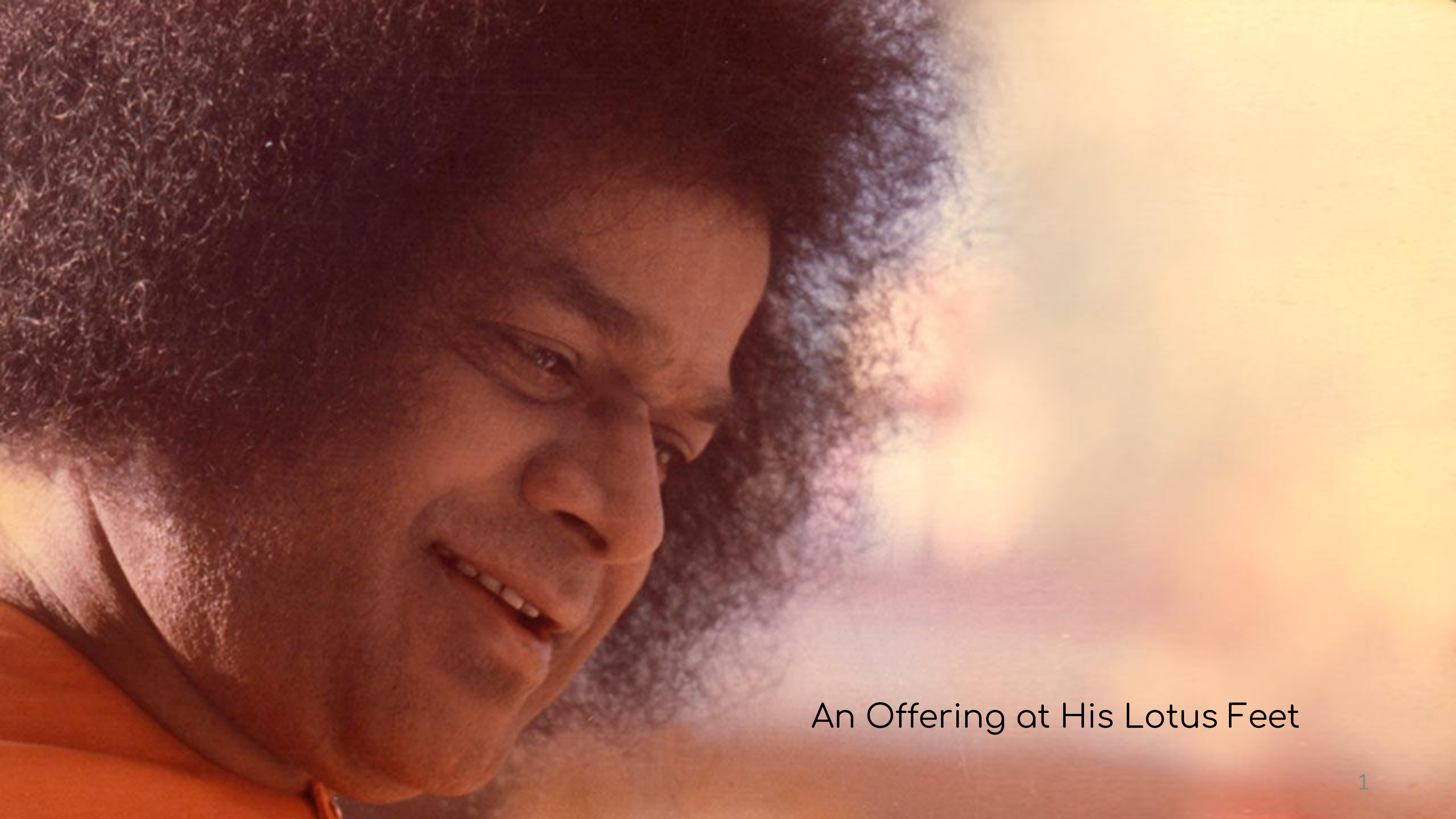
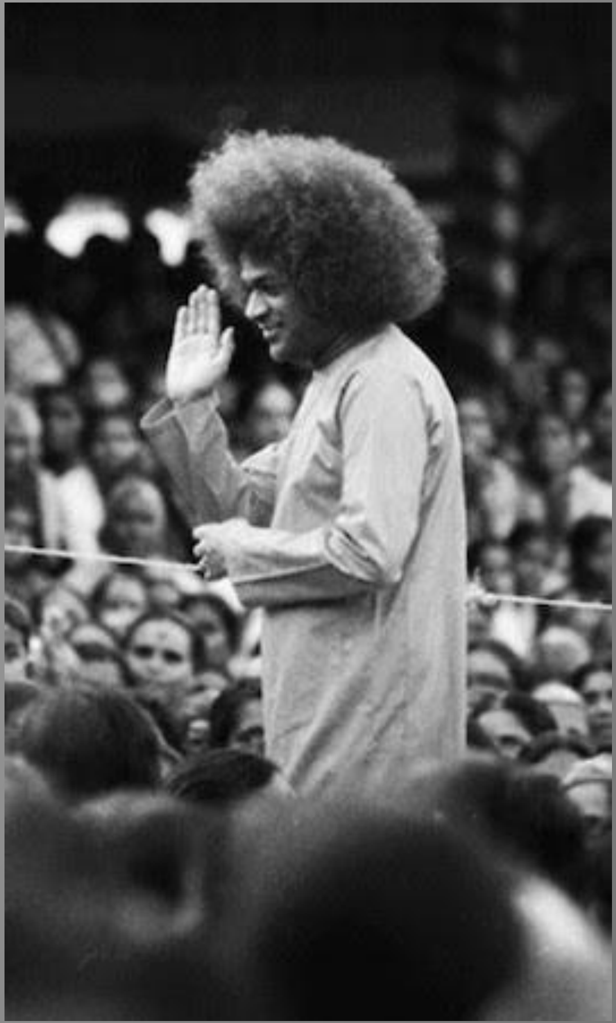

An Offering at His Lotus Feet

SRI SATHYA SAI INSTITUTE OF HIGHER LEARNING
PRASANTHI NILAYAM
सत्यं वद.धर्मं चर

A Presentation for Project
in
Masters in Technology in Computer Science
2022- 24



Hand and Associated Body Localization

Guided By-

Dr. S Balasubramanian

A Presentation by:
Bikash Ranjan Padhy
Regd. No:22554

Problem Statement

Detecting hands and finding the location of the corresponding person for each detected hand

challenges in unconstrained conditions

- multiple people in the scene
- varying overlaps
- occlusions

Introduction

Why Hand-body localization?

- Security and Surveillance: detecting the culprit in a crowded area
- Social AR: tracking of hands and bodies is essential for collaboration, interaction and communication in the same augmented environment
- Healthcare and Telemedicine: track the movements of patients for monitoring their health and remote medical assistance
- Sign language: (human-human communication) interpretation- who said what

Dataset

- ➢ unconstrained high quality images - annotations for hand locations and their corresponding body locations
- ➢ 20,490(including 1,629 test) images with bounding box annotations - more than 57K hand and 63K body instances
- ➢ varying degrees of occlusions and overlap - challenging

- ➢ annotation: an xml file for each image

- ➢ for each object annotation-file has-
  - ○ name(body/ hand)
  - ○ id(1,2,3…)
  - ○ bounding box coordinates(xmin, ymin, xmax, ymax)
  - ○ segmentation coordinates(x1, x2, x3, x4, y1, y2, y3, y4)

