

# 技术报告

## 一、赛题分析 (Competition Analysis)

CVPR 2023 Art Of Robustness Workshop Challenge - Vehicle Detection in the Physical-World 旨在鼓励参赛者构建能够对抗噪声的鲁棒目标检测器。

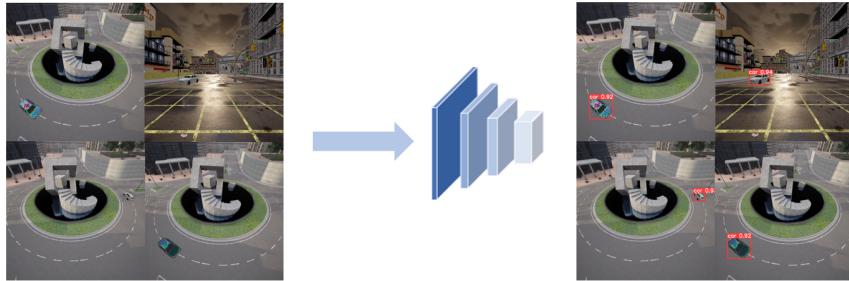


图 1 赛题任务图

### 1. 数据分析

初赛分为两个阶段，测试数据分两次释放，类似于 A/B 榜。本次比赛所有的数据集都是由 CARLA 仿真平台生成，初赛释放 600 张原始训练集用于构建样本及训练，见图 2 (a)；A 和 B 阶段分别释放了 2000 张测试集，其中包括正常样本和生成的伪装样本（对抗样本），见图 2 (b)。

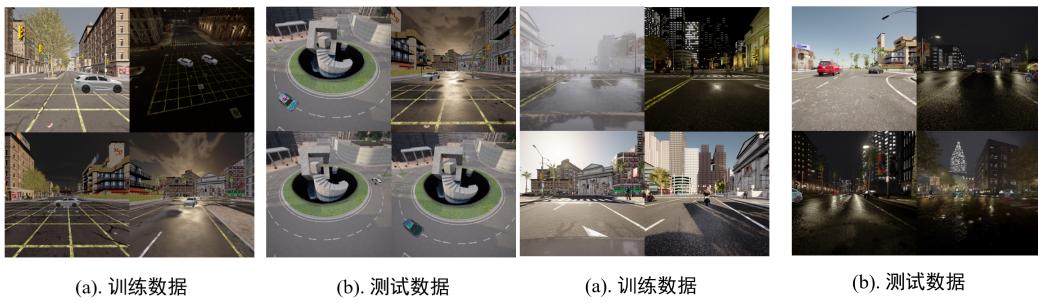


图 2 初赛数据图

(a). 训练数据 (b). 测试数据

(a). 训练数据 (b). 测试数据

图 3 复赛数据图

复赛阶段由于赛题变成鲁棒性 (Pedestrian, Vehicle, Bicycles) PVB 检测，释放的训练集中仅有 200 张图片，见图 3(a)；测试集比赛期间不释放，评估过程需提交模型后台测试后反馈。比赛结束后观察测试集情况，其数据类型包括伪装的四轮车和贴对抗补丁的四轮车、两轮车及行人，补丁类型多种多样，包括卡通图片、对抗补丁等等，并调整了补丁的透明度，见图 3(b)。

### 2. 赛题难点及突破点

**难点:** 1. 初赛阶段官方禁止使用测试集及伪标签训练，所以需要自行制作对抗伪装的车辆样本，需要复现论文以及配置渲染方法库（自测配这个渲染库还是费点事的）；2. 目标检测对抗样本生成方法参数的设置（不同 loss 权重会生成的不同攻击意义的样本），官方的配置在随机 texture 的情况下是很难达到伪装效果的，只有在使用官方制作的 texture 条件下有效；3. 复赛阶段测试集及攻击方法不可见，样本随机性较高，数据分布不一致，线上线下分说无法尽量同步。

