

基于深度学习的小目标检测算法研究

黄雪岩

学 院：电子与信息工程学院 专 业：通信工程

学 号：SZ160210103 指导教师：陈梓浩

2020 年 6 月

哈爾濱工業大學(深圳)

毕业设计（论文）

题 目 基于深度学习的小目标

检测算法研究

专 业 通信工程

学 号 SZ160210103

学 生 黄雪岩

指 导 教 师 陈梓浩

答 辩 日 期 2020 年 6 月 10 日

摘 要

目标检测是机器视觉领域内的一项研究热点，其主要任务是对图片中的物体进行定位和分类。目标检测对自动驾驶，安防监控，卫星图片分析等领域至关重要。随着近几年卷积神经网络的提出与计算机算力的提升，目标检测领域也获得了充足的发展。目前，工业界已成功将目标识别技术应用在人脸识别，文字扫描等问题上。然而，目标检测领域仍有一些尚未解决的难点，如小目标的检测率低，定位不精确，检测速度慢等。本文致力于研究小目标检测的相关问题，探讨如何在深度学习框架下提高小目标的检测率，并提出了四项改进策略。

第一，考虑到小目标尺寸小，细节特征少的特点，本文使用超分辨率生成式对抗网络对包含小目标的图片进行超分辨率重建，实现对小目标尺寸的放大以及小目标细节信息的增加。

第二，由于常见数据集包含的小目标数据量远远少于常规尺寸目标的数据量，为了避免数据量不平衡带来的模型过拟合问题，本文提出了随机扩充的数据增强方法，增加了小目标的数据量。

第三，在卷积神经网络中，浅层特征图含有较多的细节特征，适合小目标检测，而深层特征图含有较多的语义特征，适合大尺寸目标的检测。基于此特点，本文对通用目标检测模型 SSD 进行改进，增加对浅层特征层的检测模块，并适当调整锚框的大小，提高了检测模型对小目标的关注度。

第四，主流目标模型在过滤重复的候选框时通常使用非极大值抑制算法。该算法在检测目标拥挤的场景时容易过滤掉其它实例的候选框，从而发生漏检现象。为改进这一缺点，本文使用改进型非极大值抑制算法，减少了小目标的漏检概率。

本文提到的针对小目标检测的四个改进方法均通过代码进行了验证，并在文中对实验结果进行了相应的展示和分析。

关键词：目标检测；卷积网络；超分辨率；非极大值抑制

Abstract

Object detection is a research hotspot in the field of machine vision, whose main task is to locate and classify objects in pictures. Object detection is essential for autopilot, security surveillance, satellite image analysis and other fields. With the invention of convolutional networks and the advancement of computer power in recent years, the field of object detection has also gained sufficient development. The industry has successfully applied object recognition technology to face recognition, word scanning and other problems. However, there are still some unresolved problems in the field of object detection, such as low detection rate of small objects, inaccurate positioning, slow detection speed, etc. In this paper, we focus on the problems related to small object detection, explore how to improve the detection rate of small objects in a deep learning framework, and propose four improvement strategies.

First, considering the small size of objects and few detail features, this paper uses the super-resolution generative adversarial network to reconstruct the images containing small objects, so as to enlarge the size of small objects and increase the details of small objects.

Second, since the amount of small object data contained in common data sets is much less than that of regular size objects, in order to avoid the model overfitting problem caused by data imbalance, this paper proposes a data enhancement method of random expansion to increase the data volume of small objects.

Thirdly, in convolutional neural networks, the shallow feature layers tend to contain more detailed features, which are suitable for small object detection, while the deep feature layers contain more semantic features, which are suitable for large size object detection. In this paper, we improve the generalized object detection model SSD by increasing the number of detections of shallow feature layers and adjusting the size of anchor to increase the focus of the detection model on small objects.

Fourth, the mainstream object model usually uses a non-maximum suppression algorithm to filter out duplicate candidate boxes. This algorithm may filters out candidate boxes for other instances when detecting a crowded scene. To solve this drawback, this paper uses an improved non-maximum suppression algorithm.

The four improved methods for small object detection mentioned in this paper are validated by code, and the results are presented and analyzed in this paper.

Keywords: object detection, convolutional network, super-resolution, non-maximum suppression

目 录

摘要	I
ABSTRACT	II
第 1 章 绪论	1
1.1 课题背景及研究的目的和意义	1
1.1.1 目标检测领域背景	1
1.1.2 小目标检测研究背景及意义	4
1.2 国内外研究现状	5
1.2.1 目标检测领域的研究现状	5
1.2.2 小目标检测的研究现状	8
1.3 本文的主要研究内容	9
1.4 章节安排	10
第 2 章 目标检测与深度学习基础	11
2.1 目标检测基础	11
2.1.1 目标检测任务定义	11
2.1.2 目标检测评价指标	11
2.1.3 目标检测损失函数	13
2.1.4 目标检测领域常见数据集	14
2.2 深度学习基础	15
2.2.1 神经元与激活函数	15
2.2.2 卷积的定义	15
2.2.3 卷积神经网络	16
2.2.4 模型训练技巧	17
2.3 本章小结	18
第 3 章 基于 SRGAN 的图像超分辨率重建	19
3.1 引言	19
3.2 SRGAN 模型原理	19
3.3 SRGAN 对小目标的超分辨率重建	21
3.4 实验结果及分析	22
3.5 本章小结	26

第4章 基于 SSD 的小目标检测模型	27
4.1 引言	27
4.2 SSD 模型结构及算法流程.....	27
4.3 基于 SSD 模型的改进方法.....	29
4.3.1 针对小目标的数据增强.....	29
4.3.2 调整锚框尺寸.....	30
4.3.3 改进型非极大值抑制算法.....	32
4.4 实验结果及分析	33
4.5 本章小结	35
结论	37
参考文献	39
原创性声明	42
致谢	43
附录	44

第1章 绪论

1.1 课题背景及研究意义

小目标检测是面向小尺寸目标的目标检测问题。本节在介绍小目标检测的研究背景之前，先介绍目标检测领域的相关背景。

1.1.1 目标检测领域背景

在计算机视觉领域，视觉检测问题可简单分为：图像分类，目标检测，语义分割，实例分割。如图 1-1 所示：图像分类任务主要负责分类图像中的物体所属类别；目标检测任务则在目标分类的基础上再预测出物体的具体位置；语义分割任务致力于为每一个像素点划分其归属的类别；而实例分割任务则在语义分割的基础上再区分出每个实例。

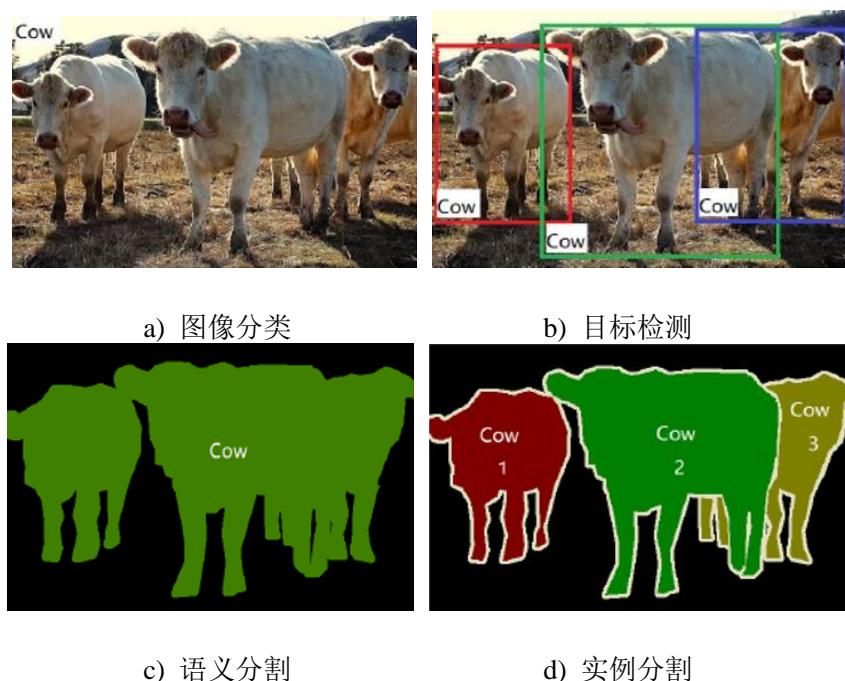


图 1-1 视觉检测问题分类^[1]

在深度学习被广泛应用之前，传统目标检测流程可分为三个阶段：候选框生成、特征向量提取、区域分类。

第一阶段，候选框搜索算法会遍历图片搜寻可能含有物体的区域。这些区域又被称作感兴趣区域（Region of Interest, ROI）。为了能够检测到不同尺寸的物体，候选框搜索算法需要通过图像金字塔的方法对图片进行缩放或者使用多尺度的搜索框进行遍历。

第二阶段，对每个候选框进行特征提取。通常会将特征向量设计成固定长度的特征向量，这样便于分类时物体相似性的计算。典型的特征向量如 HOG^[2], SIFT^[3], 等，这些特征向量被巧妙地设计出来，具有一定的噪声鲁棒性，尺度不变性，旋转不变性。

第三阶段，将得到的特征向量输入分类器得到分类结果。最典型的传统分类器是支持向量机（Support Vector Machine, SVM）。对于一个线性可分的数据集，能够对其进行分类的超平面有很多个，但是区分间隔最大的超平面是唯一的，如图 1-2 所示。支持向量机便是以这种思路设计出来的算法。SVM 具有良好的理论基础，其数学形式为凸优化问题，有唯一的最优解。SVM 以其优异的表现被广泛应用在传统的分类器中。

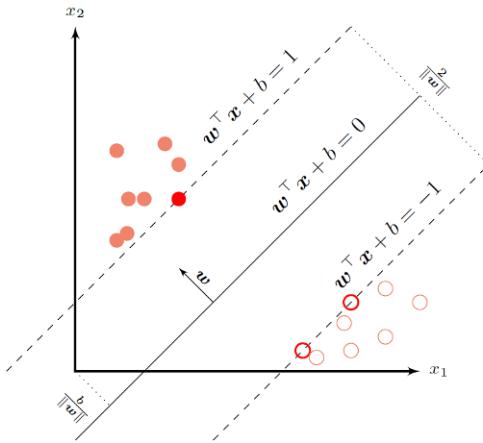


图 1-2 支持向量机

以 DPM (Deformable Part based Machine)^[4] 为代表的传统目标检测算法连续三年在 PASCAL VOC 比赛上取得冠军。随后，传统目标检测模型的发展陷入瓶颈，只能通过集成手段，如装袋法（bagging），自举法（boosting），级联学习（cascade learning）等方法，略微提高一点准确率。模型检测正确率难以提高的原因在于分类器需要有较强的特征才能有良好的表现。而传统检测算法的特征是人为设计的。这些特征通常是像素级别的底层特征，并不能满足复杂的任务要求。随着深度学习在图片分类领域的巨大成功，目标检测领域也开始尝试使用深度神经网络。

自从引入深度学习之后，目标检测领域可简单分为两个发展脉络，单阶段模型和双阶段模型^[5]，如图 1-3 所示。前文介绍过传统目标检测可分为三阶段：候选框生成、特征提取、区域分类。在深度学习的框架下，特征提取的任务交由神经网络从数据中自动学习，因此有了两阶段的目标检测模型 RCNN^[6], Fast RCN^[7], Faster RCNN^[8]。而单阶段模型如 YOLO^[9], SSD^[10]。**Error! Reference source not found.**等，直接舍弃了候选框生成模块，通过回归的方式直接输出分类和定位结果。双阶段模型和单阶段模型各有优势。一般情况下，双阶段模型的精确率较高，而单阶段模型的检测速度较快。

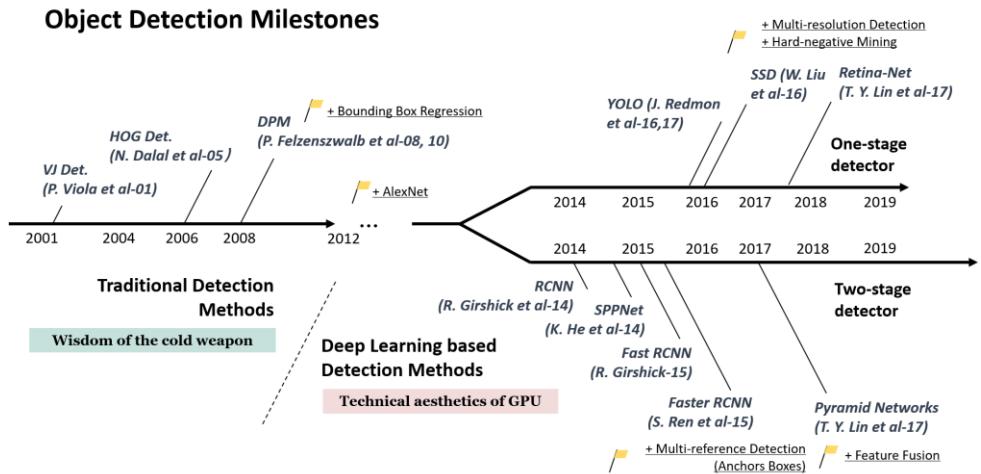


图 1-3 目标检测领域发展脉络^[4]**Error! Reference source not found.**

虽然近几年提出的目标检测模型如 Faster R-CNN, SSD 等已经能在一些公开数据集上取得很高的正确率。但现实世界的场景变化万千，如图 1-4 所示。在这些场景下，目标检测模型检测率大幅下降。目前的主流检测模型并没有很好地解决这些问题。



a) 光线干扰



b) 目标密集



c) 光线干扰



d) 目标密集

图 1-4 复杂场景的目标检测^[1]

1.1.2 小目标检测研究背景及意义

小目标检测是目标检测领域的一个难点。虽然目标检测对常见物体的识别已经达到了很高的正确率，但在识别小物体方面还远远达不到令人满意的结果。如图 1-5 所示，图片中的小物体缺乏足够的细节特征以至于模型无法将其与图片背景区分开来，于是发生了漏检现象。

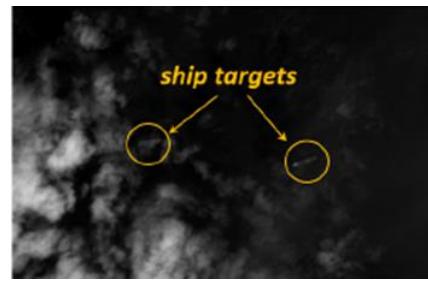
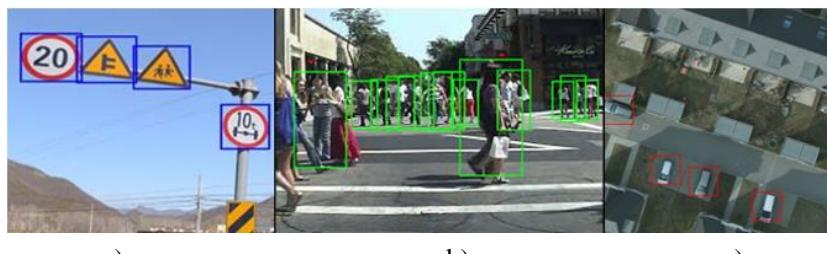


图 1-5 船只检测

通用的检测模型往往不能很好的检测出小目标，这主要有两个原因：一是小目标本身蕴含的信息量有限，二是小目标的数据量匮乏。当前的主流检测模型大多关注于物体的高层特征，而小目标本身尺寸大小就决定了它不能经过多层的卷积操作，这就导致了通用的检测算法难以适用于小目标检测。另一方面，小目标的数据集往往限定于特定场景，数据难以获得，并且人工标注的成本也非常高，因此对于小目标检测的相关研究也难以取得较大的突破。目前的主流检测模型对常规目标的识别率已经达到了很高的水平，小目标的低检测率是限制这些模型检测性能的一个重要因素。

在一些特定的场景，小目标检测起着至关重要的地位。比如自动驾驶领域，为了汽车能够正确安全地行驶，模型就必须做好信号灯检测和行人检测。这两个模型处理的数据往往就是小目标，如图 1-6 a) 和 1-6 b) 所示。此外，对于无人机航拍图和卫星摄像图，物体呈现的就是小目标的形态，如 1-6 c) 所示。因此，小目标检测在安防，自动驾驶，救援等领域具有重要的现实意义与研究价值。



a)

b)

c)

图 1-6 小目标应用场景

1.2 国内外研究现状

本节主要总结近年来国内外优秀学者在目标检测领域的工作以及其中涉及到小目标检测的部分。

1.2.1 目标检测领域的研究现状

传统目标检测和基于深度学习的检测算法最大的区别在于特征的设计上。传统方法使用的是人工设计的特征。人工设计特征的好处就是可解释性强，学术界也更热衷于研究这种有理论基础的模型。此类特征在设计之时需要考虑到以下几个方面的影响：噪声，尺度，光照，旋转等。图片提取的特征要满足同一个物体在这几种影响的作用下尽可能保持一致，并且不同物体之间的特征还要有足够的区分度。

典型的人工特征如 HOG^[2]，它由 Dadi 等人于 2005 年提出。HOG 特征的原理是在单元格内计算得到梯度并以直方图的形式统计这些梯度作为特征描述子。通过缩放图片重复计算，使得 HOG 特征有了尺度不变性。梯度的计算避免了光照强度对像素值大小的影响，通过统计直方图也在一定程度上减少了噪音的干扰。以 HOG 为基础的 DPM^[11]模型连续三年赢得了 PASCAL VOC 的目标检测比赛。如此巧妙的设计使得 HOG 特征有了强大的描述能力，但是设计这种特征需要研究人员费尽心思，还要不断通过实验调整阈值以得到最佳配置。人工设计的特征往往局限于底层像素之间的计算，无法适应更加复杂的任务，而深度卷积网络得到的特征是模型通过数据自动学习到的。虽然这类特征十分抽象，可解释性不强，但在表示能力上却远超传统人工特征。随着深度神经网络的流行，神经网络自动学习的特征逐渐取代了人工特征。

