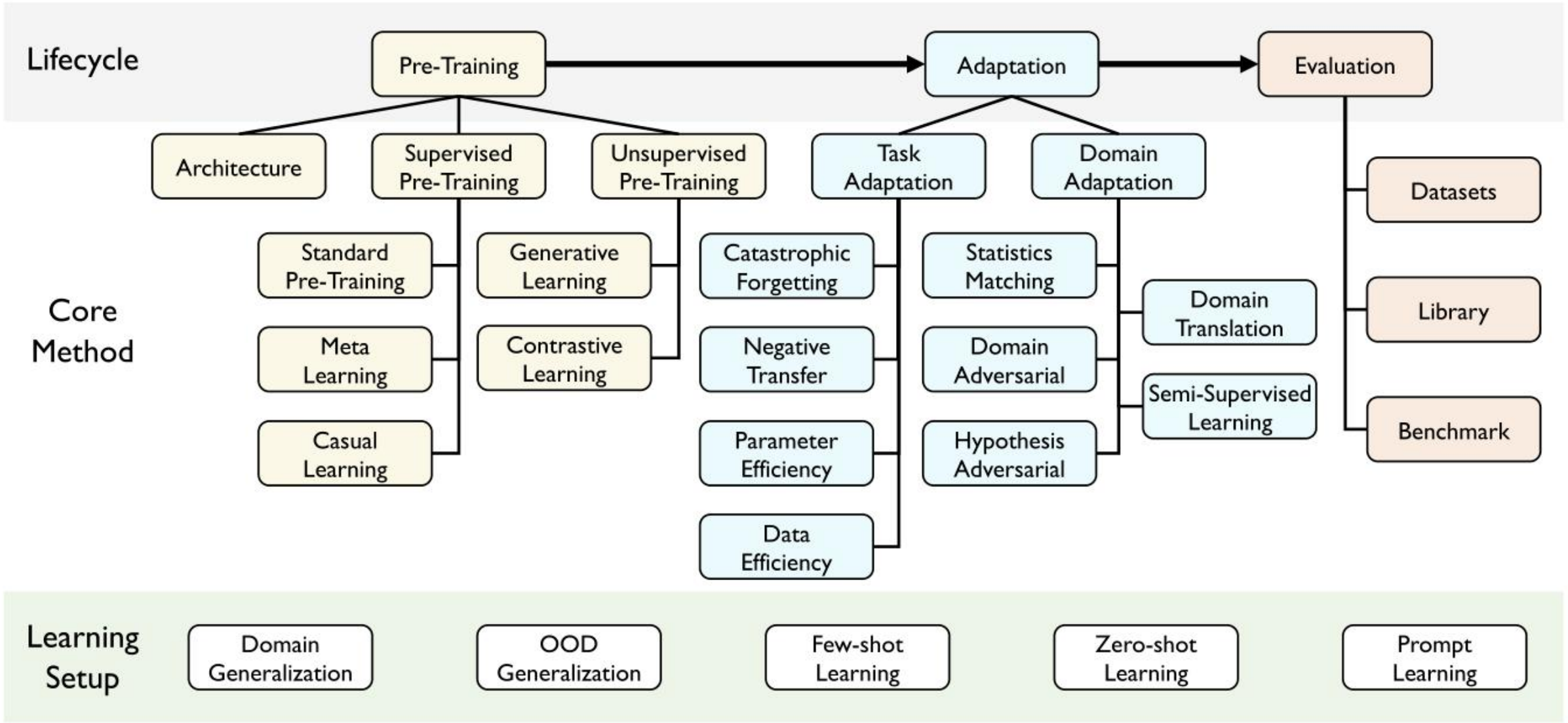
Domain Adaptation Object Detection

Transferability in Deep Learning



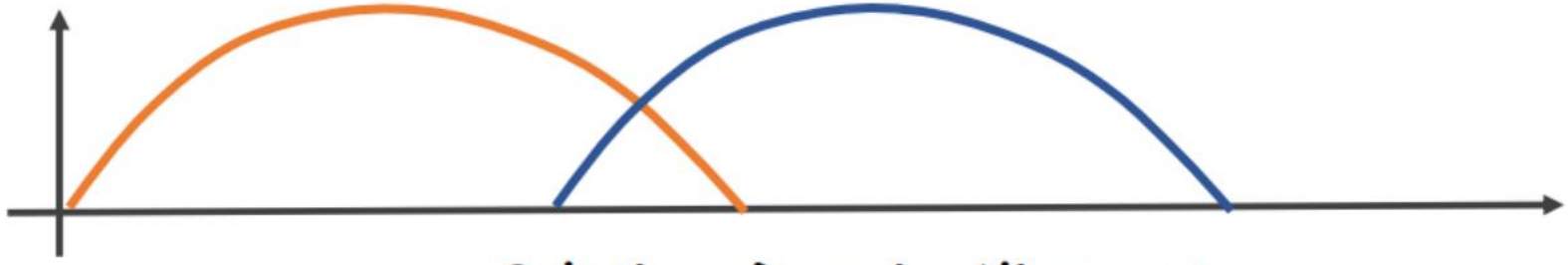
Unsupervised Domain Adaptation

Bike:

Keyboard:

Source domain
with annotations

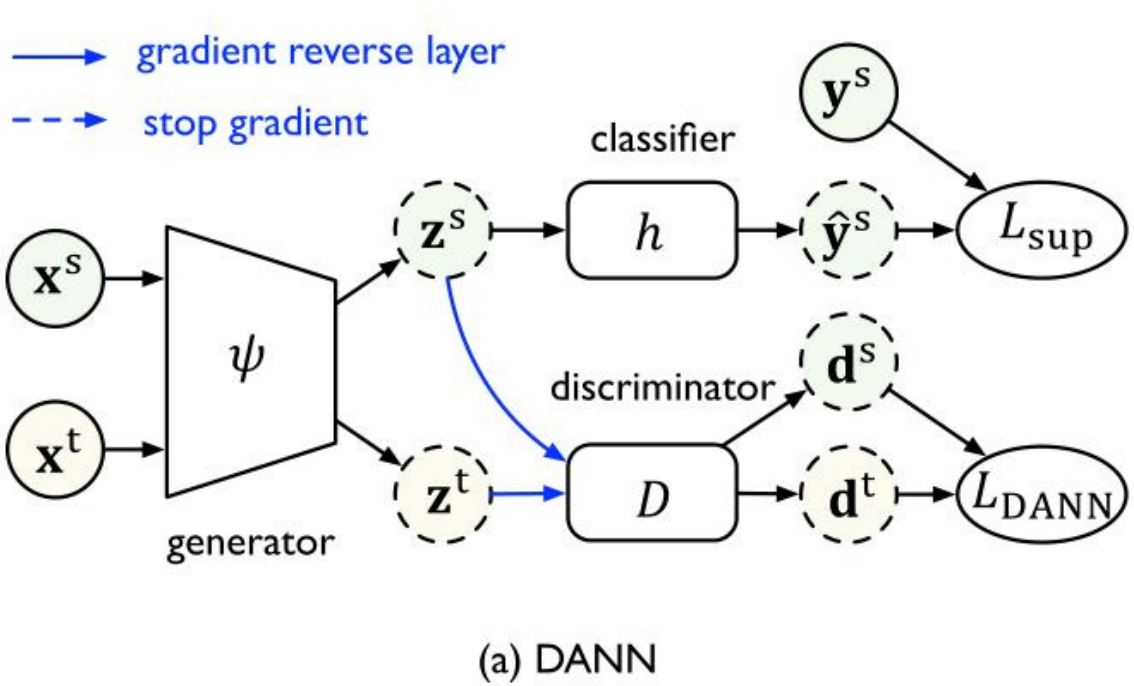
Target domain
without annotations



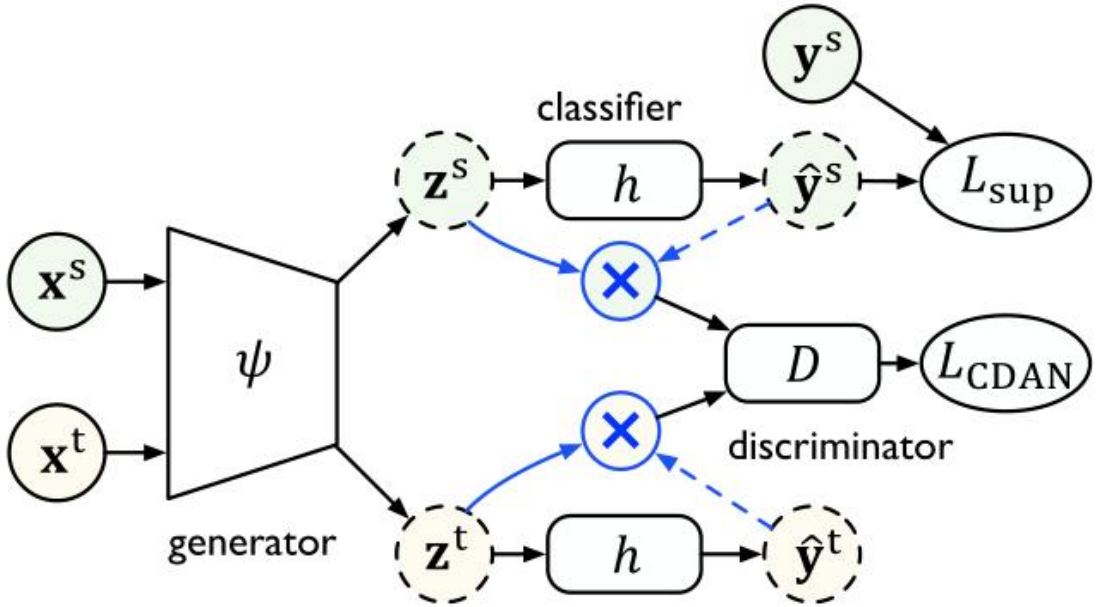
Solution: Domain Alignment

✓ **Goal**: Achieving good performance on the target domain.

Domain Adversarial Learning



(a) DANN



(b) CDAN

Domain Adaptation Object Detection

为了缓解分布偏移（Domain Shift）对于检测器性能的影响，许多工作使用域对抗自适应（Adversarial Domain Adaptation） 将检测器从有标注数据的源域迁移到只有无标注数据的目标域。对于跨域物体检测任务而言，源域上既有物体的定位框标注，也有对应的类别标注，而在目标域上，定位框和类别标注均不存在。这就带来了以下三个挑战：

一、数据挑战 Data Challenge

1. 对于物体分类任务，对抗域自适应的目标是拉近同一类物体的特征。但是对于物体检测任务而言，物体的位置未知，因此对输入数据的哪个部分进行对抗拉近成为关键问题。
2. 图像级别的全局特征自适应（Global Feature Adaptation）很容易混淆不同类别物体的特征，因为每张图片都包含多个物体。
3. 对象级别的实例特征自适应（Instance Feature Adaptation）容易混淆前景和背景的特征，因为检测器在目标域上的输出是不可靠的，大部分预测的前景实际上都是背景。
4. 像素级别的局部特征自适应（Local Feature Adaptation）适合解决低层语义级别的分布偏移，比如从晴天迁移到雾天的目标检测，但是不太适合解决高层语义的分布偏移，比如从自然风景到卡通的目标检测。

二、架构挑战 Architecture Challenge

1. 对抗域自适应方法，将域判别器和梯度反转层引入了检测器架构中，以增强特征的迁移性（Transferability），但它可能牺牲掉特征的判别性（Discriminability），而这将极大地影响检测器本身的定位和分类能力。 由于迁移性和判别性之间的矛盾，这些模块在检测架构中的放置位置对最终性能有很大影响。
2. 同时，由于检测器的架构类型非常多（比如一阶段和两阶段的检测器架构差异就非常大），将某个检测器上有效的域自适应方法拿到另外一种检测器上，可能需要花费大量的精力调整迁移模块的位置，甚至无法找到合适的位置。 因此，这些方法对不同检测架构的可扩展性并不令人满意。

三、任务挑战 Task Challenge

相比于物体分类，物体检测同时也是一个多任务学习（Multi-Task Learning）的设定。相比于监督学习，域自适应中的多任务更容易出现冲突。 比如我们的实验证实了：

物体分类任务的域自适应需要输入数据尽可能满足类簇假设，即输入的图片或者是猫，或者是狗，而不是某些中间的状态。

物体定位任务的域自适应则需要输入的数据尽可能都是前景，同时覆盖不同Intersection over Union (IoU)的数据。

DA-Faster(CVPR 2018)

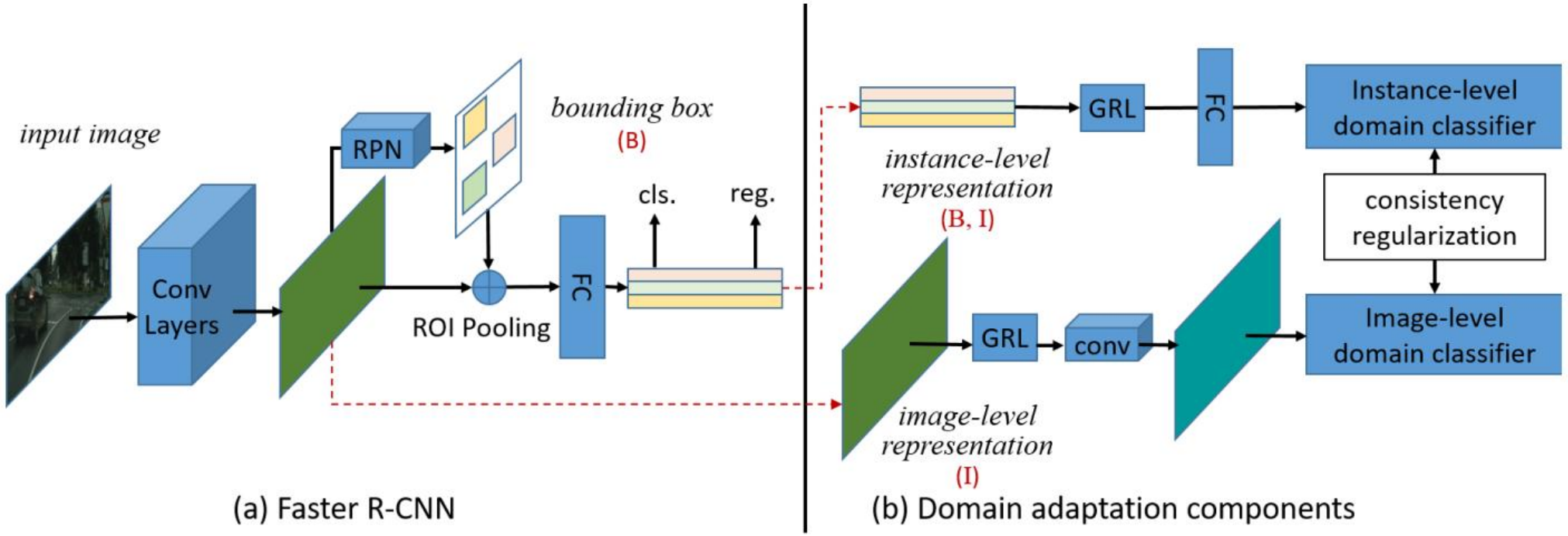


Figure 2. **An overview of our Domain Adaptive Faster R-CNN model:** we tackle the domain shift on two levels, the image level and the instance level. A domain classifier is built on each level, trained in an adversarial training manner. A consistency regularizer is incorporated within these two classifiers to learn a domain-invariant RPN for the Faster R-CNN model.