**突破点:** 1. 初赛阶段主要是从数据层面操作，本文复现了 FCA[1], DualAttentionAttack[2] 中攻击方法，并在对抗生成 loss 上做了针对 obj loss 的优化，生成对抗伪装的车辆图片，白盒场景下伪装成功率 90%+以上。虽然生成车辆的纹理颜色与测试集不太一样，但是大致攻

击原理应该是一样的，通过添加对抗样本执行对抗训练，使得模型更鲁棒，线上得分提高。（初赛阶段官方是无法验证参赛队伍是否使用的了测试集训练，排行榜分数都基本满分，大胆猜测一下应该还是有人用了测试集的）2. 复赛阶段，主要从数据和模型优化，数据层面基于官方数据集构造各种各样的对抗补丁样本，模型层面通过一些平滑约束优化使得训练的模型更平滑、更鲁棒。

## 二、解决方案 (Approach)

### 1. 总体方案框架

我们的方案在 Phase I 和 Phase II 阶段大致相同，数据上基于训练集生成对抗样本，丰富训练样本多样性；模型上使用 Yolov5[3]（无调整）；训练策略上，初赛阶段除了调整的数据增强操作，其他与官方训练、推理策略一样。复赛阶段引入 SAM 优化器[4]，通过近似优化模型 Hessian Matrix（需更高阶导数）来优化模型平滑性，使得模型权重到达最优点且所达到的最优点附近是平滑的，同时推理过程采用多尺度推理。方法框架如图所示：

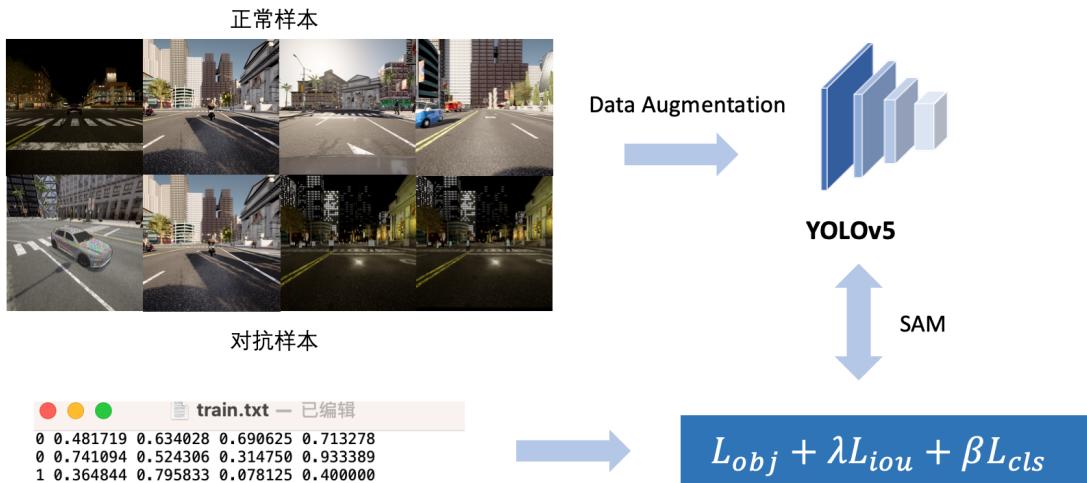


图 4 算法框架图

- 构建数据集：我们首先在 Phase I 第一阶段的训练集上训练正常的 YOLOv5l 作为白盒模型，通过 FCA[1] 攻击算法并使用 DualAttentionAttack[2] 中开源的数据集及其相关的 texture 文件生成与训练集同源的数据的对抗样本，其生成图像大小与训练集图像相同 (800, 800, 3)。

在 Phase II 阶段，由于赛题有变化，测试集中不仅 Texture-based 的对抗样本还有 Patch-based 的。我们基于 Phase II 阶段释放的训练集生成 Patch-based 对抗样本，补丁的位置我们从 Label 文件中可以获得，生成的方法将一阶段的纹理 tensor 换成补丁 tensor 即可。但 Mask 位置仅选取中心周围部分区域，patch 大小  $w = scale * W$ ,  $h = scale * H$ 。共生成了  $scale = 0.6$  样本 200 张、 $scale = 0.4$  样本 200 张， $scale = 0.4 \& smooth$  的样本 200 张，Gaussian Noise 的样本 200 张，黑白补丁样本 200 张。训练集和验证集的比例为 4:1，最优训练数据组成方式见消融实验部分。