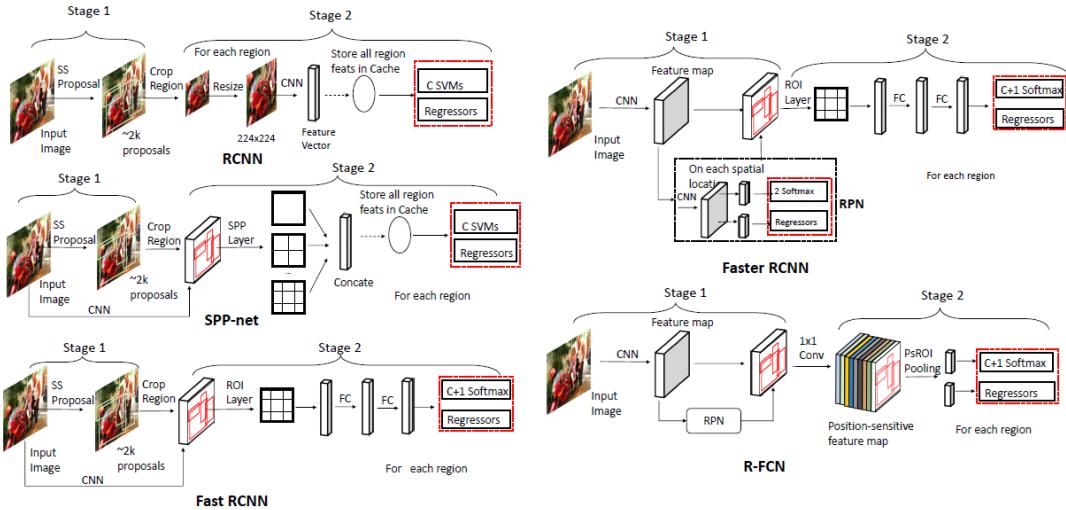
基于深度学习的目标识别模型主要分为两类：单阶段模型和双阶段模型，它们之间最主要的区别在于是否有候选框生成模块。双阶段方法延续了传统目标检测模型的思路，先将图片送入候选框生成模块后，得到可能含有物体的候选框。然后再对这些候选框进行分类和位置回归的操作。而单阶段方法则直接通过回归的方法得到目标类别。

双阶段方法中最为典型的是 Girshick^[6]于 2014 年提出的 RCNN 模型。RCNN 模型是首个将深度学习应用到目标检测领域的模型，它的创新之处在于使用深度神经网络替换了传统目标检测的特征提取模块。RCNN 首先通过 Selective Serach

的方法得到候选框，然后对这些候选框进行裁切和拉伸得到固定大小的图片，再通过神经网络对其进行特征提取，最后将得到的 4096 维特征输入多分类 SVM 得到分类结果。深度神经网络的引入使得 RCNN 获得了远超传统方法的检测正确率。由于 RCNN 每张图片都要提取近 2000 个候选框，每个候选框都要进行特征提取，因此模型运行速率十分缓慢。

He 等人经过推导发现完全可以避免重复计算候选框的特征，只对整张图片进行一次特征提取，于是提出了空间金字塔池化（Spatial Pyramid Pooling Network, SPPNet）^[12]。SPPNet 只需在最后的特征层进行 Selective Search 便可得到候选框。为了解决候选框大小不一的问题，SPPNet 提出了目标区域池化（Region of Interest pooling, ROI pooling）的手段，通过对候选框划分不同大小的网格，然后对网格使用最大池化（Max pooling）得到固定大小的输出。

借鉴了 SPPNet 的方法，Girshick 等提出了 Fast RCNN^[7]。与 SPPNet 相比，Fast RCNN 使用多个 Softmax 分类器替换掉 SVM，同时将最后的全连接层的输出与候选框的预测位置进行边界框（Bounding box）回归得到更精确的预测位置。Fast RCNN 在精度和速度上都优于 RCNN 模型。经实验发现，Fast RCNN 在运行时，传统的 Selective Search 占据了大部分时间。为了解决这一问题，Ren 等人在此 Fast RCNN 的基础上提出了 Faster RCNN^[8]，他们利用卷积网络构造了区域生成网络（Region Proposal Network, PRN）代替 Selective Search。PRN 网络利用锚框(anchor)的方法对特征层进行遍历并判断该区域是否属于目标还是背景，然后将属于目标的区域送入后续的分类和回归网络中。Faster RCNN 获得了比 Fast RCNN 更快的运行速度。由于每个 ROI 都需要经过全连接层得到分类结果，然而可能有多个 ROI 覆盖的是同一个物体，因此这种全连接层的计算就相当于重复计算了。而全连接的计算时间消耗比卷积层大得多，需要消耗大量的计算时间。根据卷积操作的特点可知，卷积网络具有位置不敏感性，即物体不管在图片哪个位置都能得到正确分类，而要进行目标检测又希望模型具有位置敏感性，才能正确框选出物体的位置。如果仅仅是移除全连接层，则物体的位置信息没法从全连接层得到，导致模型检测性能大大降低。为了改进这个缺点，Dai 等^[13]提出基于区域的全卷积网络（Region-based Fully Convolutional Network, RFCN），该模型使用位置敏感得分图代替全连接层。位置敏感得分图包含了各个种类的相对位置信息，配合位置敏感的 ROI 池化层得到既包含分类信息又包含相对空间信息的特征。由此，RFCN 网络取得了足以与 Faster RCNN 相匹配的精度还有更快的运行速度。双阶段模型结构如图 1-7 所示：

图 1-7 双阶段检测模型^[14]

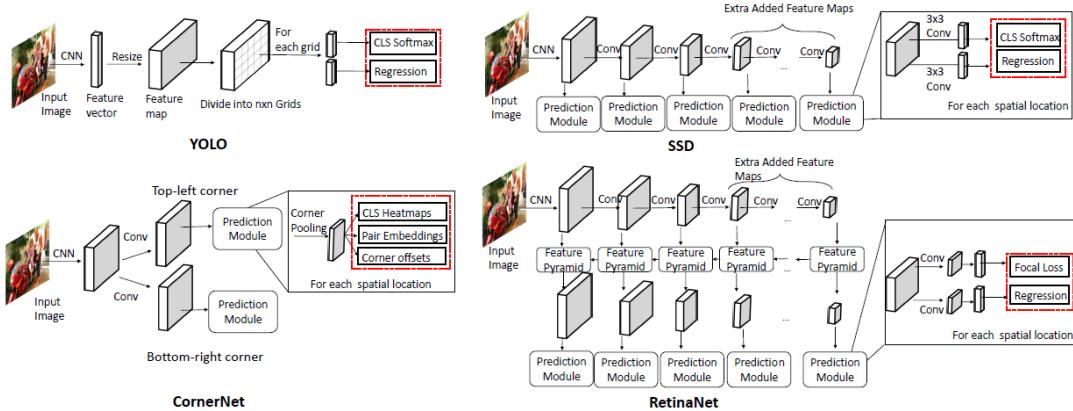
单阶段方法与双阶段方法相比少了一个候选框生成模块，它简单地认为整张图片的各个地方都可能有目标存在，然后通过遍历的方法尽可能分辨出该区域是图片背景还是目标物体。单阶段方法的最大特点在于没有了候选框上分类的过程，而是直接通过回归得到目标类别。典型的单阶段分类器如 YOLO 和 SSD。

YOLO 是 Redmon 等^[9]提出的首个单阶段目标检测模型。它将目标检测问题当成是回归问题，通过将图片均匀划分成固定大小的网格，对每个网格都当成候选框进行预测，输出分类信息和定位坐标。YOLO 的模型结构允许它进行端到端的训练。由于没有候选框生成模块，它的运行速度非常快，满足每秒 45 帧的实时检测。但运行速度增加的同时也导致了检测准确率的下降。导致准确性下降有很多原因，比如 YOLO 的多目标损失函数没有精心考虑而是直接使用了平方和；YOLO 模型的网格仅预测单个类别，不能适应物体拥挤的场景；锚框的长宽比例固定，对物体形变泛化能力较低。为了解决这些问题，Redmon 等^[15]又陆续提出了 YOLOv3。YOLOv3 在保持了 YOLO 实时性的同时还提高了目标检测的准确率。

Liu 等人借鉴 YOLO 的回归思想提出 SSD^[10]。SSD 也将图片均分成固定大小的网格，但是对每个网格设置多个不同尺度和长宽比例的锚框（anchor）以适应不同大小和形状的目标。另外，为了更好的预测不同尺度的物体，根据感受野的大小划分，SSD 在不同深度的特征图上进行网格预测。最后 SSD 使用了加权的损失函数来融合定位损失和分类损失。由此，SSD 不仅在精度上可以和双阶段的检测方法 Faster R-CNN 相媲美，在速度上达到了实时性的要求。

YOLO 和 SSD 系列都用到了锚框（anchor），Law 和 Deng^[16]认为锚框的数量众多且锚框的参数需要人工设置不利于找出最优配置，于是他们提出了不需要使用

锚框的模型 CornerNet。CornerNet 使用一个卷积网络生成两组特征图，一个负责预测左上角，另一组负责预测右下角，以两个关键点生成候选框。CornerNet 还引入了一种新型的 corner pooling 帮助网络更好地定位边界框的角度。单阶段模型结构如图 1-8 所示：

图 1-8 单阶段检测模型^[14]

1.2.2 小目标检测的研究现状

目前学界还没有给小目标一个具体的定义，因为小和大是一个相对概念。同一个物体在相机的远处和近处拍摄呈现的大小结果是不同的。图片中的小目标可能是物体自身的物理尺寸过小或者是拍摄距离较远导致的。虽然小目标的判定没有一个统一的标准，但是在特定的数据集内，小目标可以人为地给出准确定义。比如 MS COCO 数据集内规定尺寸在 32×32 之内的目标可以认为是小目标。如表 1-1 所示：

表 1-1 COCO 数据集对目标尺度的定义

	最小面积	最大面积
小尺度目标	0×0	32×32
中尺度目标	32×32	96×96
大尺度目标	96×96	$\infty \times \infty$

为了提高小目标检测的精度，各路学者提出了许多行之有效的方案。

Lin 等^[17]从损失函数方面出发，提出了 RetinaNet。RetinaNet 主要解决了单阶段模型正负样本不平衡的问题。通常情况下，一张图仅有几个目标，而产生的候选框却是成千上百，负样本对总体损失函数的影响远远大于正样本。如此一来，模型便会过分关注负样本，也更容易发生对正样本的漏检现象。RetinaNet 使用的 Focal Loss 提高了对少量样本的关注，很好地解决了正负样本不平衡的问题。

Shrivastava 等^[18]从误分类样本方面考虑，提出困难样本挖掘（Online Hard Example Mining, OHEM）方法，通过维护误分类样本池，使模型重复训练一些导致其损失较大的样本，强化了模型对小目标的关注度。

Singh^[19]设计了多尺度的检测算法（Scale Normalization for Image Pyramids, SNIP）。他们通过分析实验数据得到结论：由于卷积网络不具有尺度不变性，特定分辨率的目标数据训练得到的检测器对其他分辨率的目标识别率低。这说明检测器对训练数据有依赖性，通过大尺度目标数据训练得到的检测模型就不擅长识别小尺寸目标。基于此特点，SNIP 模型设计了三个尺度的检测器，每个检测器只对适合其自身尺寸的目标进行检测和梯度回传，然后经过映射模块得到目标在原图的检测框，最后经过非极大值抑制筛选出最后的检测结果。

Lin 等^[20]提出特征金字塔（Feature Pyramid Network, FPN）模型，它使用采样的方法融合了细节信息较多的浅层特征和语义信息较多的深层特征，以此提高模型对小目标的表达能力。

Hu 等^[21]针对在特定场景下的人脸检测提出了一系列增强小目标检测率的方法：增加目标的上下文信息；对不同尺度的图片进行检测，然后对检测结果进行非极大值抑制得到最终检测结果；选用针对特定尺度目标的候选框生成模块。

Li 等^[22]利用生成式对抗网络对小目标的特征进行训练，试图使小目标的特征与大目标所呈现的特征更相似，从而提高通用检测模型的检测性能。

Cai 等^[23]借鉴了集成学习的思路，提出了级联 RCNN（Cascade RCNN）模型，将双阶段的 RCNN 扩展到多阶段，进一步提高小目标的检测精度和位置的定位精度。

1.3 本文的主要研究内容

本文重点研究基于深度学习的小目标检测问题。通过对目标检测模型的各个检测流程进行拆解分析，结合小目标的特点，本文从四个方面对现有的目标检测模型做出改进。

首先是数据预处理部分。考虑到小目标细节特征少的特点，本文使用超分辨率生成式对抗网络对包含小目标的图片进行超分辨率重建。这一步主要是为小目标增加纹理等细节信息，让小目标有更强的特征表达。

其次是解决小目标数据量不足的问题。由于常见数据集包含的小目标数据量远少于常规尺寸目标的数据量，为了避免数据量不平衡带来的模型过拟合问题，本文提出了随机扩充的数据增强方法，增加了小目标的数据量。

第三，本文基于单阶段检测模型 SSD，通过对对其原理和模型结构的研究，增加了 SSD 模型中对浅层特征层的检测个数。在卷积神经网络中，浅层特征图往往含有较多的细节特征，适合小目标检测，而深层特征图含有较多的语义特征，适合大尺寸目标的检测。这项改进有利于提高检测模型对小目标的关注度。

第四，本文分析了检测算法的最后一个模块：预选框过滤部分。主流目标模型在过滤重复的候选框时通常使用非极大值抑制算法，该算法在检测目标拥挤的场景时容易过滤掉其它实例的候选框，从而发生漏检现象。本文使用改进型非极大值抑制算法，该方法可以有效避免这种现象发生，从而减少小目标的漏检概率。

1.4 章节安排

本文的内容结构安排如下：

第一章为绪论。该章首先对目标检测领域的发展历程做了简要介绍，比较了传统检测方法和基于深度学习的检测方法的不同之处。随后引入了小目标检测问题的研究目的与应用场景，并介绍了国内外学者在小目标检测领域的研究现状。

第二章为目标检测和深度学习的基础知识。对于目标检测，该章介绍了目标检测的数学定义，评价指标与常见的数据集。对于深度学习，该章介绍了神经网络中神经元和激活函数的概念以及卷积的定义，最后介绍了深度学习中常见的模型训练技巧。

第三章为基于超分辨率生成式对抗网络的图像增强方法。该章介绍了生成式对抗网络的原理以及如何通过生成式对抗网络得到超分辨率图片。随后对比了检测模型在不同数据集的检测性能，并对结果进行分析。

第四章为基于 SSD 模型的三个改进方法。该章介绍了针对小目标的数据增强方法，SSD 模型中锚框的基本概念与改进方法，以及改进型非极大值算法的相关原理。本文将这三种改进集成到 SSD 模型中并通过 PASCAL VOC 数据集进行测试，分析模型的检测性能。

最后为总结与展望。该部分主要对本文的工作进行总结。简要说明了本文主要的工作内容与实验结论并且指出了本文工作的一些不足之处以及未来的研究方向。

第二章：目标检测与深度学习基础

2.1 目标检测基础

2.1.1 目标检测任务定义

目标检测在学术界目前还没有一个通用的定义，但在工业界一般认为目标检测任务是在图片中找出给定类别的实例并给出其定位信息。该任务用数学符号描述如下：设 I 为一张图片， $O(I)$ 表示图片 I 内的 N 个实例及其相关描述， $O(I) = \{(c_1, d_1), \dots, (c_N, d_N)\}$ ，其中 $c_i \in C$ ， C 表示种类的集合， $d_i \in D$ ， D 表示实例描述的集合，如物体的位置，尺度大小等信息。于是目标检测可表示为：

$$G(I) = \{(c_{pred_1}, d_{pred_1}), (c_{pred_2}, d_{pred_2}), \dots\} \quad (2-1)$$

式中 G 表示检测函数。

于是，如何让 $G(I)$ 输出的信息与 $O(I)$ 尽量一致便成了目标检测领域的研究人员努力思考的一件事。

2.1.2 目标检测评价指标

交并比（Intersection over Union, IoU）是用来评价两个区域重合率的重要指标，如图 2-1 所示，其计算公式为：

$$IoU(d_{pred}, d_{gt}) = \frac{Area(d_{pred} \cap d_{gt})}{Area(d_{pred} \cup d_{gt})} \quad (2-2)$$

其中， d_{gt} 表示物体在图片的原始区域（Ground Truth）。

从公式可以看出 IoU 的值限定于 $(0, 1)$ 之间，IoU 趋近于 1 表示两个区域的重合度越大，趋近于 0 则重合度越小。

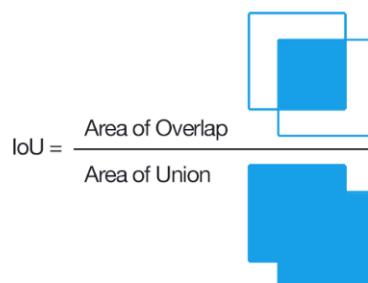


图 2-1 交并比示意图

目标检测模型会为 IoU 设定一个阈值，通常为 $\Omega=0.5$ 。在物体类别预测正确的前提下，如果预测值与真实值的 IoU 大于 0.5，则认为此次预测成功，反之则认为预测失败。可用如下公式表示：

$$\text{Predictiton} = \begin{cases} \text{Positive} & c_{pred} = c_{gt} \text{ and } \text{IoU}(d_{pred}, d_{gt}) > \Omega \\ \text{Negative} & \text{ohter} \end{cases} \quad (2-3)$$

查准率与查全率是机器学习中常用的分类指标。设置以下四种样例：真正例（True Positive, PT）、真反例（True Negative, TN）、假正例（False Positive, FP）、假反例（False Negative, FN）。其中真正例和假正例分别表示模型正确预测的正例和错误预测的正例。以图 2-2 为例，蓝色框为真实物体框，其余颜色的框都是检测模型预测的目标框。以 0.5 的 IoU 阈值为判断标准，只有绿色框能正确预测，因此绿色框为真正例（TP）。而检测模型认为紫色框含有物体，但实际为图片背景，预测失败，因此紫色框是假正例。其余颜色的预测框由于 IoU 达不到 0.5 的阈值，因此也被判定为假正例。

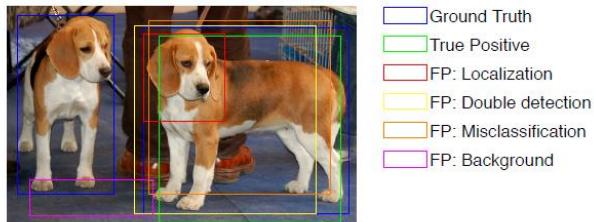


图 2-2 正例负例示意图

在此基础上，查准率（Precision）和查全率（Recall）可定义为：

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2-4)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (2-5)$$

查准率反映了对模型做出正例预测的可信度，而查全率则反映了模型找出所有正例的能力。这两个指标适用于不同场景，比如股票预测系统对查准率有较高的要求，希望模型在做出股票将会上涨的预测时具有较高的可信度。而自动驾驶领域十分看重查全率，希望模型对任何可能是行人的物体进行识别，对虚警的容忍度比较高。查准率和查全率是此消彼长的关系。查准率高的同时往往会带来漏检现象，漏检则会降低查全率，而查全率提高的同时往往会产生虚警现象，从而降低查准率。为了更好地评价一个模型的性能，综合考虑查全率和查准率后用平均查准率（Average Precision, AP）作为分类模型评价指标。通过调整 IoU 的阈值，以查全率和查准率为坐标轴，可以画出如图 2-3 所示的曲线，该曲线也叫 PR 曲线