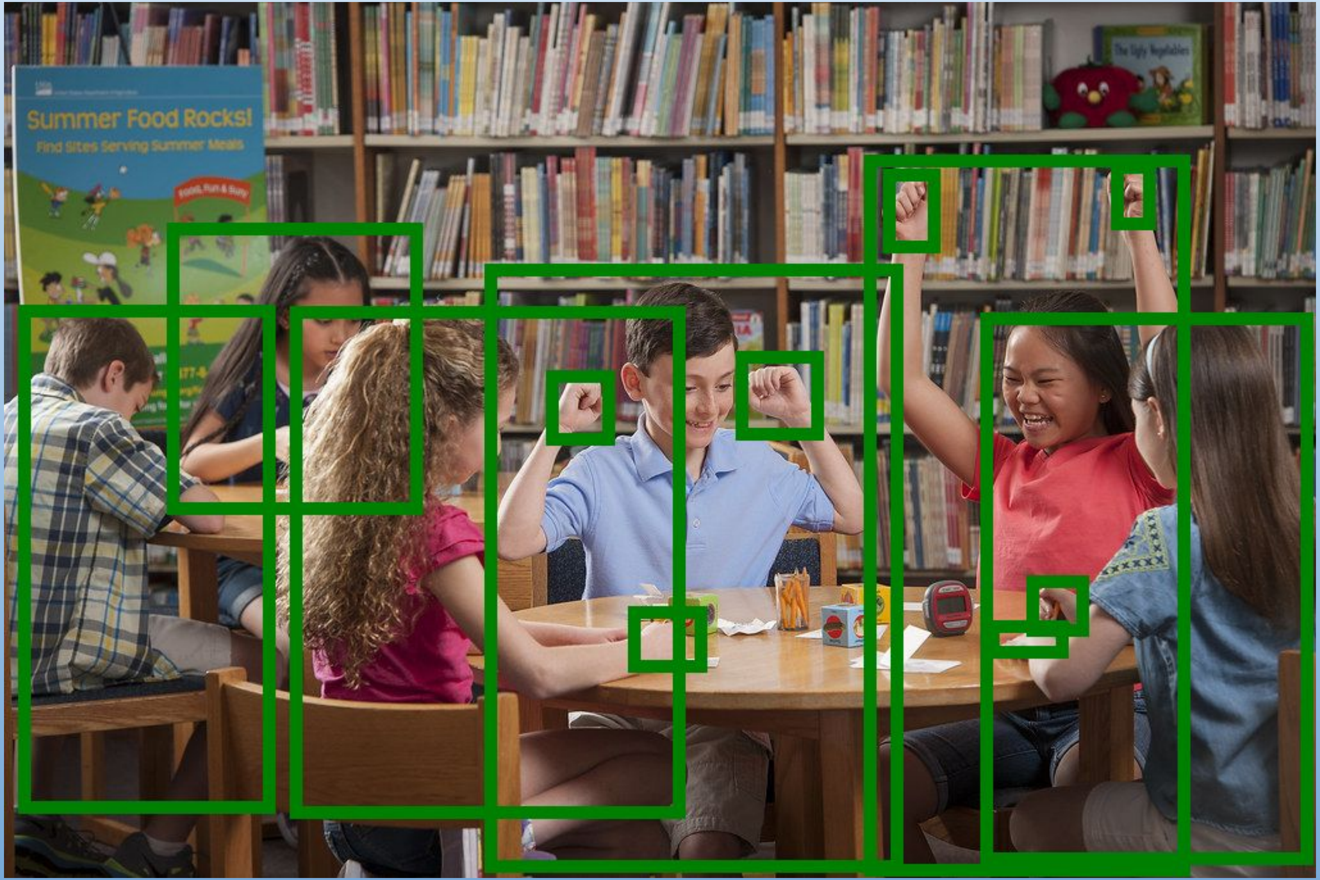
- ➢ association of body and hand - name & body id together

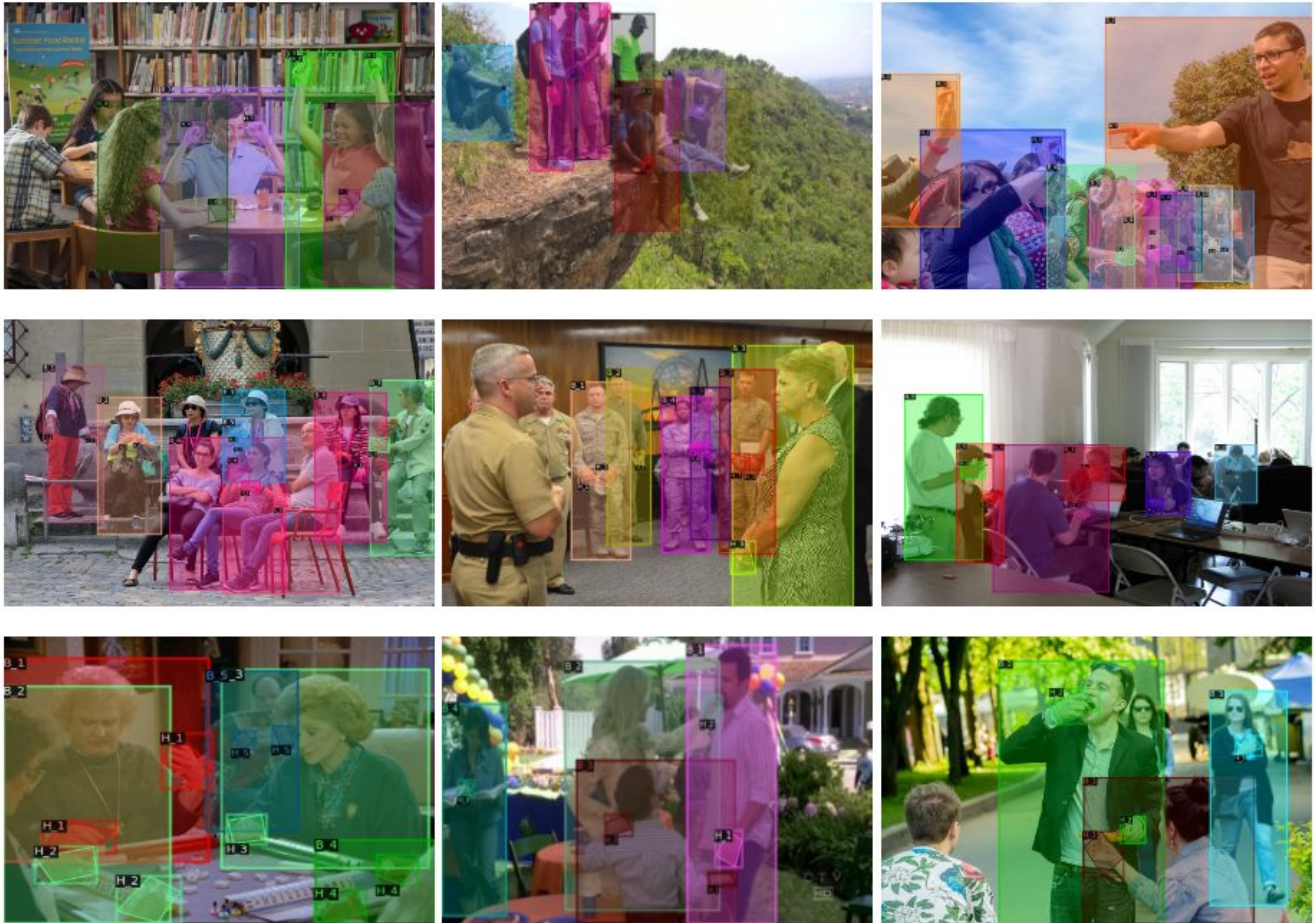
Dataset

+


```
</object>
<object>
        <name>body</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <body_id>2</body_id>
        <bndbox>
                <xmin>671</xmin>
                <ymin>119</ymin>
                <xmax>925</xmax>
                <ymax>681</ymax>
        </bndbox>
</object>
<object>
        <name>hand</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <body_id>2</body_id>
        <bndbox>
                <xmin>686</xmin>
                <ymin>130</ymin>
                <xmax>730</xmax>
                <ymax>195</ymax>
                <x1>722</x1>
                <x2>686</x2>
                <x3>694</x3>
                <x4>730</x4>
                <y1>195</y1>
                <y2>190</y2>
                <y3>130</y3>
                <y4>135</y4>
        </bndbox>
</object>
```