- 模型结构：考虑到官方模型提交有参数大小和 flops 限制，我们最终选择 YOLOv5l 网络结构，由于没有调整模型结构、参数，遂本文不会对模型结构做相应介绍，有需要的请查看 Yolov5 官网[4,5]。

● 训练：图像预处理，通过 albumentations 库实现 ResizeCrop、flip、rotation、HueSaturationValue、BrightnessContrast、CLAHE、RandomGamma、GaussNoise、GaussBlur、JpegCompress、CutOut 数据增强等。初赛阶段损失函数和优化器等都没有调整，仍采用开源配置。复赛阶段我们引入 SAM[4] 优化器，采用 SGD 基优化器。初复赛基本参数 lr = 0.01, Image-size=800, batch-size=16, epoch=30，其余参数见代码中配置文件 hyp.scratch-low.yaml。

## 2. 实现细节

复现流程：

**1. 数据准备.** 提交的文件中没有训练样本，若要训练，需将的训练集放入 datasets/phase1(2)/train/images 目录下，标签放入 datasets/phase1(2)/train/labels 目录下。Phase I 阶段 test1 图片放入 dataset/phase1/test/目录，test2 图片放入 dataset/ phase1/test2/目录。

**2. 生成标签.** 由于官方数据集提供的标签不适用于 yolov5 训练，需要生成对应标签，同时生成 train.txt 和 val.txt，代码见 prepare\_format.py 文件。

**3. 生成样本.** 我们在 FCA 开源代码基础上修改，生成对抗样本（对抗纹理、对抗补丁），生成的样本也放入 train 目录下，注意命名（需要与原样本不重名，且对应 label）。Phase I 阶段只有对抗纹理样本，Phase II 阶段有对抗补丁样本。这里暂不提交，有需要请官方联系我。

**4. 修改配置文件.** 训练之前需要修改数据配置文件 data/cvprW.yaml 或 cvprW2.yaml 中训练数据路径，cvprW2.yaml 是 Phase II 阶段配置文件。Phase I 阶段指定 path = datasets/phase1/train，Phase II 阶段修改为 path = datasets/phase2/train。

**5. 训练.** 执行 sh train.sh，可修改 GPU 块号，训练结束后在 run/phase1 (2) /train/exp 目录下查看训练权重与日志文件。两阶段训练指令不同，需注释后再运行。

**参数设置:** 部分参数设置已在训练方法中说明，其他见代码中 data 目录下配置文件以及训练命令指定参数。

**环境配置:** 其余 Python 依赖包见 requirements.txt。

操作系统	Ubuntu 20.04.2 LTS
操作系统内核	5.4.0-110-generic x86_64
Python	3.8.5
GPU	GeForce RTX 3090

## 三、 消融实验 (Ablation Study)

### 1. 初赛阶段

初赛阶段没有做太多尝试，在加入对抗样本后分数直接到了 99+，实验结果如表 1 所示：

表 1 初赛数据组成消融实验

Dataset	Local Metric				Online Score
	Precision	Recall	mAP_0.5	mAP_0.5:0.95	
Train600	0.99938	0.99320	0.99500	0.88700	0.92314
Train600+Adv500	0.99999	0.99971	0.99500	0.92900	<b>0.99912</b>

### 2. 复赛阶段

复赛阶段将从三方面做消融实验：训练数据组成，优化器参数，后处理阈值：

● 首先是针对训练数据集组成的消融实验，其中 T1-adv 表示 Phase I 训练集 100 张以及生成的对抗纹理样本 100 张，T2 表示 Phase II 训练集 200 张，P 表示由 T2 生成的对抗补丁样本 200 张，S 表示由 T2 生成的较小对抗补丁样本 200 张，SS 表示在较小补丁基础上加入平滑约束的样本 200 张，GS 表示高斯噪声补丁 200 张，BW 表示黑白补丁 200 张。不同数据集组成消融实验如表 2 所示，默认情况 epoch=30，batch-size=16，IOU\_th(I)=0.6，Score\_th(S)=0.1，默认不采用多尺度推理 (TTA)。