(Precision Recall Curve)。 曲线与坐标轴包围的面积作为平均查准率 AP 的定义, AP 通常介于 (0, 1) 之间。另外, 考虑到多种类分类器有各自的查全率和查准率, 通常会对各种类的平均查准率取平均值得到 mAP (mean Average Precision)。

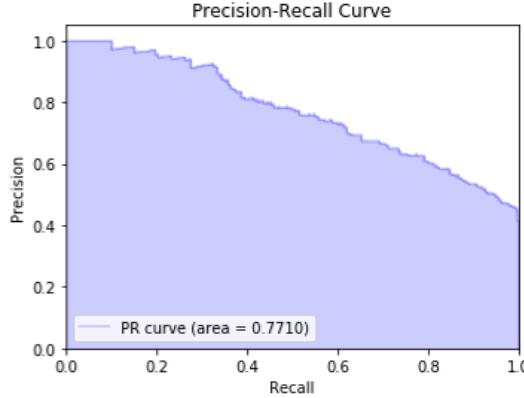


图 2-3 PR 曲线

2.1.3 目标检测损失函数

目标检测有分类和定位两项基本任务, 因此也具有分类和定位两种损失。分类损失通常使用交叉熵 (Cross Entropy):

$$CE(c_{pred}, c_{gt}) = -\sum_i c_{gt_i} \log(c_{pred_i}) \quad (2-6)$$

式中, c_{gt} 表示目标真实类别, c_{pred} 为预测类别, 计算公式如 (2-7) 所示。

$$c_{pred_i} = \text{softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad (2-7)$$

分类损失函数一般不用均方误差 (Mean Squire Error, MSE)。原因在于分类问题输出分类概率时会使用 softmax 函数, 而 softmax 函数与 MSE 结合得到的数学形式不满足凸优化定义。而使用交叉熵损失函数则不存在这个问题。

为了增加对难分类物体的关注度, Lin 等人提出了 Focal Loss 损失函数。Focal Loss 通过参数 α 和 γ 增加了错分类物体的损失权重, 公式如下:

$$L_{FL} = -\alpha(1-p_i)^\gamma \log(p_i) \quad (2-8)$$

对于定位任务, 参与目标定位的参数有 4 个: (x_i, y_i, w_i, h_i) , 分别代表左上角的坐标和物体的长宽。对于连续变量的损失函数可以简单使用均方误差 MSE, 但 MSE 由于含有平方操作, 等效于对离群点有更大的关注度, 不利于训练出更精确

的预测值。为了解决这个问题，Faster RCNN 和 SSD 模型都使用了平滑的绝对值损失函数（Smooth L1 Loss），公式如下：

$$\text{SmoothL1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{other} \end{cases} \quad (2-9)$$

另外还可以将分类损失和定位损失通过加权结合在一起得到多任务损失函数。使用多任务损失能够对分类和位置回归做联合训练。多任务损失函数公式如下：

$$L(p, u, t^u, v) = L_{cls}(p, u) + \lambda[u \geq 1]L_{reg}(t^u, v) \quad (2-10)$$

其中 p, u 分别代表物体的真实分类和预测分类， t, v 分别代表物体的真实坐标信息和预测坐标信息。

2.1.4 目标检测领域常见数据集

近几年机器学习领域的高速发展离不开丰富的数据集作为支撑。本节主要介绍目标检测领域常见的三个数据集 PASCAL VOC, ILSVRC, MS-COCO。它们都是计算机视觉领域比赛专用的数据集。这些数据集数据量够足够大，覆盖面广，各有各的特点。目标检测领域的模型常常以这些数据集的检测结果作为模型运行性能的依据。

PASCAL VOC 数据集的名字来源于 PASCAL Visual Object Classes Challenge，这是一个计算机视觉领域十分重要的比赛。比赛内容包括目标分类、目标检测、语义分割和动作检测。PASCAL VOC 数据集有两个版本 VOC07 和 VOC12，前者含有 5000 张图片和 12000 个带有标注的目标，后者含有 11000 张图片和 27000 个带有标注的目标。VOC 数据集包含了 20 种生活中常见的目标，如猫、狗、船、飞机等。但随着其它更丰富的数据集的出现，如 ILSVRC 和 MS-COCO, PASCAL VOC 逐渐成为新模型试验时使用的数据集。

ILSVRC 源于 ImageNet Large Scale Visual Recognition Challenge，这个比赛从 2010 年开始举办直到 2017 年才停止。ILSVRC 数据集包含了 200 个种类，其数据量远远超过了 PASCAL VOC 数据集，包含了 51 万张图片和 53 万个带标注的目标。

MS-COCO 是目标检测领域目前难度系数最高的数据集，它包含的目标类别少于 ILSVRC 数据集，但有更多的目标个数。以 2017 年的 MS-COCO-17 数据集为例，它包含 80 个类别的 16 万张图片和 89 万个带标注的目标。MS-COCO 数据集的优点是它的目标标注信息不仅仅是矩形框，还包括精确到像素的实例分割信息，这些信息也更有利提高目标检测模型的精确度。相对来说，MS-COCO 数据集中目标会更密集，检测难度也更大。

2.2 深度学习基础

2.2.1 神经元与激活函数

神经网络由一层层的神经元结构组成。神经元模型根据生物学上的神经元特性设计，借鉴了神经元的激活过程。假设一个神经元接收 d 个输入信号 x_1, x_2, \dots, x_d ，以加权和的形式表示神经元所获得的信号：

$$z = \sum_{i=1}^d w_i x_i + b = \mathbf{w} \cdot \mathbf{x} + b \quad (2-11)$$

其中 w_i 为信号 x_i 对应的权重， b 为偏置项。

信号 z 经过一个非线性函数 $f(\cdot)$ 得到神经元的活跃值。这类非线性函数 $f(\cdot)$ 被称为激活函数（Activation Function）。这类函数通常都经过精心设计。常见的激活函数如 Logistic 函数，定义如下：

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \quad (2-12)$$

其输出值限定在 (0, 1) 之间，函数曲线如同 S 型，中间梯度大，两边梯度趋于零，函数处处可导。Logistic 函数以其良好的函数特性，在早期的神经网络应用广泛。然而 Logistic 函数也有着计算复杂的缺点，并且在 x 轴两边梯度趋近零的特点也容易出现梯度消失的现象。基于此原因，深度学习模型通常使用修正线性单元（Rectified Linear Unit, ReLU）激活函数，定义如下：

$$\text{ReLU}(x) = \max(0, x) \quad (2-13)$$

ReLU 函数计算十分简单，并且右半边的梯度始终保持为 1，在一定程度上缓解了梯度消失的问题，同时也加快了神经网络的收敛速度。虽然 ReLU 函数在原点不可导，但这个缺点在工程应用上几乎可以忽略。ReLU 函数左半边梯度为零的特性也导致了神经元输出结果为零之后就无法通过梯度下降算法更新自身参数，这种现象也被称为死亡 ReLU 问题。于是出现了几种 ReLU 函数的变种：Leaky ReLU、PReLU、ELU 等。激活函数的非线性大大增强了多层神经网络的表达能力。

2.2.2 卷积的定义

机器学习领域的卷积和傅里叶变换使用的卷积是两种不同的计算方法。机器学习领域的卷积操作是用特定的矩阵模板对另一矩阵的对应元素位置进行相乘并求和的一种操作，数学符号记为 \otimes 。图 2-4 是卷积操作的一个示例。

$$\begin{array}{|c|c|c|c|c|} \hline
 1 & 1 & 1 & 1 & 1 \\ \hline
 -1 & 0 & -3 & 0 & 1 \\ \hline
 2 & 1 & 1 & -1 & 0 \\ \hline
 0 & -1 & 1 & 2 & 1 \\ \hline
 1 & 2 & 1 & 1 & 1 \\ \hline
 \end{array}
 \otimes
 \begin{array}{|c|c|c|} \hline
 1 & 0 & 0 \\ \hline
 0 & 0 & 0 \\ \hline
 0 & 0 & -1 \\ \hline
 \end{array}
 =
 \begin{array}{|c|c|c|} \hline
 0 & -2 & -1 \\ \hline
 2 & 2 & 4 \\ \hline
 -1 & 0 & 0 \\ \hline
 \end{array}$$

图 2-4 卷积操作示意图

卷积操作在数学形式上，实际是一种简化版的全连接。卷积和全连接都是一组线性变换，但是卷积操作与全连接相比参数更加稀疏，并且非零元素实际上是共享的。这两个特点让卷积神经网络的参数大大减少。与此同时，同一套参数的复用也更有利 于捕捉局部特征。卷积的这种特性特别适合于图像处理。与全连接相比，卷积操作参数更少，计算量更小也更容易训练。虽然这种处理会使得模型的拟合能力理论上会不如全连接网络，但是从效果来看是值得做出这种折中的。然而，卷积操作并不具有尺度不变性还有旋转不变性。这两个缺陷在主流模型中有很大程度上是通过大量的数据训练进行弥补的，强行让卷积网络“记住”同一物体不同尺度和旋转时的特征。

2.2.3 卷积神经网络

卷积神经网络（Convolutional Neural Network, CNN）是一种结合了卷积操作的深层前馈神经网络。卷积神经网络一般是由卷积层（convolutional layer），池化层（Pooling layer）和全连接层（fully connected layers）交叉堆而成的前馈神经网络，并使用反向传播算法进行训练。

卷积层有两个主要参数：卷积核与卷积步长。卷积核的大小决定了参与卷积计算的元素的数量。卷积步长决定了卷积核在特征层上滑动间隔。感受野的概念如图 2-5 所示，感受野一般会随着网络层数的增加扩大。

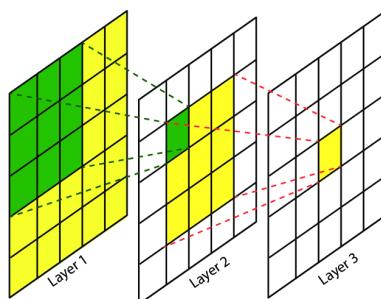


图 2-5 感受野示意图

池化层（Pooling Layer）作用是进行特征选择，降低特征数量。常见的池化操作有最大池化（Max Pooling）和平均池化（Average Pooling）。以最大池化为例，该操作会将上一层的特征图划分均等大小的网格，然后提取每个网格中的最大值组成一张新的特征图。这种操作有三个好处：一是保持上层特征不变，在一定程度上允许特征的微小位移；二是特征降维，减少输入下层的数据量；三是在一定程度上防止了过拟合现象。不过池化层并不是卷积神经网络必不可少的部件。利用卷积层也能实现类似的效果，比如采取步长为 2 的卷积操作，也能顺利减少近一半的特征图。

全连接层（fully connected layers）可以看成是传统的多层前馈神经网络，在整个卷积网络中起到分类器的作用。神经网络中卷积层，池化层，激活函数层的操作是将原始数据映射到隐形特征空间，而全连接层则负责将这隐形特征映射到样本的标记空间。全连接层由于其全连接的特性，它的权值参数往往占了大部分的模型整体参数。卷积神经网络结构如图 2-6 所示。

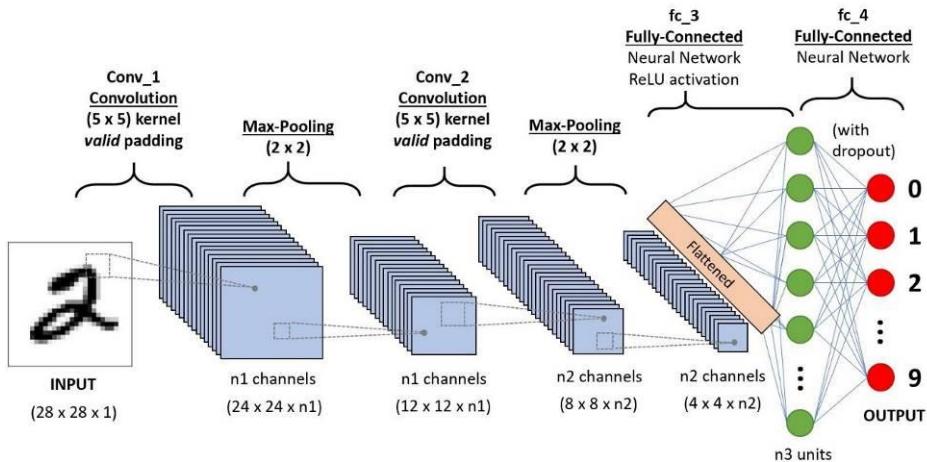


图 2-6 卷积神经网络模型结构^[24]

2.2.4 模型训练技巧

在理论上，模型的表达能力由模型的组成结构决定。但能否让模型发挥它的理论能力还得看工程上如何实现它。要得到一个与理论预期相符的模型需要搭配一些训练技巧。训练神经网络模型的技巧大多是通过试验得到的，往往是通过试验发现某些技巧十分有效，随后才反思其背后的理论依据。常见的训练技巧有随机打乱训练数据，使用指数衰减的学习率，选择基于动量的梯度下降算法，以及批处理化等。

训练更深层的神经网络一直是提高深度学习模型性能的重要手段之一，但直接增加模型层数又常常会导致梯度消失现象。为了使深层网络模型训练时保持稳定，Google 公司提出了批处理化操作（Batch Normalization, BN）。其算法流程如表 2-1 所示。批处理的目标就是使神经输出层的数据尽量保持同样的均值和方差。即使深层网络的响应或梯度很小，也可以通过批处理化将其尺度放大。此操作缓解了深层网络可能带来的梯度消失问题。另外，神经网络训练收敛速度慢和梯度爆炸问题也可以通过批处理化来解决。批处理化操作以其良好且普适的应用效果，被广泛应用在深度学习模型中。

表 2-1 批规范化算法

算法 1 批规范化算法 BN

输入：批处理（mini-batch） 输入 $x: B = \{x_1, \dots, x_m\}$

输出：规范化后的网络响应 $y_i = BN_{\gamma, \beta}(x_i)$

1: $\mu_\beta \leftarrow \frac{1}{m} \sum_{i=1}^m x_i$ // 计算批处理数据均值

2: $\sigma_\beta^2 \leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_\beta)^2$ // 计算批处理数据方差

3: $\hat{x}_i \leftarrow \frac{x_i - \mu_\beta}{\sqrt{\sigma_\beta^2 + \epsilon}}$ // 规范化

4: $y_i \leftarrow \gamma \hat{x}_i + \beta$ // 尺度变换和偏移

5: return 学习的参数 γ 和 β

2.3 本章小结

本章分为目标检测基础与深度学习基础两个部分。对于目标检测基础部分，本文详细介绍了目标检测的定义，评价指标，损失函数和该领域常见的测试数据集。对于深度学习基础部分，本文简要介绍了神经网络中神经元和激活函数的概念，介绍了卷积的定义，还介绍了卷积网络的组成，以及深度学习模型的训练技巧。这些理论知识为理解后续的模型结构的设计打下了坚实的基础。

第三章：基于 SRGAN 的图像超分辨率重建

3.1 引言

小目标检测精度低的一个重要原因就是目标尺寸与常规物体相比太小了。从卷积的数学定义中可知卷积神经网络并不具有尺度不变性。基于此原因，如何解决小目标的尺度问题也就成了研究小目标检测绕不开的一道坎。为了提高模型对小目标的检测率，一个朴素的想法就是放大图片进行检测，但是 Hu 等人通过实验说明了单纯放大图片对提高小目标的检测率并没有太大助益^[21]。以信息论的观点来看，一张图片的信息量是固定的，经过线性放大图片并不会增加其信息量，因此单纯放大图片起到的作用不大。本文尝试在此基础上对放大的图片进行图像增强，通过使用超分辨率生成式对抗网络（Super-Resolution Generative Adversarial Network, SRGAN）^[25]实现小目标的图像增强目标，提高模型对小目标的检测率。

3.2 SRGAN 模型原理

超分辨率的定义是通过软件或是硬件的方法提高原本图像分辨率。传统的超分辨率图像复原是通过多张在同一场景上拍摄的低分辨率图片实现高分辨率重建，其本质是以时间带宽换取空间分辨率。随着深度学习在图像处理领域的广泛应用，Christian 等提出了基于单张图片的超分辨率生成式对抗网络 SRGAN。虽然 SRGAN 在过程上是基于单张图片实现超分辨率，但模型的具体参数实际上是通过大量图片数据训练得到的。因此在本质上，模型在图像生成的过程中间接使用了大量的图片信息。基于深度学习的超分辨率图像复原的效果也往往要比传统方法好得多。

SRGAN 源于生成式对抗网络（Generative Adversarial network, GAN）^[26]。GAN 的发展历史并不长，由 Goodfellow 在 2014 年提出。GAN 基于博奕论场景，生成器必须与判别器竞争。生成器（Generator）直接产生样本 $x = G(z; \theta^{(g)})$ ，其中 z 为随机噪声， $\theta^{(g)}$ 表示生成器的网络参数。而判别器（discriminator）作为对手，则试图区分当前样本是从生成器得到的还是从原始样本集得到的。判别器通过判别函数 $D(x, \theta^{(d)})$ 来给出自己的判断结果。通常输出结果在 $(0, 1)$ 之间，输出值越接近于 1 表示判别器认为当前样本更可能来自于原始数据集。生成器和判别器之间是零和博奕，它们对应的是两个优化过程。通常情况下用交叉熵作为分类问题

的损失函数。对于生成器，希望其生成更逼真的样本，对应公式为 (3-1)。对于判别器，希望其有更强的判别能力，其损失函数对应公式 (3-2)。

$$\min_G V(D, G) \triangleq E_{z \sim P_z(z)} [\log(1 - D(G(z)))] \quad (3-1)$$

$$\max_D V(D, G) \triangleq E_{x \sim P_{data}(x)} [\log D(x)] + E_{z \sim P_z(z)} [\log(1 - D(G(z)))] \quad (3-2)$$

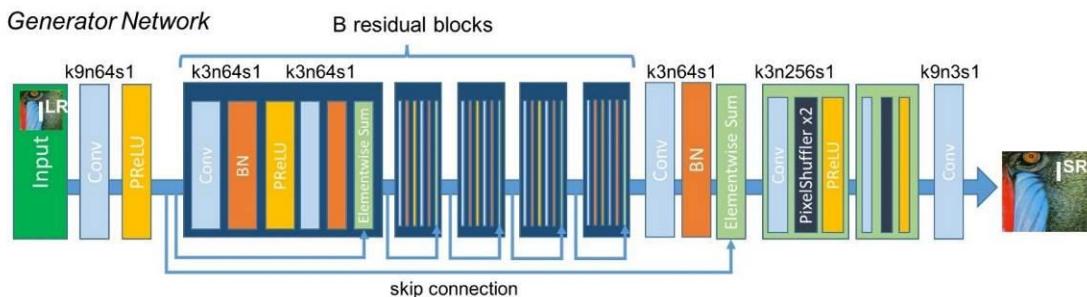
综合两者，可以得到 GAN 网络整体的损失函数：

$$\min_G \max_D V(D, G) \triangleq E_{x \sim P_{data}(x)} [\log D(x)] + E_{z \sim P_z(z)} [\log(1 - D(G(z)))] \quad (3-3)$$

