计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2019,28(10):8−14 [doi: 10.15888/j.cnki.csa.007101]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

目标检测数据集半自动生成技术研究①

孙晓璇[1,2], 张　磊[1], 李　健[1]

[1](中国科学院 计算机网络信息中心, 北京 100190)
[2](中国科学院大学 计算机与控制学院, 北京 100049)
通讯作者: 孙晓璇, E-mail: sunxiaoxuan@cnic.cn

摘　要: 目标检测广泛使用于计算机视觉领域. 在不同的场景中, 我们需要使用不同的数据集训练模型. 但是, 人工生成数据集标签非常耗时. 本文提出一种半自动的方法生成数据集标签, 然后按照图像相似度设置的阈值自动筛选, 最后保留符合要求的图像和对应的标签作为最终的数据集. 实验表明, 该方法可以提高数据集生成标签的速度, 同时确保了准确率.

关键词: YOLOv3; SSD; 差异值哈希; 半自动生成训练集

引用格式: 孙晓璇,张磊,李健.目标检测数据集半自动生成技术研究.计算机系统应用,2019,28(10):8–14. http://www.c-s-a.org.cn/1003-3254/7101.html

Research on Semi-Automatic Generation Technology of Object Detection Datasets

SUN Xiao-Xuan[1,2], ZHANG Lei[1], LI Jian[1]

[1](Computer Network Information Center, Chinese Academy of Sciences, Beijing 100190, China)
[2](School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: Object detection is widely used in the field of computer vision. In different occasions, we need to use different training set to train the model. However, manually generating label is very time consuming. This study proposed a semi-automatic method to generate labels for dataset, then automatically filter them according to the threshold set by image similarity, lastly retain the required images and corresponding labels as the final dataset. Experiments show that the method can both improve the speed and ensure accuracy rate of generating labels for dataset.

Key words: YOLOv3; SSD; difference Hash; semi-auto produce training set

随着科技的进步与发展, 计算机视觉迅速发展. 其中, 目标检测是计算机视觉领域的一个重要分支, 它的主要任务是找出图像中的目标对象的位置信息, 判断目标框中对象的类别, 并为检测到的所有目标在图像上画出预测边框. 随着深度学习相关技术的快速发展, 计算机视觉领域也不断取得突破[1]. 基于深度学习的目标检测算法主要分为两类: 一类是 RCNN 系列, 如 FastRCNN[2]、FasterRCNN 等 two-stage 方式, 另一类是 YOLO[3]、SSD[4]等 end-to-end 的 one-stage 方式. 与传统方法[5–9]相比, 使用深度学习算法进行目标检测不仅减少误检、漏检, 还更好的适应复杂场景, 在检测速度方面也有较大的提升. 针对 one-stage 方式的算法, 如果我们需要训练目标检测模型但是没有现成的数据集, 传统的做法是使用 LabelImg 等软件人工标注图像[10]中目标对象的位置, 并标注对象的类别. 对于数据集中大量的图像, 制作、处理数据集是繁琐耗时的工作. 除了传统人工标注的做法外, 还可以使用 OpenCV 阈值过滤的方法[11], 通过提取指定灰度值范围内的部分获

① 基金项目: 中国科学院科技服务网络计划 (KFJ-STS-QYZD-058)
Foundation item: Science and Technology Service Network Plan of Chinese Academy of Sciences (KFJ-STS-QYZD-058)
收稿时间: 2019-03-25; 修改时间: 2019-04-18; 采用时间: 2019-04-19; csa 在线出版时间: 2019-10-15

得目标位置, 但是这种方式对图像的要求比较高, 如果图像中的目标对象与背景颜色相近或者背景比较复杂的时候, 就难以提取出图像中的对象.