表 2 虽然算不上单一变量的消融实验，但也能显示出数据集组成、多尺度推理，阈值的对于结果的影响，我们在比赛过程中也是择优后进一步优化。

● 其次针对训练优化层面（模型平滑）的消融实验，是在上述最好 T2+P+S+SS+GS 基础上引入了 SAM 优化器，以 SGD 代表原最优版本。表 3 中实验通过调整 SAM 中 rho 参数，以及训练迭代 epoch 观察得分情况。表 3 中实验结果，epoch=30，S=0.01，I=0.6，默认采用 TTA。

表 2 复赛数据组成消融实验

Dataset	Local Metric				Online Score
	Precision	Recall	mAP_0.5	mAP_0.5:0.95	
T2	0.78823	0.75723	0.80698	0.40343	49.58
T2+P	0.91577	0.90640	0.93190	0.61188	63.48
T2+P+T1-adv	0.89904	0.86343	0.91261	0.62382	65.19
T2+P+T1-adv+S	0.96572	0.90677	0.97422	0.71123	69.05 (TTA)
T2+P+S+SS	0.95879	0.93489	0.97790	0.74847	73.92 (TTA)
T2+P+S+SS+T1-adv	0.97481	0.95823	0.98657	0.76574	74.51 (TTA, S=0.05)
<b>T2+P+S+SS+GS (Best)</b>	0.97004	0.96089	0.98585	0.79020	74.86 (TTA, S=0.05)
T2+P+S+SS+T1-adv+GS	0.97584	0.94904	0.98517	0.75485	74.06 (TTA, S=0.05)
T2+P+S+SS+GS+BW	0.97789	0.97213	0.98905	0.80895	72.01 (TTA, S=0.05)
P+S+SS+GS	0.98038	0.9889	0.99357	0.77359	71.84 (TTA, S=0.05)

表 3 优化器及参数消融实验

Optimizer	Local Metric				Online Score
	Precision	Recall	mAP_0.5	mAP_0.5:0.95	
SGD	0.97004	0.96089	0.98585	0.79020	75.61
SAM(rho=0.05)	0.95466	0.88983	0.96166	0.62116	75.76
SAM(rho=0.1)	0.92845	0.90915	0.95787	0.60649	75.81
SAM(rho=0.2)	0.90309	0.87614	0.93094	0.56678	76.87
<b>SAM(rho=0.2,epoch=50)</b>	0.91985	0.9052	0.96198	0.61691	<b>78.36</b>
SAM(rho=0.2,epoch=80)	0.94796	0.91100	0.97187	0.66850	77.71
SAM(rho=0.5,epoch=50)	0.94654	0.90184	0.96250	0.60218	78.28

● 最后是关于后处理阈值的消融实验，其实也不算是消融实验，仅在上述最优模型基础上修改推理时 IOU 和 Score 阈值，实验结果见表 4。标红部分是我们团队线上最高分数。

表 3 优化器及参数消融实验

	IOU-th	Score-th	Online Score
T2+P+S+SS+GS	0.01	0.6	78.36
SAM(rho=0.2,epoch=50)	0.001	0.6	78.5
	0.001	0.5	<b>78.65</b>

## 四、参考文献 (Reference)

- [1]. Wang D, Jiang T, Sun J, et al. Fca: Learning a 3d full-coverage vehicle camouflage for multi-view physical adversarial attack[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2022, 36(2): 2414-2422.
- [2]. Wang J, Liu A, Yin Z, et al. Dual attention suppression attack: Generate adversarial camouflage in physical world[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 8565-8574.
- [3]. <https://github.com/ultralytics/yolov5>.
- [4]. Foret P, Kleiner A, Mobahi H, et al. Sharpness-aware minimization for efficiently improving generalization[J]. arXiv preprint arXiv:2010.01412, 2020.