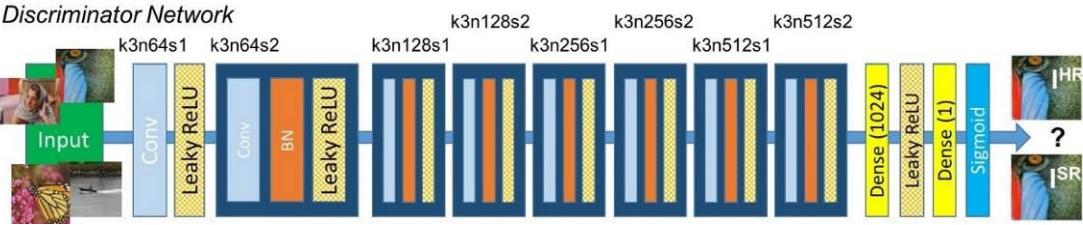
GAN 的原理简单，但是工程上训练困难。因为这两个网络的优化目标正好相反，训练过程一般是交替进行，先固定一方的参数然后训练另一方。判别器的任务比生成器简单，因此更容易训练，但如果判别器太强，则生成网络不容易学到有用的信息，不利于生成网络的提升。因此如何保持判别器和生成器的微妙平衡是训练 GAN 的关键。

超分辨率生成式对抗网络是第一将 GAN 应用到超分辨率重建领域的深度学习算法。在超分辨率重建领域，常见的模型都使用均方误差作为损失函数，这样做虽然能取得较高峰值信噪比，但是同时也存在高频细节缺失的问题。SRGAN 更改了损失函数，将其替换为对抗损失 (perceptual loss) 和内容损失 (content loss)，其中内容损失为生成图与原图在卷积层中的特征图的均方误差。深层卷积提取得到的特征图通常更加抽象，含有更强的语义性，因此减少内容损失有利于生成的图片在视觉上更靠近真实图。

SRGAN 中生成器模型如图 3-1 a) 所示，判别器模型如图 3-1 b) 所示。生成器的主要任务是输入一张低分辨率的图片，经过卷积网络和多个残差块 (residual blocks) 后生成一张高分辨率的图片。而判别器的工作便是判别输入的图片是来源于生成器还是来源于原始数据集。输入图片在判别器中经过卷积层和激活函数层提取高阶的语义特征，通过设置卷积的步长来降低数据的维度，最终经过一个二元分类器输出判别结果。



a) 生成器结构图



b) 判别器结构图

图 3-1 SRGAN 结构图^[25]

SRGAN 的图像生成效果如图 3-2 a)所示：



a) 双线性内插法

b) SRGAN

c) 原图

图 3-2 图像复原对比^[25]

3.3 SRGAN 对小目标的超分辨率重建

为了体现 SRGAN 的图像增强效果，本文从 PASCAL 数据集中提取一张照片进行实验。图 3-3 a)是高分辨率的原图，图 3-3 b)是对原图进行 4 倍下采样的低分辨率图片，图 3-3 c)是对低分辨率图片进行双线性内插法得到放大 4 倍的图片，图 3-3 d)是 SRGAN 生成的图片。关于双线性内插法的详细算法流程可在附录三中查看。相比直接对像素进行复制的放大方法，通过双线性内插法得到的图像整体观感较为平滑，像素值较为连贯，但是没能恢复出图片细节，整体观感仍然是模糊的。通过比对可以发现 SRGAN 对图片的恢复效果要远远好于双线性内插法。SRGAN 能够对图片放大 4 倍同时进行细节信息的增强。由于训练集中包含大量动物的图片，SRGAN 对眼睛，嘴巴等面部细节的修复能力较强，这些细节对检测小目标而言至关重要。

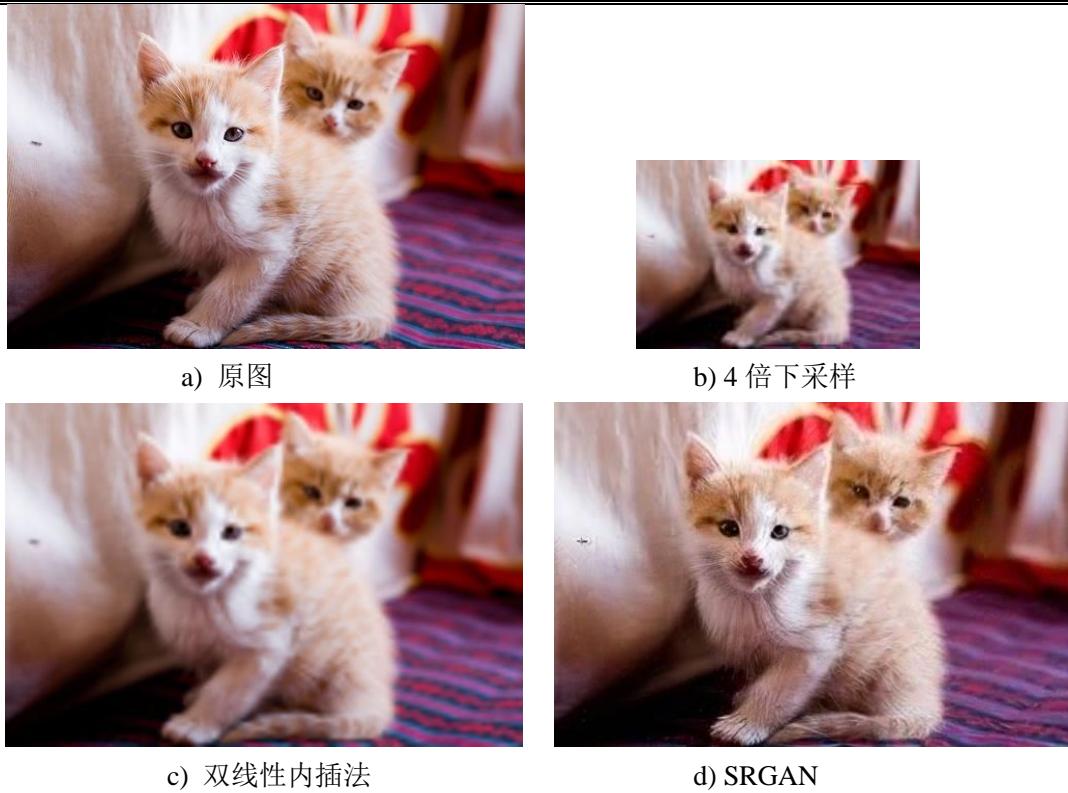


图 3-3 SRGAN 输出图片对比

3.4 实验结果及分析

为了全面检验 SRGAN 处理后的数据是否能够提高检测模型对小目标的检测效果，本文做了以下实验进行对比。本文采用 PASCAL VOC 数据集作为训练集和测试集。PASCAL VOC 数据集有 4592 张图片，其中包含了 20 种生活中常见的目标，包括轮船，飞机，猫，狗等，适合用来评估模型的检测性能。首先，本文通过数据集中带有的标注信息，计算出图片中各类目标的相对大小，结果如表 3-1 所示。实验过程中，本文将重点关注列表前几项面积相对较小的种类。

表 3-1 VOC 数据集数据分析

类别	实例个数	平均相对面积
瓶子	468	0.0559
盆栽	480	0.1175
椅子	756	0.1255
电视	308	0.1278
羊	242	0.1282
人	4527	0.1637

表 3-1（续表）

类别	实例个数	平均相对面积
牛	244	0.1706
船	263	0.1815
鸟	459	0.1979
汽车	1201	0.2117
自行车	337	0.2444
飞机	285	0.2447
摩托车	325	0.2740
公交	213	0.3078
马	348	0.3087
餐桌	206	0.3531
狗	489	0.3590
火车	282	0.3602
沙发	239	0.3629
猫	358	0.4209

本文以目标检测领域主流的单阶段检测模型 SSD 作为检测器。首先，本文对 VOC 数据进行 4 倍下采样得到低分辨率图片，然后通过 SRGAN 得到复原的超分辨率图片。依次对比 SSD 模型对不同分辨率图片的检测性能，并将实验结果记录于表 3-2 中。受限于计算机硬件的运算能力，本文训练的 SSD 模型的性能与 SSD 模型原作者训练出的模型的性能还有一定的差距。表中 LR, SR, HR 分别表示使用低分辨率图片、SRGAN 生成的超分辨率图片、高分辨率的原图。原作者权重表示 SSD 作者在网上公开的模型权重，该权重经由高性能计算机通过长时间训练得到，自训练权重表示本文从随机权重开始训练 SSD 后最终得到的权重。

表 3-2 不同权重模型对不同分辨率图片的检测效果对比

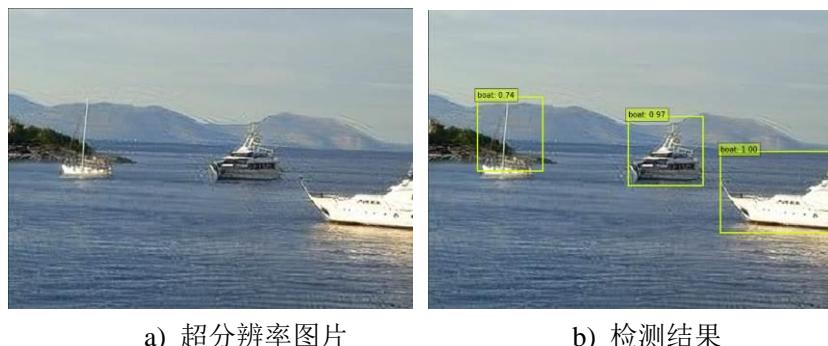
模型	检测数据	检测指标(mAP)
SSD+原作者权重	LR	48.57%
SSD+原作者权重	SR	54.96%
SSD+原作者权重	HR	77.43%
SSD+自训练权重	LR	39.69%
SSD+自训练权重	SR	46.27%
SSD+自训练权重	HR	60.75%

根据表 3-2 可以发现，自己训练的模型在性能上离原作者训练的模型还有很大的差距。以检测原图的 mAP 为参考指标，两者的检测性能有着 17%mAP 的差距。通过分析，可以确定问题原因在于训练时长上。为了达到最佳的检测性能，原作者在高性能的计算机上对模型训练了近一周的时间直到损失函数逐渐趋于一个平稳值。计算机的性能则直接影响到训练时长，性能越强大的计算机搭配大容量的 GPU 可以并行处理数据，缩短训练时长。考虑到训练时长，本文并没有对模型进行长时间的训练。虽然实验环境不足以支持训练高精度的模型，但这并不影响本文对小目标检测问题的研究。

对比模型在不同数据集的表现，低分辨率图片经过 SRGAN 模型的超分辨率复原后可以提高模型大约 7% mAP 的检测性能。经过 SRGAN 复原的图片增加了目标的细节信息，从而有效提高了 SSD 模型的检测性能。这一结论在两种不同权重的 SSD 模型上都能成立。如图 3-4 与图 3-5 所示，通过对比低分辨率图片和超分辨率图片的检测结果，可以看出细节信息对小目标检测的重要性。虽然 SRGAN 处理后的图片比低分辨率图片在模型检测上表现得更好，但相比高分辨率的原图还是有很大的差距。因此使用 SRGAN 对图像进行增强的方法还有很大的提升空间。



图 3-4 低分辨率图片检测情况



a) 超分辨率图片

b) 检测结果

图 3-5 超分辨率图片检测情况

为了进一步分析 SRGAN 对小目标检测率的影响，本文提取了相对面积较小的前十个类别的检测情况。对比低分辨率图片和超分辨率图片在这些类别上的检测

结果。为保证实验结果的准确性，本次检测使用同一个 SSD 模型，使用 SSD 原作者的模型权重。检测结果如表格 3-4 所示：

表 3-4 低分辨率图片与超分辨率图片检测结果

AP	使用低分辨率图片	使用超分辨率图片
瓶子	20.3	16.45
盆栽	21.04	26.24
椅子	28.29	35.24
电视	45.32	47.63
羊	41.75	54.49
人	54.08	54.29
牛	42.72	52.47
船	32.49	42.30
鸟	38.87	52.15
汽车	60.81	61.93

通过对比可以发现：在使用超分辨率图片后，大部分种类的平均查准率（AP）有一定的提升，这说明对图片进行超分辨率重建后确实有利小目标的检测率。但是对于特定的种类，如瓶子这一种类的平均查准率反而有较大的下降。对于瓶子种类，其检测平均查准率下降的原因可能是 SRGAN 模型在训练过程中，训练的数据集中并没有足够的瓶子目标，因此导致 SRGAN 不仅不能对瓶子进行图像复原，反而导致其形状特征发生改变，这一现象可通过图 3-6 查看。



a) 低分辨率图片检测结果 b) 超分辨率图片检测结果

图 3-6 对比低分辨率图片与超分辨率图片的检测结果

另外，如图 3-7 所示，可以观察到，瓶子这一种类在图片中的尺寸较小并且其出现的场景往往是多个瓶子聚集在一起，不同瓶子的外观也有不同颜色的包装，瓶身的反光等多种原因导致了瓶子这一类别的检测率很低。

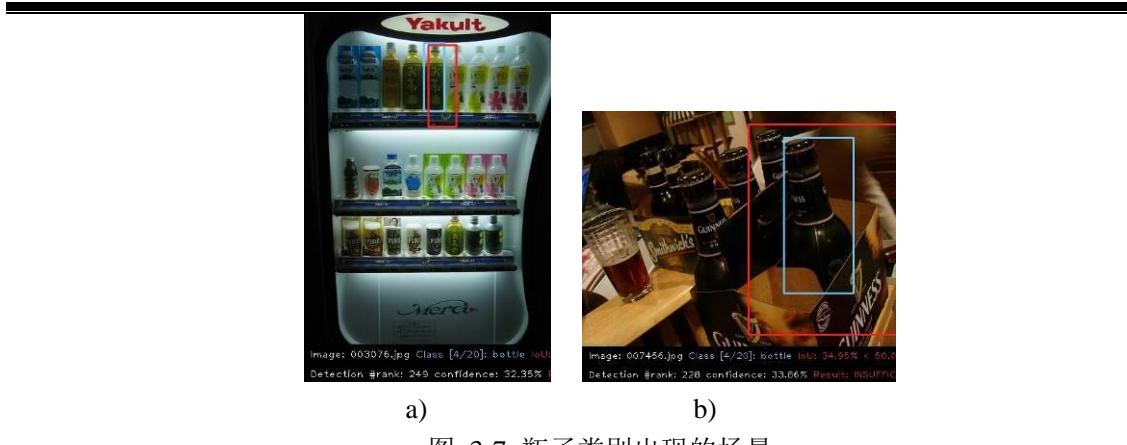


图 3-7 瓶子类别出现的场景

3.5 本章小结

本章从数据集的预处理出发，思考如何放大小目标而不损失其细节信息。本章首先介绍了 SRGAN 的基础原理与设计思路。通过 SRGAN 的处理，可以对图片放大四倍的同时增强图片的细节信息。为了体现 SRGAN 的实际应用效果，本文对原图片进行了四倍下采样，得到低分辨率图片，然后通过 SRGAN 将低分辨率图片转换为超分辨率图片。通过对低分辨率图片与超分辨率图片的检测效果，得到结论：SRGAN 能提高模型对小目标的检测性能。

第四章：基于 SSD 的小目标检测模型

4.1 引言

本章以 SSD 模型为基础展开对小目标检测问题的研究。在第三章，本文提出了使用超分辨率生成式对抗网络对图像进行图像恢复，保证小目标在放大的同时增加其细节信息，相关实验也验证了该方法的有效性。本章则从 SSD 模型的模型结构和训练策略入手，探讨如何修改 SSD 模型使其更加适合小目标检测问题。最后通过实验去检验这些改进对模型检测性能的影响。

4.2 SSD 模型结构及算法流程

SSD^[10]模型全称为（Single Shot MultiBox Detector），其最大的创新点在于它选择在不同深度的特征图上进行目标检测。这样做的好处在于，模型能同时兼顾浅层特征的细节信息和深层特征的语义信息。根据卷积神经网络的特点，不同深度的特征图适合识别不同尺度的目标。浅层特征图涉及更多的像素操作，往往具有更多的特征细节，适合识别小尺度目标。而深层特征图往往更抽象，含有更多的语义信息，适合识别大尺寸目标。

SSD 模型的结构如图 4-1，它以经典的分类模型 VGG 作为主干网络，并且替换其中的全连接层。在训练时，SSD 模型直接使用预训练好的 VGG 权重，VGG 网络则起到特征提取的作用。除此之外，为了得到更深层的语义特征，SSD 又增加了四个尺寸依次减小的卷积层。接着，SSD 在特定的卷积层提取的特征图上进行网格，执行分类和定位任务。

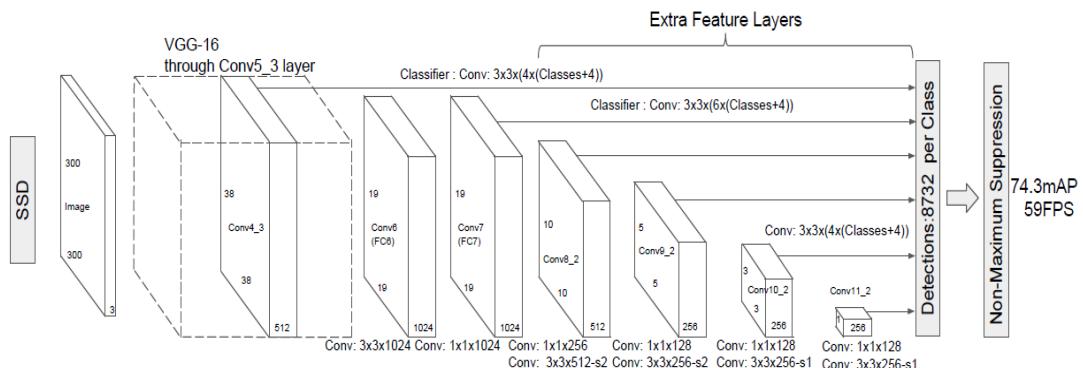


图 4-1 SSD 模型结构^[10]

根据特征图所处的不同深度，划分不同大小的网格。越深层的特征图语义特征较强，往往适合识别大尺寸物体，因此划分的网格尺寸也越大。反之，越浅层的特征图划分的网格尺寸就比较小而且密集。对特征图进行均等网格划分后，检测器在这些网格上滑动并将网格内的特征送入检测器，得到分类和定位结果。