作者通过对齐image-level 和 instance-level 特征的方式来实现无监督域适应，基于Faster RCNN实现一个end-to-end系统。

主要解决两个问题： - image-level shift：如光照、图片风格等 - instance-level shift ： 如物体外貌、大小等

$$L = L_{det} + \lambda(L_{img} + L_{ins} + L_{cst})$$

SWDA(CVPR 2019)

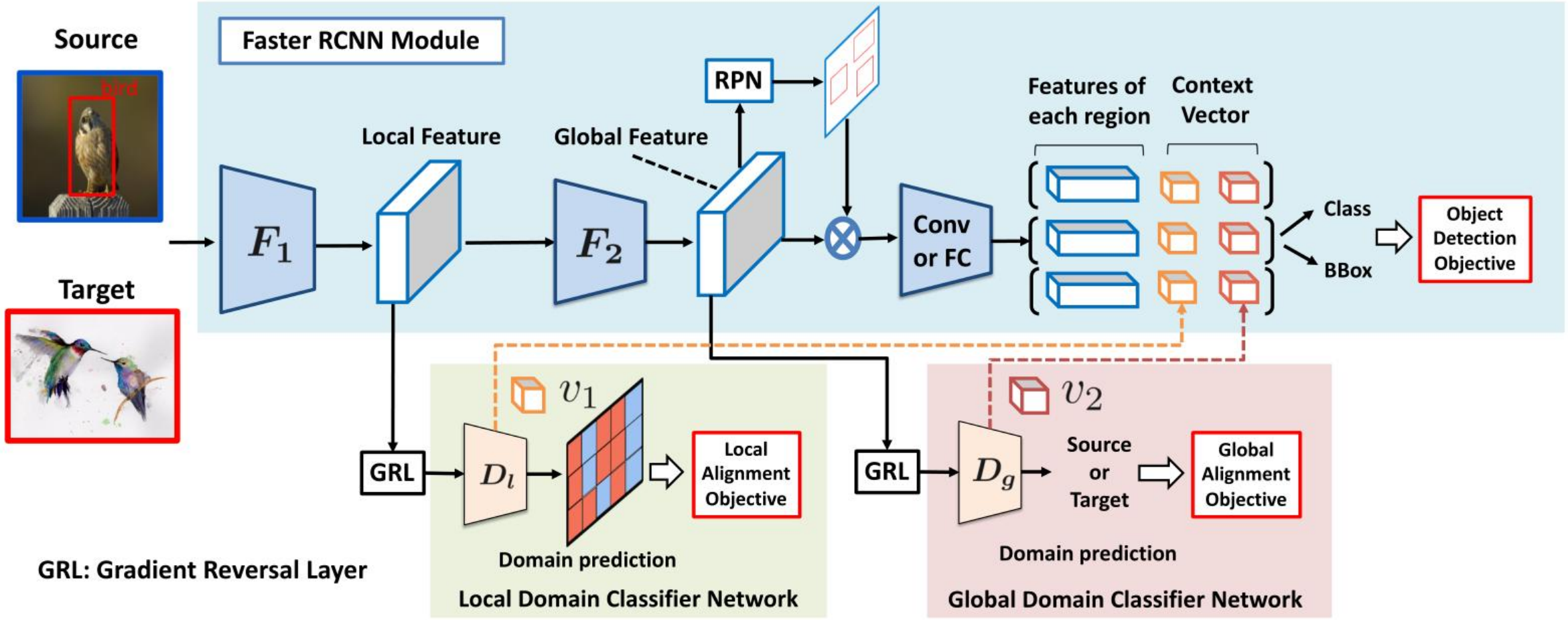


Figure 3. Proposed Network Architecture. Our method performs strong-local alignment by a local domain classifier network and weak-global alignment by a global domain classifier. The context vector is extracted by the domain classifiers and is concatenated in the layer before the final fully connected layer.

$$\mathcal{L}_{adv}(F, D) = \mathcal{L}_{loc}(F_1, D_l) + \mathcal{L}_{global}(F, D_g)$$

$$\max_{D} \min_{F,R} \mathcal{L}_{cls}(F, R) - \lambda \mathcal{L}_{adv}(F, D)$$

SCL(arXiv 2019)