Datapoint Visualisation

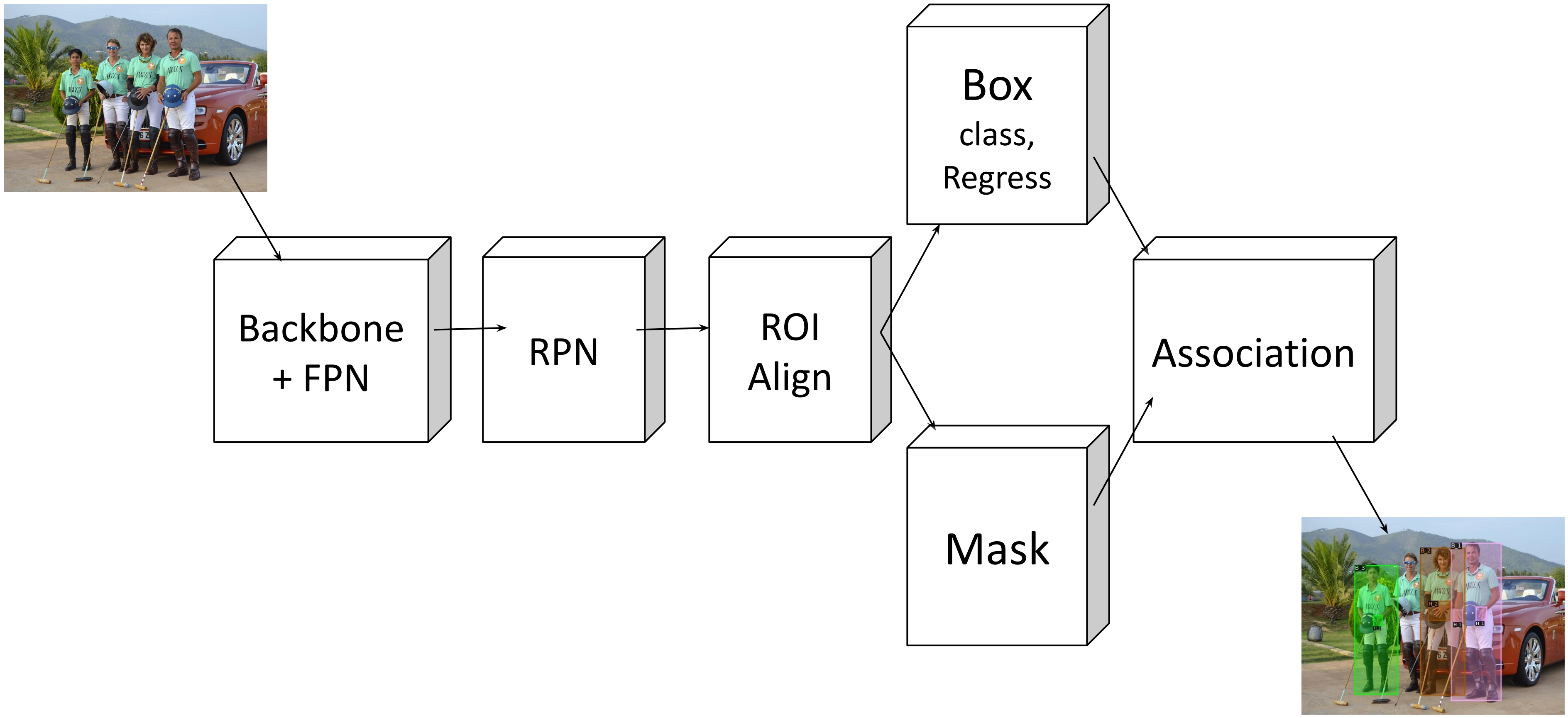




Dataset

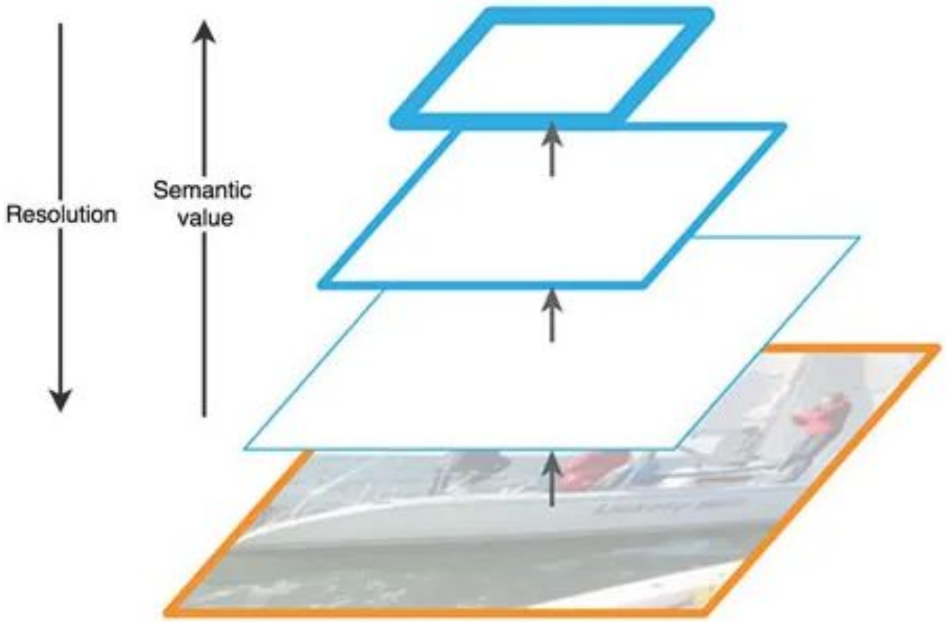
[Link](#) to the dataset

Model Framework

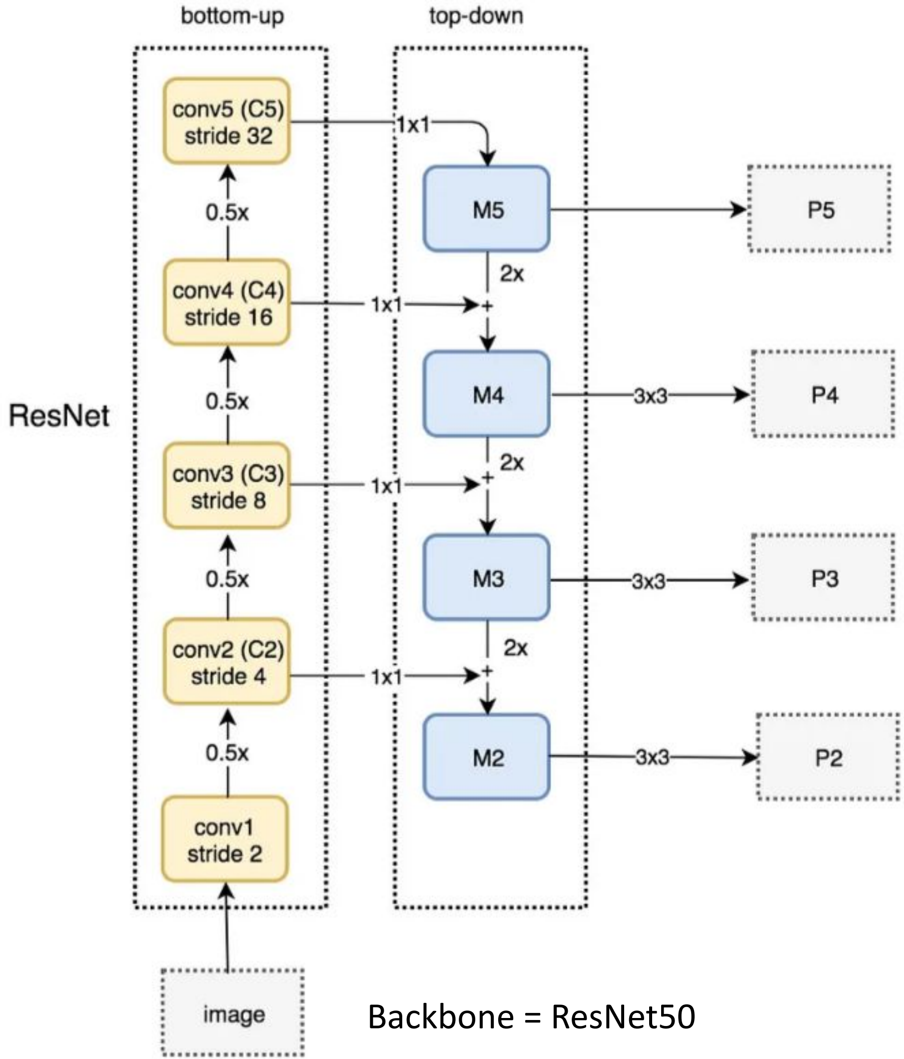


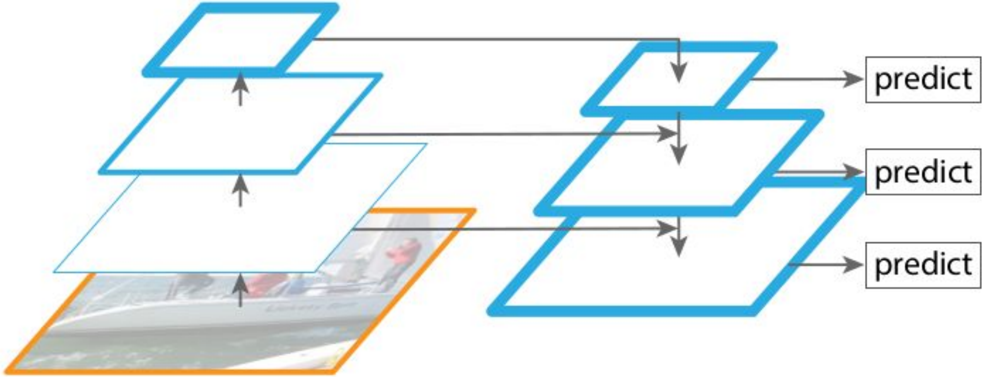
Feature Pyramid Network

- a feature extractor
- multi-scale feature maps
- feature fusion-marginal extra cost