基于以上原因, 本文提出一种方法, 通过使用官方提供的已训练的目标检测模型, 对输入图像批量检测, 实现自动定位图像中的目标物体, 记录物体坐标位置、类别信息作为标签, 以此制作自己的数据集. 以 YOLOv3 算法和 SSD 算法为例, 使用已训练的目标检测模型, 对数据集中的图像进行初步的目标检测, 记录检测目标在图像中的位置信息, 数据集中图像的类别是已知的, 此时可以为数据集中的图像生成相应的标签. 这种方法可以帮助我们高效地生成模型训练所需要的数据集, 但是由于已训练模型的数据集与自己的数据集不同, 导致检测目标定位不够准确, 所以生成的数据集质量不可预估. 为了提高数据集质量, 本文使用多种不同的深度学习目标检测算法对自己的数据集进行目标检测生成标签, 并对检测结果图像进行裁剪、缩放操作, 再进行图像相似度的对比, 实现高效生成标签以及筛选符合预期效果图像的功能.

1 算法介绍

1.1 YOLOv3 算法

YOLOv3 是一个融合多种先进算法思想构成的目标检测网络, 通过在 COCO 和 ImageNet 数据集上联合训练, 得到能够检测出 80 种类别物体的模型. YOLOv3 采用 darknet-19 与 ResNet[12]中残差思想融合的 darknet-53 为网络主干, 网络兼具 ResNet 的较高准确率和 YOLO 系列检测速度快的特点, 同时还避免了网络深度过深导致梯度消失. 此外, YOLOv3 可以在 DarkNet、OpenCV、TensorFlow 等多种框架中实现. 本论文中使用基于 DarkNet 实现的 YOLOv3 算法, 使用 Python 和 C++两种语言完成实验.

YOLOv3 运用了类似 FPN 的思想, 将原始图像缩放到 416×416 大小, 然后分别在 13×13、26×26、52×52 这 3 种尺寸的特征图上做目标检测, 最后融合 3 个检测结果, 这样使得 YOLOv3 在小目标的检测效果上有所提升. YOLOv3 的每一个网格单元使用 3 个 anchor, anchor 所在网格在图像中的相对位置为 (c_x, c_y), 每个 anchor 预测 3 个边界框, 边界框的长和宽分别为 p_w 和 p_h. 每个网格单元的每个边界框产生 4 个位置预测值 (t_x, t_y, t_w, t_h) 和 1 个置信度. σ 指 Sigmod 函数.

$$\begin{cases} b_x = \sigma(t_x) + c_x \\ b_y = \sigma(t_y) + c_y \\ b_w = p_w e^{tw} \\ b_h = p_h e^{th} \end{cases}$$

YOLOv3, 使用多个独立的 logistics 代替 Softmax 分类器, 这样使每个预测框可以单独分类, 即使是重叠物体, 模型也可以检测出来.

1.2 SSD 算法

SSD 算法与 YOLO 系列算法相同都是 one-stage 的算法, 都具有检测速度快的特点. SSD 算法融合了 Fast-RCNN 的 anchor 思想, 同时对每个 default box 还生成不同横纵比的候选框, 这样减小了漏检的可能. SSD 算法采用预训练模型 VGG16 和额外的卷积层, 实现在不同层次的特征图中进行目标检测, 相当于在不同尺寸的特征图上多次执行目标检测, 高层特征图检测大物体, 低层特征图检测小物体, 最后综合各层次的检测结果, 可以更准确的识别图像中的目标物体. SSD 算法在不同层次的特征图上使用更小的卷积核预测, 在许多不同横纵比的 defaultbox 中, 选择出满足条件的一些 prior box 与 ground truth box 匹配, 最终得到预测框, 具有比 YOLOv3 算法更高的精度. 但是, SSD 和 YOLOv3 与多数目标检测算法相同, 对小尺寸物体检测效果不太好. 对于 prior box 等参数需要提前人工设置, 而且参数的设置对检测的结果会产生影响.

2 数据集介绍

2.1 COCO 数据集

COCO[13]是微软团队提供的一个可以用来进行图像识别的数据集. COCO 中有 80 个对象类别和各种场景类型的图像, 现在有 3 种标注类型: 目标实例、目标上的关键点、图像文字描述, 可以在目标检测、实例分割、目标关键点追踪等任务中使用.

