中国科学技术大学
University of Science and Technology of China

目标检测SOTA

Co-DETR

原理+代码

目录

为什么能成为SOTA?

传统目标检测算法和新兴端到端目标检测算法的集大成者

强有力的骨干网络 + 多头辅助训练

一对一：一个GT对应一个正样本

只选最匹配正样本的计算损失

一对多：一个GT对应多个正样本

选择多个较为匹配的正样本计算损失

Co-DETR

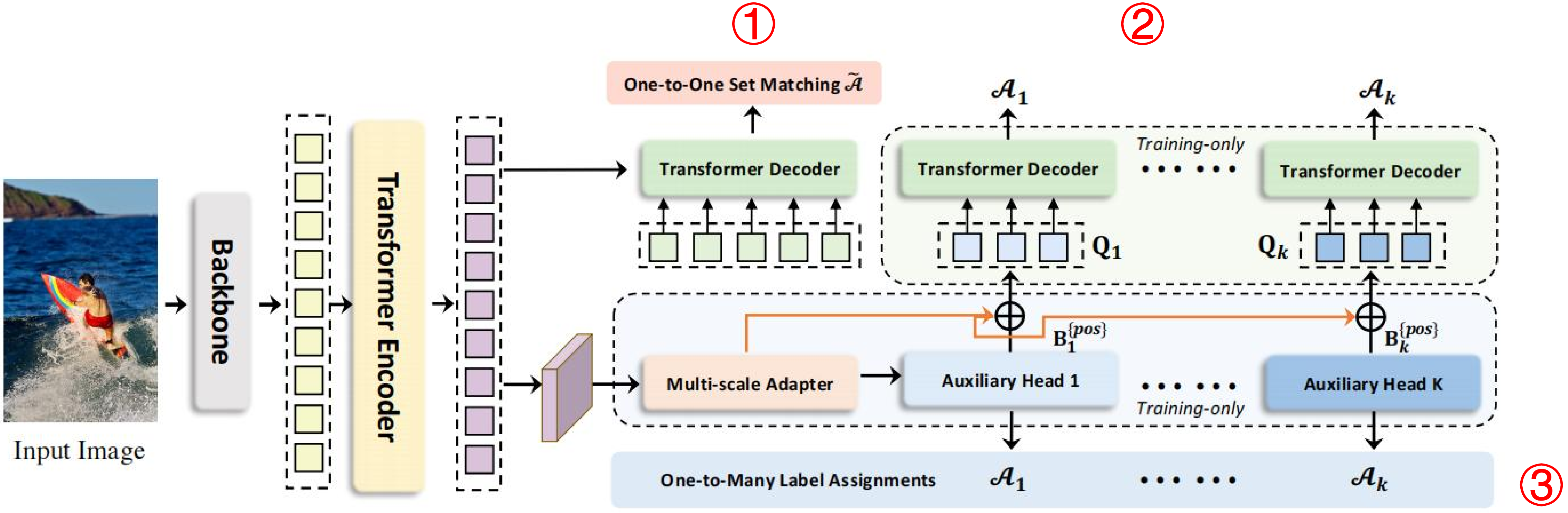


Figure 4. **Framework of our Collaborative Hybrid Assignment Training.** The auxiliary branches are discarded during evaluation.

$$\mathcal{L}^{global} = \sum_{l=1}^{L}(\widetilde{\mathcal{L}}_l^{dec} + \lambda_1 \sum_{i=1}^{K} \mathcal{L}_{i,l}^{dec} + \lambda_2 \mathcal{L}^{enc}), \quad (6)$$