为了更好得到分类和定位结果，SSD 引入了类似双阶段模型 Faster RCNN 中锚框（anchor box）的概念。锚框的引入主要是为了处理同一网格内可能存在多个目标的情况。单阶段模型在定位时通常使用的是回归的方法，它们基于这样一种强假设：当预测框和真实框足够近时，它们之间的坐标变换是线性的。因此，为了更精准的定位目标，模型就得给出不同长宽比例的锚框以适应不同长宽比例的目标。此外，由于一个锚框对应一个目标，多个锚框的设计也同时解决了多目标在同一网格的情况。

SSD 模型的损失函数为多任务损失：

$$L(x, c, l, g) = \frac{1}{N} (L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \quad (4-1)$$

其中 L_{conf} 为交叉熵损失，公式为(2-6)， L_{loc} 为 Smooth L1 损失函数，公式为(2-9)， α 的取值通常为 1， x, c, g, l 分别表示图片数据，数据类别，预测的目标定位信息，真实的目标定位信息。

在训练时，大量锚框的引入也生成了大量的负例，而正例的数据非常少。正负例的不平衡会导致模型难以得到充分训练。于是 SSD 采用了困难样本挖掘（Hard Example Mining, HEM）^[27]，该算法可以对负例样本的置信度进行排序，控制正负样本的比例在 1:3 之间。

SSD 模型的流程如图 4-2 所示。首先，将图片输入特征提取网络，得到多张特征图。然后在不同深度的特征图上通过两个 3×3 卷积层分别得到定位信息和分类信息。最后将各层的预测结果进行汇总，通过非极大值抑制筛选重复预测框，输出最终预测结果。

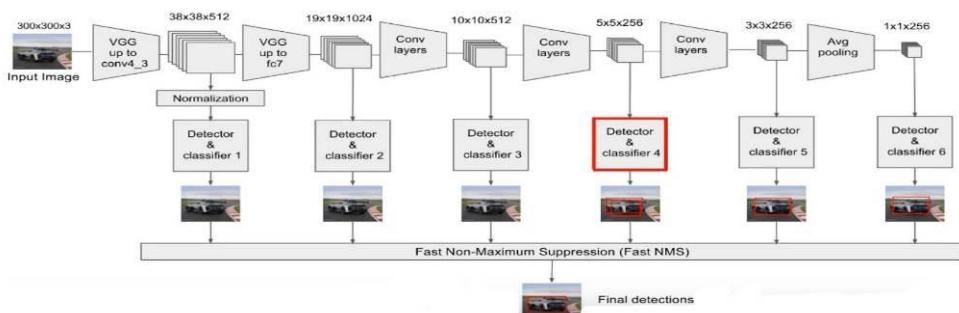


图 4-2 SSD 检测流程

4.3 基于 SSD 模型的改进方法

上一节简单介绍了 SSD 模型结构和算法流程，其中可以看出 SSD 模型是面对常见目标的通用检测模型。虽然 SSD 对常见目标的检测率达到了当前主流模型的顶尖水平，但其对小目标的检测率还远远不能令人满意。为了适应小目标检测问题，SSD 模型还需要进行一些特定的改进。结合当前国内外学者对小目标检测问题提出的解决思路，本文从三个方面对 SSD 模型进行了改进，分别是针对小目标的数据增强，增加模型的多尺度信息，采用改进的非极大值抑制算法。

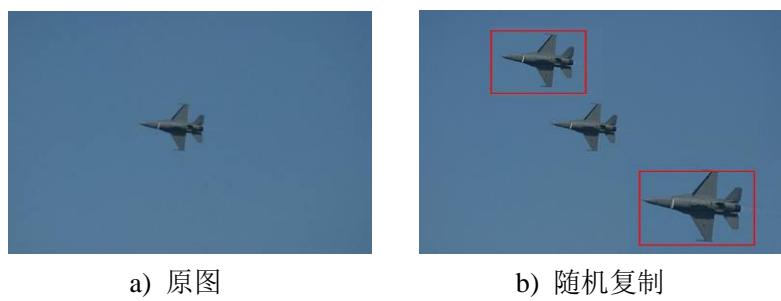
4.3.1 针对小目标的数据增强

数据增强（Data Augmentation）是指对训练数据进行变换以解决训练数据不平衡导致的模型过拟合现象。深度学习领域的模型需要大量的数据对模型进行训练。当数据量不足时，为了使模型损失函数值减小，重复使用同一批数据对模型进行训练就容易使模型过拟合。过拟合的模型往往对训练集数据的检测率高，而对测试集数据的检测率低。这样的模型泛化能力不够强，现实应用价值不够高。为了缓解这一困境，数据增强方法可以在保证图像可识别的前提下，对训练数据进行扩充。在计算机视觉领域，常见的数据增强方法有：水平翻转，饱和度调整，亮度调整，随机角度旋转，随机剪裁等，如图 4-3 所示。这些数据增强操作缓解了模型训练数据量不足的困难，提高了模型的泛化能力。



图 4-3 常见的数据增强方法

在前文的分析中可以知道，模型对小目标识别率不高的一个重要原因就是训练图片中包含小目标的图片数量相比正常尺寸的目标少得多。这样就导致了数据量的不平衡性，导致模型过分关注于识别常规尺寸的目标。为了解决这个问题，本文参考 Kisantal 等^[28]的论文，设计了针对小目标的数据增强方法。根据数据集的标记信息，将图片中的目标进行复制，并随机乘以一个尺度因子 a ， $a \in (0.5, 1.5)$ ，粘贴在原图的其他角落。需要注意的是，在粘贴过程中，需要保证粘贴的图片不能覆盖住原有的小目标。结果如图 4-4 所示。这样操作在一定程度上缓解了小目标与中尺寸和大尺寸目标的数量的不平衡性。



a) 原图

b) 随机复制

图 4-4 针对小目标的数据增强方法

4.3.2 调整锚框尺寸

SSD 模型选择在不同深度的特征图上进行目标检测。一般来说，随着卷积神经网络层数的增加，越深层的卷积层提取的特征就越具有语义性，也更加适合检测。但对于小目标检测而言，网络层数增加反而会降低识别率。小目标本身信息量有限，因此对目标检测模型必须借助小目标的细节特征对其进行识别。然而卷积神经网络用卷积操作实现特征提取的同时往往也会丢失小目标的细节信息。如图 4-5 所示，以杯子图像为例，浅层特征如图 4-5 a) 所示，浅层特征还保留了大量的细节特征。而到了最后，深层卷积层提取的特征就几乎不可识别了，此时细节特征已经丢失。对于大尺度目标而言，其特征足够丰富，细节的丢失对检测结果影响不大。但对小目标而言，它们的区分度来源往往就是这些细节特征。

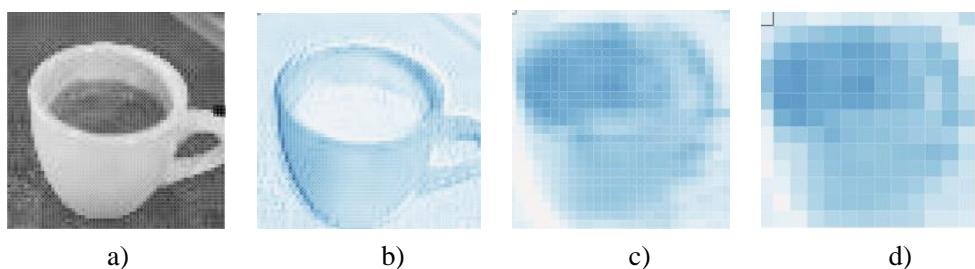


图 4-5 神经网络中不同深度的特征图

SSD 挑选了 6 个不同深度的特征图，然后对特征图划分出大小均等的网格。越低层的特征图划分的网格就越密集，越深层的特征图划分的网格就越稀疏。这样设计有其合理性。底层的特征图主要负责识别小目标，而小目标的尺寸决定了网格必须足够密集，这样才能覆盖图片的各个角落。而深层的特征图主要负责识别大目标，大目标占据了图片的大部分面积，因此网格可以相对较大，以减少不必要的计算。SSD 给出了对应不同特征图的锚框尺度计算公式：

$$s_k = s_{\min} + \frac{s_{\max} - s_{\min}}{m-1}(k-1), \quad k \in [1, m] \quad (4-2)$$

其中 $s_{\min} = 0.2$, $s_{\max} = 0.9$, m 表示选取的特征图的数量, k 表示第 k 个特征图。

将图片大小乘对应的尺度便得到了默认的锚框的大小，以 300×300 的图片为例，最小的特征图对应的尺度系数为 0.2，因此默认锚框的大小为 60×60 。该锚框大小也说明 SSD 模型能够识别尺寸大约为 60×60 的小目标。而每个网格还对应了 6 种不同长宽比的锚框，长宽比参数为 $a_r \in \{1, 2, 3, \frac{1}{2}, \frac{1}{3}\}$ ，对应的长和宽计算公式分别为(4-3), (4-4)。不同尺寸的锚框适合识别不同形状的目标。

$$w_k^a = s_k \sqrt{a_r} \quad (4-3)$$

$$h_k^a = s_k / \sqrt{a_r} \quad (4-4)$$

在模型训练时，锚框必须与真实框的交并比大于 0.5 才被认为是正例，否则以负例处理。以图 4-6 为例，当网格为 8×8 时，对于猫这一类别，左下角的网格中有两个锚框（蓝色）与目标的重叠比例满足阈值，因此被当成正例，其余框则被当成负例。当目标类别是狗时， 8×8 的网格对应的锚框相对较小，与目标的重叠比例不满足阈值，因此都被当成负例，只有在 4×4 的网格上才能找到满足重叠比例的锚框。锚框的尺寸和长宽比例尽可能适应各个目标常见的尺寸形态，保证在某一特征图的锚框内能较精确地框选住目标。

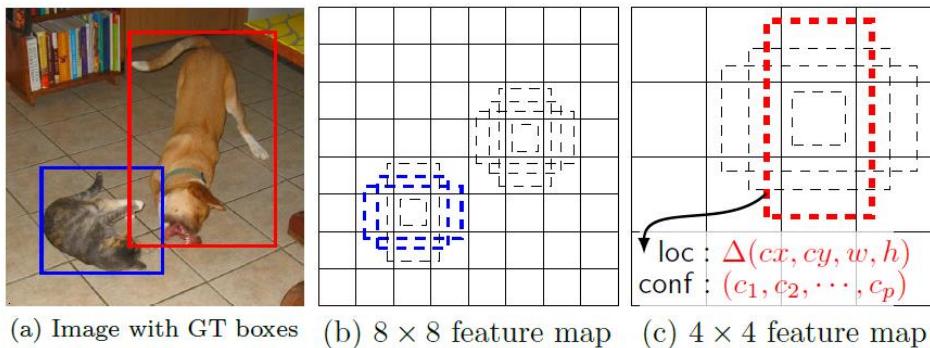


图 4-6 锚框示意图^[10]

正如上文所说的，SSD 最小的默认锚框尺寸为 60×60 ，这对小目标检测而言是远远不够的。根据 COCO 数据集的定义，小目标尺寸介于 32×32 之间。因此，为了提高小目标的检测率，有必要对 SSD 模型的 anchor 设置得更小。同时，为了不影响 SSD 模型原有的检测精度，本文设想在 conv3_3 卷积层提取的特征图上进行目标检测。同时将锚框的最小尺寸比例 s_{min} 设为 0.1。这样设计的结果是默认锚框的大小为 30×30 ，更适合应用在小目标的检测场景。

另外，小目标对位置偏差的容忍度比大目标要低得多。由于小目标本身的面积就较小，同样大小的位置偏差很容易使真实框与预测框的交并比（IoU）小于阈值，而这种问题对大尺度目标的影响就相对较小。因此，本文也对 SSD 模型中关于小目标预测的 IoU 阈值调整为 0.3，而不是通常的 0.5。

4.3.3 改进型非极大值抑制算法

非极大值抑制（Non-Maximum Suppression）是目标检测领域十分常见的候选框过滤算法。一般情况下，SSD 模型中不同尺度大小，不同长宽比例的锚框可能检测到同一个目标，随后便产生了多个重复的预测框。如图 4-7 所示。

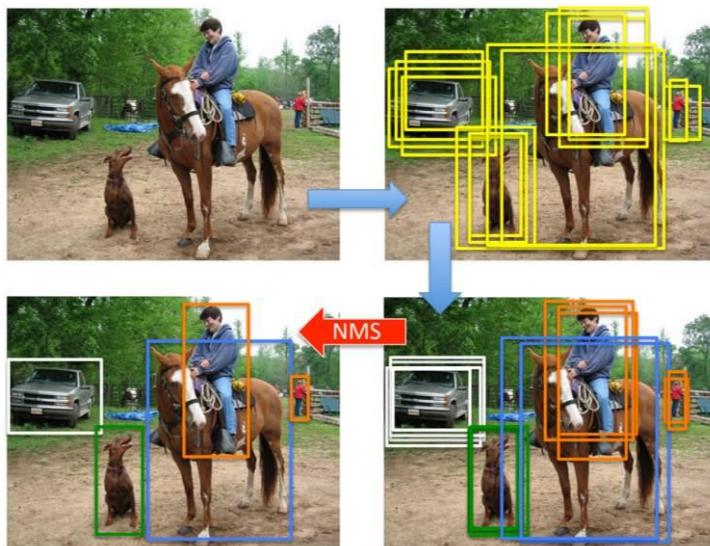


图 4-7 非极大值抑制算法流程

如何从多个重复的预测框中选出最佳的预测框成了一个难题。非极大值抑制采用贪心算法的策略，算法流程可参考附录四。首先，根据预测框的置信度 s 的大小进行排列，选出分数最大的预测框 b_i ，然后将 b_i 移除候选框队列，接着以 b_i 为参照物，如果有其他预测框与 b_i 的交并比超过某一阈值 t ，则将其从候选队列中移除。重复此过程，直到候选框队列为空。

相比直接按分数挑选候选框的算法而言，极大值抑制算法是个十分有效的方法，它可以筛选掉高置信度的重复框，保证小区域内仅有某一种类的一个高置信度的候选框。当然，不同种类的目标需要各自独立执行非极大值抑制算法，仅同一种类的候选框才会因为交并比过大而被筛选掉。但这样做也有一个缺点，那就是容易发生密集目标的漏检现象。如图 4-8 所示，以黄框为基准，绿框是重复框，而红框则是另一目标的预测框。在执行非极大值抑制算法时，容易将红框也当成是重复框而移除。这一现象在以瓶子为主要目标的图片中经常发生，这也是瓶子这一类别的识别率低的原因之一。

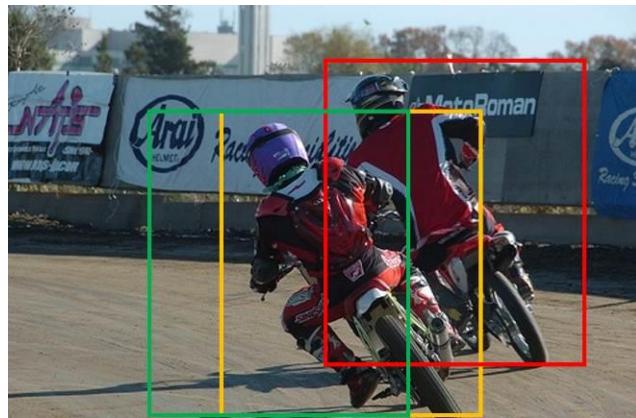


图 4-8 重复预测框与重叠预测框

为了解决这一问题，本文采用了 soft NMS^[29] 算法。soft NMS 算法相比于 NMS 只进行了一处改动。NMS 对重叠比例超过某一阈值的候选框直接移除。而 soft NMS 则通过公式 (4-5)，将重叠比例高的候选框的置信度进行下调，并且仍然将其留在候选框序列中。soft NMS 很巧妙地将置信度下调的比例与重叠比例挂钩，重叠比例越大的候选框分数下调得越多。这样既能筛选出重复的候选框，又能避免筛选其它邻近目标的候选框。soft NMS 算法简单且易于实现，它的参数不需要通过数据学习得到，可以直接集成到现有模型中。

$$s_i = \begin{cases} s_i & IoU(M, b_i) < t \\ s_i \times (1 - IoU(M, B_i)) & IoU(M, b_i) \geq t \end{cases} \quad (4-5)$$

4.4 实验结果及分析

本文根据上节涉及到的三个方面对 SSD 模型进行了改进，并通过低分辨率图片 (LR)，超分辨率图片 (SR)，和 VOC 原图 (HR) 对模型进行性能检测，得到了表 4-1 所示的实验结果。

表 4-1 改进版 SSD 模型检测性能对比

模型	检测数据	检测指标 (mAP)
SSD+自训练权重	LR	39.69%
SSD+自训练权重	SR	46.27%
SSD+自训练权重	HR	60.75%
SSD+改进版权重	LR	41.43%
SSD+改进版权重	SR	47.84%
SSD+改进版权重	HR	61.36%

从中可以得到结论：在检测数据集为低分辨率图片，超分辨率图片和高分辨率图片的情况下，改进后的 SSD 模型的表现都要好于未改动的模型。虽然检测性能仅仅提高了 1% mAP，但在同等的训练时长的情况下，三种数据集的测试结果都可以验证这一结论。

以超分辨率图片检测为参考，具体分析改进版 SSD 模型和原版 SSD 模型在小目标类别的检测结果，如表 4-2 所示。在前十种小目标类别的检测中，有八种类别的检测率都得到了提高。这说明改进后的 SSD 模型确实比原版 SSD 模型在小目标检测问题上有更好的表现。然而改进后模型对瓶子和牛的检测性能却有所下降。这种情况不仅在低分辨率图片出现了，在超分辨率图片也有类似的情况。出现这种情况的原因可能是模型训练不充分。改进的 SSD 模型相比原版 SSD 多了一层待检测的特征层，这样的改进增加了大量的候选框，降低漏检率的同时也增加了误检的可能。而沙发这一目标的相对大小较大，改进后的 SSD 模型更关注小目标的检测过程，在训练不充分的前提下有可能导致大目标的检测率下降。

表 4-2 改进版 SSD 模型与原版 SSD 模型的小目标检测结果对比

类别	原版 SSD	改进版 SSD
	mAP (%)	mAP (%)
瓶子	17.41	16.45
盆栽	14.89	14.89
椅子	22.30	23.06
电视	44.85	47.73
羊	45.98	48.50
人	47.23	49.58
牛	39.00	38.19
船	31.29	35.52
鸟	41.03	42.69
汽车	56.46	59.78