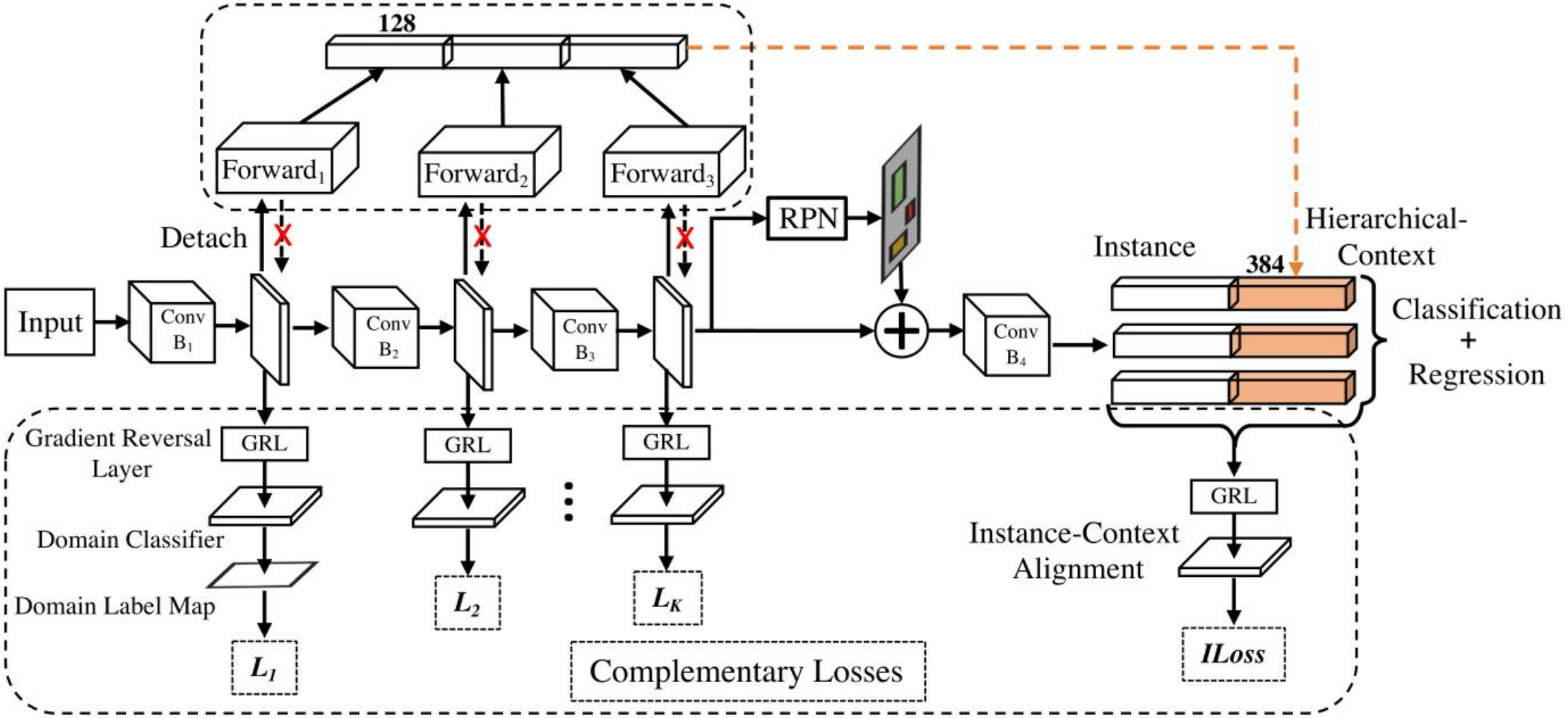
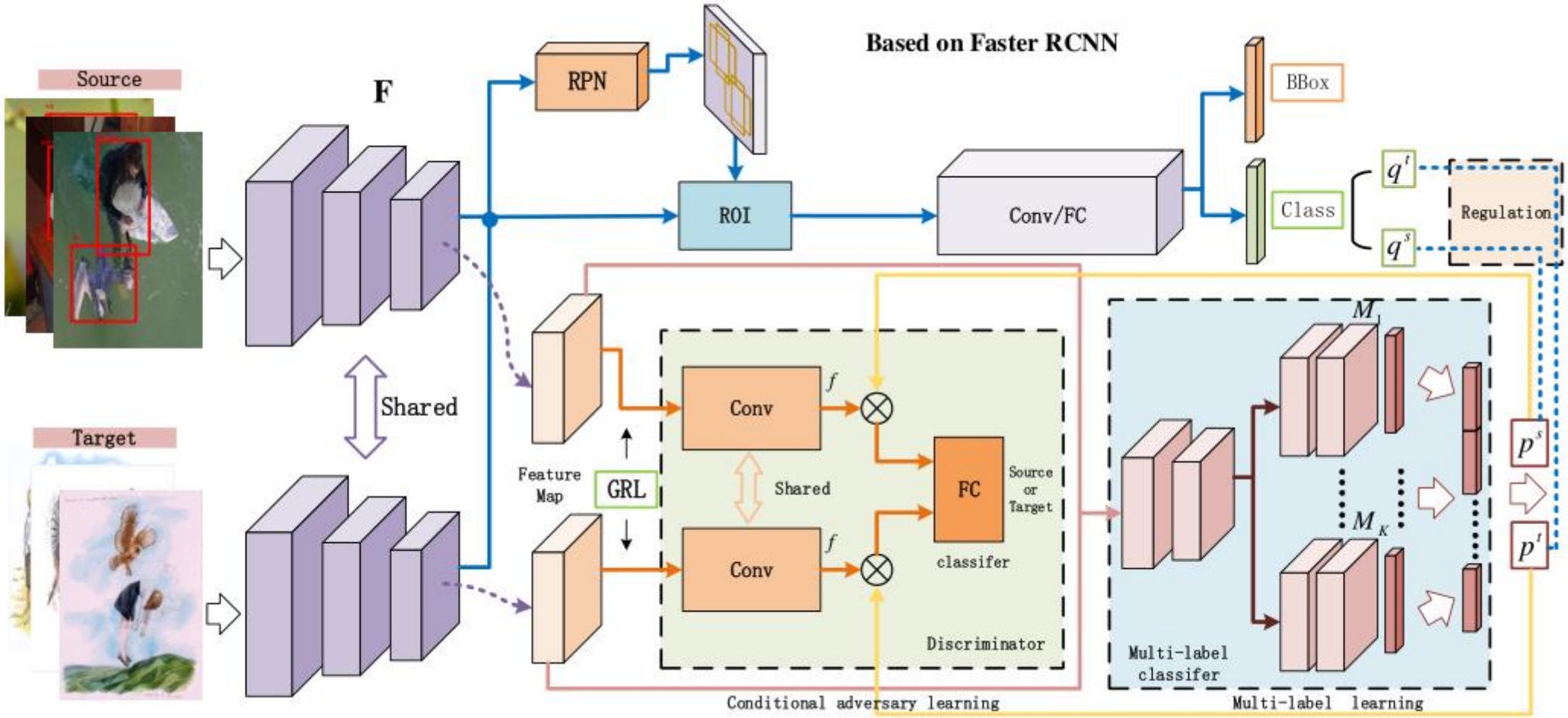


Figure 2: Overview of our SCL framework. More details please refer to Section 2.

$$\min_{\mathcal{F},\mathbf{R}} \max_{\mathbf{D}} \mathcal{L}_{det}(\mathcal{F}(\mathbf{Z}), \mathbf{R}) - \lambda \mathcal{L}_{\mathbf{SCL}}(\mathcal{F}(\mathbf{Z}), \mathbf{D})$$

MCAR(ECCV 2020)



$$\mathcal{L}_{all} = \mathcal{L}_{\text{det}} + \lambda \mathcal{L}_{adv} + \mu \mathcal{L}_{multi} + \varepsilon \mathcal{L}_{kl}$$

$$\min_{F} \max_{D} \quad \mathcal{L}_{all}$$

CFFA(CVPR 2020)

