Yolov3 实验总结

邹睿石[1] 陈一鸣[1] 雒俊为[1] 何征昊[1] 刘毅[1] 姚钧辰[1] 贾俊杰[1]
（1.同济大学电子与信息工程学院计算机科学与技术系 上海 201804）

关键词： YoloV3 小样本训练 Darknet 百度飞桨

0 引言

作为本学期的最后一个实验，完成本实验的过程贯穿了整个学期。

首先，为了更好的理解“目标检测”问题，我们在学期初完成的综述论文便选择了“目标检测”作为探究议题。其中的第二部分，粗略的概述了二次、一次检测方法（R-CNN 系列、Yolo 系列）及解决深层次网络收敛问题的 ResNet 方法的特点、突破和局限。

其次，我们基于 Yolov3 作者 Joseph Redmon 所编写的 Darknet 深度学习框架进行了预实验，并从中获得了一定深度学习经验，包括和模型训练相关的训练集的标定、测试集的构建方法、关键参数的选定、优化器的选择，以及 AiStudio 平台、Notebook 及 Linux 终端使用的经验。

基于上述前期准备，我们在 Baidu Paddle 框架下实施本实验。解决了 Paddle 框架及 AiStudio 平台下的特定问题后，我们得到了基于自主采集的数据集的，能够具备一定检测准确率，针对车、行人和自行车目标的 Yolov3 模型。

1 Yolov3 模型简述

Yolo 代表着“You Only Look Once”，明确的指出了模型最大的特征。相比于 R-CNN 方法，Yolo 模型不需要提取 RoI 区域，而是通过将图片分割为 7*7 的区域进行特征预测。

Yolo 对 7*7 区域中的每一个区域进行类概率的预测，形成类概率图。继而，和生成的锚定框结合，形成最终预测。

2 Darknet 框架下的预实验及经验总结

为加深对于 Yolo 模型及其训练方法的理解，我们首先尝试利用 AiStudio 中的 GPU 资源，尝试实践以 Darknet 为框架的 Yolov3。由于本项目的文档不甚完善，目前属性被设置为私有，如需查看，可随时开放访问。

（https://aistudio.baidu.com/aistudio/projectdetail/2595118）

由于此部分实践仅仅是为正式实验做准备，因此很多设置并不完善。数据集方面，预实验中仅采用了 40 张标注图片进行训练；标注方面，没有采用作业要求的“人（person）、车（vehicle）、自行车（bicycle）”分类，而是采用了“行人（pedestrian）、骑车人（cyclist）、车辆（car）”的标注策略。因此，在预实验中有很多和正式实验不同的地方。

然而，正由于此，也给后续实验带来了宝贵经验。具体如下：

1. 小样本训练测试集的构造

对于仅有 40 张图片的情形，如果抽取其 10%的数据用于测试，则构成的测试集代表性较差，难以训练出有效的模型。在 Darknet 框架中运行，训练过程中会显示大量的-nan 提示，即 IoU 过低，模型难以收敛的情形。

为了解决这个问题，我们将数据既用于训练，又用于测试，即提取 100%的数据用于测试。这一方法，可以使 Darknet 模型生成一个在有限数据集上性能较好的模型（图 1，图 2 成功的案例）。