在目标检测领域，评价模型性能指标除了 AP 之外，往往还会查看 PR 曲线（Precision-Recall curve）。PR 曲线是以查准率为纵轴，查全率为横轴绘制出的曲线。本文选取了相对面积较小的瓶子与盆栽两个类别与相对面积较大的猫和狗两个种类的 PR 曲线进行对比。其余种类的 PR 曲线可在附录五中查看。通过比对可以发现，大尺寸目标的查准率和查重率基本都保持较高的水平，而小目标种类的查准率随着查全率的增加迅速下降。即使把 IoU 的阈值调整到零，只要预测框与真实框有稍许重叠就判定为正例，查全率也仅能维持在 20%-30% 之间，这说明众多的小目标被漏检了。虽然改进后的 SSD 模型相比未改进的 SSD 而言，在小目标种类的检测上性能有所提高，但提升效果还不能令人满意，还有很大的发展空间。

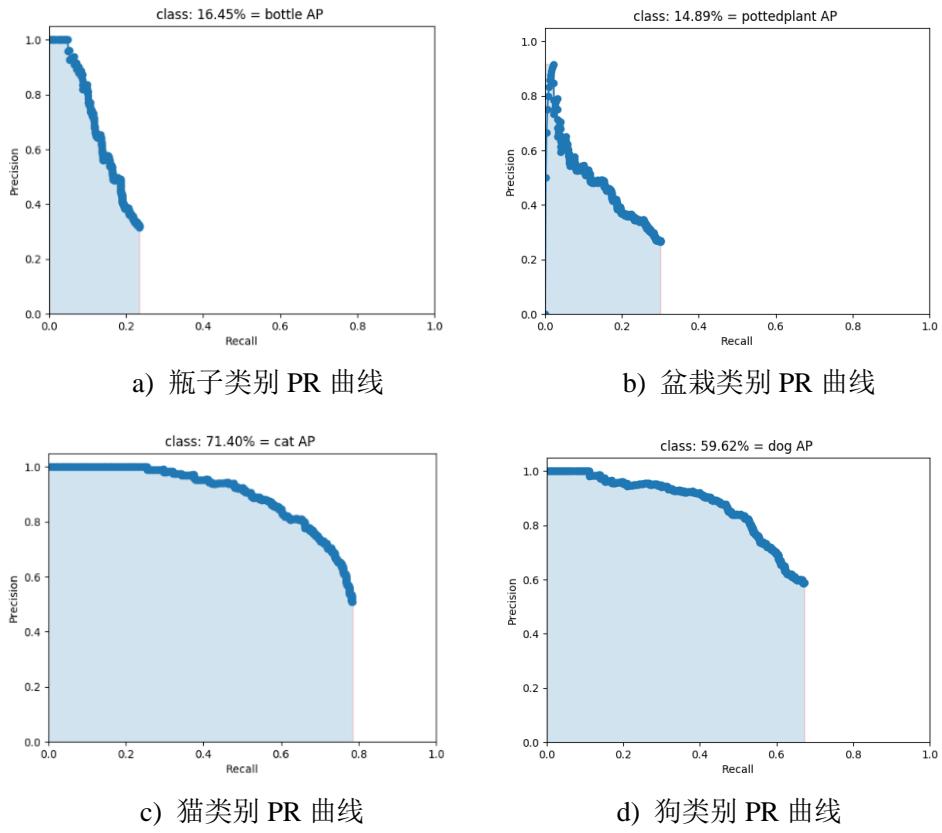


图 4-11 四个类别的 PR 曲线

4.5 本章小结

本章基于单阶段检测模型 SSD 进行了三个部分的改进，以适应小目标检测场景。首先，为了缓解小目标数量少导致的模型过拟合问题，本文采用了目标扩充的数据增强算法。其次，考虑到浅层特征图往往含有更多的细节特征，本文为 SSD 模型增加了针对小目标检测的浅层特征图，在该特征图上划分更密集的网格以适

应小目标检测问题。最后，本文采用了改进型非极大值抑制算法以缓解密集目标的漏检现象。本文将以上三个改进集成到 SSD 模型中，通过 PASCAL VOC 进行了性能测试，对比原版 SSD 模型与改进后的 SSD 模型的检测结果。观察到改进版本的 SSD 对大部分种类的检测率有所提高，得出结论：本文对 SSD 模型的三个改进措施确实能有效提高小目标的检测率。

结 论

基于小目标场景的目标检测是一项十分具有挑战性的研究课题，同时它也含有巨大的实际应用价值和理论价值。本文基于深度神经网络对小目标检测问题展开研究，并针对单阶段目标检测模型 SSD 提出了四种相应的改进措施，提高了模型对小目标的检测率。这些改进措施都具有一定的普适性，可以简单移植到不同模型上。具体的工作内容及研究结果如下：

(1) 在数据预处理部分，本文提出了使用超分辨率生成式对抗网络对小目标进行图像超分辨率重建的方法。该方法在放大小目标的同时也增加其细节纹理信息。通过实验表明，使用超分辨率图片有利于小目标的识别，其中目标检测的性能指标可提升大约 7% mAP。

(2) 针对小目标数据量不足的问题。本文提出了随机扩充的数据增强方法。根据小目标的位置信息，在图片中对小目标进行随机复制。该方法缓解了小目标数据量不平衡带来的模型过拟合问题，并且具有通用性，可直接在不同模型上使用。

(3) 针对卷积神经网络的特点，本文提出了基于单阶段目标检测模型 SSD 的小目标检测改进方法。在神经网络中，浅层特征往往具有更多的细节特征。针对这一特点，本文对模型 SSD 进行了改进，增加对浅层特征层的检测个数，并适当调小锚框的尺寸，提高了检测模型对小目标的关注度。

(4) 针对预选框过滤算法，本文替换了原有的非极大值抑制算法，使用了改进型非极大值抑制算法。该算法根据重叠面积的大小对候选框的置信度进行相应下调，避免了由于目标密集而导致候选框被删除的现象，有效减少小目标的漏检概率。

受限于实验环境与研究时长，本文还存在一些不足之处：本文仅针对 SSD 模型做出了一些优化措施，没有对比使用其他的主流检测模型如 Faster RCNN，YOLOv3 等；本文使用的数据集为 PASCAL VOC 数据集，该数据集中包含的小目标数量相对较少，不利于模型的充分训练。

为了进一步提高小目标的检测性能，今后的小目标检测研究可以尝试以下两个思路：

(1) 基于端到端的图片超分辨率检测模型。本文在第三章中提出的方法是利用 SRGAN 模型对数据进行图像超分辨重建，增加小目标的细节信息，然后再将图片输入模型进行检测。在这个过程中，数据处理和模型检测是相互独立的。结合深度学习领域端到端的训练方法，可以设计一个多任务检测模型，利用检测结果反馈给生成器，让生成器尽量生成更适合检测模型的图片数据。

(2) 重新设置小目标与大目标的定位损失函数。小目标对定位的准确性要高于大尺度目标，同样大小的位置偏差对小目标和大目标的影响是不一样的。小目标本身的面积就较小，同样的位置偏差很容易就使真实框与预测框的交并比小于阈值，这种问题对大尺度目标的影响就相对较小。因此需要设计一种合理的损失函数，增加对小目标定位误差的惩罚权重。

参考文献

- [1] Wu, Xiongwei, Doyen Sahoo, and Steven CH Hoi. Recent advances in deep learning for object detection[M]. Neurocomputing (2020).
- [2] Dadi, Harihara Santosh, and GK Mohan Pillutla. Improved face recognition rate using HOG features and SVM classifier[J]. IOSR Journal of Electronics and Communication Engineering 11.04 (2016): 34-44.
- [3] Zhou, Huiyu, Yuan Yuan, and Chunmei Shi. Object tracking using SIFT features and mean shift[J]. Computer vision and image understanding 113.3 (2009): 345-352.
- [4] Agarwal S , Terrail J O D , Jurie, Frédéric. Recent Advances in Object Detection in the Age of Deep Convolutional Neural Networks[J]. 2018.
- [5] 曹燕, 李欢, 王天宝.基于深度学习的目标检测算法研究综述[J].计算机与现代化, 2020(05): 63-69.
- [6] Girshick, Ross, et al. Region-based convolutional networks for accurate object detection and segmentation[J]. IEEE transactions on pattern analysis and machine intelligence 38.1 (2015): 142-158.
- [7] Wang X , Shrivastava A , Gupta A . A-Fast-RCNN: Hard Positive Generation via Adversary for Object Detection[C]// 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2017.
- [8] Ren, Shaoqing, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[J]. Advances in neural information processing systems. 2015.
- [9] Redmon, Joseph, et al. You only look once: Unified, real-time object detection[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
- [10] Liu W , Anguelov D , Erhan D , et al. SSD: Single Shot MultiBox Detector[C]// European Conference on Computer Vision. Springer International Publishing, 2016.
- [11] Yan, Junjie, et al. The fastest deformable part model for object detection[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014.
- [12] He, Kaiming, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[J]. IEEE transactions on pattern analysis and machine intelligence 37.9 (2015): 1904-1916.

-
- [13] Dai, Jifeng, et al. R-fcn: Object detection via region-based fully convolutional networks[C]. Advances in neural information processing systems. 2016.
 - [14] Agarwal, Shivang, Jean Ogier Du Terrail, and Frédéric Jurie. Recent advances in object detection in the age of deep convolutional neural networks[M]. arXiv preprint arXiv: 1809.03193 (2018).
 - [15] Redmon, Joseph, and Ali Farhadi. Yolov3: An incremental improvement[M]. arXiv preprint arXiv: 1804.02767 (2018).
 - [16] Law, Hei, and Jia Deng. Cornernet: Detecting objects as paired keypoints[C]. Proceedings of the European Conference on Computer Vision (ECCV). 2018.
 - [17] Lin, Tsung-Yi, et al. Focal loss for dense object detection[C]. Proceedings of the IEEE international conference on computer vision. 2017.
 - [18] Shrivastava, Abhinav, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
 - [19] Singh, Bharat, and Larry S. Davis. An analysis of scale invariance in object detection snip[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.
 - [20] Lin , Tsung-Yi , et al. Feature pyramid networks for object detection[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
 - [21] Hu, Peiyun, and Deva Ramanan. Finding tiny faces[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
 - [22] Jianan , et al. Perceptual generative adversarial networks for small object detection[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
 - [23] Cai, Zhaowei, and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.
 - [24] Krizhevsky A , Sutskever I , Hinton G . ImageNet Classification with Deep Convolutional Neural Networks[C]// NIPS. Curran Associates Inc. 2012.
 - [25] Ledig, Christian, et al. Photo-realistic single image super-resolution using a generative adversarial network[C]. Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
 - [26] Goodfellow I . NIPS 2016 Tutorial: Generative Adversarial Networks[J]. 2016.

- [27] 陈巧媛, 陈莹. 基于困难样本三元组损失的多任务行人再识别[J]. 计算机辅助设计与图形学学报, 2019, 31(07):1156-1165.
- [28] Kisantal M , Wojna Z , Murawski J , et al. Augmentation for small object detection[J]. 2019.
- [29] Bodla, Navaneeth, et al. Soft-NMS--improving object detection with one line of code[C]. Proceedings of the IEEE international conference on computer vision. 2017.

哈尔滨工业大学（深圳）本科毕业设计（论文）原创性 声明

本人郑重声明：在哈尔滨工业大学（深圳）攻读学士学位期间，所提交的毕业设计（论文）《基于深度学习的小目标检测算法研究》，是本人在导师指导下独立进行研究工作所取得的成果。对本文的研究工作做出重要贡献的个人和集体，均已在文中以明确方式注明，其它未注明部分不包含他人已发表或撰写过的研究成果，不存在购买、由他人代写、剽窃和伪造数据等作假行为。

本人愿为此声明承担法律责任。

作者签名：

日期： 年 月 日

致 谢

时光荏苒，犹如白驹过隙。四年的大学时光仿佛仍历历在目。我从来没有后悔选择来到哈尔滨工业大学（深圳）。感谢学校给我提供的学习机会和学习资源，也感谢在学习和生活中遇到的各位老师和同学，我的成长离不开你们的帮助。

在本次毕业设计的过程中，感谢陈梓浩老师对我的充分理解与肯定，感谢鹏城实验室 SLAM 项目组提供的设备支持，感谢有一群志同道合的室友能陪我一起熬夜改代码。

特别要感谢我的父母和姐姐。他们在背后坚定地支持我的各种选择，衷心希望我能学有所成。相信在未来，我也不会辜负他们的期待，成为值得他们骄傲的人。

附录

附录一：外文文献翻译

SSD：单次检测器

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy,

摘要：

本文提出了一种使用单个深度神经网络来检测图像中的目标的方法。本文的方法命名为 SSD，将边界框的输出空间离散化为不同长宽比的一组默认框和并缩放每个特征映射的位置。在预测时，网络会在每个默认框中为每个目标类别的出现生成分数，并对框进行调整以更好地匹配目标形状。此外，网络还结合了不同分辨率的多个特征映射的预测，自然地处理各种尺寸的目标。相对于需要目标提出的方法，SSD 非常简单，因为它完全消除了提出生成和随后的像素或特征重新采样阶段，并将所有计算封装到单个网络中。这使得 SSD 易于训练和直接集成到需要检测组件的系统中。PASCAL VOC, COCO 和 ILSVRC 数据集上的实验结果证实，SSD 对于利用额外的目标提出步骤的方法具有竞争性的准确性，并且速度更快，同时为训练和推断提供了统一的框架。对于 300×300 的输入，SSD 在 VOC2007 测试中以 59FPS 的速度在 Nvidia Titan X 上达到 74.3% 的 mAP，对于 512×512 的输入，SSD 达到了 76.9% 的 mAP，优于参照的最先进的 Faster R-CNN 模型。与其他单阶段方法相比，即使输入图像尺寸较小，SSD 也具有更高的精度。

关键词：实时目标检测，卷积神经网络

1 引言

目前最先进的目标检测系统是以下方法的变种：假设边界框，每个框重采样像素或特征，并应用一个高质量的分类器。自从选择性搜索通过在 PASCAL VOC, COCO 和 ILSVRC 上所有基于 Faster R-CNN 的检测都取得了当前领先的结果，这种流程在检测基准数据上流行开来。尽管这些方法准确，但对于嵌入式系统而言，这些方法的计算量过大，即使是高端硬件，对于实时应用而言也太慢。通常，这些方法的检测速度是以每帧秒 (SPF) 度量，甚至最快的高精度检测器，Faster R-CNN，仅以每秒 7 帧 (FPS) 的速度运行。已经有很多尝试通过处理检测流程中的每个阶段来构建更快的检测器（参见第 4 节中的相关工作），但是到目前为止，显著提高的速度仅以显著降低的检测精度为代价。

本文引入了 SSD，这是一种针对多个类别的单次检测器，比先前的先进的单次检测器（YOLO）更快，并且准确得多，事实上，与执行显式区域提出和池化的更慢的技术具有相同的精度（包括 Faster R-CNN）。

SSD 的核心是预测固定的一系列默认边界框的类别分数和边界框偏移，使用更小的卷积滤波器应用到特征映射上。

为了实现高检测精度，本文根据不同尺度的特征映射生成不同尺度的预测，并通过纵横比明确分开预测。这些设计功能使得即使在低分辨率输入图像上也能实现简单的端到端训练和高精度，从而进一步提高速度与精度之间的权衡。

实验包括在 PASCAL VOC, COCO 和 ILSVRC 上评估具有不同输入大小的模型的时间和精度分析，并与最近的一系列最新方法进行比较。

2 单次检测器（SSD）

2.1 模型

SSD 方法基于前馈卷积网络，该网络产生固定大小的边界框集合，并对这些边界框中存在的目标类别实例进行评分，然后进行非极大值抑制步骤来产生最终的检测结果。早期的网络层基于用于高质量图像分类的标准架构（在任何分类层之前被截断），本文将其称为基础网络。然后，本文将辅助结构添加到网络中以产生具有以下关键特征的检测：

用于检测的多尺度特征映射。本文将卷积特征层添加到截取的基础网络的末端。这些层在尺寸上逐渐减小，并允许在多个尺度上对检测结果进行预测。用于预测检测的卷积模型对于每个特征层都是不同的（查阅 Overfeat[4]和 YOLO[5]在单尺度特征映射上的操作）。

用于检测的卷积预测器。每个添加的特征层（或者任选的来自基础网络的现有特征层）可以使用一组卷积滤波器产生固定的检测预测集合。这些在图 2 中的 SSD 网络架构的上部指出。对于具有 p 通道的大小为 $m \times n$ 的特征层，潜在检测的预测参数的基本元素是 $3 \times 3 \times p$ 的小核得到某个类别的分数，或者相对于默认框坐标的形状偏移。在应用卷积核的 $m \times n$ 的每个位置，它会产生一个输出值。边界框偏移输出值是相对每个特征映射位置的相对默认框位置来度量的（查阅 YOLO[5]的架构，该步骤使用中间全连接层而不是卷积滤波器）。

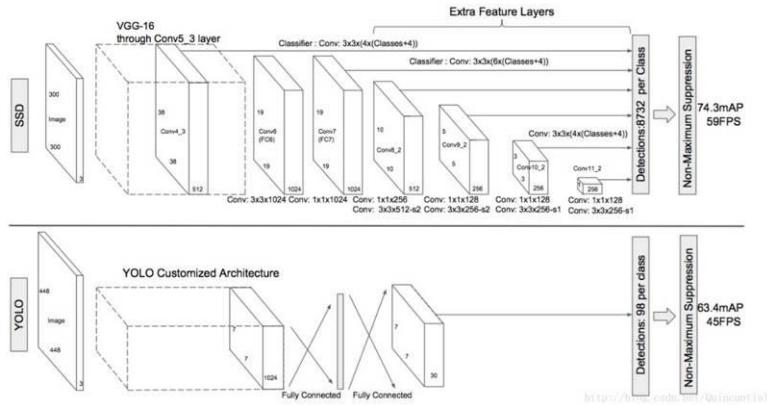


图 1：两个单次检测模型的比较：SSD 和 YOLO[5]。

本文的 SSD 模型在基础网络的末端添加了几个特征层，它预测了不同尺度和长宽比的默认边界框的偏移量及其相关的置信度。300×300 输入尺寸的 SSD 在 VOC2007 test 上的准确度上明显优于 448×448 的 YOLO 的准确度，同时也提高了速度。