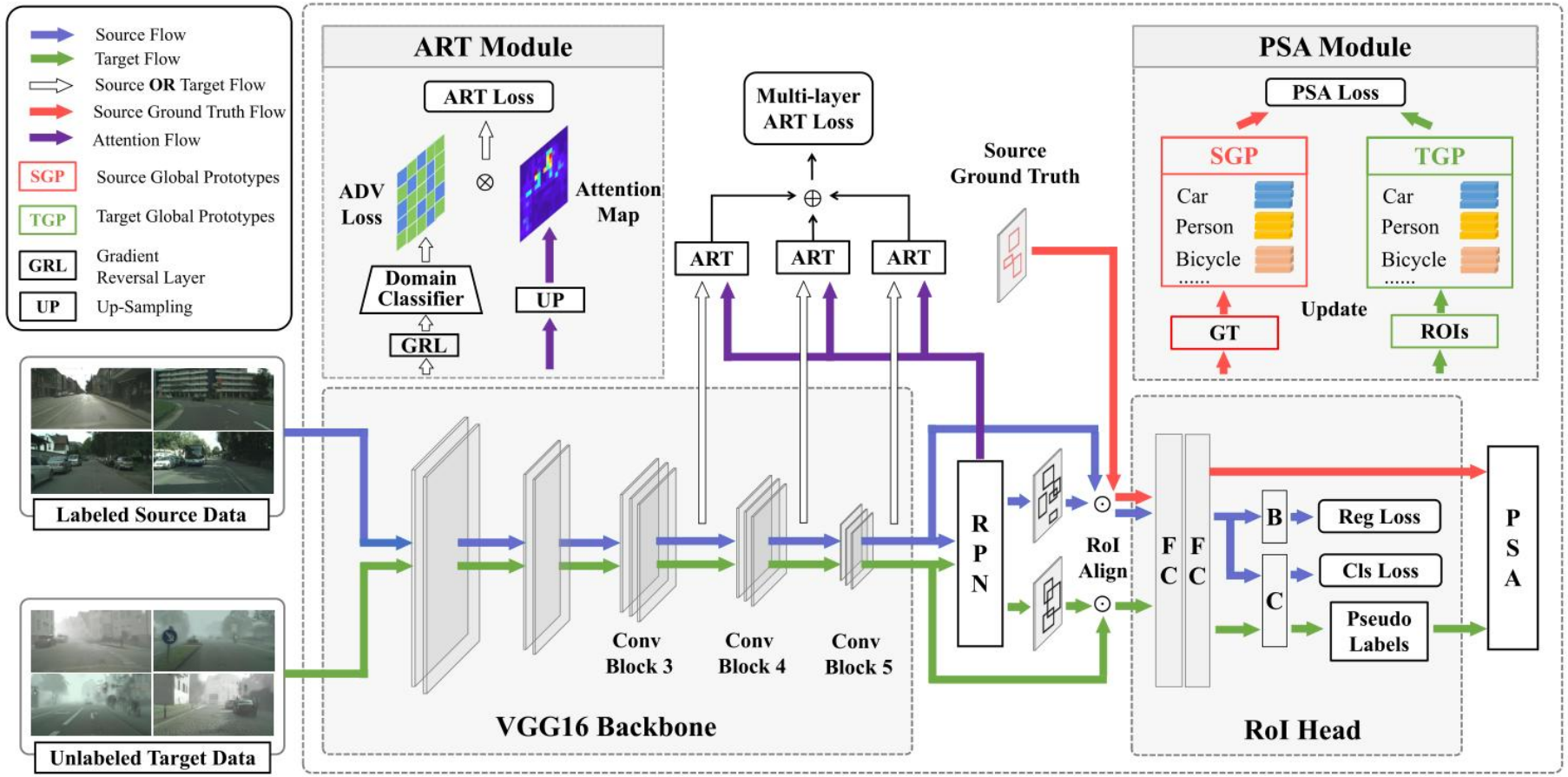
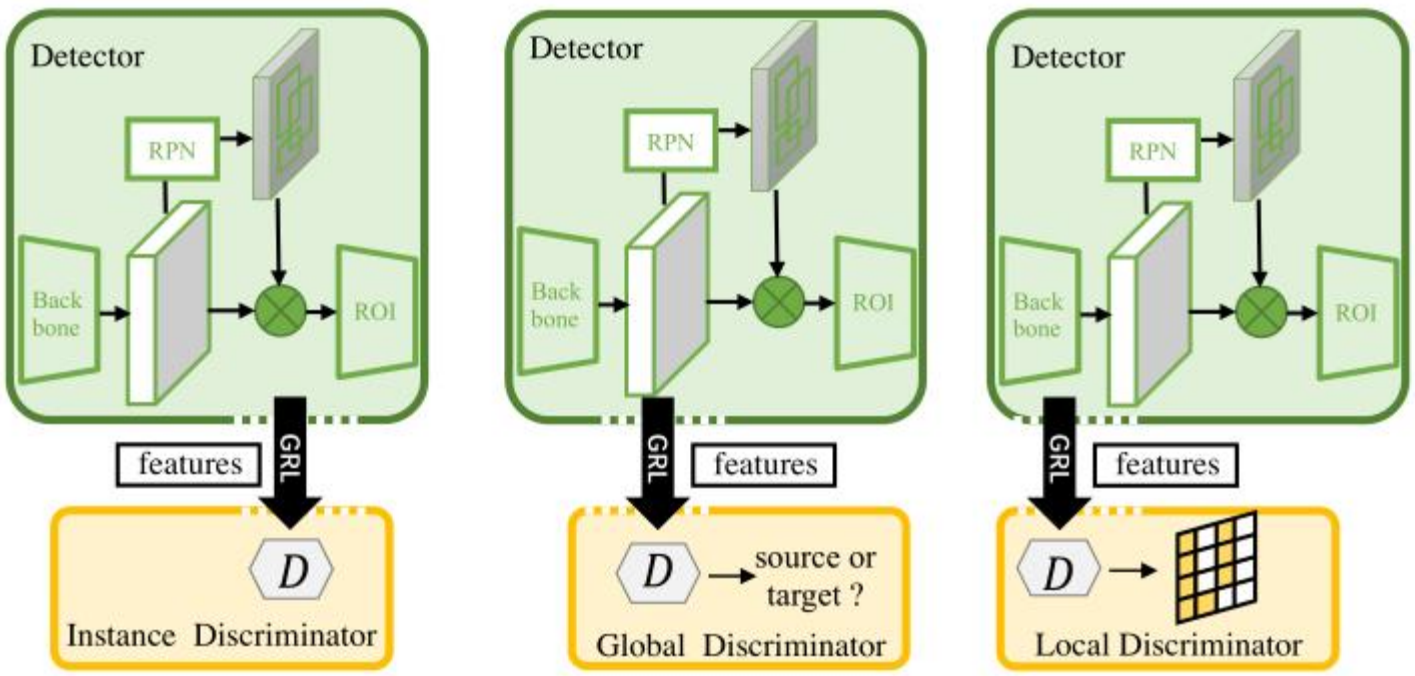


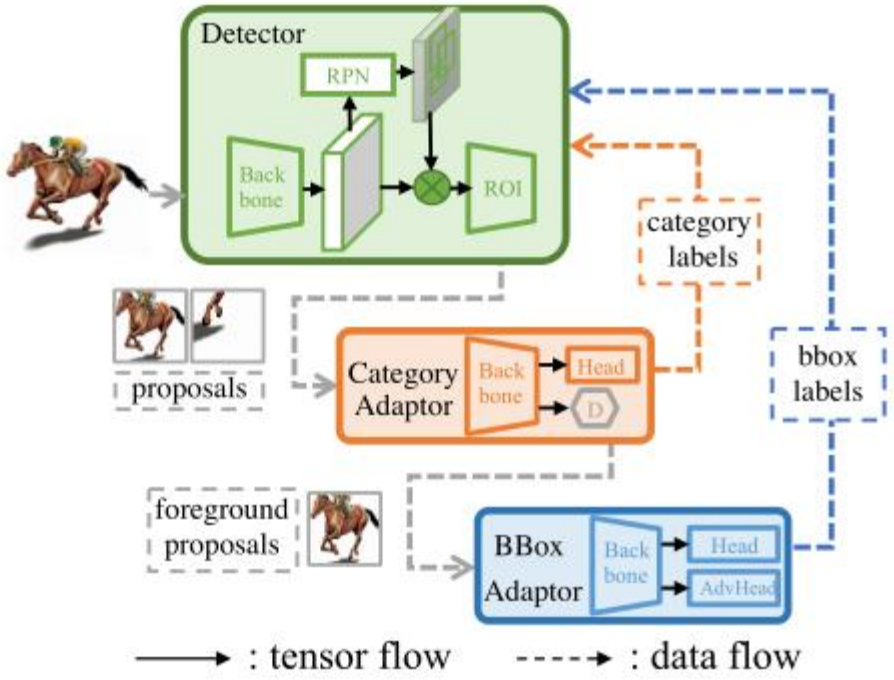
Figure 2. Overview of the proposed feature adaptation framework. We address the problem of domain shift on foreground regions by coarse-to-fine scheme with the ART and PSA modules. First, we utilize the attention map learned from the RPN module to localize foregrounds. Combined with multiple domain classifiers, the ART module puts more emphasis on aligning feature distributions of foreground regions, which achieves a coarse-grained adaptation in a category-agnostic way. Second, the PSA module makes use of ground truth labels (for source) and pseudo labels (for target) to maintain global prototypes for each category, and delivers fine-grained adaptation on foreground regions in a category-ware mode.

$$\mathcal{L}_{total} = \mathcal{L}_{det} + \lambda_1 \mathcal{L}_{ART} + \lambda_2 \mathcal{L}_{PSA}$$

D-adapt(ICLR 2022)



(a) Instance adapt (b) Global adapt (c) Local adapt



(d) Decouple adapt

D-adapt(ICLR 2022)

Algorithm 1: D-adapt Training Pipeline.