- better semantics + higher resolution ft.maps



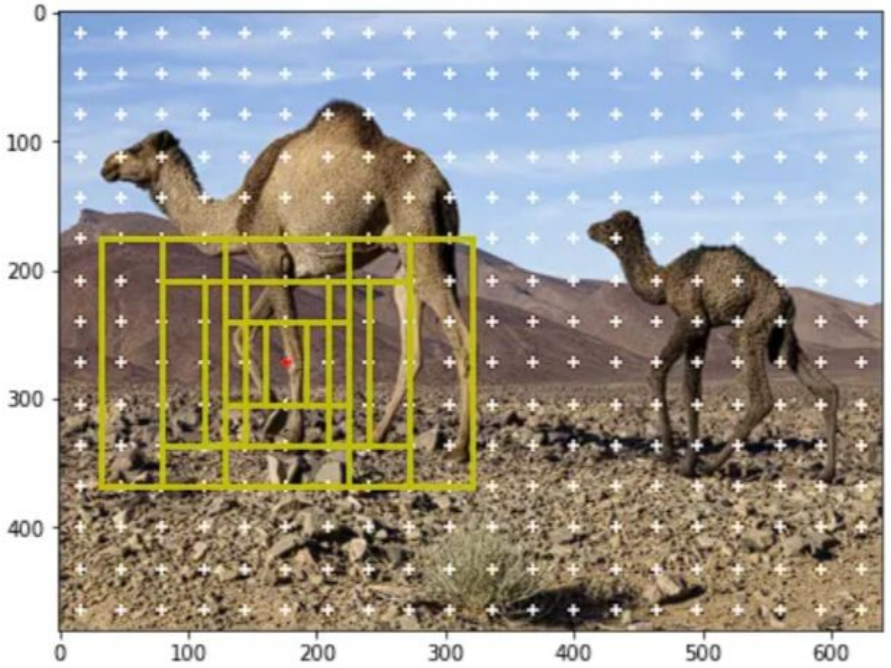
Feature Pyramid Network

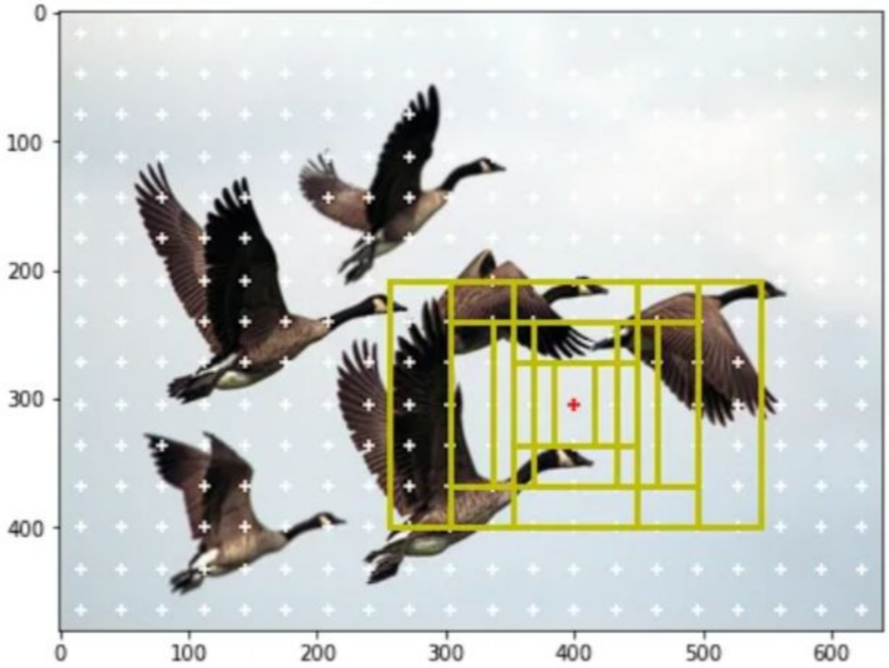




Backbone = ResNet50

Region Proposal Network





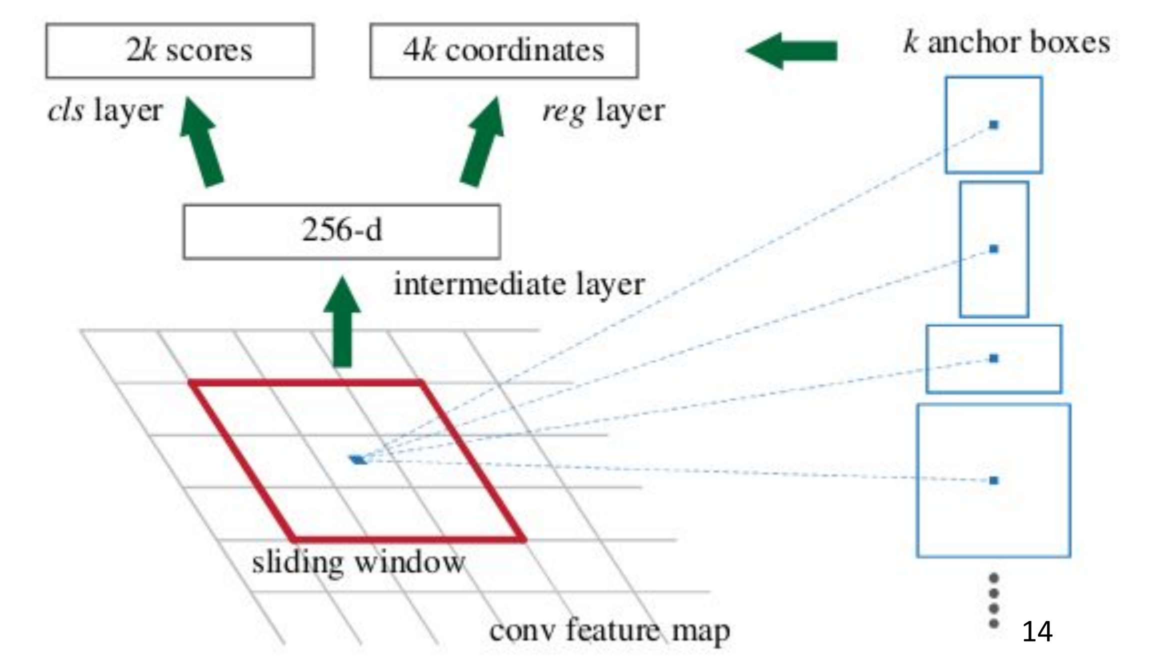
Anchor boxes - various sizes and aspect ratios in the image grid (for visualisation)