2.2 ImageNet 数据集

ImageNet[14]数据集是用于计算机视觉领域的大型图像数据库, 数据集有 1400 多万张图图像, 涵盖 2 万多个类别, 超过百万的图像有明确的类别标注和图像中物体位置的标注, 可以在图像分类 (CLS)、目标定位 (LOC)、目标检测 (DET)、视频目标检测 (VID) 等任务中使用.

2.3 ILSVRC 数据集

ILSVRC[14]数据集是 ImageNet 数据集的子集,

ILSVRCCLS-LOC 是 ImageNet 中图像分类和目标定位的数据集子集, 其中包括 1000 个对象类别.

2.4 实验数据集

实验数据集中只包含图像, 这些图像均来自网络及实际工作过程中长期积累的图像数据.

2.5 数据集格式

YOLOv3 的数据集包括两部分: 标签 labels 和图像 images, 其中, labels 文件夹中每一个文本文件中存放的是图像中目标对象的坐标位置信息, 每张图像对应一个 txt 标签 (目标定位) 文件, 要求图像名称与目标定位的 txt 标签文件名称必须相同, 并且 labels 文件夹和 images 文件夹在同一个目录下. YOLOv3.txt 标签文件的数据格式如下:

类别编号	中心点 x 相对坐标	中心点 y 相对坐标	目标框相对宽度 w	目标框相对高度 h

当检测框 boundingbox 的左下、右上坐标分别为 $(x_1, y_1)(x_2, y_2)$, 图像的宽和高分别为 W, H, 此时可以计算:

$$\begin{cases} x = ((x_1 + x_2)/2.0)/W \\ y = ((y_1 + y_2)/2.0)/H \\ w = (x_2 - x_1)/W \\ h = (y_2 - y_1)/H \end{cases}$$

SSD 的数据集同样包括标签和图像两部分. JPEGImages 文件夹下存放图像, Annotations 文件夹下存放 xml 格式的标签文件, 并且要求图像的名称与标签的名称必须一致. SSD 算法中, 还有一个 ImageSets 文件夹下的 main 文件夹中存放 test.txt、val.txt、train.txt、trainval.txt 4 个文件, 这 4 个文件中分别存放用来测试、验证、训练、训练和验证的图像文件的文件名列表. SSD 标签文件格式如下:

类别编号	X_{min} 坐标	Y_{min} 坐标	X_{max} 坐标	Y_{max} 坐标

然后, 将上面的标签文件处理成 xml 文件的数据格式如图 1. 与图 1 对应的图像如图 2.

但是, xml 格式的标签不能直接在 SSD 算法中使用, 所以需要对标签文件做处理, 将 xml 格式的标签文件转换成 TensorFlow 统一使用的数据存储格式——tfrecord 格式. tfrecord 格式文件主要通过 Example 数据结构存储, 格式如图 3.

```
<?xml version="1.0"?>
- <annotation>
    <folder>VOC2007</folder>
    <filename>000036.jpg</filename>
  - <source>
      <database>The VOC2007 Database</database>
      <annotation>PASCAL VOC2007</annotation>
      <image>flickr</image>
      <flickrid>340478761</flickrid>
    </source>
  - <owner>
      <flickrid>jmp45fr</flickrid>
      <name>?</name>
    </owner>
  - <size>
      <width>332</width>
      <height>500</height>
      <depth>3</depth>
    </size>
    <segmented>0</segmented>
  - <object>
      <name>dog</name>
      <pose>Unspecified</pose>
      <truncated>0</truncated>
      <difficult>0</difficult>
    - <bndbox>
        <xmin>27</xmin>
        <ymin>79</ymin>
        <xmax>319</xmax>
        <ymax>344</ymax>
      </bndbox>
    </object>
</annotation>
```

图 1 SSD 算法 xml 标签文件格式

图 2 图像与图 1 文件对应 (332×500)

