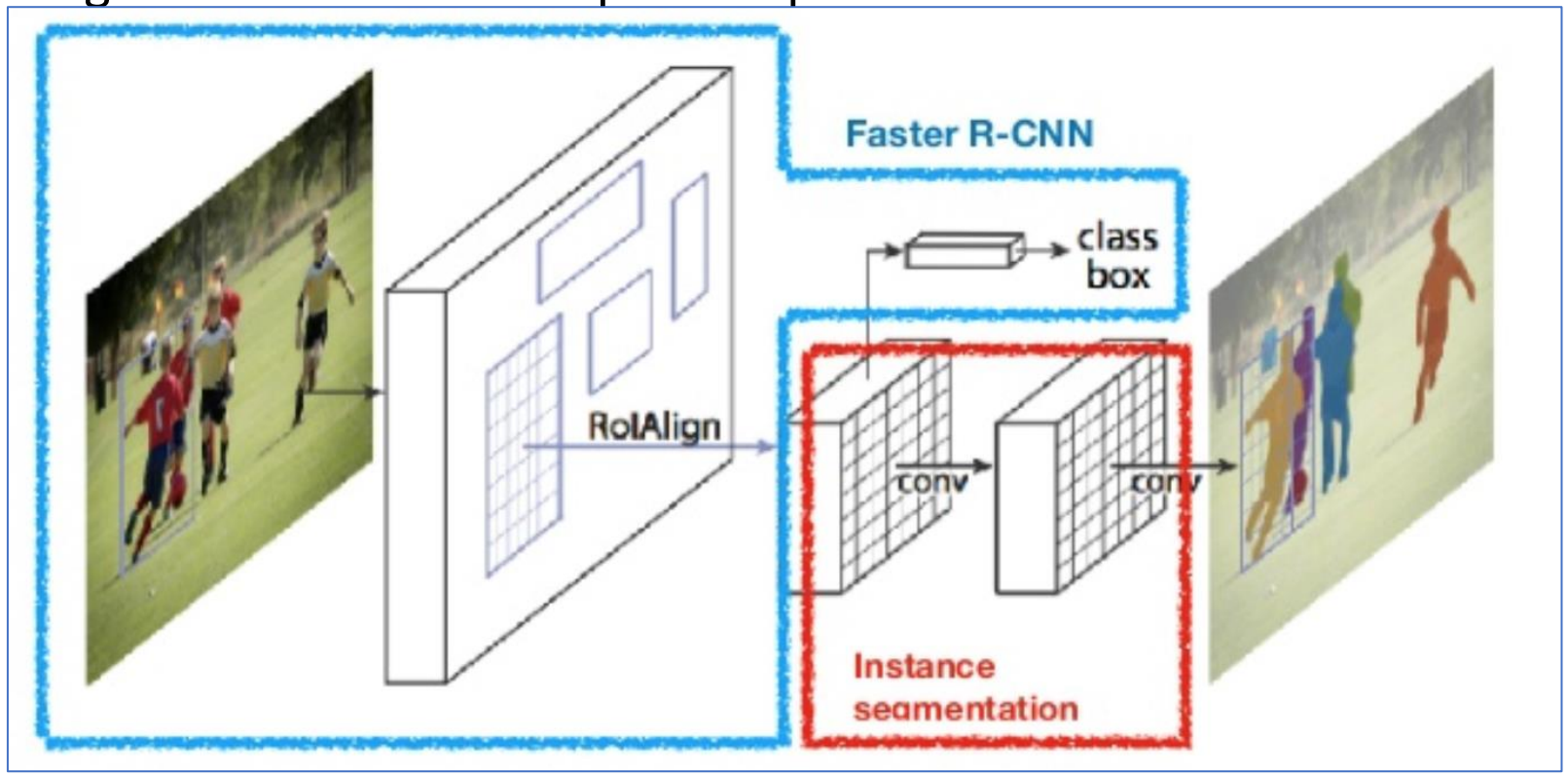
Mask-RCNN

Mask R-CNN

- Mask R-CNN (He et al., 2017) extends Faster R-CNN to pixel-level image segmentation.
- The key point is to decouple the classification and the pixel-level mask prediction tasks.
- Based on the framework of Faster R-CNN, it added a third branch for predicting an object mask in parallel with the existing branches for classification and localization
- The mask branch is a small fully-connected network applied to each RoI, predicting a segmentation mask in a pixel-to-pixel manner.