input : Source domain $\mathcal{D}_s$ and target domain $\mathcal{D}_t$,
number of iterations T
output: Cross-domain object detector G^{det}

initialize the object detector G^{det} by optimizing with $\mathcal{L}_s^{\text{det}}$;
for $t \leftarrow 1$ **to** T **do**
  generate proposals $\mathcal{D}_s^{\text{prop}}$ and $\mathcal{D}_t^{\text{prop}}$ for each sample in $\mathcal{D}_s$ and $\mathcal{D}_t$ by G^{det};
  for *each mini-batch in* $\mathcal{D}_s^{prop}$ *and* $\mathcal{D}_t^{prop}$ **do**
    train the category adaptor G^{cls};
  end
  generate category label for each proposal in $\mathcal{D}_t^{\text{prop}}$;
  generate foreground proposals $\mathcal{D}_s^{\text{fg}}$ and $\mathcal{D}_t^{\text{fg}}$ from $\mathcal{D}_s^{\text{prop}}$ and $\mathcal{D}_t^{\text{prop}}$;
  for *each mini-batch in* $\mathcal{D}_s^{fg}$ *and* $\mathcal{D}_t^{fg}$ **do**
    train the bounding box adaptor G^{reg};
  end
  generate bounding box label for each proposal in $\mathcal{D}_t^{\text{fg}}$;
  train the object detector G^{det} by optimizing with $\mathcal{L}_t^{\text{det}}$;
end

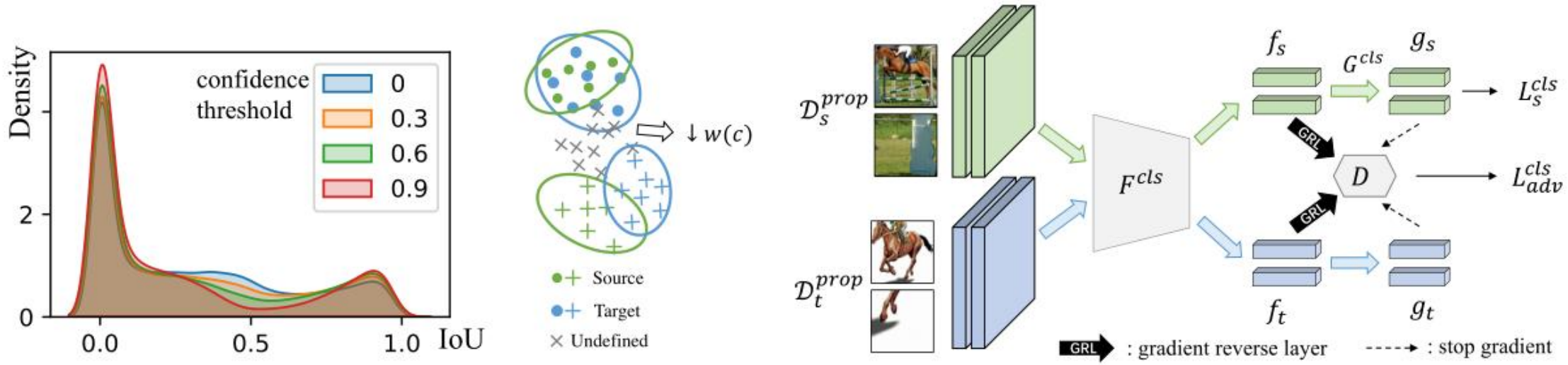
D-adapt(ICLR 2022)

$\mathcal{D}_s = \{(\mathbf{X}_s^i, \mathbf{B}_s^i, \mathbf{Y}_s^i)\}_{i=1}^{n_s}$ $\mathbf{X}_s^i$ is the image, $\mathbf{B}_s^i$ is the bounding box coordinates, and $\mathbf{Y}_s^i$ is the categories

$$\mathcal{L}_s^{\text{det}} = \mathbb{E}_{(\mathbf{X}_s, \mathbf{B}_s, \mathbf{Y}_s) \in \mathcal{D}_s} \mathcal{L}_{\text{cls}}^{\text{rpn}} + \mathcal{L}_{\text{reg}}^{\text{rpn}} + \mathcal{L}_{\text{cls}}^{\text{roi}} + \mathcal{L}_{\text{reg}}^{\text{roi}}.$$

$$\mathcal{L}_t^{\text{det}} = \mathbb{E}_{(\mathbf{X}_t, \mathbf{b}_t^{\text{det}}, \mathbf{y}_t^{\text{cls}}, \mathbf{b}_t^{\text{reg}}) \in \mathcal{D}_t^{\text{prop}}} \mathcal{L}_{\text{cls}}^{\text{roi}}(\mathbf{X}_t, \mathbf{b}_t^{\text{det}}, \mathbf{y}_t^{\text{cls}}) + \mathcal{L}_{\text{cls}}^{\text{roi}}(\mathbf{X}_t, \mathbf{b}_t^{\text{reg}}, \mathbf{y}_t^{\text{cls}})$$
$$+ \mathbb{I}_{\text{fg}}(\mathbf{y}_t^{\text{cls}}) \cdot \mathcal{L}_{\text{reg}}^{\text{roi}}(\mathbf{X}_t, \mathbf{b}_t^{\text{det}}, \mathbf{b}_t^{\text{reg}}),$$

D-adapt(ICLR 2022)



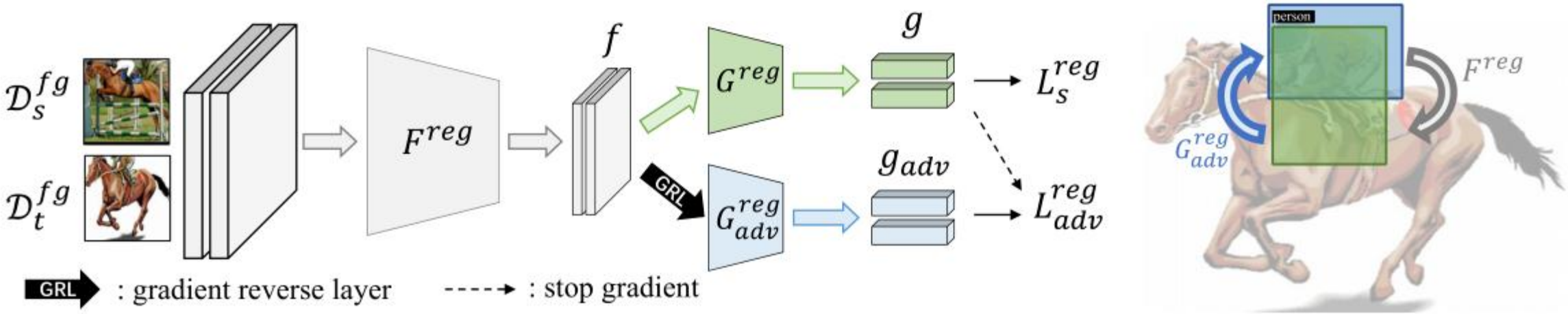
(a) IoU distribution of proposals (b) Discretization (c) Architecture of the category adaptor

Figure 2: Category adaptation (best viewed in color). **(a)** The IoU distribution of the proposals from Foggy Cityscapes. When we increase the confidence threshold from 0 to 0.9, undefined proposals (proposals with IoU between 0.3 and 0.7) will decrease. **(b)** Proposals with lower confidence will be assigned a lower weight in the adaptation. **(c)** The discriminator D is trained to separate the source-domain proposals from the target-domain proposals for each class independently, while the feature extractor F^{cls} is encouraged to fool D.