① ② ③

https://zhuanlan.zhihu.com/p/715082004

DETR

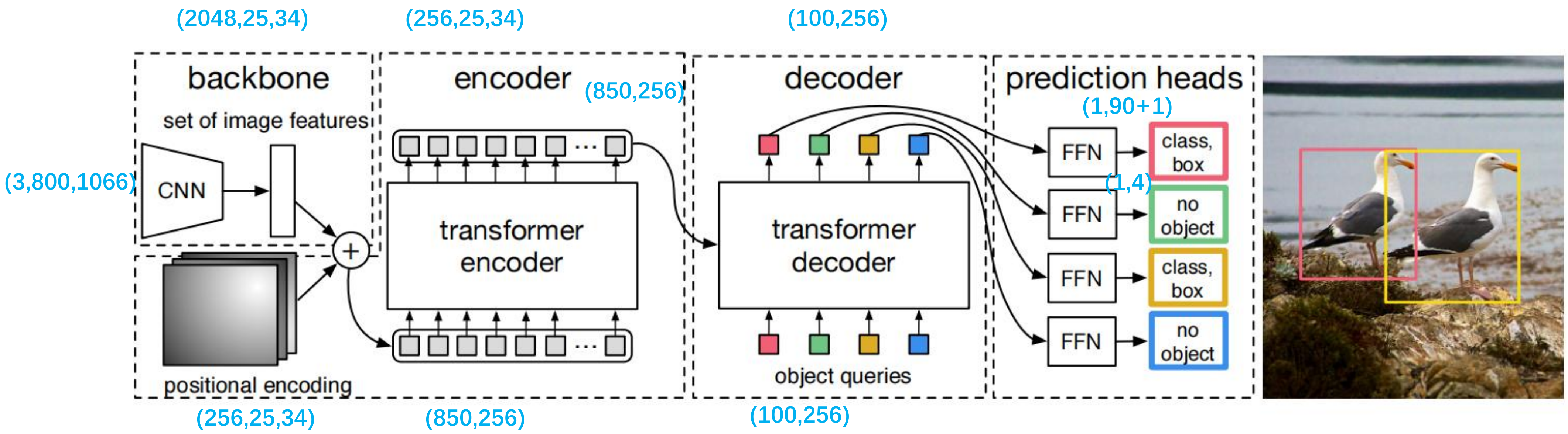
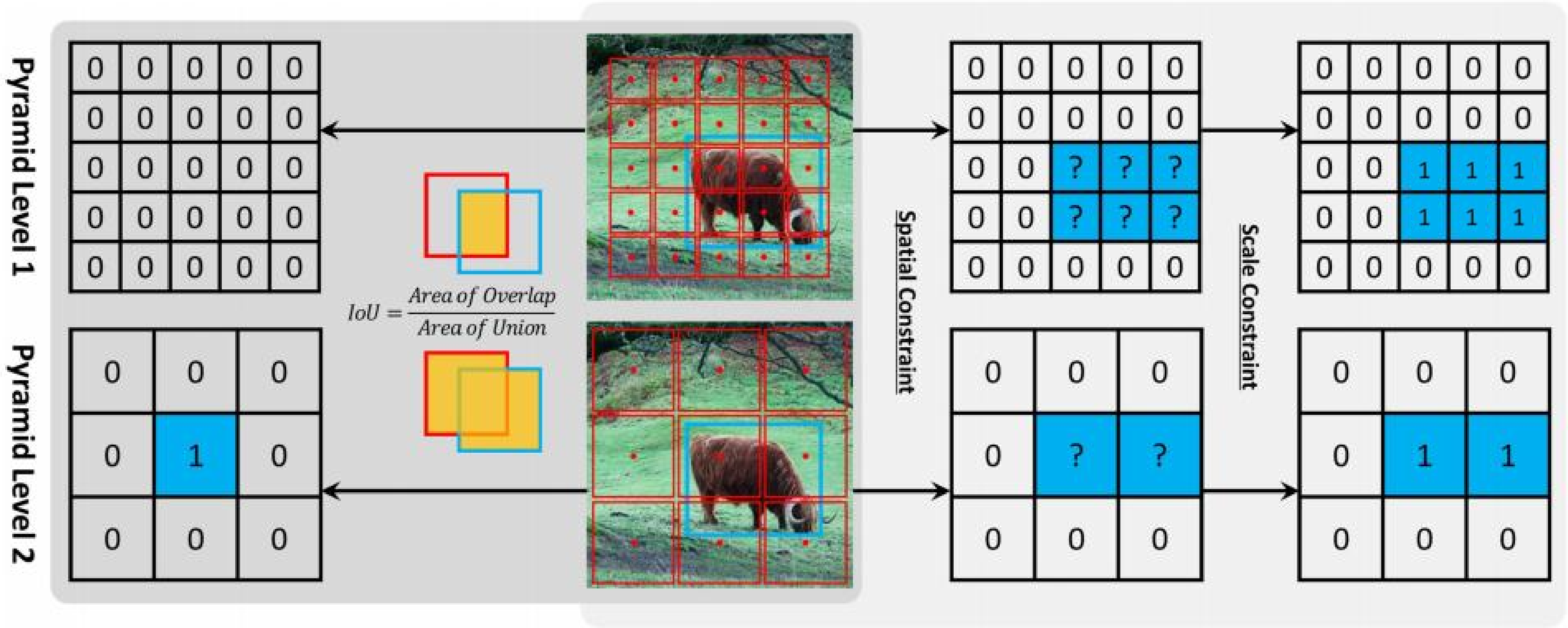