默认边界框和长宽比。对于网络顶部的多个特征映射，本文将一组默认边界框与每个特征映射单元相关联。默认边界框以卷积的方式平铺特征映射，以便每个边界框相对于其对应单元的位置是固定的。在每个特征映射单元中，本文预测单元中相对于默认边界框形状的偏移量，以及指出每个边界框中存在的每个类别实例的类别分数。具体而言，对于给定位置处的 k 个边界框中的每一个，本文计算 c 个类别分数和相对于原始默认边界框形状的 4 个偏移量。这导致在特征映射中的每个位置周围应用总共 $(c+4)k$ 个滤波器，对于 $m \times n$ 的特征映射取得 $(c+4)kmn$ 个输出。有关默认边界框的说明，请参见图 1。本文的默认边界框与 Faster R-CNN[2] 中使用的锚边界框相似，但是本文将它们应用到不同分辨率的几个特征映射上。在几个特征映射中允许不同的默认边界框形状让本文有效地离散可能的输出框形状的空间。

2.2 训练

训练 SSD 和训练使用区域提出的典型检测器之间的关键区别在于，需要将真实信息分配给固定的检测器输出集合中的特定输出。在 YOLO[5]的训练中、Faster R-CNN[2]和 MultiBox[7]的区域提出阶段，一些版本也需要这样的操作。一旦确定了这个分配，损失函数和反向传播就可以应用端到端了。训练也涉及选择默认边界框集合和缩放进行检测，以及难例挖掘和数据增强策略。

匹配策略。在训练过程中，本文需要确定哪些默认边界框对应实际边界框的检测，并相应地训练网络。对于每个实际边界框，本文从默认边界框中选择，这些框会在位置、长宽比和尺度上变化。本文首先将每个实际边界框与具有最好的 Jaccard 重叠（如 MultiBox[7]）的边界框相匹配。与 MultiBox 不同的是，本文将默认边界框匹配到 Jaccard 重叠高于阈值（0.5）的任何实际边界框。这简化了学习问题，允许网络为多个重叠的默认边界框预测高分，而不是要求它只挑选具有最大重叠的一个边界框。

训练目标函数。SSD 训练目标函数来自于 MultiBox 目标，但扩展到处理多个目标类别。设 $x_{ij}^p = \{1, 0\}$ 是第 i 个默认边界框匹配到类别 p 的第 j 个实际边界框的指示器。在上面的匹配策略中，本文有 $\sum_i x_{ij}^p \geq 1$ 。总体目标损失函数是定位损失 (loc) 和置信度损失 (conf) 的加权和

$$L(x, c, l, g) = \frac{1}{N} (L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \quad (1)$$

其中 N 是匹配的默认边界框的数量。如果 N=0，则将损失设为 0。定位损失是预测框 (l) 与真实框 (g) 参数之间的 Smooth L1 损失[6]。类似于 Faster R-CNN[2]，本文回归默认边界框 (d) 的中心偏移量 (cx, cy) 和其宽度 (w)、高度 (h) 的偏移量。

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^N \sum_{m \in \{cx, cy, w, h\}} x_{ij}^k smooth_{L1}(l_i^m - \hat{g}_j^m) \quad (2)$$

$$\hat{g}_j^h = \log\left(\frac{g_j^h}{d_i^h}\right), \quad \hat{g}_j^w = \log\left(\frac{g_j^w}{d_i^w}\right), \quad \hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx}) / d_i^w, \quad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy}) / d_i^h$$

置信度损失是在多类别置信度 (c) 上的 softmax 损失。

$$L_{conf}(x, c) = - \sum_{i \in Pos}^N x_{ij}^p \log(\hat{c}_i^p) - \sum_{i \in Neg} \log(\hat{c}_i^0) \quad where \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)} \quad (3)$$

为默认边界框选择尺度和长宽比。为了处理不同的目标尺度，一些方法[4, 9]建议处理不同尺寸的图像，然后将结果合并。然而，通过利用单个网络中几个不同层的特征映射进行预测，本文可以模拟相同的效果，同时还可以跨所有目标尺度共享参数。以前的工作[10, 11]已经表明，使用低层的特征映射可以提高语义分割的质量，因为低层会捕获输入目标的更多细节。同样，[12]表明，从特征映射上添加全局上下文池化可以有助于平滑分割结果。受这些方法的启发，本文使用较低和较高的特征映射进行检测。

难例挖掘。在匹配步骤之后，大多数默认边界框为负例，尤其是当可能的默认边界框数量较多时。这在正的训练实例和负的训练实例之间引入了显著的不平衡。本文不使用所有负例，而是使用每个默认边界框的最高置信度损失来排序它们，并挑选最高的置信度，以便负例和正例之间的比例至多为 3: 1。本文发现这会导致更快的优化和更稳定的训练。

数据增强。为了使模型对各种输入目标大小和形状更鲁棒，每张训练图像都是通过以下选项之一进行随机采样的：

- (1) 使用整个原始输入图像。
- (2) 采样一个图像块，使得与目标之间的最小 Jaccard 重叠为 0.1, 0.3, 0.5, 0.7 或 0.9。
- (3) 随机采样一个图像块。

每个采样图像块的大小是原始图像大小的[0.1, 1]，长宽比在 12 和 2 之间。如果实际边界框的中心在采用的图像块中，本文保留实际边界框与采样图像块的重叠部分。在上述采样步骤之后，除了应用类似于文献[14]中描述的一些光度变形之外，将每个采样图像块调整到固定尺寸并以 0.5 的概率进行水平翻转。

3. 实验结果

基础网络。本文的实验全部基于 VGG16，它是在 ILSVRC CLS-LOC 数据集上预先训练的。类似于 DeepLab-LargeFOV，本文将 fc6 和 fc7 转换为卷积层，从 fc6 和 fc7 中重采样参数，将 pool5 从 $2 \times 2 - s2$ 更改为 $3 \times 3 - s1$ ，并使用空洞算法来填补这个“小洞”。本文删除所有的丢弃层和 fc8 层。本文使用 SGD 对得到的模型进行微调，初始学习率为 10^{-3} ，动量为 0.9，权重衰减为 0.0005，批数据大小为 32。每个数据集的学习速率衰减策略略有不同，本文将在后面详细描述。

3.1 PASCAL VOC2007

在这个数据集上，本文在 VOC2007 test(4952 张图像)上比较了 Fast R-CNN[6] 和 FAST R-CNN[2]。所有的方法都在相同的预训练好的 VGG16 网络上进行微调。图 2 显示了 SSD300 模型的架构细节。本文使用 conv4_3, conv7 (fc7), conv8_2, conv9_2, conv10_2 和 conv11_2 来预测位置和置信度。本文在 conv4_3 上设置了尺度为 0.1 的默认边界框。本文使用“xavier”方法[20]初始化所有新添加的卷积层的参数。对于 conv4_3, conv10_2 和 conv11_2，本文只在每个特征映射位置上关联了 4 个默认边界框——忽略 13 和 3 的长宽比。对于所有其它层，本文像 2.2 节描述的那样放置了 6 个默认边界框。

Method	data	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast [6]	07	66.9	74.5	78.3	69.2	53.2	36.6	77.3	78.2	82.0	40.7	72.7	67.9	79.6	79.2	73.0	69.0	30.1	65.4	70.2	75.8	65.8
Fast [6]	07+12	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
Faster [2]	07	69.9	70.0	80.6	70.1	57.3	49.9	78.2	80.4	82.0	52.2	75.3	67.2	80.3	79.8	75.0	76.3	39.1	68.3	67.3	81.1	67.6
Faster [2]	07+12	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
Faster [2]	07+12+COCO	78.8	84.3	82.0	77.7	68.9	65.7	88.1	88.4	88.9	63.6	86.3	70.8	85.9	87.6	80.1	82.3	53.6	80.4	75.8	86.6	78.9
SSD300	07	68.0	73.4	77.5	64.1	59.0	38.9	75.2	80.8	78.5	46.0	67.8	69.2	76.6	82.1	77.0	72.5	41.2	64.2	69.1	78.0	68.5
SSD300	07+12	74.3	75.5	80.2	72.3	66.3	47.6	83.0	84.2	86.1	54.7	78.3	73.9	84.5	85.3	82.6	76.2	48.6	73.9	76.0	83.4	74.0
SSD300	07+12+COCO	79.6	80.9	86.3	79.0	76.2	57.6	87.3	88.2	88.6	60.5	85.4	76.7	87.5	89.2	84.5	81.4	55.0	81.9	81.5	85.9	78.9
SSD512	07	71.6	75.1	81.4	69.8	60.8	46.3	82.6	84.7	84.1	48.5	75.0	67.4	82.3	83.9	79.4	76.6	44.9	69.9	69.1	78.1	71.8
SSD512	07+12	76.8	82.4	84.7	78.4	73.8	53.2	86.2	87.5	86.0	57.8	83.1	70.2	84.9	85.2	83.9	79.7	50.3	77.9	73.9	82.5	75.3
SSD512	07+12+COCO	81.6	86.6	88.3	82.4	76.0	66.3	88.6	88.9	89.1	65.1	88.4	73.6	86.5	88.9	85.3	84.6	59.1	85.0	80.4	87.4	81.2

表 1：PASCAL VOC2007 test 检测结果。Fast 和 Faster R-CNN 都使用最小维度为 600 的输入图像。两个 SSD 模型使用完全相同的设置除了它们有不同的输入大小（300×300 和 512×512）。很明显更大的输入尺寸会导致更好的结果，并且更大的数据同样有帮助。数据：“07”：VOC2007 trainval，“07+12”：VOC2007 和 VOC2012 trainval 的联合。“07+12+COCO”：首先在 COCO trainval35k 上训练然后在 07+12 上微调。

4. 结论

本文介绍了 SSD，一种快速的单次多类别目标检测器。本文模型的一个关键特性是使用网络顶部多个特征映射的多尺度卷积边界框输出。这种表示使本文能够高效地建模可能的边界框形状空间。本文通过实验证明，在给定合适训练策略的情况下，大量仔细选择的默认边界框会提高性能。本文构建的 SSD 模型比现有的方法至少要多一个数量级的边界框预测采样位置，尺度和长宽比[5, 7]。本文证明了给定相同的 VGG-16 基础架构，SSD 在准确性和速度方面与其对应的最先进的目标检测器相比毫不逊色。在 PASCAL VOC 和 COCO 上，本文的 SSD512 模型的性能明显优于最先进的 Faster R-CNN[2]，而速度提高了 3 倍。本文的实时 SSD300 模型运行速度为 59FPS，比目前的实时 YOLO[5]更快，同时显著提高了检测精度。

附录:二：外文文献原文

arXiv:1512.02325v5 [cs.CV] 29 Dec 2016

SSD: Single Shot MultiBox Detector

Wei Liu¹, Dragomir Anguelov², Dumitru Erhan³, Christian Szegedy³,
Scott Reed⁴, Cheng-Yang Fu¹, Alexander C. Berg¹

¹UNC Chapel Hill ²Zoox Inc. ³Google Inc. ⁴University of Michigan, Ann Arbor

¹wliu@cs.unc.edu, ²drago@zoox.com, ³{dumitru,szegedy}@google.com,
⁴reedscot@umich.edu, ¹{cyfu,aberg}@cs.unc.edu

Abstract. We present a method for detecting objects in images using a single deep neural network. Our approach, named SSD, discretizes the output space of bounding boxes into a set of default boxes over different aspect ratios and scales per feature map location. At prediction time, the network generates scores for the presence of each object category in each default box and produces adjustments to the box to better match the object shape. Additionally, the network combines predictions from multiple feature maps with different resolutions to naturally handle objects of various sizes. SSD is simple relative to methods that require object proposals because it completely eliminates proposal generation and subsequent pixel or feature resampling stages and encapsulates all computation in a single network. This makes SSD easy to train and straightforward to integrate into systems that require a detection component. Experimental results on the PASCAL VOC, COCO, and ILSVRC datasets confirm that SSD has competitive accuracy to methods that utilize an additional object proposal step and is much faster, while providing a unified framework for both training and inference. For 300×300 input, SSD achieves 74.3% mAP¹ on VOC2007 test at 59 FPS on a Nvidia Titan X and for 512×512 input, SSD achieves 76.9% mAP, outperforming a comparable state-of-the-art Faster R-CNN model. Compared to other single stage methods, SSD has much better accuracy even with a smaller input image size. Code is available at: <https://github.com/weiliu89/caffe/tree/ssd>.

Keywords: Real-time Object Detection; Convolutional Neural Network

1 Introduction

Current state-of-the-art object detection systems are variants of the following approach: hypothesize bounding boxes, resample pixels or features for each box, and apply a high-quality classifier. This pipeline has prevailed on detection benchmarks since the Selective Search work [1] through the current leading results on PASCAL VOC, COCO, and ILSVRC detection all based on Faster R-CNN[2] albeit with deeper features such as [3]. While accurate, these approaches have been too computationally intensive for embedded systems and, even with high-end hardware, too slow for real-time applications.

¹ We achieved even better results using an improved data augmentation scheme in follow-on experiments: 77.2% mAP for 300×300 input and 79.8% mAP for 512×512 input on VOC2007. Please see Sec. 3.6 for details.

Often detection speed for these approaches is measured in seconds per frame (SPF), and even the fastest high-accuracy detector, Faster R-CNN, operates at only 7 frames per second (FPS). There have been many attempts to build faster detectors by attacking each stage of the detection pipeline (see related work in Sec. 4), but so far, significantly increased speed comes only at the cost of significantly decreased detection accuracy.

This paper presents the first deep network based object detector that does not resample pixels or features for bounding box hypotheses *and* is as accurate as approaches that do. This results in a significant improvement in speed for high-accuracy detection (59 FPS with mAP 74.3% on VOC2007 test, vs. Faster R-CNN 7 FPS with mAP 73.2% or YOLO 45 FPS with mAP 63.4%). The fundamental improvement in speed comes from eliminating bounding box proposals and the subsequent pixel or feature resampling stage. We are not the first to do this (cf [4,5]), but by adding a series of improvements, we manage to increase the accuracy significantly over previous attempts. Our improvements include using a small convolutional filter to predict object categories and offsets in bounding box locations, using separate predictors (filters) for different aspect ratio detections, and applying these filters to multiple feature maps from the later stages of a network in order to perform detection at multiple scales. With these modifications—especially using multiple layers for prediction at different scales—we can achieve high-accuracy using relatively low resolution input, further increasing detection speed. While these contributions may seem small independently, we note that the resulting system improves accuracy on real-time detection for PASCAL VOC from 63.4% mAP for YOLO to 74.3% mAP for our SSD. This is a larger relative improvement in detection accuracy than that from the recent, very high-profile work on residual networks [3]. Furthermore, significantly improving the speed of high-quality detection can broaden the range of settings where computer vision is useful.

We summarize our contributions as follows:

- We introduce SSD, a single-shot detector for multiple categories that is faster than the previous state-of-the-art for single shot detectors (YOLO), and significantly more accurate, in fact as accurate as slower techniques that perform explicit region proposals and pooling (including Faster R-CNN).
- The core of SSD is predicting category scores and box offsets for a fixed set of default bounding boxes using small convolutional filters applied to feature maps.
- To achieve high detection accuracy we produce predictions of different scales from feature maps of different scales, and explicitly separate predictions by aspect ratio.
- These design features lead to simple end-to-end training and high accuracy, even on low resolution input images, further improving the speed vs accuracy trade-off.
- Experiments include timing and accuracy analysis on models with varying input size evaluated on PASCAL VOC, COCO, and ILSVRC and are compared to a range of recent state-of-the-art approaches.

2 The Single Shot Detector (SSD)

This section describes our proposed SSD framework for detection (Sec. 2.1) and the associated training methodology (Sec. 2.2). Afterwards, Sec. 3 presents dataset-specific model details and experimental results.

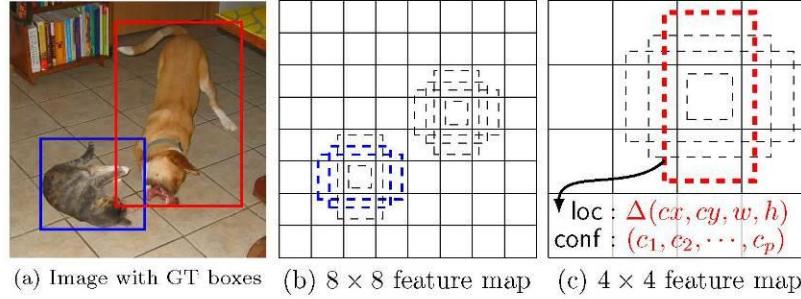


Fig. 1: SSD framework. (a) SSD only needs an input image and ground truth boxes for each object during training. In a convolutional fashion, we evaluate a small set (e.g. 4) of default boxes of different aspect ratios at each location in several feature maps with different scales (e.g. 8×8 and 4×4 in (b) and (c)). For each default box, we predict both the shape offsets and the confidences for all object categories ((c_1, c_2, \dots, c_p)). At training time, we first match these default boxes to the ground truth boxes. For example, we have matched two default boxes with the cat and one with the dog, which are treated as positives and the rest as negatives. The model loss is a weighted sum between localization loss (e.g. Smooth L1 [6]) and confidence loss (e.g. Softmax).

2.1 Model

The SSD approach is based on a feed-forward convolutional network that produces a fixed-size collection of bounding boxes and scores for the presence of object class instances in those boxes, followed by a non-maximum suppression step to produce the final detections. The early network layers are based on a standard architecture used for high quality image classification (truncated before any classification layers), which we will call the base network². We then add auxiliary structure to the network to produce detections with the following key features:

Multi-scale feature maps for detection We add convolutional feature layers to the end of the truncated base network. These layers decrease in size progressively and allow predictions of detections at multiple scales. The convolutional model for predicting detections is different for each feature layer (*cf* Overfeat[4] and YOLO[5] that operate on a single scale feature map).

Convolutional predictors for detection Each added feature layer (or optionally an existing feature layer from the base network) can produce a fixed set of detection predictions using a set of convolutional filters. These are indicated on top of the SSD network architecture in Fig. 2. For a feature layer of size $m \times n$ with p channels, the basic element for predicting parameters of a potential detection is a $3 \times 3 \times p$ *small kernel* that produces either a score for a category, or a shape offset relative to the default box coordinates. At each of the $m \times n$ locations where the kernel is applied, it produces an output value. The bounding box offset output values are measured relative to a default

² We use the VGG-16 network as a base, but other networks should also produce good results.