3 图像相似度算法

图像相似度检测[15]是对多张图像的内容相似程度进行分析, 目前广泛应用在图像搜索、图像过滤、目标识别等领域. 图像相似度对比的方法有很多, 包括: SIFT 算法[16]、图像直方图相似度、哈希的图像相似度等方法. 图像直方图方式最简单, 但是图像转为直方图丢失大量信息, 误判概率比较大. SIFT 算法复杂, 会产生大量 SIFT 特征向量, 但是能够识别图像不超过 25% 的变形. 根据实验的需求, 选择了介于两者算法性

能、复杂度之间的哈希图像相似度检测算法. 基于哈希的图像相似度检测算法包括: 均值哈希算法、感知哈希算法和差异值哈希算法. 以上三种相似度检测算法分别适用于不同的应用场景. 差异值哈希算法在识别效果方面优于均值哈希算法, 不如感知哈希算法, 但是在识别速度方面优于感知哈希算法[17]. 本论文中, 采用差异值哈希算法[18], 可以在使用较少的时间实现较好的效果. 使用差异值哈希算法, 首先需要对图像进行缩放, 可以选择缩放为 8×8、16×16、32×32 不同尺寸的图像, 然后, 将图像灰度化, 接下来是对图像差异值的计算, 以缩放尺度 9×8(差异值哈希得到 8×8 的图像需要缩放成 9×8, 同样 16×16 的需要缩放为 17×16) 为例, 通过分别对 9×8 图像的每一行中相邻的两个像素进行相减得到每个像素间的差异值, 最后得到 8×8 的图像哈希值. 最终, 通过两张图像哈希值的汉明距离计算, 得到图像的相似度.

```
Example Message {
    Features{
        feature{
            key:"name"
            value:{
                bytes_list:{
                    value:"dog"
                }
            }
        }
        feature{
            key:"size"
            value:{
                int64_list:{
                    value:332
                    value:500
                    value:3
                }
            }
        }
        feature{
            key:"bndbox"
            value:{
                int64_list:{
                    value:27
                    value:79
                    value:319
                    value:344
                }
            }
        }
        feature{
            key:"data"
            value:{
                bytes_list:{
                    value:0xbe
                    value:0xb2
                    ...
                    value:0x3
                }
            }
        }
    }
}
```

图 3 tfrecord 格式

4 架构与流程

目标检测数据集自动生成架构及执行流程如图 4 所示, 分为如下 3 个阶段.

阶段一: 获取原始标注图像. 使用基于 DarkNet 的 YOLOv3 算法和基于 TensorFlow 的 SSD 算法, 以及在 COCO 数据集上训练的 YOLOv3 模型和在 ILSVRC CLS-LOC 数据集预训练的 SSD 模型. 首先, 将原始数据集中的图像输入 YOLOv3 模型, 检测后得到带 boundingbox 的图像和位置坐标文件, 然后, 对坐标文件进行初筛, 丢弃未检测到目标的图像. 同时, 将原始数据集中的图像输入到已经训练好的 SSD 模型, 同上, 保留适当的图像集和位置坐标文件, 并且对两种方式获得的检测图像按照 boundingbox 进行裁剪.

阶段二: 图像相似度比较. 利用差异哈希算法, 对阶段一裁剪后的两个图像集中同名的图像逐一进行相似度检测, 根据相似度结果, 按照预先设置的阈值, 保留阈值范围的图像.

阶段三: 人工图像筛查. 通过对原始图像与检测图像的缩略图快速对比, 对阶段二得到的图像进行人工筛选, 选择正确的图像与相应的坐标文件, 得到最终的数据集. 获得的数据集, 作为将来目标检测模型训练的训练数据集, 使新训练的模型能够正确分类. 对于未检测到目标的图像, 根据模型在实际场景中是否达到预期效果, 决定是否对图像进行人工标注, 加入训练数据集, 增强模型目标检测的能力.

在 SSD 算法中, 传统数据集制作需要人工完成: 图像目标框标注、类别标注并生成 xml 标签文件、xml 文件格式转换为 tfrecord 格式等工作. 在 YOLOv3 算法中, 传统数据集制作需要人工完成: 图像目标框标注、类别标注并生成 txt 文件等步骤. 对比本文方法, 现在只需要快速简单的人工筛查即可替代传统复杂、耗时的工作. 使用这种方法生成数据集代替人工标注图像, 不仅加快了标注速度, 而且引入相似度对比、人工筛查的步骤, 更好的保证了最终数据集的质量.