图2：DETR 使用传统的 CNN 主干网络来学习输入图像的 2D 表示。模型将该表示展平，并补充位置编码，然后将其传递给Transformer编码器。Transformer解码器以少量固定数量的学习位置嵌入（称之为对象查询）作为输入，并且还关注编码器的输出。将解码器的每个输出嵌入传递给一个共享的前馈网络（FFN），该网络预测一个检测（类别和边界框）或“无对象”类别。

如何定义正样本？



(a) RetinaNet (b) FCOS

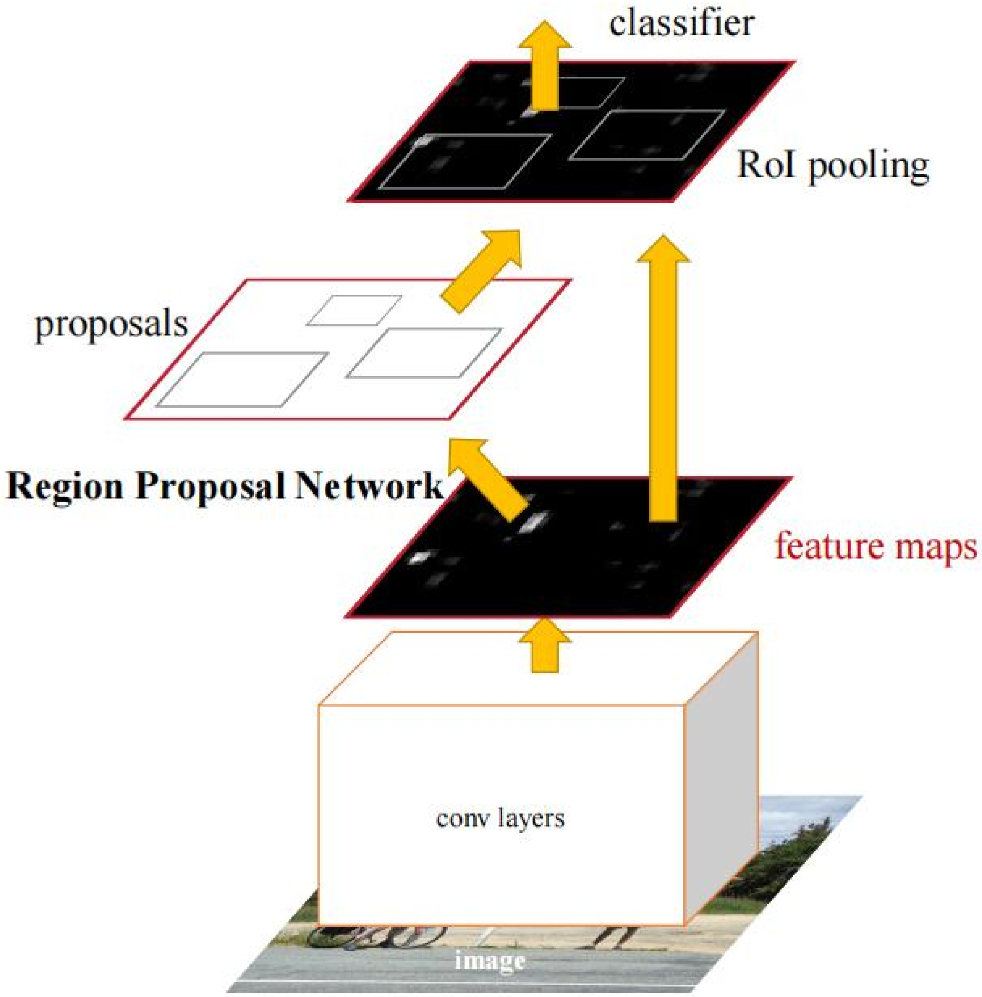
一对一 vs. 一对多

- **一对多标签分配**
  - 训练阶段，一个真实边界框可以作为多个框候选项的正样本。
  - 在基于Anchor的经典检测器中，例如Faster-RCNN和RetinaNet，样本选择是由预定的IoU阈值和Anchor与标注框之间的IoU引导的。
  - Anchor-free的FCOS利用中心先验，将每个边界框中心附近的空间位置视为正样本。
  - 自适应机制被纳入一对多标签分配中，以克服固定标签分配的局限性。ATSS通过统计学上的动态IoU值对锚点进行自适应选择。
- **一对一集合匹配**
  - 作为基于transformer的检测器的先驱，DETR将一对一集合匹配方案整合到目标检测中，并执行完全端到端的目标检测。一对一集合匹配策略首先通过匈牙利匹配计算全局匹配成本，并为每个真实边界框分配只有一个最小匹配成本的正样本。

FasterRCNN

由粗到细（coarse to fine）

粗粒度：类别只区分前景和背景



细粒度：区分具体类别

ATSS

依据手动设置阈值筛选正负样本

-->依据数据分布自适应确定阈值

（阈值 = 均值 + 标准差）

算法 1

第 1 ~ 2 行，进行初始化。

第 3 ~ 6 行，对于图像上的每个真实边界框 g，首先找出其候选正例；在每个金字塔层级上，根据 L2 距离选择 k 个中心最接近真实边界框 g 中心的锚点。假设有 L 个特征金字塔层级，真实边界框 g 将有 k × L 个候选正例。

第 7行，计算这些候选者与真实边界框 g 之间的 IoU 作为 D_g 。

第 8~ 9 行，计算其均值和标准差 m_g 和 v_g 。

第 10 行，有了上述统计数据，就可以获得此真实边界框 g 的 IoU 阈值 t_g 。

第 11 ~ 15 行，选择 IoU 大于或等于阈值 t_g 的候选者作为最终正例。值得注意的是，我们还限制正例的中心位于真实边界框内，如第 12 行所示。此外，如果一个锚点被分配给多个真实边界框，则选择 IoU 最高的。

第 17 行，其余的是负例。

Algorithm 1 Adaptive Training Sample Selection (ATSS)

Input:

$\mathcal{G}$ is a set of ground-truth boxes on the image
$\mathcal{L}$ is the number of feature pyramid levels
$\mathcal{A}_i$ is a set of anchor boxes from the i_{th} pyramid levels
$\mathcal{A}$ is a set of all anchor boxes
k is a quite robust hyperparameter with a default value of 9

Output:

$\mathcal{P}$ is a set of positive samples
$\mathcal{N}$ is a set of negative samples

1: **for** each ground-truth $g \in \mathcal{G}$ **do**
2: build an empty set for candidate positive samples of the ground-truth g: $\mathcal{C}_g \leftarrow \varnothing$;
3: **for** each level $i \in [1, \mathcal{L}]$ **do**
4: $\mathcal{S}_i \leftarrow$ select k anchors from $\mathcal{A}_i$ whose center are closest to the center of ground-truth g based on L2 distance;
5: $\mathcal{C}_g = \mathcal{C}_g \cup \mathcal{S}_i$;
6: **end for**
7: compute IoU between $\mathcal{C}_g$ and g: $\mathcal{D}_g = IoU(\mathcal{C}_g, g)$;
8: compute mean of $\mathcal{D}_g$: $m_g = Mean(\mathcal{D}_g)$;
9: compute standard deviation of $\mathcal{D}_g$: $v_g = Std(\mathcal{D}_g)$;
10: compute IoU threshold for ground-truth g: $t_g = m_g + v_g$;
11: **for** each candidate $c \in \mathcal{C}_g$ **do**
12: **if** $IoU(c, g) \geq t_g$ and center of c in g **then**
13: $\mathcal{P} = \mathcal{P} \cup c$;
14: **end if**
15: **end for**
16: **end for**
17: $\mathcal{N} = \mathcal{A} - \mathcal{P}$;
18: **return** $\mathcal{P}, \mathcal{N}$;