$$\max_D \mathcal{L}_{\text{adv}}^{\text{cls}} = \mathbb{E}_{\mathbf{x}_s \sim \mathcal{D}_s^{\text{prop}}} w(\mathbf{c}_s) \log[D(\mathbf{f}_s, \mathbf{g}_s)] + \mathbb{E}_{\mathbf{x}_t \sim \mathcal{D}_t^{\text{prop}}} w(\mathbf{c}_t) \log[1 - D(\mathbf{f}_t, \mathbf{g}_t)]$$

$$\min_{F^{\text{cls}}, G^{\text{cls}}} \mathbb{E}_{(\mathbf{x}_s, \mathbf{y}_s^{\text{gt}}) \sim \mathcal{D}_s^{\text{prop}}} \mathcal{L}_{\text{CE}}(G^{\text{cls}}(\mathbf{f}_s), \mathbf{y}_s^{\text{gt}}) + \lambda \mathcal{L}_{\text{adv}}^{\text{cls}}$$

D-adapt(ICLR 2022)



(a) Architecture of the bounding box adaptor (b) Minimax on IoU

Figure 3: Bounding box adaptation (best viewed in color). Box adaptor has three parts: feature generator F^{reg}, regressor G^{reg} and adversarial regressor $G_{\text{adv}}^{\text{reg}}$. $G_{\text{adv}}^{\text{reg}}$ learns to maximize the target disparity by moving two predicted boxes far from each other while F^{reg} learns to minimize the target disparity by making two predicted boxes overlap as much as possible.

D-adapt(ICLR 2022)

$$\min_{F^{\text{reg}},G^{\text{reg}}} \mathcal{L}_s^{\text{reg}} = \mathbb{E}_{(\mathbf{x}_s,\mathbf{y}_s^{\text{gt}},\mathbf{b}_s^{\text{gt}},\mathbf{b}_s^{\text{det}})\sim\mathcal{D}_s^{\text{fg}}} \sum_{i\in\{x,y,w,h\}} \text{smooth}_{L_1}(t_i^u - v_i)$$

where $t = G^{\text{reg}} \circ F^{\text{reg}}(\mathbf{x}_s)$ is the regression prediction, $u = \mathbf{y}_s^{\text{gt}}$ is ground truth category, v is the ground truth bounding box offsets calculated from $\mathbf{b}_s^{\text{gt}}$ and $\mathbf{b}_s^{\text{det}}$

$$\begin{aligned}\max_{G_{\text{adv}}^{\text{reg}}} \mathcal{L}_{\text{adv}}^{\text{reg}} =& \mathbb{E}_{(\mathbf{x}_t,\mathbf{y}_t^{\text{cls}})\sim\mathcal{D}_t^{\text{fg}}} \text{smooth}_{L_1}(G_{\text{adv}}^{\text{reg}} \circ F^{\text{reg}}(\mathbf{x}_t)^{\mathbf{y}_t^{\text{cls}}}, G^{\text{reg}} \circ F^{\text{reg}}(\mathbf{x}_t)^{\mathbf{y}_t^{\text{cls}}}) \\ & - \mathbb{E}_{(\mathbf{x}_s,\mathbf{y}_s^{\text{gt}})\sim\mathcal{D}_s^{\text{fg}}} \text{smooth}_{L_1}(G_{\text{adv}}^{\text{reg}} \circ F^{\text{reg}}(\mathbf{x}_s)^{\mathbf{y}_s^{\text{gt}}}, G^{\text{reg}} \circ F^{\text{reg}}(\mathbf{x}_s)^{\mathbf{y}_s^{\text{gt}}}).\end{aligned}$$

$$\min_{F^{\text{reg}}} \mathcal{L}_s^{\text{reg}} + \eta\mathcal{L}_{\text{adv}}^{\text{reg}}$$

D-adapt(ICLR 2022)

Stage 1: Source-domain pre-training

Stage 2: Category adaptation

Stage 3: Bounding box adaptation

Stage 4: Target-domain pseudo-label training

Stage 5: Repeat steps 2 to 4 T times

D-adapt(ICLR 2022)

Table 1: Results from PASCAL VOC to Clipart (ResNet101).

	aero	bcycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	hrs	bike	prsn	plnt	sheep	sofa	train	tv	mAP
Source Only	35.6	52.5	24.3	23.0	20.0	43.9	32.8	10.7	30.6	11.7	13.8	6.0	36.8	45.9	48.7	41.9	16.5	7.3	22.9	32.0	27.8
DA-Faster [8]	15.0	34.6	12.4	11.9	19.8	21.1	23.2	3.1	22.1	26.3	10.6	10.0	19.6	39.4	34.6	29.3	1.0	17.1	19.7	24.8	19.8
BDC-Faster [43]	20.2	46.4	20.4	19.3	18.7	41.3	26.5	6.4	33.2	11.7	26.0	1.7	36.6	41.5	37.7	44.5	10.6	20.4	33.3	15.5	25.6
WST-BSR [27]	28.0	64.5	23.9	19.0	21.9	64.3	43.5	16.4	42.0	25.9	30.5	7.9	25.5	67.6	54.5	36.4	10.3	31.2	57.4	43.5	35.7
SWDA [43]	26.2	48.5	32.6	33.7	38.5	54.3	37.1	18.6	34.8	58.3	17.0	12.5	33.8	65.5	61.6	52.0	9.3	24.9	54.1	49.1	38.1
MAF [18]	38.1	61.1	25.8	**43.9**	40.3	41.6	40.3	9.2	37.1	48.4	24.2	13.4	36.4	52.7	57.0	**52.5**	18.2	24.3	32.9	39.3	36.8
SCL [45]	44.7	50.0	33.6	27.4	42.2	55.6	38.3	19.2	37.9	**69.0**	30.1	**26.3**	34.4	67.3	61.0	47.9	21.4	26.3	50.1	47.3	41.5
CRDA [53]	28.7	55.3	31.8	26.0	40.1	63.6	36.6	9.4	38.7	49.3	17.6	14.1	33.3	74.3	61.3	46.3	22.3	24.3	49.1	44.3	38.3
HTCN [5]	33.6	58.9	34.0	23.4	**45.6**	57.0	39.8	12.0	39.7	51.3	21.1	20.1	39.1	72.8	63.0	43.1	19.3	30.1	50.2	51.8	40.3
ATF [19]	41.9	**67.0**	27.4	36.4	41.0	48.5	42.0	13.1	39.2	75.1	33.4	7.9	41.2	56.2	61.4	50.6	**42.0**	25.0	53.1	39.1	42.1
Unbiased [35]	30.9	51.8	27.2	28.0	31.4	59.0	34.2	10.0	35.1	19.6	15.8	9.3	41.6	54.4	52.6	40.3	22.7	28.8	37.8	41.4	33.6
D-adapt	**56.4**	63.2	**42.3**	40.9	45.3	**77.0**	**48.7**	**25.4**	**44.3**	58.4	**31.4**	24.5	**47.1**	**75.3**	**69.3**	43.5	27.9	**34.1**	**60.7**	**64.0**	**49.0**

D-adapt(ICLR 2022)

南京航空航天大学 Nanjing University of Aeronautics and Astronautics

Adaptation between dissimilar domains

Table 2: Results from VOC to Comic (ResNet-101). Oracle results are obtained by training on labeled data in the target domain.