Region Proposal Network

- Anchor Boxes: of different scales and aspect ratios - at different spatial locations across the feature map grid

- Predicts two key properties - each anchor box: objectness score and bounding box regression deltas; 2 separate convolutional layers

- Objectness Score: using a convolutional layer - the likelihood of an anchor containing an object



Region Proposal Network

- Bounding Box Regression Deltas: Another convolutional layer predicts - each anchor - changes in the box's location, width, and height(dx1, dy1, dx2, dy2)
- Non-Maximum Suppression (NMS): filter out redundant or overlapping anchors - based on their objectness scores
- Positive & Negative: positive anchors(1) - contain objects; negative anchors(0) - background anchors
- BB Regression Targets: calculated for positive anchors - ground-truth box it is matched with
- Loss Computation: for predictions and targets; binary cross-entropy loss for objectness scores and a smooth L1 loss for bounding box regression

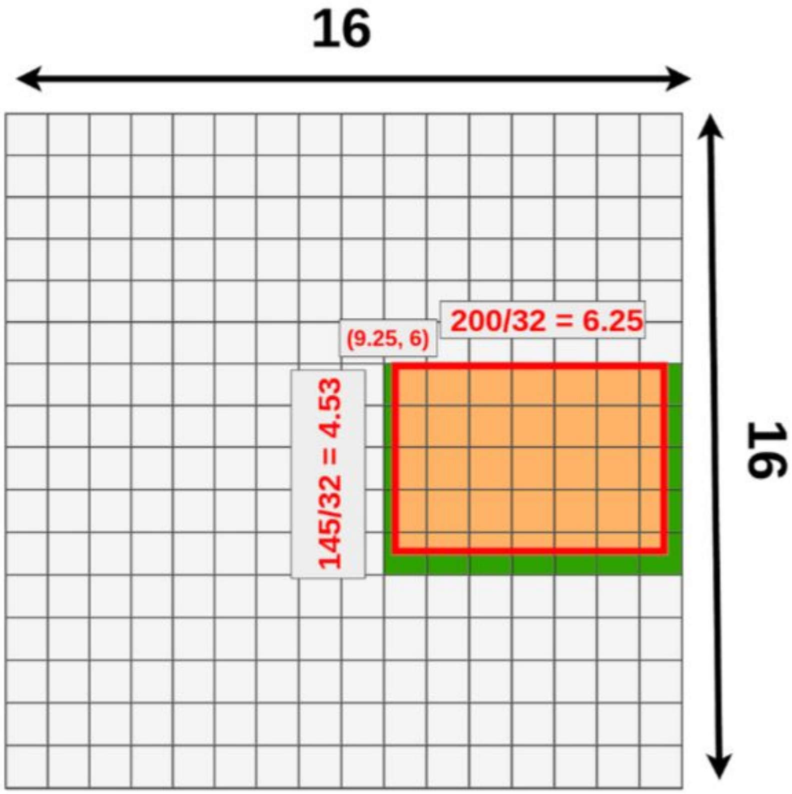
ROI Pooling

1. Input Features and Region Proposals:
   a. 2D feature maps from CNN backbone
   b. Region proposals, generated by the Region Proposal Network (RPN)

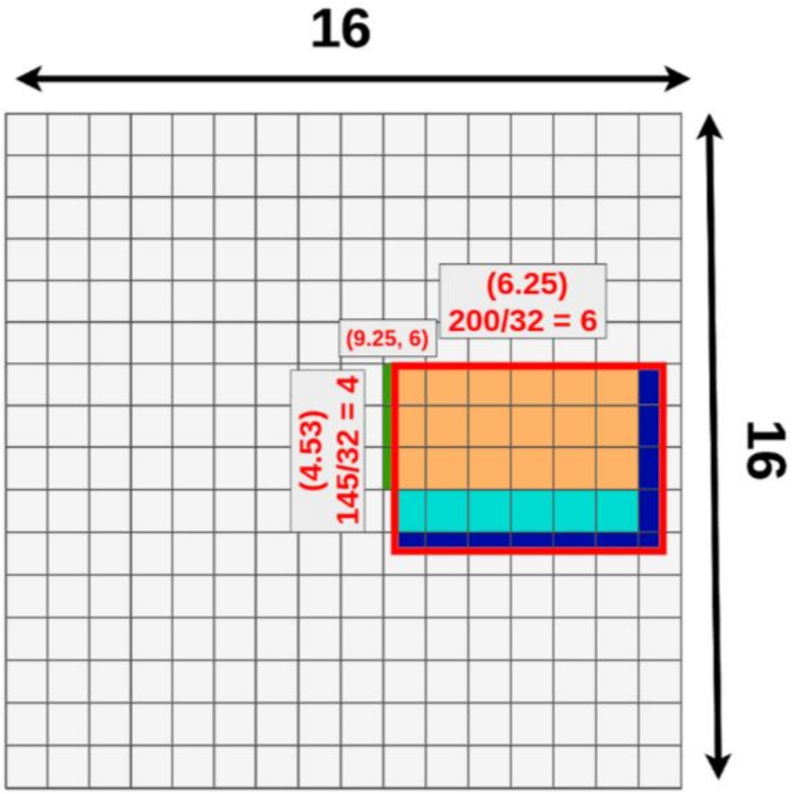
2. Adaptive Pooling:
   a. output size - fixed grid (e.g., 7x7 cells)
   b. extract a single value from the corresponding region in the input feature map

3. Output Feature Map:
   a. fixed-sized feature maps (e.g., 7x7) for each region proposal
   b. flattened and passed through fully connected layers
   c. perform object classification and bounding box regression

image input size - 512x512x3 | feature map size - 16x16x512 | scale factor is 32



Left: ROI Align



Right: ROI Pooling

ROI Align

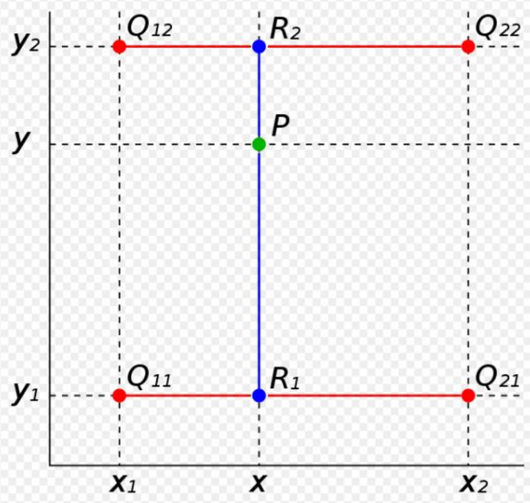
1. Sub-pixel Sampling:
   ❌ Instead of dividing the RoI into a grid and performing pooling,
   ✅ apply bilinear interpolation to the feature map; more accurate alignment with feature map

2. Precise Localization: capable of interpolating features from arbitrary spatial locations within the RoI -> more precise localization

3. Better Handling of Small RoIs: doesn't discard spatial information unlike RoI pooling

Bilinear Interpolation

Suppose that we want to find the value of the unknown function f at the point (x, y). It is assumed that we know the value of f at the four points $Q_{11} = (x_1, y_1)$, $Q_{12} = (x_1, y_2)$, $Q_{21} = (x_2, y_1)$, and $Q_{22} = (x_2, y_2)$.