Mask R-CNN is Faster R-CNN model with image segmentation

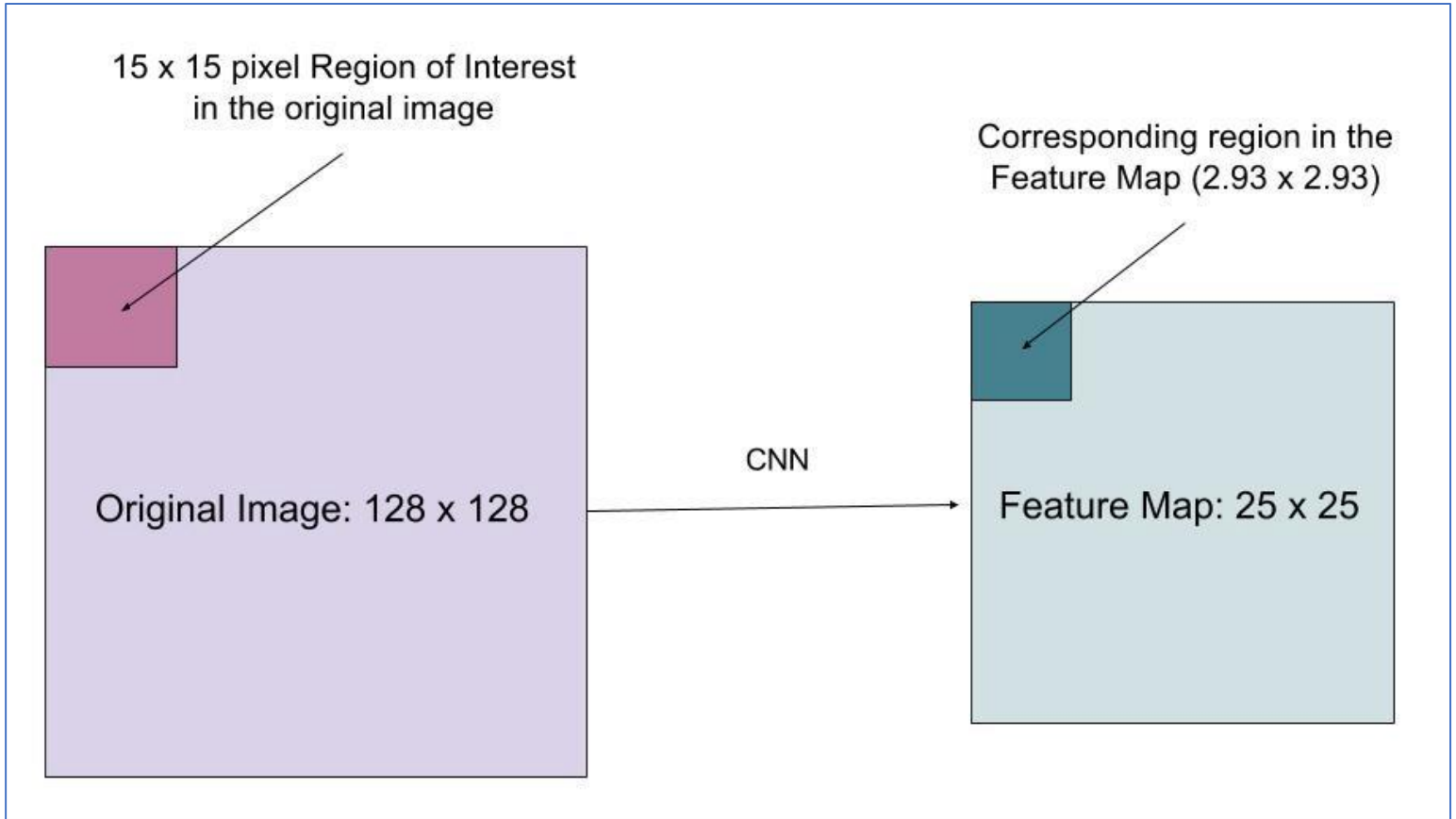
Mask R-CNN

- Because pixel-level segmentation requires much more fine-grained alignment than bounding boxes, mask R-CNN improves the RoI pooling layer (named “RoIAlign layer”) so that RoI can be better and more precisely mapped to the regions of the original image.