5 实验过程

以基于 DarkNet 的 YOLOv3 为例, 需要修改网络结构 (.cfg) 文件的 batch、division 等参数存放在 cfg 中, 将指明训练集/测试集路径、类别数量、类别名称等信息的文件存放在 data 中.

5.1 实现方法

首先, 将待生成标签的图像输入到已训练的模型

中, 检测图像中目标物体的位置, 通过在 DarkNet 中绘制预测框命令前添加命令, 将图片路径、预测框的相对坐标以及目标框相对宽高写入文件, 输出的文件格式如图 5(a) 所示. 然后将图像的类别编号追加到上述的坐标位置文件中, 如图 5(b) 所示, 坐标位置文件的名称与图像名称一致.

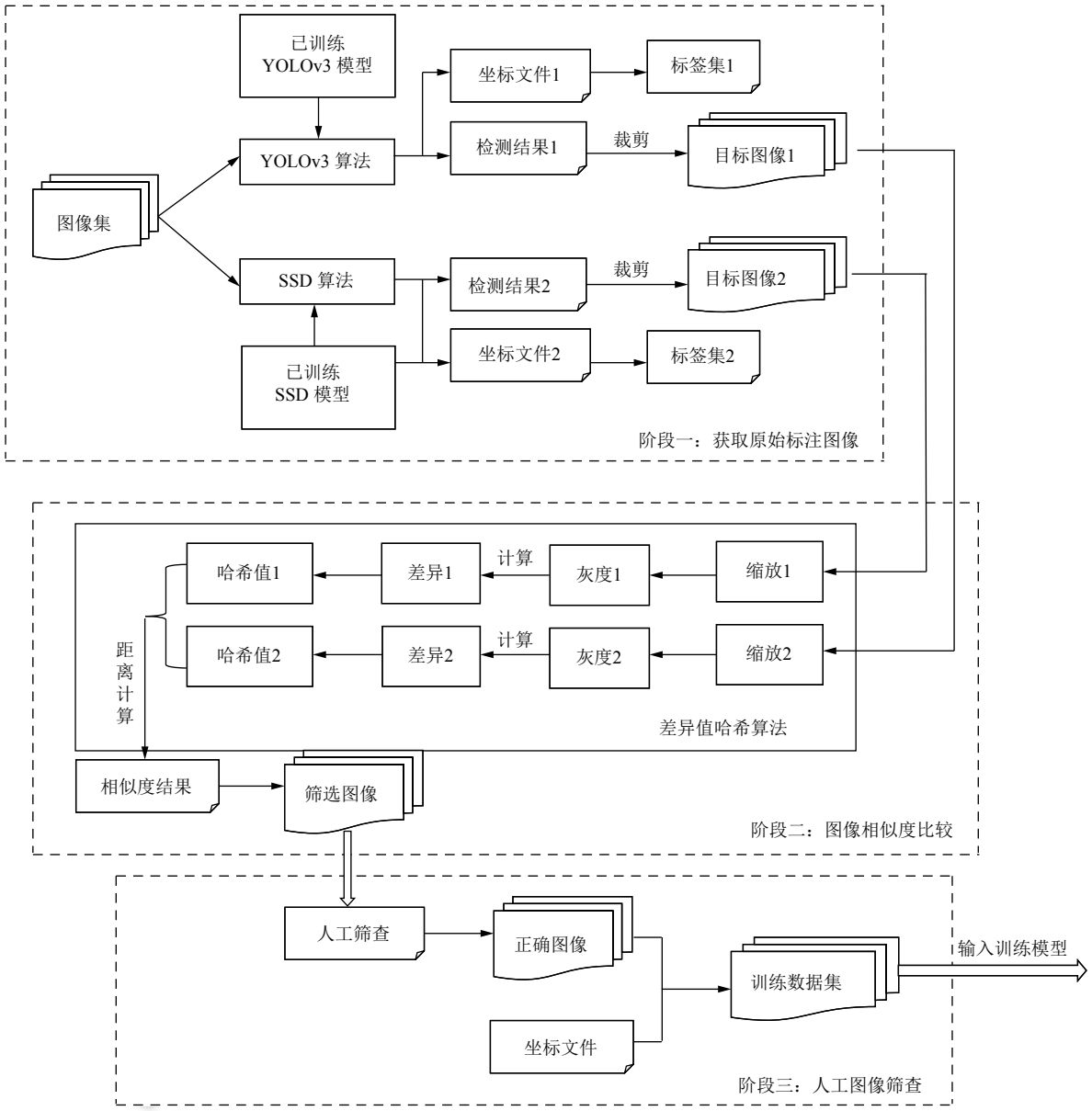

图 4 目标检测数据集生成架构与流程

然后, 使用基于 TensorFlow 的 SSD 算法, 直接使用 SSD[4]论文中提到的在 ILSVRC CLS-LOC 数据集上预训练的 SSD 模型, 将待生成标签的图像输入, 然后把得到检测结果中目标框的坐标位置等信息整理成 xml 格式, 最终通过 Python 脚本生成 tfrecord 格式标签.

通过前面两步, 得到两组检测结果图像. 最后, 对以上两组结果图像按照目标框进行裁剪, 如图 6 和图 7, 对比两组裁剪后图像的相似度为 17. 输入图像大小对差异值相似度有影响[17], 经过对 8×8、16×16、32×32 不同尺寸图像的相似度检测结果对比, 8×8 的图像在减少计算量的同时较好的保留图像信息. 将裁剪后的图像统一大小为 9×8, 再计算相似度, 根据实验测试, 相似度阈值设置为 30, 当相似度小于阈值的时候, 认为两张图像相似, 图像和标签是有效的.

./1/69dcc552-dcdf-4d7c-ba77-2e76dc4821f8.jpg
0.508680 0.497835 0.563919 0.820717
./1/c29ffc64-289c-479c-8d89-f12266c96f51.jpg
0.535448 0.529205 0.942825 0.894189