We first do linear interpolation in the x-direction. This yields

$$f(x, y_1) = \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}),$$
$$f(x, y_2) = \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}).$$

$$\begin{aligned}
f(x, y) &= \frac{y_2 - y}{y_2 - y_1} f(x, y_1) + \frac{y - y_1}{y_2 - y_1} f(x, y_2) \\
&= \frac{y_2 - y}{y_2 - y_1} \left(\frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21}) \right) + \frac{y - y_1}{y_2 - y_1} \left(\frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22}) \right) \\
&= \frac{1}{(x_2 - x_1)(y_2 - y_1)} \big(f(Q_{11})(x_2 - x)(y_2 - y) + f(Q_{21})(x - x_1)(y_2 - y) + f(Q_{12})(x_2 - x)(y - y_1) + f(Q_{22})(x - x_1)(y - y_1) \big) \\
&= \frac{1}{(x_2 - x_1)(y_2 - y_1)} \begin{bmatrix} x_2 - x & x - x_1 \end{bmatrix} \begin{bmatrix} f(Q_{11}) & f(Q_{12}) \\ f(Q_{21}) & f(Q_{22}) \end{bmatrix} \begin{bmatrix} y_2 - y \\ y - y_1 \end{bmatrix}.
\end{aligned}$$

$$P \approx \frac{7.5-6.5}{7.5-6.5}\left(\frac{10.5-9.94}{10.5-9.5}0.1+\frac{9.94-9.5}{10.5-9.5}0.2\right)+\frac{6.5-6.5}{7.5-6.5}\left(\frac{10.5-9.94}{10.5-9.5}1+\frac{9.94-9.5}{10.5-9.5}0.7\right)$$

First dot(Out of 4 dots) value calculation in the first cell of the 3x3 output


6.25
4.53
3x3 RoIAlign
0.1
0.14
0.2
0.21
0.51
0.3
0.4
0.5
0.6
0.8
1
0.7
0.2
0.6
0.1
0.4
0.9
0.8
0.7
0.3
0.5
0.2
0.5
0.2
0.5
1
0.7
0.1
0.1
0.8

Association

1. Body Bboxes and Hand Bboxes overlap in dataset - Bbox covers the entire body- including the hand

2. During Testing, the Body Bbox covers the hand - Overlap

3. For each Hand Bbox, check overlap with all Body Bbox in the image - Bipartite Matching for areas

Association

1. Each body box is a candidate for a hand, and the hand box itself too - if there are no bodies but only hands in the image

2. Hungarian Algorithm

Worker \ Task	Clean bathroom	Sweep floors	Wash windows
Alice	**$8**	$4	$7
Bob	$5	$2	**$3**
Dora	$9	**$4**	$8

Code Snippet

```python
def get_model_segmentation(num_classes):
    # load an instance segmentation model pre-trained on COCO
    model = torchvision.models.detection.maskrcnn_resnet50_fpn(weights="DEFAULT")

    # get number of input features for the classifier
    in_features = model.roi_heads.box_predictor.cls_score.in_features
    # replace the pre-trained head with a new one
    model.roi_heads.box_predictor = FastRCNNPredictor(in_features, num_classes)

    # now get the number of input features for the mask classifier
    in_features_mask = model.roi_heads.mask_predictor.conv5_mask.in_channels

    hidden_layer = 256
    # and replace the mask predictor with a new one
    model.roi_heads.mask_predictor = MaskRCNNPredictor(in_features_mask, hidden_layer, num_classes)

    return model
```

Code Snippet

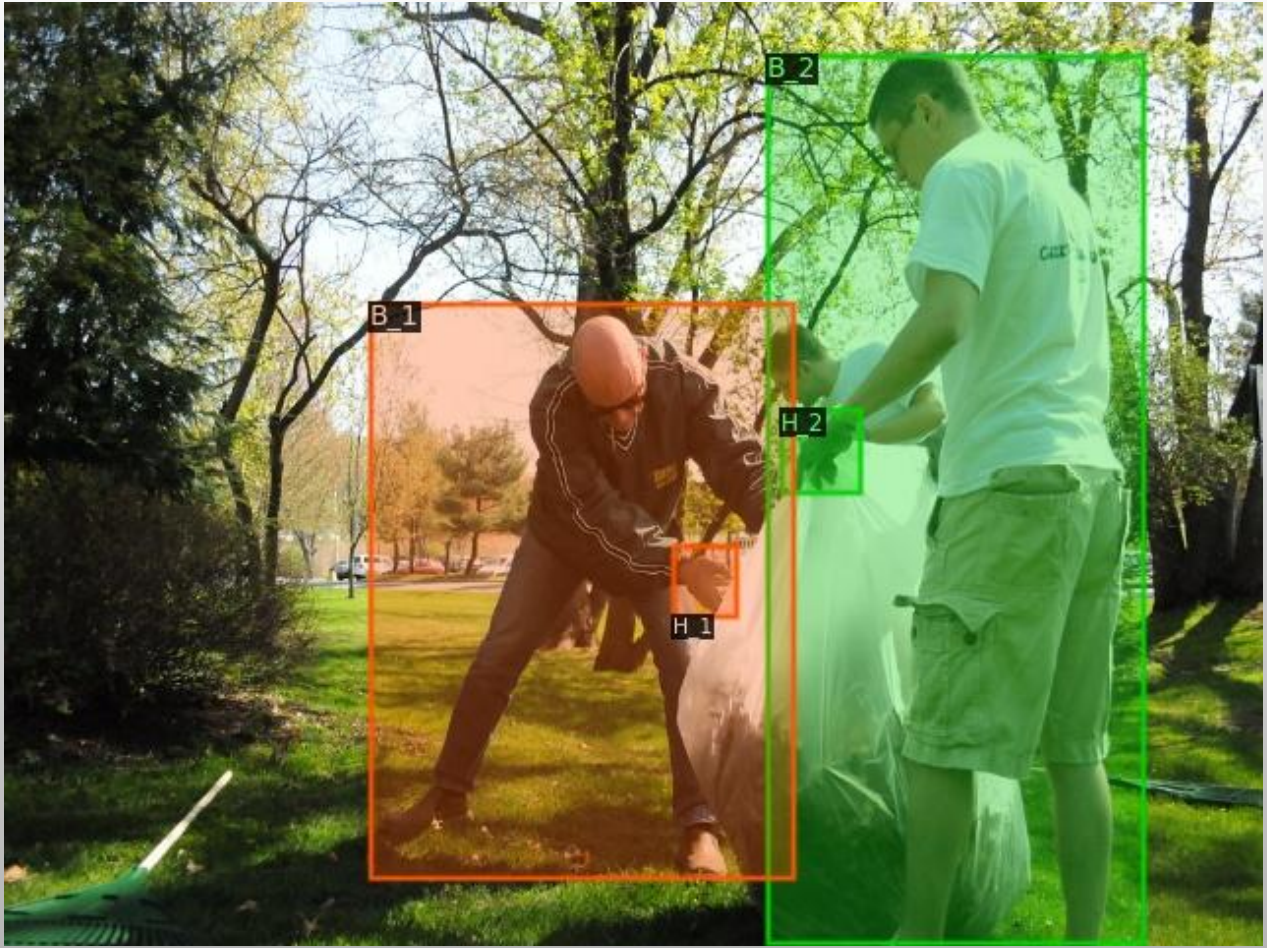
```
pred_overlap = F.sigmoid(pred_overlap)
overlap_mask = (pred_overlap > 0.1).float() ## overlap_mask= 0 if pred_overlap<= 0.1

scores = pred_overlap * scores_positional_density * overlap_mask

scores = torch.cat([scores, scores], dim=1) ## make it a "2-d square matrix" to use hungrian algo on it
scores_numpy = scores.detach().to("cpu").numpy() ## transfer to cpu, as numpy-array
row_ind, col_ind = linear_sum_assignment(-scores_numpy) ## minus to get the max score and not the min score

col_ind = (col_ind % (num_bodies+1)) + 1 ## index back to 1 from 0
row_ind, col_ind = torch.from_numpy(row_ind).to(device),\
torch.from_numpy(col_ind).to(device) ## transfer back to device(cuda)

pred_body_ids_for_bodies = torch.arange(1, num_bodies+1).to(device)
pred_body_ids_for_hands = torch.FloatTensor([num_bodies+1] * num_hands).to(device) ## a row tensor
pred_body_ids_for_hands[row_ind] = col_ind.float() ## match the hand indices with the respective body indices
pred_body_ids[hand_indices] = pred_body_ids_for_hands
pred_body_ids[body_indices] = pred_body_ids_for_bodies.float()

pred_instances[0]["pred_body_ids"] = pred_body_ids

return pred_instances
```