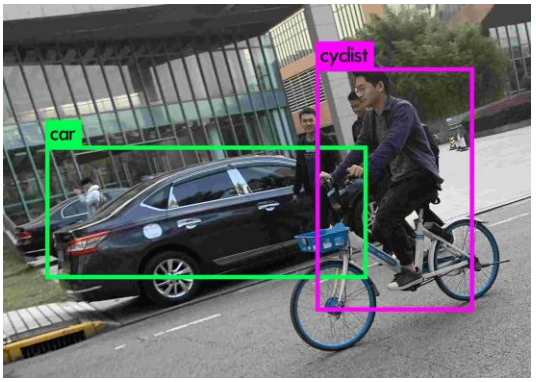

图 1 成功的案例

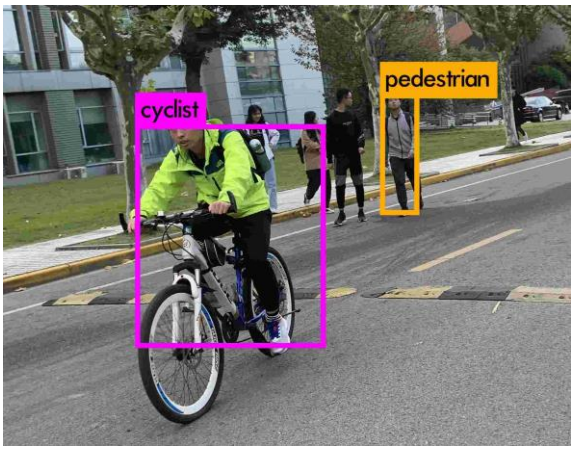

图 2　成功的案例

当然，这一设计也有其缺陷。由于训练集和测试集相同，模型很容易“过拟合”，即对于已有数据表现较好，而对新数据的特征表现较差的现象。其具体的实例，将在第 3 部分更详细的给出。

2.　标注策略设计

预实验中，设计的标注策略为“行人（pedestrian）、骑车人（cyclist）、车辆（car）”。根据实验结果进行分析，这一策略并不十分成功，极易出现“假阳性”的情况（图 3 消失的骑车人；图 4 行人骑车人混乱）。

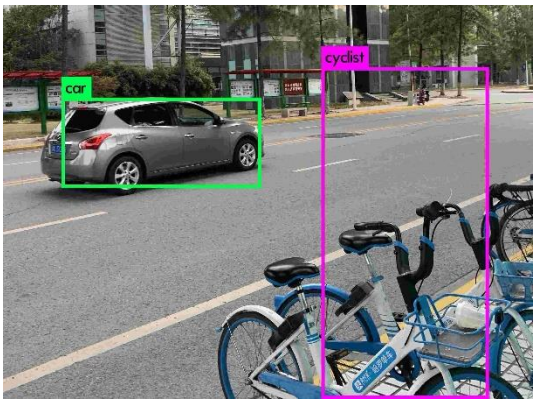

图 3　消失的骑车人

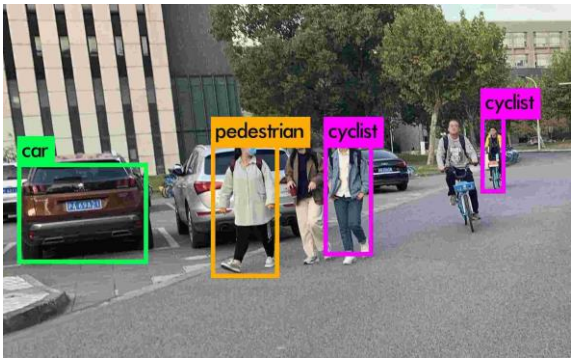

图 4　行人骑车人混乱

我们认为，这一现象的发生与模型本身的设计有关。“骑车人”可以在语义上分解为“自行车且行人”，一旦任一组成部分的置信度超过设定值，或两者置信度和预期相颠倒，便会出现“假阳性”的存在。