https://zhuanlan.zhihu.com/p/713749238

这张图说明了啥？

ATSS的Encoder层很强，Co-Deformable-DETR次之，Deformable-DETR太挫

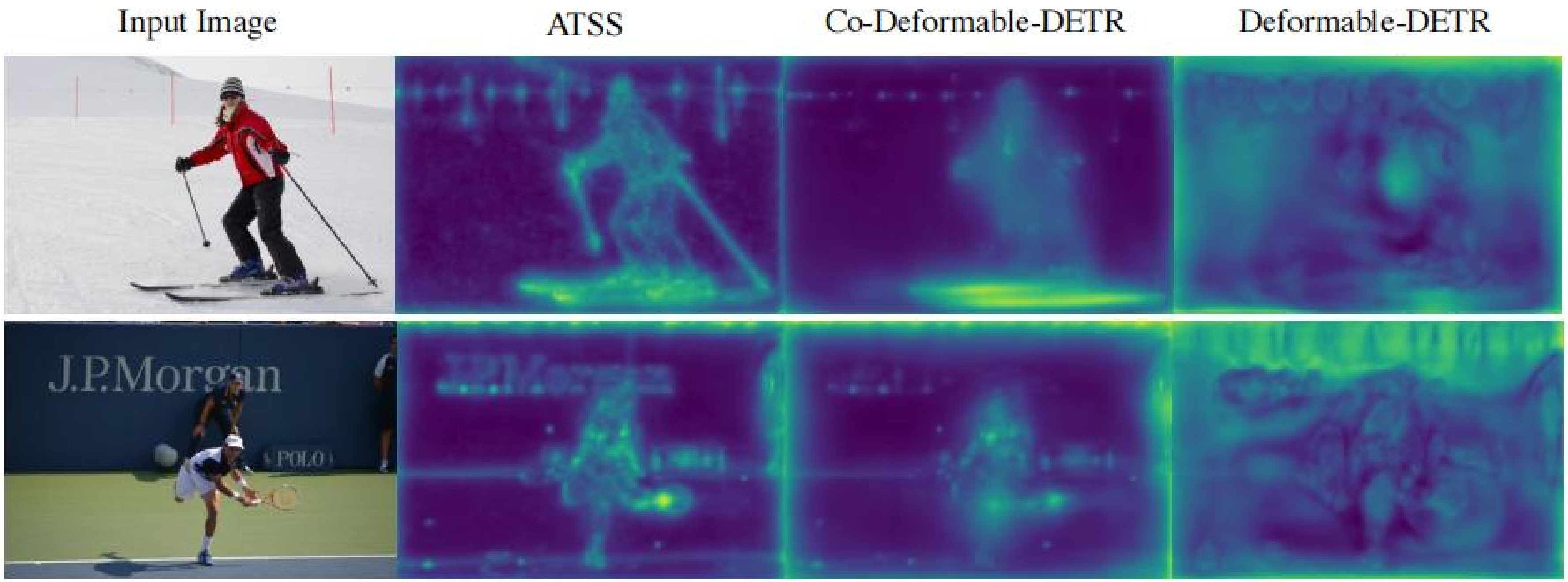


Figure 3. Visualizations of discriminability scores in the encoder.

Encoder loss

Head i	Loss $\mathcal{L}_i$	Assignment $\mathcal{A}_i$		
		$\{pos\}$, $\{neg\}$ Generation	$\mathbf{P}_i$ Generation	$\mathbf{B}_i^{\{pos\}}$ Generation
Faster-RCNN [27]	cls: CE loss, reg: GIoU loss	$\{pos\}$: IoU(proposal, gt)>0.5 $\{neg\}$: IoU(proposal, gt)<0.5	$\{pos\}$: gt labels, offset(proposal, gt) $\{neg\}$: gt labels	positive proposals (x_1, y_1, x_2, y_2)
ATSS [41]	cls: Focal loss reg: GIoU, BCE loss	$\{pos\}$:IoU(anchor, gt)>(mean+std) $\{neg\}$: IoU(anchor, gt)<(mean+std)	$\{pos\}$: gt labels, offset(anchor, gt), centerness $\{neg\}$: gt labels	positive anchors (x_1, y_1, x_2, y_2)
RetinaNet [21]	cls: Focal loss reg: GIoU Loss	$\{pos\}$: IoU(anchor, gt)>0.5 $\{neg\}$: IoU(anchor, gt)<0.4	$\{pos\}$: gt labels, offset(anchor, gt) $\{neg\}$: gt labels	positive anchors (x_1, y_1, x_2, y_2)