Method	bike	bird	car	cat	dog	prsn	mAP
Source Only	32.5	12.0	21.1	10.4	12.4	29.9	19.7
DA-Faster [8]	31.1	10.3	15.5	12.4	19.3	39.0	21.2
SWDA [43]	36.4	21.8	29.8	15.1	23.5	49.6	29.4
MCAR [58]	47.9	20.5	37.4	20.6	24.5	**53.6**	33.5
Instance Adapt	39.5	17.7	26.5	27.3	22.4	48.4	30.3
Global Adapt	31.9	15.7	30.3	21.3	17.1	37.9	25.7
D-adapt	**52.4**	**25.4**	**42.3**	**43.7**	**25.7**	53.5	**40.5**
Oracle	42.2	35.3	31.9	46.2	40.9	70.9	44.6

Adaptation from synthetic to real images

Table 3: Sim10k to Cityscapes.

Method	Backbone	AP on Car
Source Only	VGG-16	34.6
DA-Faster [8]		38.9
BDC-Faster [43]		31.8
SWDA [43]		40.1
MAF [18]		41.1
Selective DA [61]		43.0
CDN [48]		49.3
HTCN* [5]		42.5
CFFA [59]		43.8
ATF [19]		42.8
CADA [20]		49.0
MeGA-CDA [52]		44.8
UMT* [10]		43.1
D-adapt		**50.3**
Oracle		69.7
Source-only	ResNet101	41.8
CADA [20]		51.2
D-adapt		**51.9**
Oracle		70.4

Adaptation between similar domains

Table 4: Results from Cityscapes to Foggy Cityscapes.

Method	Backbone	prsn	rider	car	truck	bus	train	mcycle	bcycle	MAP
Source only	VGG-16	25.1	32.7	31.0	12.5	23.9	9.1	23.7	29.1	23.4
DA-Faster [8]		25.0	31.0	40.5	22.1	35.3	20.2	20.0	27.1	27.7
BDC-Faster [43]		26.4	37.2	42.4	21.2	29.2	12.3	22.6	28.9	27.5
SW-DA [43]		36.2	35.3	43.5	30.0	29.9	42.3	32.6	24.5	34.3
Selective DA [61]		33.5	38.0	48.5	26.5	39.0	23.3	28.0	33.6	33.8
DD-MRL* [28]		30.8	40.5	44.3	27.2	38.4	34.5	28.4	32.2	34.5
CADA [20]		41.9	38.7	56.7	22.6	41.5	26.8	24.6	35.5	36.0
CRDA [53]		32.9	43.8	49.2	27.2	45.1	36.4	30.3	34.6	37.4
CFFA [59]		34.0	46.9	52.1	30.8	43.2	29.9	34.7	37.4	38.6
ATF [19]		34.6	47.0	50.0	23.7	43.3	38.7	33.4	38.8	38.7
MCAR [58]		32.0	42.1	43.9	31.3	44.1	**43.4**	**37.4**	36.6	38.8
HTCN* [20]		33.2	47.5	47.9	**31.6**	**47.4**	40.9	32.3	37.1	39.8
D-adapt		43.1	51.8	58.1	26.3	36.8	14.6	32.2	42.0	38.1
D-adapt*		**44.9**	**54.2**	**61.7**	25.6	36.3	24.7	37.3	**46.1**	**41.3**
Oracle		47.4	40.8	66.8	27.2	48.2	32.4	31.2	38.3	41.5
Source-only	ResNet101	33.8	34.8	39.6	18.6	27.9	6.3	18.2	25.5	25.6
CADA [20]		41.5	43.6	57.1	29.4	44.9	39.7	29.0	36.1	40.2
D-adapt		**42.8**	**48.4**	56.8	31.5	42.8	37.4	**35.2**	42.4	42.2
D-adapt*		40.8	47.1	**57.5**	**33.5**	**46.9**	**41.4**	33.6	**43.0**	**43.0**
Oracle		44.7	43.9	64.7	31.5	48.8	44.0	31.0	36.7	43.2

D-adapt(ICLR 2022)

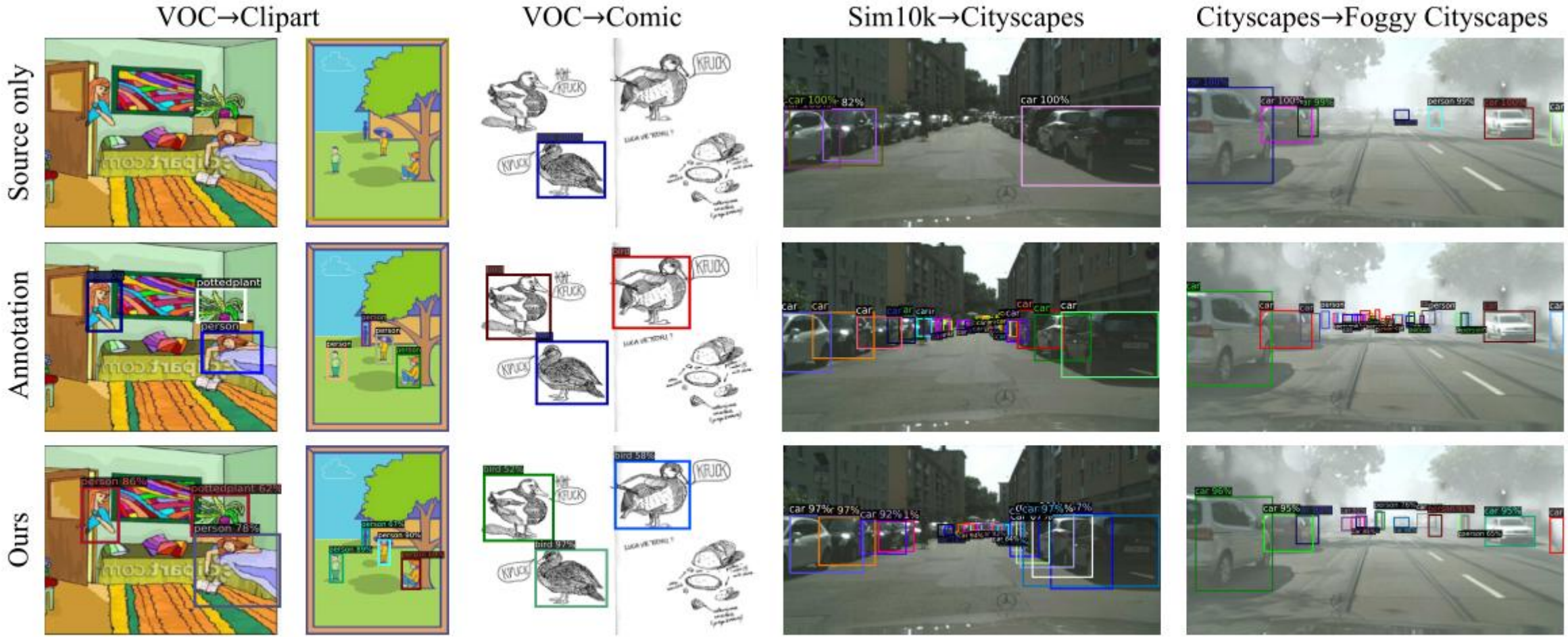


Figure 4: Qualitative results on the target domain.

D-adapt(ICLR 2022)

总结：

1. 不同任务合适的域自适应数据分布是存在差异的。此前的方法主要关注了物体检测中的分类性能，而往往忽视了物体检测作为一个多任务学习问题所面临的挑战，D-adapt考虑了物体检测的回归适应。
2. 在实际部署中，可以通过使用更强大的适配器来进一步提高检测性能，而不会引入任何计算开销，因为适配器可以在推理过程中被移除（解耦）。

不足：

1. 相比于物体分类，物体边界框定位依然是一个还没有解决的非常完善的问题。
2. 物体检测中天然存在着类别上的长尾分布（Long Taild Distribution）问题，同时不同数据域上的类别分布还不一致，导致了对抗域自适应的鲁棒性变差。

Thanks