当然，需要说明的是，即使将语义分成“自行车”、“人”的方式，也可能存在“指鹿为马”的现象，但这种情况是模型优化不充分导致的，而非由于不好的定义所造成的。

（本环节由邹睿石完成）

3　Paddle 框架下实验筹备过程

1.　构建技术文档

根据 Darknet 的实践经验，撰写一份技术文档，要求组员以此为参考对数据进行采集和标注。技术文档截图如下（图 5 技术文档）：

Object Detection 数据采集和标注要求

1. **数据标注标准名称**
   bicycle
   person
   vehicle
2. **标注详情**
   bicycle：主要为有人骑行的自行车、摩托车（两轮），辅以不重叠的静止自行车；
   person：包括但不限于行人、骑车人；
   vehicle：“小轿车”，不要对中巴、大巴、卡车等车辆进行标注。
3. **分工、编号及数据要求**
   由于操作过程中需要严格的号码排序，本次采用的编码要求如下：
   将采集的数据通过 python 程序（老师的参考资料里有，也可以自己写），将图片转化为“四位编号”（0001-9999）；
   标注后，确保生成的.xml 文件的编号与相应图片编号匹配。

为了方便合并采集的数据，**编号区段采用分配制**。如果有组员希望投入更多精力，标注更多数据，请务必告知我，再分配未使用的区段：

雒俊为	0001-0200
陈一鸣	0201-0400
何征昊	0401-0600
刘毅	0601-0800
姚钧辰	0801-1000
贾俊杰	1001-1200
邹睿石	1201+

200 张图片中，应当有>30 张自己采集的数据，剩余可以选择从网络中查找或自己采集。采集的场景也不局限于校园内，如果有同学朋友能帮忙的照片也可以。

注意：每人分配了 200 张图片的编号，请务必**标注满 200 张图片!!!**

4. **标注方法及如何发送**
   通过老师给的 labelImg.exe 标注。要注意，此程序不需要安装，打开需要约 10s 时间；可以选择照片存储路径和.xml 文件存储路径，路径没有要求（通过 Open Dir 和 Change Save Dir 按钮实现）；程序需要在**全英文目录下运行**。

进入程序后，选择 Create RectBox 标注。第一次标注需要输入类别名称，请**直接复制本文档中的类别名**，防止拼写错误。通过 Next Image 和 Prev Image 翻页。

标注完毕后，不要忘记**点击 Save** 以保存在你指定的目录里。

标注完毕后，将**原图片文件**和**标注好的.xml 文件**分成**两个文件夹**，打包上传百度**网盘**（非会员上传速度不限），将链接发送到 QQ 群中。

提交时间为：2021.11.19 晚 23:59 前

图 5　技术文档

（本环节由邹睿石完成）

2.　采集、标注数据

小组共采集了 1200 张图片作为数据集，其内容包括手机拍摄采集结果、行车记录仪截图及网络图片。采集数据及标注结果如下（图 6 数据集）：

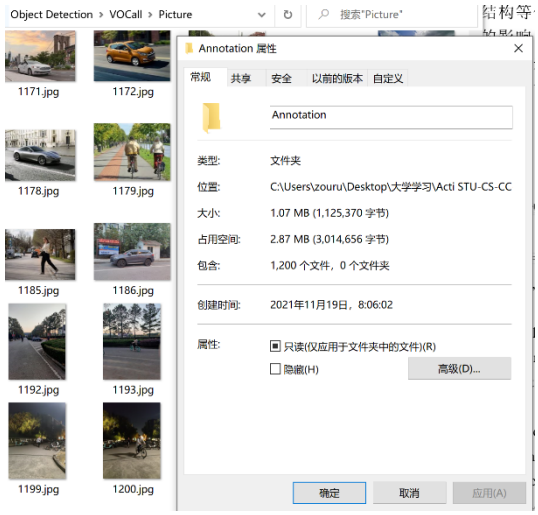

图 6　数据集

（本环节由全体成员完成）

3.　平台报错的解决

Invoke Operator xxx failed：此错误是由于模型参数改变，而选用了继续训练模式而导致的。解决方案是将 continue_train 设置为 False。

Cannot find lslm/xxx.jpg　xxx：此错误是由于“解析”数据集时的中断符号错误设置造成的。将隔断数据的方式设置为 Tab，再次导入，即可解决问题。

