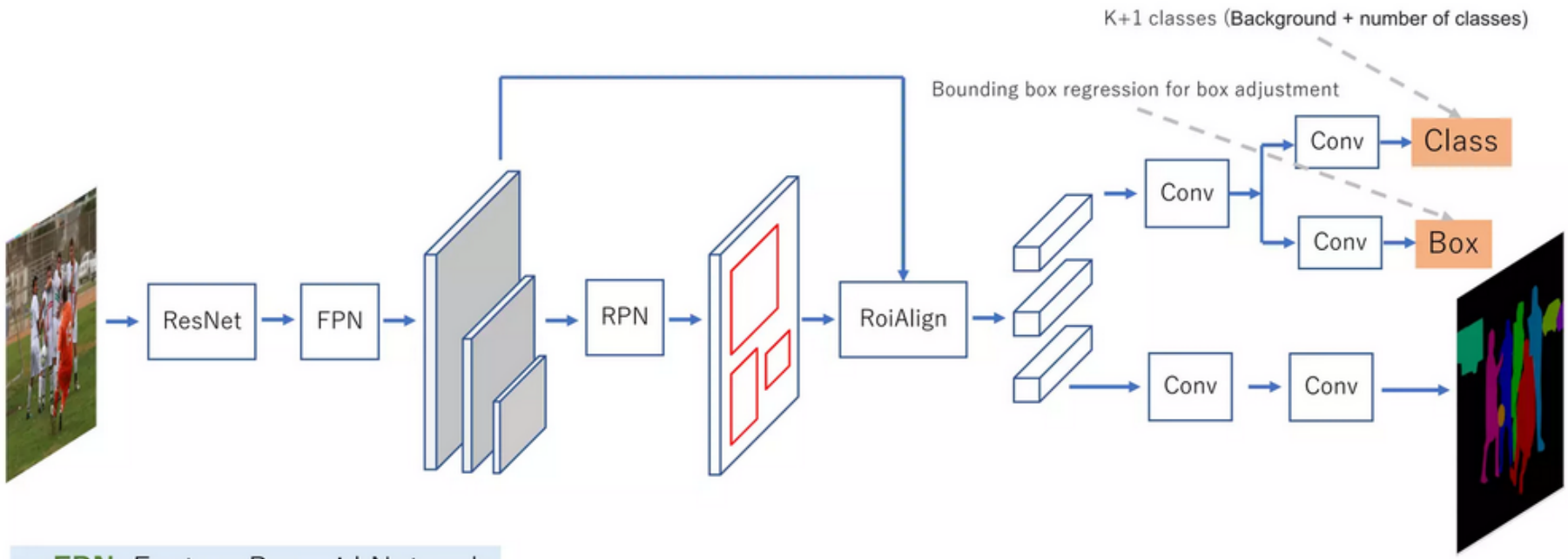
Mask R-CNN [2]

- Mask R-CNN stands for **Mask Region-based Convolutional Neural Network**.
- State-of-the-art algorithm for **Instance Segmentation**.
- Mask R-CNN extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition.



Mask R-CNN

MaskRCNN architecture



- **FPN**: Feature Pyramid Network
- **RPN**: Region Proposal Network
- **RoI**: Region of Interest

Mask R-CNN

Key concepts and components of Mask R-CNN:

- Feature Pyramid Network (FPN):
  - To extract features from different scales of the input image.
  - FPN enhances the network's ability to detect objects at various scales by incorporating features from multiple levels of the feature hierarchy.
- Region Proposal Network (RPN):
  - Generates a set of candidate bounding boxes (regions of interest) that may contain objects.
  - These candidate boxes are generated at multiple scales and aspect ratios to capture objects of various sizes and shapes within the image.
- Region of Interest (RoI) Align: Improves RoIPool from Faster R-CNN with RoIAlign
  - The quantization-free layer, called RoIAlign is introduced to fix the misalignment, that faithfully preserves exact spatial locations.
  - RoIAlign layer removes the harsh quantization of RoIPool, properly aligning the extracted features with the input.
  - RoIAlign improves mask accuracy by relative 10% to 50%.
  - Uses bilinear interpolation.

Mask R-CNN

Key concepts and components of Mask R-CNN: Continues....

- Classification and Regression Head:
  - Similar to Faster R-CNN, to predict the class label and refine the bounding box coordinates for each region of interest.
  - This head consists of fully connected layers followed by softmax activation for classification and linear regression for bounding box refinement.
- Mask Head:
  - The key innovation of Mask R-CNN is the addition of a mask prediction head, which operates in parallel with the classification and regression head.
  - The mask head takes the features extracted from each region of interest and generates a binary mask for each class, indicating the pixel-wise segmentation of objects.

Loss functions

- For each sampled RoI, a multi-task loss is applied:

$$L = L_{cls} + L_{loc} + L_{mask}$$

where

- L_{cls} is classification loss
- L_{loc} is bounding-box regression loss
- L_{mask} is mask loss

- The final loss is calculated as mean of loss over samples

Classification loss L_{cls}

- For a RoI, denotes:
  - u : true class of the RoI
  - $p = (p_0, \ldots, p_K)$: predicted probability distribution over K+1 classes
- The classification loss L_{cls} for a RoI is a log-loss calculated as:

$$L_{\text{cls}}(p, u) = -\log p_u$$

Bounding-box regression loss L_{loc}

- For a RoI, denotes:
  - u: true class of the RoI
  - $v = (v_x, v_y, v_w, v_h)$: true bounding-box regression targets of the RoI
  - $t^u = (t^u_x, t^u_y, t^u_w, t^u_h)$: predicted bounding-box regression for the class u.
- The bounding-box regression loss L_{loc} for the RoI is calculated as:

$$L_{loc}(t^u, v) = \sum_{i \in \{x,y,w,h\}} \text{smooth}_{L_1}(t^u_i - v_i)$$

where

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

31

Mask loss L_{mask}

- For a RoI, denotes:
  - u true class of the RoI
  - Q, P^u the true mask and the predicted mask for the class u of the RoI respectively ($Q_{ij} \in \{0, 1\}, P^u_{ij} \in [0, 1]$)
- The mask loss L_{mask} for the RoI is the **average binary cross-entropy** loss, calculated as:

$$L_{mask} = -\frac{1}{m^2} \sum_{i,j} [Q_{ij} \, log P^u_{ij} + (1 - Q_{ij}) \, log(1 - P^u_{ij})]$$

References I

Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian.

Centernet: Keypoint triplets for object detection.

In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6568–6577, 2019.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick.

Mask r-cnn.

In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.