Predictions by Mask R-CNN on COCO test set

RoIAlign

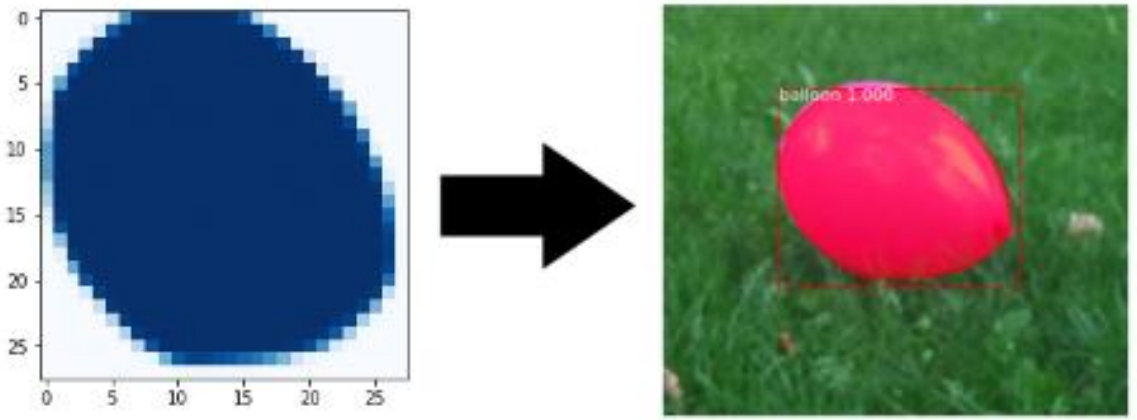
- The RoIAlign layer is designed to fix the location misalignment caused by quantization in the RoI pooling
- RoIAlign ensures that the extracted features can be properly aligned with the input pixels.
- Bilinear interpolation is used for computing the floating-point location values in the input.



*A region of interest is mapped **accurately** from the original image onto the feature map without rounding up to integers*

Mask R-CNN – Segmentation Masks

- The mask network is the addition that the Mask R-CNN paper introduced to the Faster R-CNN framework for object detection
- The mask branch is a convolutional network that takes the positive regions selected by the ROI classifier and generates masks for them
- The generated masks are low resolution: 28x28 pixels. But they are *soft* masks, represented by float numbers, so they hold more details than binary masks. The small mask size helps keep the mask branch light.
- During training, we scale down the ground-truth masks to 28x28 to compute the loss, and during inferencing we scale up the predicted masks to the size of the ROI bounding box and that gives us the final masks, one per object.