（本环节由邹睿石完成）

4.　缓慢的进度

将 1200 张图片的数据集导入模型后进行训练，效果并不理想（表 1 训练次数与时间和 Loss 的关系）。

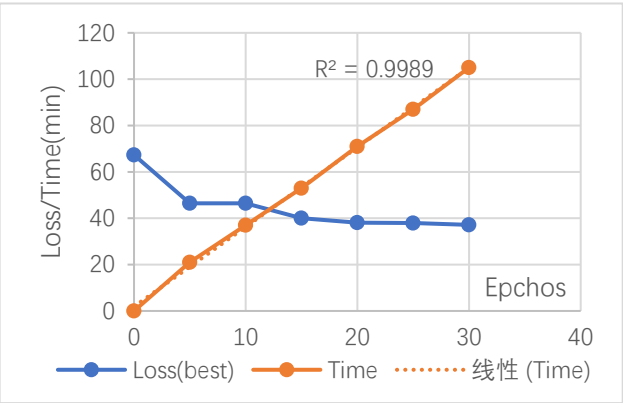

表 1 训练次数与时间和 Loss 的关系

首先，训练耗时过长。仅进行 30 次训练，在 GPU 环境下，耗时接近 2h。从 R^2 值可以看出，每一次训练的耗时基本相等，故并非因为网络等其它因素造成。

其次，loss 值难收敛。表 1 中的实例是在经过一定轮数的训练后继续训练 30 次得到的结果，loss 值再进一步训练 30 次后仍旧保持在 40 左右。根据 Darknet 训练的经验，这一数值下并不能产生有效的预测结果（图 7 高 loss 值(>20)情况下的预测）。

图 7 高 loss 值(>20)情况下的预测

虽然训练效率低下，结果不甚理想，但至此，基于 Paddle 框架的 Yolov3 已然可以成功运行，并完成了对于数据的标注及数据集的替换。

（本环节由全体成员完成）

4　Paddle 框架 Yolov3 模型及参数优化

1.　解决训练耗时过长的问题

由于神经网络的训练对算力需求很高，我首先考虑到耗时问题可能是 GPU 未参与计算导致的。

为验证这一猜想，我在终端使用“watch -n 0.1 -d nvidia-smi”指令检测每 0.1s 的 GPU 的使用情况。我发现，在运行 1200 张图片数据集时，在几乎所有时间内，GPU 均没有被调用。由此判断，训练耗时长并非是 GPU 未被调用导致的，而是 AiStudio 平台羸弱的文件 IO 能力阻塞了 GPU 性能的发挥。

为解决这一问题，我尝试减少训练集的图片数量，并减小 batch 的大小。然而，这一举措并未显著减少训练的耗时。

我又尝试减小图片占用空间。通过 Caesium 免费图片压缩软件，将数据集中的图片压缩至 20%大小。然而，这也没能显著降低训练的耗时。

为保持一定的数据量，同时加速训练的进程，在不得已的条件下，只能通过降低分辨率的方式实现。为了尽可能保留图像的信息，我们选择了 800*600 作为图片的分辨率。经实验，这一方案可以大幅加速训练的速度。

同时，在进行这一操作时，我们意外发现了数据集训练过程中难收敛的原因。我们所采集的数据集中，有大量数据的分辨率小于设定的800*600，图片非常模糊。模糊的图片将影响图片特征的提取，影响其正确性的检验。

当然，需要说明的是，在 Darknet 框架下训练时，我并没有对图像进行降低分辨率处理，而其效率仍旧可观。在约 1.5h 的训练后，即可达到前面展示的水平。其中的具体原因，有待进一步探讨。

2. 模型优化：梯度下降优化器的选择

原模型使用的是 SGD 优化器。在尝试通过压缩图片加速进程的过程中，我使用了约 30 张图片的数据集，并仅标记了“车”类。在以上实验条件下，记录数据（表 2 SGD 优化器）。