Results

Results

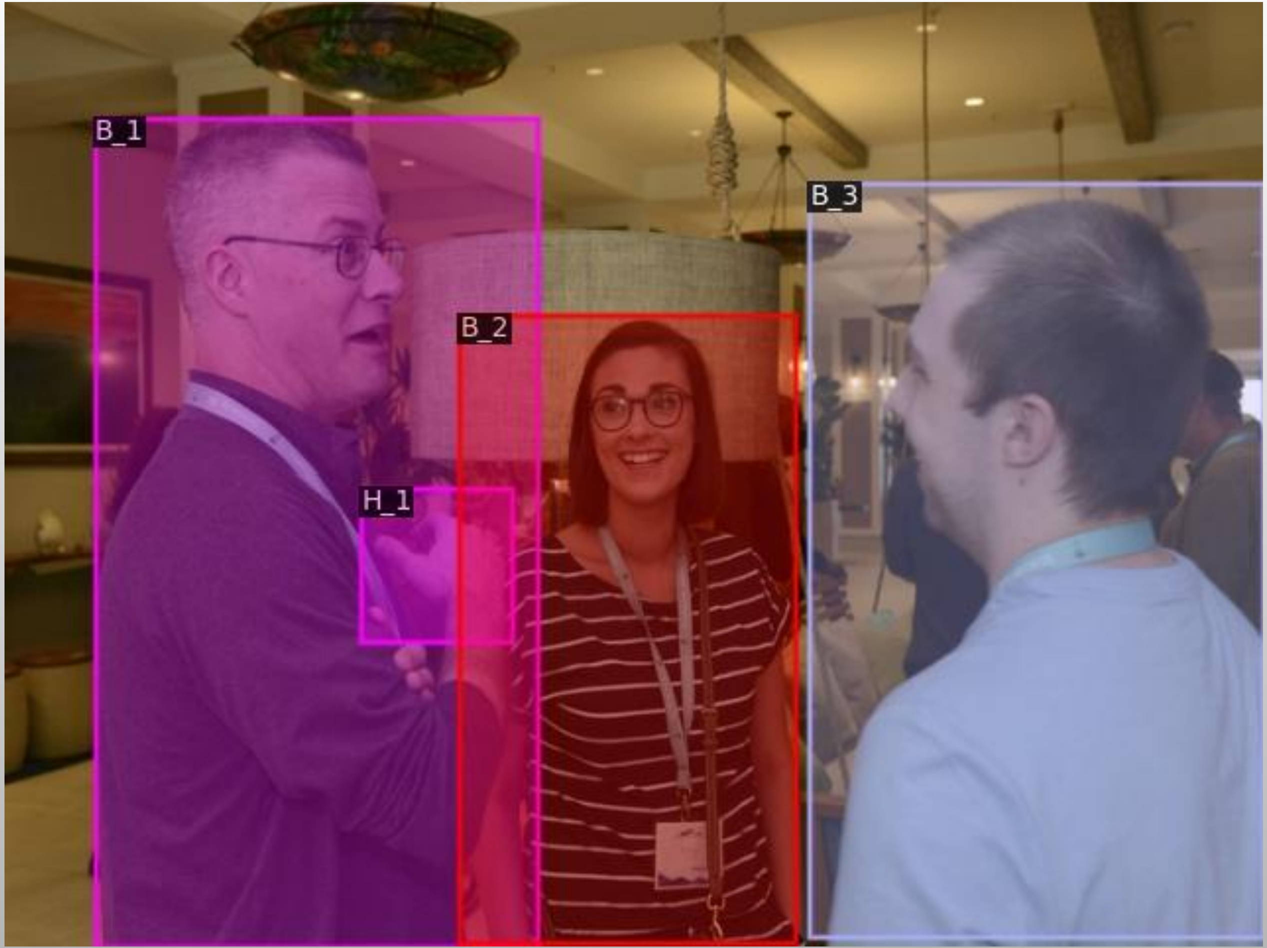
Results

Results



Results

Results



Future Work

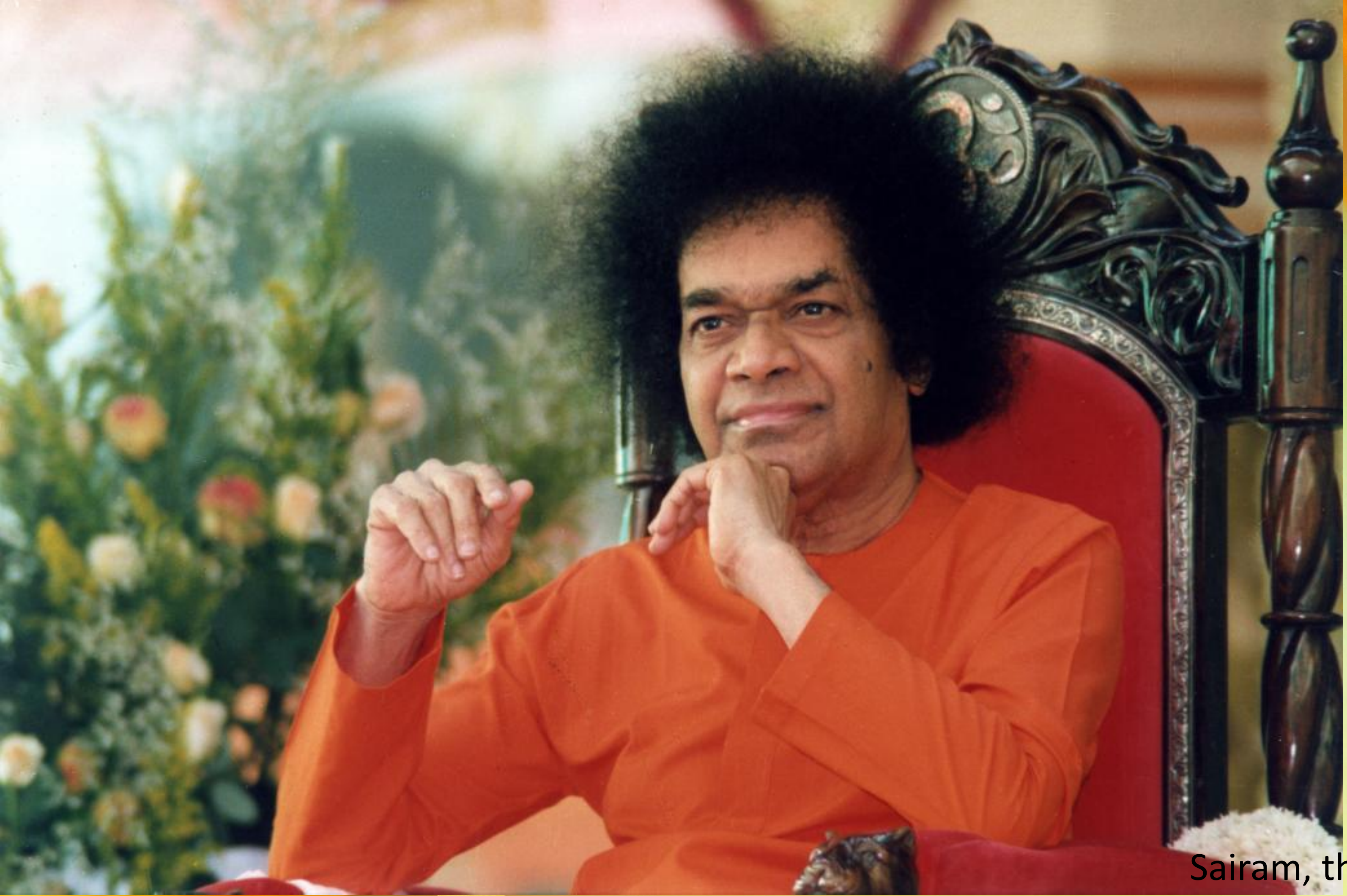
- Making the association part learnable
- Try other models- YOLO