./1/c29ffc64-289c-479c-8d89-f12266c96f51.jpg
0.457206 0.535079 0.885151 0.868892
./1/bing_755_121.jpg
0.481500 0.451958 0.228212 0.676705
./1/1265a047ef0g213-s270.jpg
0.488264 0.558415 0.823716 0.685284

(a) 图像路径和对应图像的坐标位置

1 0.481500 0.451958 0.228212 0.676705

(b) 坐标文件 bing_755_121.txt 的内容

图 5 (a) 图像路径和对应图像的坐标位置 (b) 坐标文件 bing_755_121.txt 的内容

图 6 YOLOv3 检测结果裁剪图

图 7 SSD 检测结果裁剪图

5.2 实验结果

为了避免大量选取 COCO 数据集中包括的物种, 导致测试准确率虚高. 本实验选择猴类图像 3797 张、鹿类 3530 张、兔子类 2407 张、松鼠类 2122 张、老虎 227 张以及狼 126 张, 总共选取了约 12 000 张哺乳类动物的图像, 为其生成标签. 实验分别选择: 1) 单独使用 YOLOv3 算法, 2) 单独使用 SSD 算法, 3) 同时使用 YOLOv3 和 SSD 算法以及相似度对比 3 种方法, 为图像生成标签. 为了实验结果有对比性, 3 组实验使用相同的数据集. 具体实验结果如表 1 所示.

表 1 3 种方式生成数据集标签情况对比

检测方法	图像数目	检测数目	检测率 (%)	检测准确率 (%)
YOLOv3	12 104	8869	73.27	82.6
SSD	12 104	8953	73.97	82.74
YOLOv3+SSD+相似度对比	8738	7656	87.32	84.42

使用 YOLOv3+SSD+相似度对比方法时, 数据集选择 YOLOv3 和 SSD 中均检测到对象的图像集作为数据集. 检测数目为经过相似度筛选后保留的图像数目. 检测率为图像检测数目占图像数目的比例. 检测准确率为准确图像占检测数目的比例.