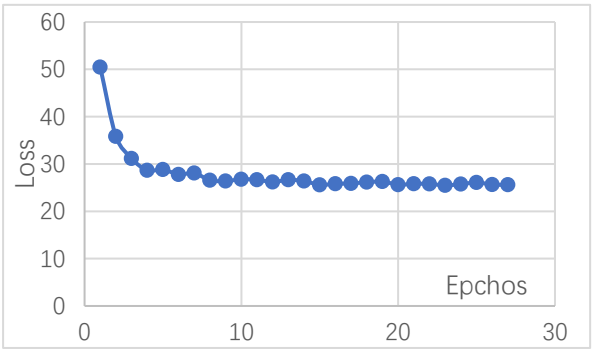

表 2 SGD 优化器（单一类标，压缩）

可以发现，在经过 10 次训练后，Loss 下降非常缓慢。这一现象，在 Darknet 框架训练过程中并没有显著体现。为了进一步加速训练的进程，我将优化器替换为 Darknet 框架中的 Momentum，并使用了与 Darknet 框架训练相同的训练集进行尝试。（表 3 Momentum 优化器）

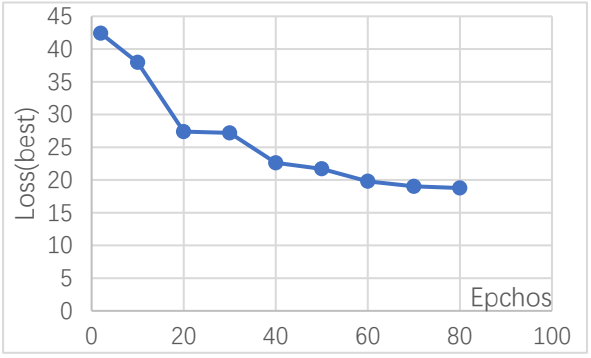

表 3 Momentum 优化器（Darknet 标注，低分辨率）

改用 Momentum 后效果并不显著。进一步查询资料，我采取 AdaDelta 优化器实施训练。AdaDelta 具备超参数自动调整的效果，可以一定程度避免数据陷入“鞍点”。（表 4 AdaDelta 优化器）

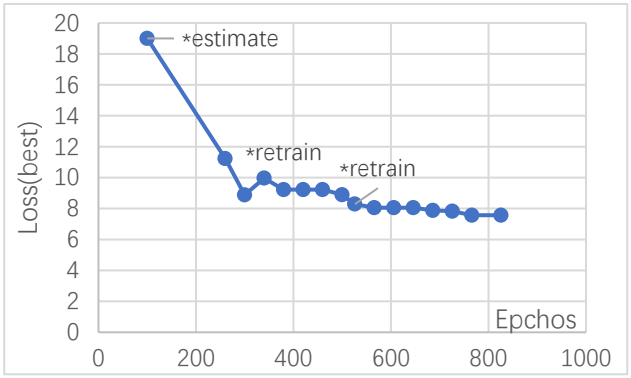

表 4 AdaDelta 优化器（正常标注，低分辨率）

由于彼时对平台的操作尚不熟练，前 260 次训练数据出现遗失。由于本次数据集经过了降低分辨率的处理，训练速度较快，因此训练了超过 800 次。实践证明，数据收敛慢可能更多是由于训练次数过少导致的。在训练截止时，最佳 Loss 约为 7.4，将此结果固化为最终模型。

3. 参数标定

锚点的选择：更改为与 Yolov3 相同的通用参数；

```
"yolo_cfg": {
"input_size": [3, 448, 448],
"anchors": [10, 13, 16, 30, 33, 23, 30, 61, 62,
45, 59, 119, 116, 90, 156, 198, 373, 326],
"anchor_mask": [[6, 7, 8], [3, 4, 5], [0, 1, 2]]
}
```

有效、忽略阈值和非极大值抑制：适当增大有效阈值、提高合并效果；

```
"ignore_thresh": 0.7,
"valid_thresh": 0.2,
"nms_thresh": 0.2,
```