FCOS [32]	cls: Focal Loss reg: GIoU, BCE loss	$\{pos\}$: points inside gt center area $\{neg\}$: points outside gt center area	$\{pos\}$: gt labels, ltrb distance, centerness $\{neg\}$: gt labels	FCOS point (cx, cy) $w = h = 8 \times 2^{2+j}$

Table 1. **Detailed information of auxiliary heads.** The auxiliary heads include Faster-RCNN [27], ATSS [41], RetinaNet [21], and FCOS [32]. If not otherwise specified, we follow the original implementations, *e.g.*, anchor generation.

$$\mathbf{P}_i^{\{pos\}}, \mathbf{B}_i^{\{pos\}}, \mathbf{P}_i^{\{neg\}} = \mathcal{A}_i(\hat{\mathbf{P}}_i, \mathbf{G}), \tag{1}$$

$$\mathcal{L}_i^{enc} = \mathcal{L}_i(\hat{\mathbf{P}}_i^{\{pos\}}, \mathbf{P}_i^{\{pos\}}) + \mathcal{L}_i(\hat{\mathbf{P}}_i^{\{neg\}}, \mathbf{P}_i^{\{neg\}}), \tag{2}$$

$$\mathcal{L}^{enc} = \sum_{i=1}^{K} \mathcal{L}_i^{enc} \tag{3}$$

获得预测结果和GT，依据不同方法（ATSS、FCOS等）
生成不同的对齐方式，标记正样本和负样本

Decoder loss

- 传统DETR正查询过少导致Decoder中的交叉注意力学习效率低，Co-DETR引入多个辅助头扩充正查询，给定第i个辅助头部中的正坐标集合$B_i^{\{pos\}}$，可以通过以下方式生成额外的定制化正查询

$$\mathbf{Q}_i = \mathrm{Linear}(\mathrm{PE}(\mathbf{B}_i^{\{pos\}})) + \mathrm{Linear}(\mathrm{E}(\{\mathcal{F}_*\}, \{pos\})). \tag{4}$$