Acknowledgements

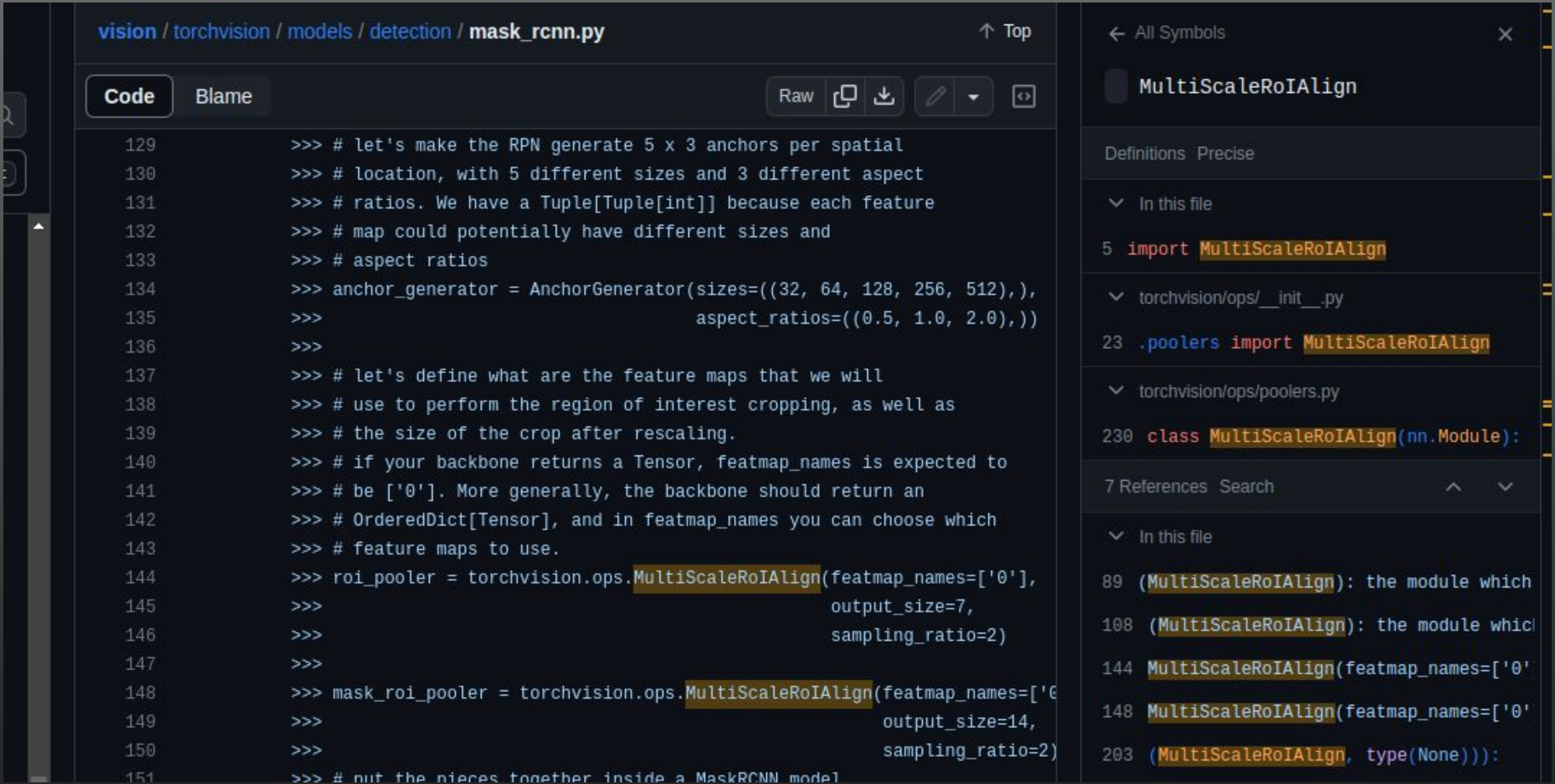
Swami
Dr. S Balasubramanian
Parents
DMACS
Internet

References

- Narasimhaswamy, T. Nguyen, M. Huang, M. Hoai. Whose Hands Are These Hand Detection and Hand-Body Association. In CVPR 2022.
- Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In Proceedings of the International Conference on Computer Vision, 2017.
- R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014.
- S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal net- works. In NIPS, 2015.
- T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In CVPR, 2017.
- R. Girshick. Fast R-CNN. In ICCV, 2015.
- Facebook Research - Detectron2



Sairam, thank You



Pytorch’s mask_rcnn model uses MultiScaleRoIAlign for box and mask

RegionProposal Network

vision / torchvision / models / detection / faster_rcnn.py

Top

Code Blame

Raw

```
43   class FasterRCNN(GeneralizedRCNN):
164      def __init__(
```

```
            raise ValueError("num_classes should be None when box_predictor is specified")
    else:
        if box_predictor is None:
            raise ValueError("num_classes should not be None when box_predictor is not spec

    out_channels = backbone.out_channels

    if rpn_anchor_generator is None:
        rpn_anchor_generator = _default_anchorgen()
    if rpn_head is None:
        rpn_head = RPNHead(out_channels, rpn_anchor_generator.num_anchors_per_location()[0]

    rpn_pre_nms_top_n = dict(training=rpn_pre_nms_top_n_train, testing=rpn_pre_nms_top_n_te
    rpn_post_nms_top_n = dict(training=rpn_post_nms_top_n_train, testing=rpn_post_nms_top_r

    rpn = RegionProposalNetwork(
        rpn_anchor_generator,
        rpn_head,
        rpn_fg_iou_thresh,
        rpn_bg_iou_thresh,
        rpn_batch_size_per_image,
        rpn_positive_fraction,
        rpn_pre_nms_top_n,
        rpn_post_nms_top_n,
        rpn_nms_thresh,
        score_thresh=rpn_score_thresh,
    )

    if box_roi_pool is None:
        box_roi_pool = MultiScaleRoIAlign(featmap_names=["0", "1", "2", "3"], output_size=
```

All Symbols

RegionProposalNetwork

Definitions Precise

In this file

`20 RegionProposalNetwork, RPNHead`

torchvision/models/detection/rpn.py

`113 class RegionProposalNetwork(torch`

5 References Search

In this file

`234 rpn = RegionProposalNetwork(`

...t_models_detection_negative_samples.py

test/test_onnx.py

Search for this symbol