梯度下降优化器：三种设置

```
"sgd_strategy": {
"learning_rate": 0.001,
"lr_epochs": [10, 18, 26, 40],
"lr_decay": [0.2, 0.1, 0.05, 0.01, 0.001]
},
"momentem_strategy":{
"learning_rate": 0.002,
"momentum": 0.9
},
"adadelta_strategy":{
"learning_rate": 0.001,
"epsilon": 1.0e-6,
"rho": 0.95
}
```

5 模型预测试

共设置四种测试工况：

单一目标（图 8 图 9 图 10）；

平凡情况（图 11 图 12）

复杂情况（图 13 训练集 图 14 测试集）；

扩展情况（图 15）；

单一目标，意图检验在简单环境下能否正确

识别车、自行车、人，经测试性能良好；

平凡工况下表现正常，但也有特殊情况。如图 12，在训练集中，我们规定不标注面包车，但模型中仍能将其识别出来；

复杂工况通过对比呈现，可见模型对于复杂工况有一定处理能力，但并不理想；

扩展情况选用了某艺术展的室内环境，可见本模型泛化能力较差。这可能是由于训练集和测试集相同导致的。

图 8 自行车

图 9 车辆

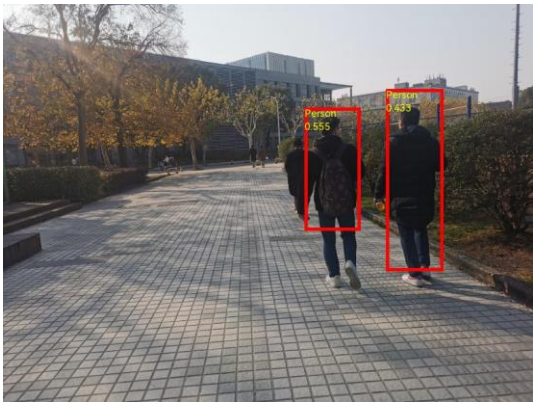

图 10 行人

图 11 平凡情况 1

图 12 平凡情况 2

图 13 复杂情况（训练集）

图 14 复杂情况（测试集）

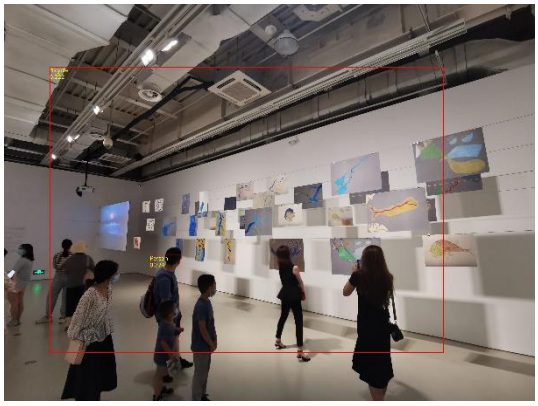

图 15 扩展情况

6 增强模型泛化能力

基于上述经验，我们重新筛选图片，将图片压缩至 448*448，重新编写了标注要求，并制作了数据集。新的数据集中共有 781 个项目。

同时，为进一步扩大数据量，我们和 Group14 进行了数据集的合并，合并后共有 1296 张图片。其中，包含 936 个汽车目标，873 个自行车目标及 1393 个行人目标。

在这次训练中，不再采用相同的数据集、训练集进行训练。通过随机数生成器，共产生 267 张图片用于评估模型。这 267 张图片并未从训练中剔除。测试集中包含 162 个汽车目标，186 个自行车目标及 320 个行人目标。