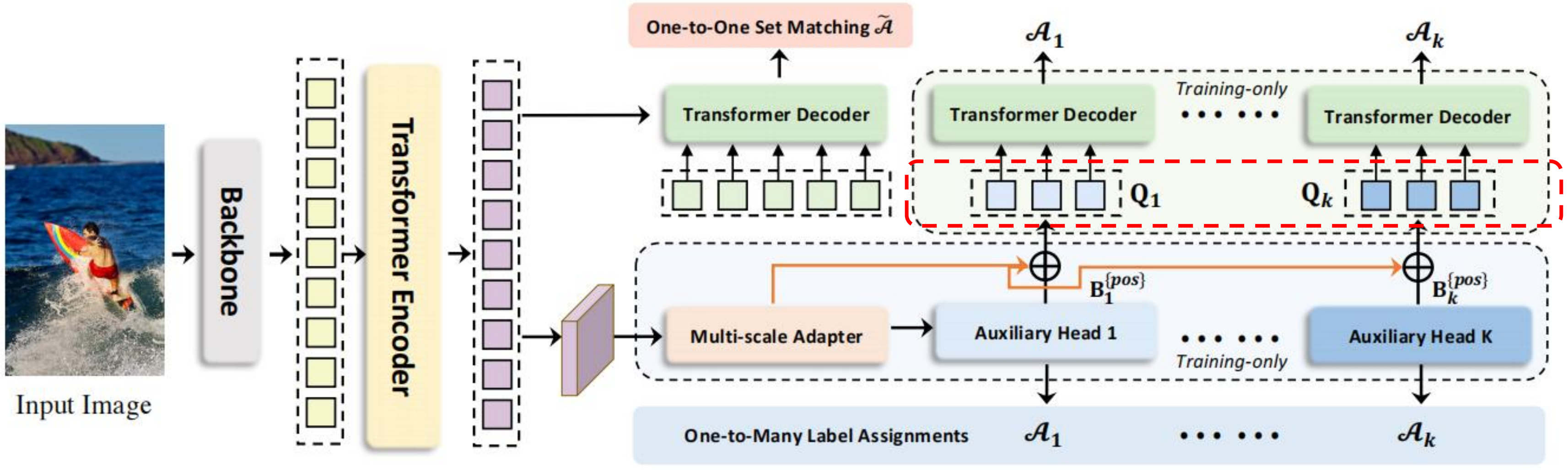


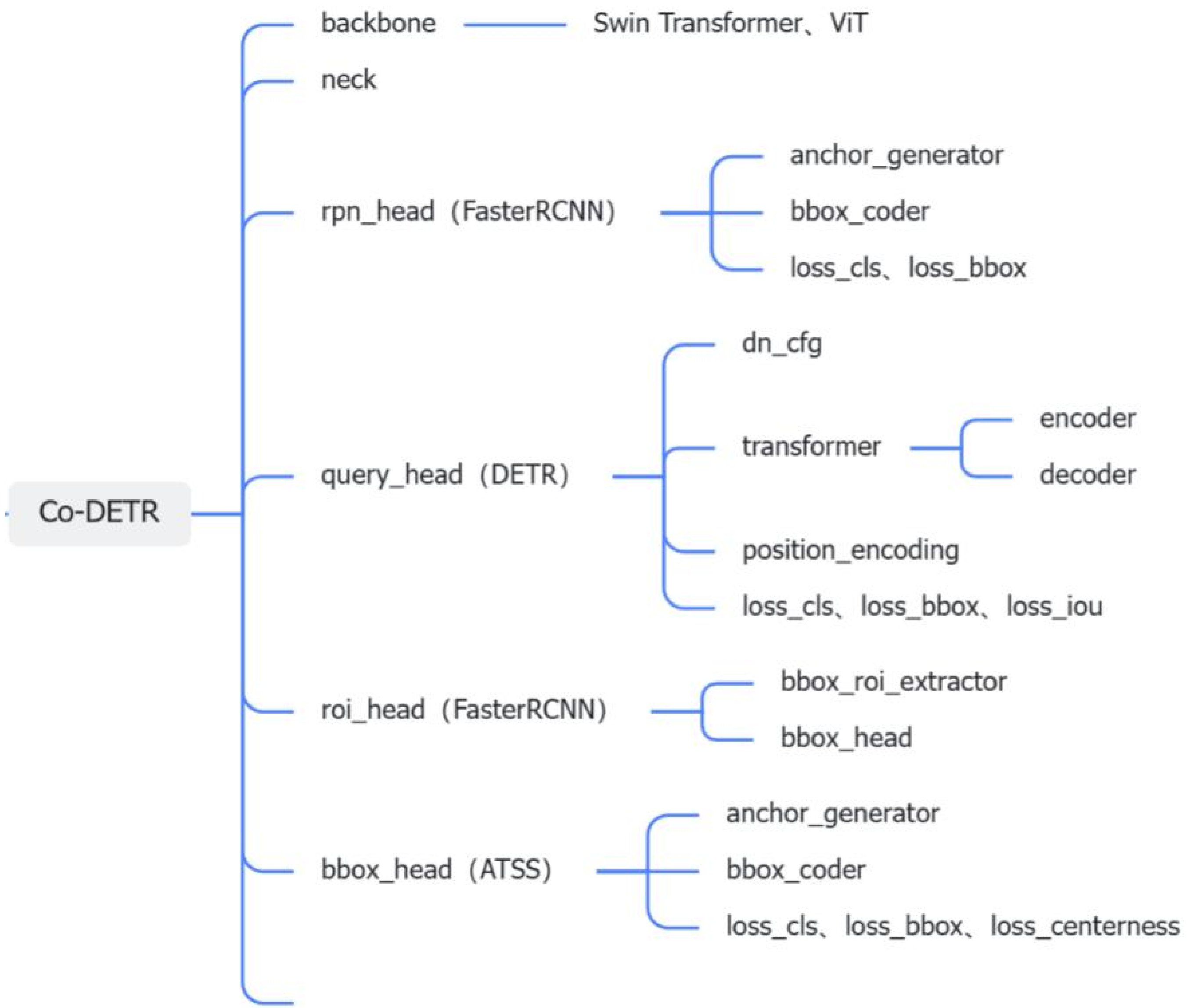
Figure 4. **Framework of our Collaborative Hybrid Assignment Training.** The auxiliary branches are discarded during evaluation.

$$\mathcal{L}_{i,l}^{dec} = \widetilde{\mathcal{L}}(\widetilde{\mathbf{P}}_{i,l}, \mathbf{P}_i^{\{pos\}}). \tag{5}$$

调试环境

- github地址
  - https://github.com/Sense-X/Co-DETR
- 调试问题
  - KeyError: 'CoDETR is not in the models registry'
    - 参考 https://github.com/Sense-X/Co-DETR/issues/93
      - ① pip install -e .
      - ② pip uninstall projects

配置文件


Co-DETR
backbone
Swin Transformer、ViT
neck
rpn_head（FasterRCNN）
anchor_generator
bbox_coder
loss_cls、loss_bbox
query_head（DETR）
dn_cfg
transformer
encoder
decoder
position_encoding
loss_cls、loss_bbox、loss_iou
roi_head（FasterRCNN）
bbox_roi_extractor
bbox_head
bbox_head（ATSS）
anchor_generator
bbox_coder
loss_cls、loss_bbox、loss_centerness

谢谢观看