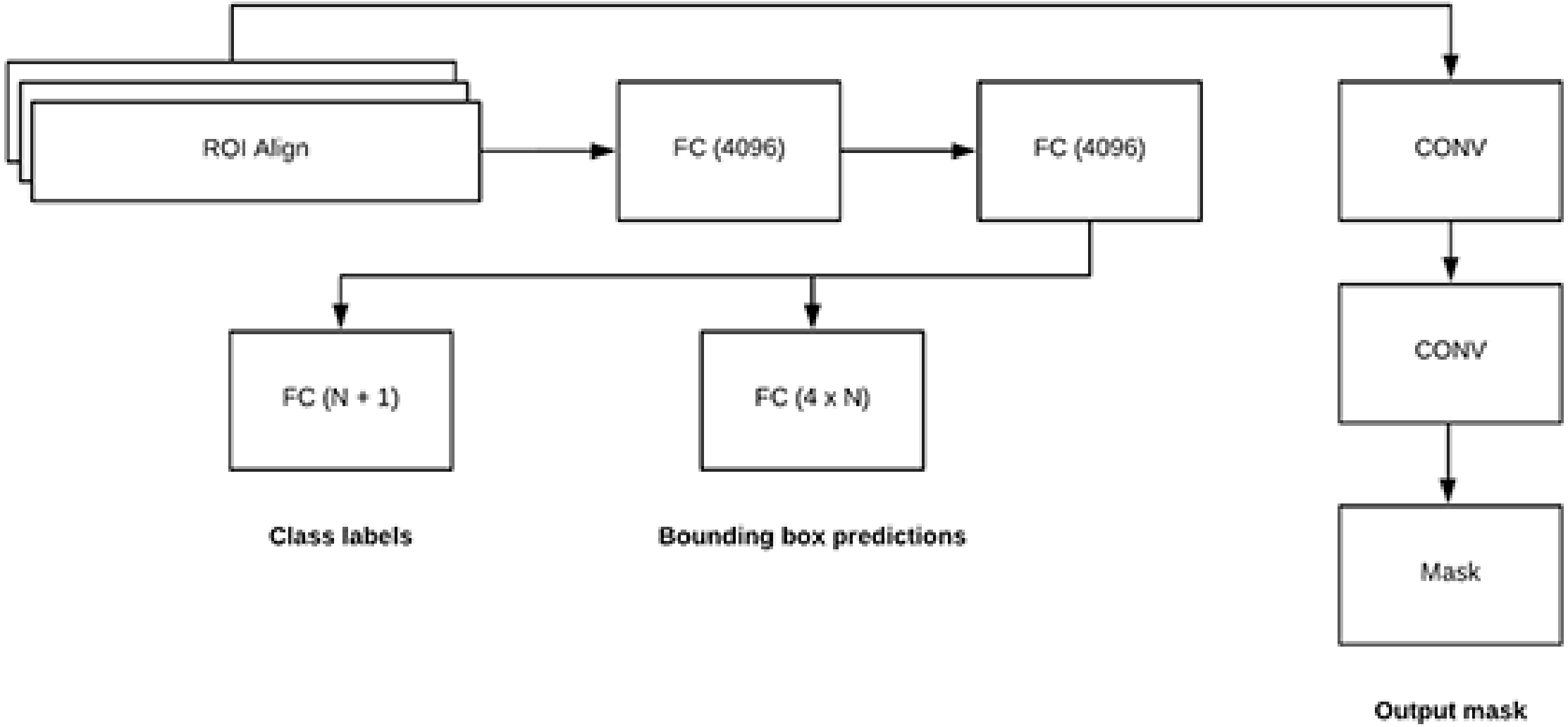


Mask R-CNN

- Fast R-CNN
  - We input an image and associated ground-truth bounding boxes
  - Extract the feature map
  - Apply ROI pooling and obtain the ROI feature vector
  - And finally, use the two sets of fully-connected layers to obtain (1) the class label predictions and (2) the bounding box locations for each proposal.
- Faster R-CNN
  - introduces the **Region Proposal Network (RPN)** that bakes region proposal *directly* into the architecture, alleviating the need for the Selective Search algorithm
- Mask R-CNN
  - Replaces the ROI Pooling module with a more accurate ROI Align module
  - Inserts an additional branch out of the ROI Align module
  - This additional branch accepts the output of the ROI Align and then feeds it into two CONV layers.
  - The output of the CONV layers is the mask itself.

https://www.pyimagesearch.com/2018/11/19/mask-r-cnn-with-opencv/

Mask R-CNN


ROI Align
FC (4096)
FC (4096)
CONV
CONV
FC (N + 1)
FC (4 x N)
Mask
Class labels
Bounding box predictions
Output mask

Mask R-CNN

- The Faster R-CNN/Mask R-CNN architectures leverage a Region Proposal Network (RPN) to generate regions of an image that *potentially* contain an object.
- Each of these regions is ranked based on their "objectness score" (i.e., how likely it is that a given region could potentially contain an object) and then the top *N* most confident objectness regions are kept.
- In the original Faster R-CNN publication Girshick et al. set *N=2,000*, but in practice, we can get away with a much smaller N, such as *N={10, 100, 200, 300}* and still obtain good results.
- He et al.(mask R-CNN) set *N=300* in **their publication**
- Each of the 300 selected ROIs go through three parallel branches of the network:
  - Label prediction
  - Bounding box prediction
  - Mask prediction
- During prediction, each of the 300 ROIs go through **non-maxima suppression** and the top 100 detection boxes are kept, resulting in a 4D tensor of *100 x L x 15 x 15* where *L* is the number of class labels in the dataset and *15 x 15* is the size of each of the *L* masks.