最终，经过约 280 次训练，总耗时约 8.5h，完成 Yolov3 新模型的训练。固化预测模型时，Loss 值约为 8（表 5）。

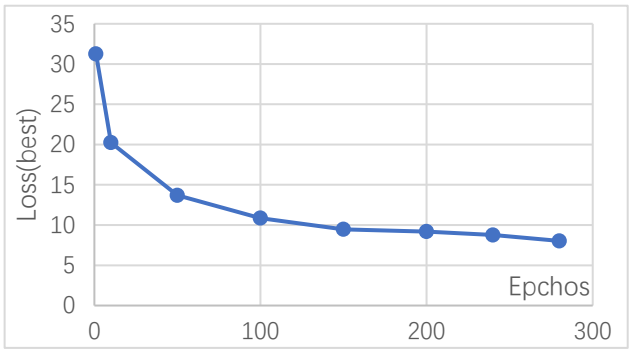

表 5 Adadelta 优化器大数据集 Loss

在发布的挑战集上评估模型的效果。作为对比，我们使用小样本训练出来的模型作为参照。

需要说明的是，总目标量是在我们的标注标准下给出的，和实际目标量有一定差距。除此以外，True Positive 代表正确识别的结果，Failed Cases 代表未识别出目标，False Positive 则代表识别出物体，但不属此类。

结果如下（表 6 小数据集 表 7 大数据集）：

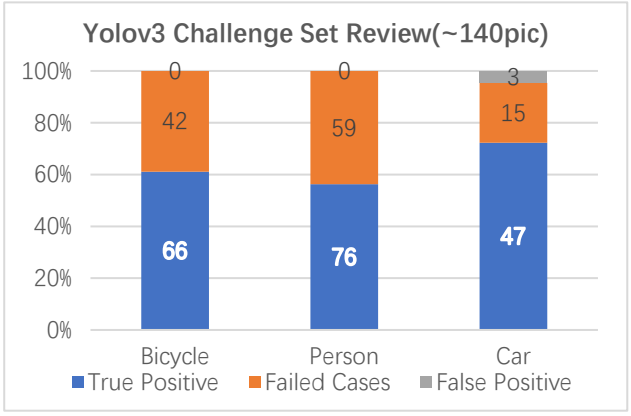

表 6 小数据集

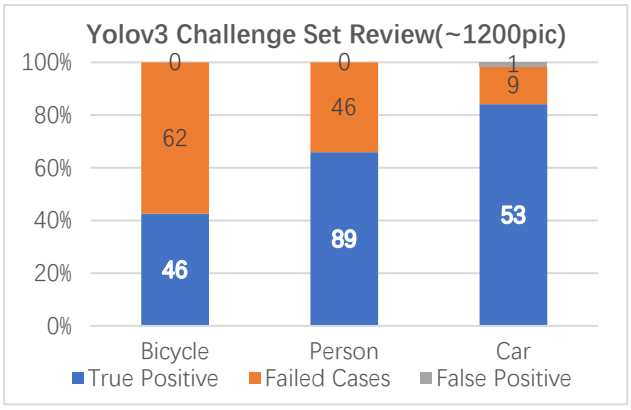

表 7 大数据集

可以看出，在 Person 和 Car 两类中，正确识别的概率均有了一定提升，且错误识别的数量也减少了。然而，在 Bicycle 类中，新模型甚至不如老模型效果好。

究其原因，我们认为，由于老模型的测试集和训练集是一一对应的，且挑战集的环境和训练集的环境类似，因此并没有出现显著的泛化难题。除此以外，老模型测试集中，有 120 个汽车目标，127 个自行车目标，158 个行人目标，在测试集上图片数量与新模型相当，由于是经过精心筛选的 120 张图片，其防止过拟合的性能可能更好，最终导致了这个现象的出现。除此以外，也可能是由于训练次数仍然较少导致的。

虽然模型性能各有优劣，但新模型在绘制预测框上的精准度比老模型有显著的提升。由于时间缘故，我们本次并没有计算 mAP。但是，这种提升仍可以在平台提交结果 result 文件夹中预测效果看出。