6 结论与展望

通过实验对比, 可以直观的看出, 当使用 YOLOv3+SSD+相似度对比的方法, 不仅在生成数据集的速度上有提升, 而且对目标物体检测的准确率也有一定的提升. 但是, 这种方法存在不足: 筛选图像的质量依赖相似度阈值的设置, 如果阈值设置过小, 准确率率提升不明显, 设置过大, 导致大量的图像和标签被丢弃, 最终训练的数据量变小, 所以, 此后还需要寻找一种阈值设置的标准和方法. 另外, 检测图像时可能存在两种算法出现相同错误的情况, 导致该方法效果下降, 所以通过加入人工筛查来克服这个缺点. 如果最终数据集过小, 可以适当增大阈值的设置, 之后通过人工筛选过滤掉不符合要求的图像.

参考文献

1 LeCun Y, Bengio Y, Hinton G. Deep learning. Nature, 2015, 521(7553): 436–444. [doi: 10.1038/nature14539]
2 Girshick R. Fast R-CNN. 2015 IEEE International Conference on Computer Vision (ICCV). Santiago, Chile. 2015. 1440–1448.
3 Redmon J, Farhadi A. YOLOv3: An incremental improvement. Computer Vision and Pattern Recognition. arXiv: 1804.02767, 2018.
4 Liu W, Anguelov D, Erhan D, *et al*. SSD: Single shot multiBox detector. European Conference on Computer Vision. Cham, Switzerland. 2016. 21–37.

5 Lowe DG. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 2004, 60(2): 91–110. [doi: 10.1023/B:VISI.0000029664.99615.94]
6 Schapire RE, Singer Y. Improved boosting algorithms using confidence-rated predictions. Machine Learning, 1999, 37(3): 297–336. [doi: 10.1023/A:1007614523901]
7 Dalal N, Triggs B. Histograms of oriented gradients for human detection. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. San Diego, CA, USA. 2005. 886–893.
8 Ojala T, Pietikäinen M, Harwood D. A comparative study of texture measures with classification based on featured distributions. Pattern Recognition, 1996, 29(1): 51–59. [doi: 10.1016/0031-3203(95)00067-4]
9 Ng PC, Henikoff S. SIFT: Predicting amino acid changes that affect protein function. Nucleic Acids Research, 2003, 31(13): 3812–3814. [doi: 10.1093/nar/gkg509]
10 殷帅, 胡越黎, 刘思齐, 等. 基于 YOLO 网络的数据采集与标注. 仪表技术, 2018, (12): 22–25.
11 不用人工打标制作目标检测数据集的方法. https://blog.csdn.net/u011983997/article/details/80089418. [2018-04-26].
12 He KM, Zhang XY, Ren SQ, *et al*. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, NV, USA. 2016. 770–778.
13 Chen XL, Fang H, Lin TY, *et al*. Microsoft COCO captions: Data collection and evaluation server. Computer Vision and Pattern Recognition. arXiv: 1504.00325, 2015.
14 Deng J, Dong W, Socher R, *et al*. ImageNet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition. Miami, FL, USA. 2009. 248–255.
15 王法强, 张宏志, 王鹏, 等. 计算机视觉中相似度学习方法的研究进展. 智能计算机与应用, 2019, (1): 149–152. [doi: 10.3969/j.issn.2095-2163.2019.01.035]
16 Lowe DG. Object recognition from local scale-invariant features. Proceedings of the International Conference on Computer Vision. Corfu, Greece. 1999. 20–27.
17 黄嘉恒, 李晓伟, 陈本辉, 等. 基于哈希的图像相似度算法比较研究. 大理大学学报, 2017, 2(12): 32–37. [doi: 10.3969/j.issn.2096-2266.2017.12.007]
18 尹玉梅, 彭艺, 祁俊辉, 等. 基于双重 Hash 的图像相似检索算法研究. 信息通信技术, 2019, (1): 33–38.