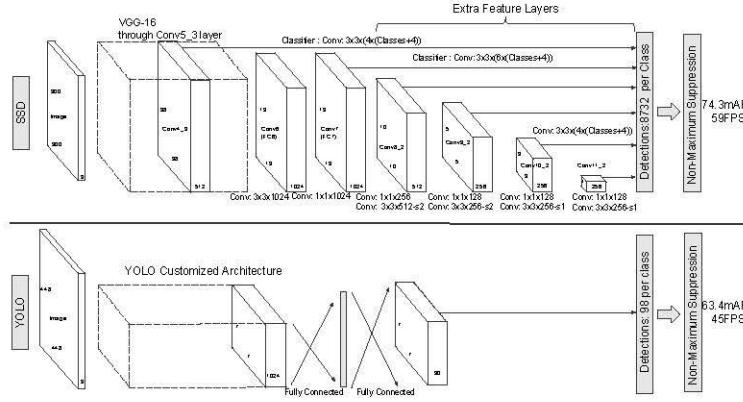


Fig. 2: A comparison between two single shot detection models: SSD and YOLO [5]. Our SSD model adds several feature layers to the end of a base network, which predict the offsets to default boxes of different scales and aspect ratios and their associated confidences. SSD with a 300×300 input size significantly outperforms its 448×448 YOLO counterpart in accuracy on VOC2007 test while also improving the speed.

box position relative to each feature map location (*cf* the architecture of YOLO[5] that uses an intermediate fully connected layer instead of a convolutional filter for this step).

Default boxes and aspect ratios We associate a set of default bounding boxes with each feature map cell, for multiple feature maps at the top of the network. The default boxes tile the feature map in a convolutional manner, so that the position of each box relative to its corresponding cell is fixed. At each feature map cell, we predict the offsets relative to the default box shapes in the cell, as well as the per-class scores that indicate the presence of a class instance in each of those boxes. Specifically, for each box out of k at a given location, we compute c class scores and the 4 offsets relative to the original default box shape. This results in a total of $(c + 4)k$ filters that are applied around each location in the feature map, yielding $(c + 4)kmn$ outputs for a $m \times n$ feature map. For an illustration of default boxes, please refer to Fig. 1. Our default boxes are similar to the *anchor boxes* used in Faster R-CNN [2], however we apply them to several feature maps of different resolutions. Allowing different default box shapes in several feature maps let us efficiently discretize the space of possible output box shapes.

2.2 Training

The key difference between training SSD and training a typical detector that uses region proposals, is that ground truth information needs to be assigned to specific outputs in the fixed set of detector outputs. Some version of this is also required for training in YOLO[5] and for the region proposal stage of Faster R-CNN[2] and MultiBox[7]. Once this assignment is determined, the loss function and back propagation are applied end-to-end. Training also involves choosing the set of default boxes and scales for detection as well as the hard negative mining and data augmentation strategies.

Matching strategy During training we need to determine which default boxes correspond to a ground truth detection and train the network accordingly. For each ground truth box we are selecting from default boxes that vary over location, aspect ratio, and scale. We begin by matching each ground truth box to the default box with the best jaccard overlap (as in MultiBox [7]). Unlike MultiBox, we then match default boxes to any ground truth with jaccard overlap higher than a threshold (0.5). This simplifies the learning problem, allowing the network to predict high scores for multiple overlapping default boxes rather than requiring it to pick only the one with maximum overlap.

Training objective The SSD training objective is derived from the MultiBox objective [7,8] but is extended to handle multiple object categories. Let $x_{ij}^p = \{1, 0\}$ be an indicator for matching the i -th default box to the j -th ground truth box of category p . In the matching strategy above, we can have $\sum_i x_{ij}^p \geq 1$. The overall objective loss function is a weighted sum of the localization loss (loc) and the confidence loss (conf):

$$L(x, c, l, g) = \frac{1}{N} (L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \quad (1)$$

where N is the number of matched default boxes. If $N = 0$, we set the loss to 0. The localization loss is a Smooth L1 loss [6] between the predicted box (l) and the ground truth box (g) parameters. Similar to Faster R-CNN [2], we regress to offsets for the center (cx, cy) of the default bounding box (d) and for its width (w) and height (h):

$$\begin{aligned} L_{loc}(x, l, g) &= \sum_{i \in Pos}^N \sum_{m \in \{cx, cy, w, h\}} x_{ij}^k \text{smooth}_{L1}(l_i^m - \hat{g}_j^m) \\ \hat{g}_j^{cx} &= (g_j^{cx} - d_i^{cx}) / d_i^w \quad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy}) / d_i^h \\ \hat{g}_j^w &= \log \left(\frac{g_j^w}{d_i^w} \right) \quad \hat{g}_j^h = \log \left(\frac{g_j^h}{d_i^h} \right) \end{aligned} \quad (2)$$

The confidence loss is the softmax loss over multiple classes confidences (c).

$$L_{conf}(x, c) = - \sum_{i \in Pos}^N x_{ij}^p \log(\hat{c}_i^p) - \sum_{i \in Neg} \log(\hat{c}_i^0) \quad \text{where} \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)} \quad (3)$$

and the weight term α is set to 1 by cross validation.

Choosing scales and aspect ratios for default boxes To handle different object scales, some methods [4,9] suggest processing the image at different sizes and combining the results afterwards. However, by utilizing feature maps from several different layers in a single network for prediction we can mimic the same effect, while also sharing parameters across all object scales. Previous works [10,11] have shown that using feature maps from the lower layers can improve semantic segmentation quality because the lower layers capture more fine details of the input objects. Similarly, [12] showed that adding global context pooled from a feature map can help smooth the segmentation results.

Motivated by these methods, we use both the lower and upper feature maps for detection. Figure 1 shows two exemplar feature maps (8×8 and 4×4) which are used in the framework. In practice, we can use many more with small computational overhead.

Feature maps from different levels within a network are known to have different (empirical) receptive field sizes [13]. Fortunately, within the SSD framework, the default boxes do not necessarily need to correspond to the actual receptive fields of each layer. We design the tiling of default boxes so that specific feature maps learn to be responsive to particular scales of the objects. Suppose we want to use m feature maps for prediction. The scale of the default boxes for each feature map is computed as:

$$s_k = s_{\min} + \frac{s_{\max} - s_{\min}}{m-1}(k-1), \quad k \in [1, m] \quad (4)$$

where s_{\min} is 0.2 and s_{\max} is 0.9, meaning the lowest layer has a scale of 0.2 and the highest layer has a scale of 0.9, and all layers in between are regularly spaced. We impose different aspect ratios for the default boxes, and denote them as $a_r \in \{1, 2, 3, \frac{1}{2}, \frac{1}{3}\}$. We can compute the width ($w_k^a = s_k \sqrt{a_r}$) and height ($h_k^a = s_k / \sqrt{a_r}$) for each default box. For the aspect ratio of 1, we also add a default box whose scale is $s'_k = \sqrt{s_k s_{k+1}}$, resulting in 6 default boxes per feature map location. We set the center of each default box to $(\frac{i+0.5}{|f_k|}, \frac{j+0.5}{|f_k|})$, where $|f_k|$ is the size of the k -th square feature map, $i, j \in [0, |f_k|]$. In practice, one can also design a distribution of default boxes to best fit a specific dataset. How to design the optimal tiling is an open question as well.

By combining predictions for all default boxes with different scales and aspect ratios from all locations of many feature maps, we have a diverse set of predictions, covering various input object sizes and shapes. For example, in Fig. 1, the dog is matched to a default box in the 4×4 feature map, but not to any default boxes in the 8×8 feature map. This is because those boxes have different scales and do not match the dog box, and therefore are considered as negatives during training.

Hard negative mining After the matching step, most of the default boxes are negatives, especially when the number of possible default boxes is large. This introduces a significant imbalance between the positive and negative training examples. Instead of using all the negative examples, we sort them using the highest confidence loss for each default box and pick the top ones so that the ratio between the negatives and positives is at most 3:1. We found that this leads to faster optimization and a more stable training.

Data augmentation To make the model more robust to various input object sizes and shapes, each training image is randomly sampled by one of the following options:

- Use the entire original input image.
- Sample a patch so that the *minimum* jaccard overlap with the objects is 0.1, 0.3, 0.5, 0.7, or 0.9.
- Randomly sample a patch.

The size of each sampled patch is $[0.1, 1]$ of the original image size, and the aspect ratio is between $\frac{1}{2}$ and 2. We keep the overlapped part of the ground truth box if the center of it is in the sampled patch. After the aforementioned sampling step, each sampled patch is resized to fixed size and is horizontally flipped with probability of 0.5, in addition to applying some photo-metric distortions similar to those described in [14].

3 Experimental Results

Base network Our experiments are all based on VGG16 [15], which is pre-trained on the ILSVRC CLS-LOC dataset [16]. Similar to DeepLab-LargeFOV [17], we convert fc6 and fc7 to convolutional layers, subsample parameters from fc6 and fc7, change pool5 from $2 \times 2 - s_2$ to $3 \times 3 - s_1$, and use the *a trous* algorithm [18] to fill the “holes”. We remove all the dropout layers and the fc8 layer. We fine-tune the resulting model using SGD with initial learning rate 10^{-3} , 0.9 momentum, 0.0005 weight decay, and batch size 32. The learning rate decay policy is slightly different for each dataset, and we will describe details later. The full training and testing code is built on Caffe [19] and is open source at: <https://github.com/weiliu89/caffe/tree/ssd>.

3.1 PASCAL VOC2007

On this dataset, we compare against Fast R-CNN [6] and Faster R-CNN [2] on VOC2007 test (4952 images). All methods fine-tune on the same pre-trained VGG16 network.

Figure 2 shows the architecture details of the SSD300 model. We use conv4_3, conv7 (fc7), conv8_2, conv9_2, conv10_2, and conv11_2 to predict both location and confidences. We set default box with scale 0.1 on conv4_3³. We initialize the parameters for all the newly added convolutional layers with the “xavier” method [20]. For conv4_3, conv10_2 and conv11_2, we only associate 4 default boxes at each feature map location – omitting aspect ratios of $\frac{1}{3}$ and 3. For all other layers, we put 6 default boxes as described in Sec. 2.2. Since, as pointed out in [12], conv4_3 has a different feature scale compared to the other layers, we use the L2 normalization technique introduced in [12] to scale the feature norm at each location in the feature map to 20 and learn the scale during back propagation. We use the 10^{-3} learning rate for 40k iterations, then continue training for 10k iterations with 10^{-4} and 10^{-5} . When training on VOC2007 trainval, Table 1 shows that our low resolution SSD300 model is already more accurate than Fast R-CNN. When we train SSD on a larger 512×512 input image, it is even more accurate, surpassing Faster R-CNN by 1.7% mAP. If we train SSD with more (i.e. 07+12) data, we see that SSD300 is already better than Faster R-CNN by 1.1% and that SSD512 is 3.6% better. If we take models trained on COCO trainval35k as described in Sec. 3.4 and fine-tuning them on the 07+12 dataset with SSD512, we achieve the best results: 81.6% mAP.

To understand the performance of our two SSD models in more details, we used the detection analysis tool from [21]. Figure 3 shows that SSD can detect various object categories with high quality (large white area). The majority of its confident detections are correct. The recall is around 85-90%, and is much higher with “weak” (0.1 jaccard overlap) criteria. Compared to R-CNN [22], SSD has less localization error, indicating that SSD can localize objects better because it directly learns to regress the object shape and classify object categories instead of using two decoupled steps. However, SSD has more confusions with similar object categories (especially for animals), partly because we share locations for multiple categories. Figure 4 shows that SSD is very sensitive to the bounding box size. In other words, it has much worse performance on smaller

³ For SSD512 model, we add extra conv12_2 for prediction, set s_{min} to 0.15, and 0.07 on conv4_3.

Method	data	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	motorbike	person	plant	sheep	sofa	train	tv
Fast [6]	07	66.9	74.5	78.3	69.2	53.2	36.6	77.3	78.2	82.0	40.7	72.7	67.9	79.6	79.2	73.0	69.0	30.1	65.4	70.2	75.8	65.8
Fast [6]	07+12	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
Faster [2]	07	69.9	70.0	80.6	70.1	57.3	49.9	78.2	80.4	82.0	52.2	75.3	67.2	80.3	79.8	75.0	76.3	39.1	68.3	67.3	81.1	67.6
Faster [2]	07+12	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
Faster [2]	07+12+COCO	78.8	84.3	82.0	77.7	68.9	65.7	88.1	88.4	88.9	63.6	86.3	70.8	85.9	87.6	80.1	82.3	53.6	80.4	75.8	86.6	78.9
SSD300	07	68.0	73.4	77.5	64.1	59.0	38.9	75.2	80.8	78.5	46.0	67.8	69.2	76.6	82.1	77.0	72.5	41.2	64.2	69.1	78.0	68.5
SSD300	07+12	74.3	75.5	80.2	72.3	66.3	47.6	83.0	84.2	86.1	54.7	78.3	73.9	84.5	85.3	82.6	76.2	48.6	73.9	76.0	83.4	74.0
SSD300	07+12+COCO	79.6	80.9	86.3	79.0	76.2	57.6	87.3	88.2	88.6	60.5	85.4	76.7	87.5	89.2	84.5	81.4	55.0	81.9	81.5	85.9	78.9
SSD512	07	71.6	75.1	81.4	69.8	60.8	46.3	82.6	84.7	84.1	48.5	75.0	67.4	82.3	83.9	79.4	76.6	44.9	69.9	69.1	78.1	71.8
SSD512	07+12	76.8	82.4	84.7	78.4	73.8	53.2	86.2	87.5	86.0	57.8	83.1	70.2	84.9	85.2	83.9	79.7	50.3	77.9	73.9	82.5	75.3
SSD512	07+12+COCO	81.6	86.6	88.3	82.4	76.0	66.3	88.6	88.9	89.1	65.1	88.4	73.6	86.5	88.9	85.3	84.6	59.1	85.0	80.4	87.4	81.2

Table 1: **PASCAL VOC2007 test detection results.** Both Fast and Faster R-CNN use input images whose minimum dimension is 600. The two SSD models have exactly the same settings except that they have different input sizes (300×300 vs. 512×512). It is obvious that larger input size leads to better results, and more data always helps. Data: "07": VOC2007 trainval, "07+12": union of VOC2007 and VOC2012 trainval. "07+12+COCO": first train on COCO trainval35k then fine-tune on 07+12.

objects than bigger objects. This is not surprising because those small objects may not even have any information at the very top layers. Increasing the input size (e.g. from 300×300 to 512×512) can help improve detecting small objects, but there is still a lot of room to improve. On the positive side, we can clearly see that SSD performs really well on large objects. And it is very robust to different object aspect ratios because we use default boxes of various aspect ratios per feature map location.

3.2 Model analysis

To understand SSD better, we carried out controlled experiments to examine how each component affects performance. For all the experiments, we use the same settings and input size (300×300), except for specified changes to the settings or component(s).

	SSD300				
more data augmentation?		✓	✓	✓	✓
include $\{\frac{1}{2}, 2\}$ box?	✓		✓	✓	✓
include $\{\frac{1}{3}, 3\}$ box?	✓			✓	✓
use atrous?	✓	✓	✓		✓
VOC2007 test mAP	65.5	71.6	73.7	74.2	74.3

Table 2: Effects of various design choices and components on SSD performance.

Data augmentation is crucial. Fast and Faster R-CNN use the original image and the horizontal flip to train. We use a more extensive sampling strategy, similar to YOLO [5]. Table 2 shows that we can improve 8.8% mAP with this sampling strategy. We do not know how much our sampling strategy will benefit Fast and Faster R-CNN, but they are likely to benefit less because they use a feature pooling step during classification that is relatively robust to object translation by design.

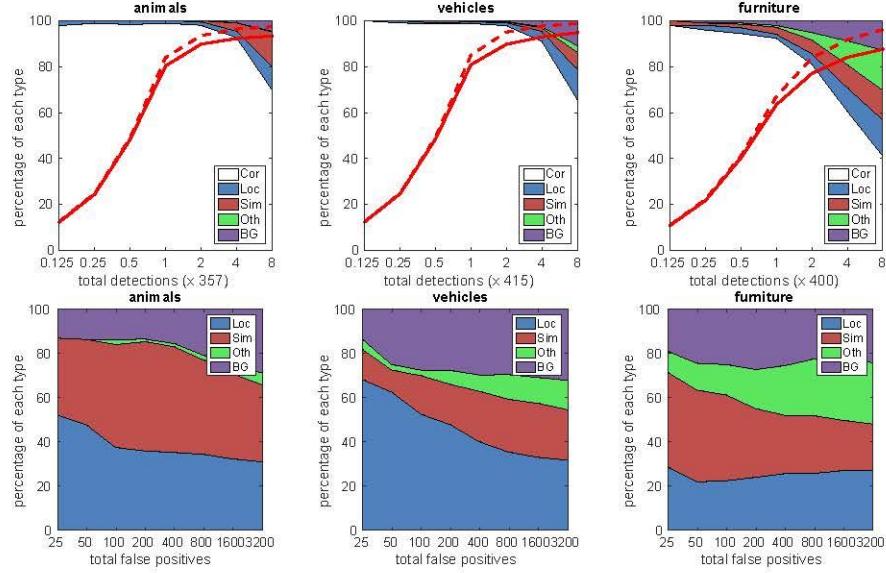


Fig. 3: **Visualization of performance for SSD512 on animals, vehicles, and furniture from VOC2007 test.** The top row shows the cumulative fraction of detections that are correct (Cor) or false positive due to poor localization (Loc), confusion with similar categories (Sim), with others (Oth), or with background (BG). The solid red line reflects the change of recall with strong criteria (0.5 jaccard overlap) as the number of detections increases. The dashed red line is using the weak criteria (0.1 jaccard overlap). The bottom row shows the distribution of top-ranked false positive types.

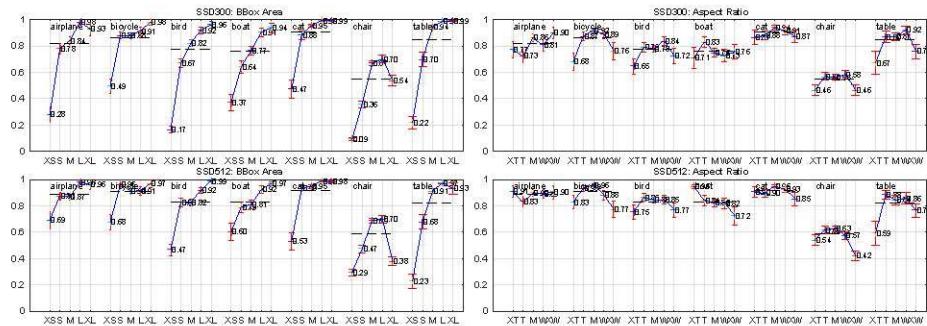


Fig. 4: **Sensitivity and impact of different object characteristics on VOC2007 test set using [21].** The plot on the left shows the effects of BBox Area per category, and the right plot shows the effect of Aspect Ratio. Key: BBox Area: XS=extra-small; S=small; M=medium; L=large; XL =extra-large. Aspect Ratio: XT=extra-tall/narrow; T=tall; M=medium; W=wide; XW =extra-wide.

More default box shapes is better. As described in Sec. 2.2, by default we use 6 default boxes per location. If we remove the boxes with $\frac{1}{3}$ and 3 aspect ratios, the performance drops by 0.6%. By further removing the boxes with $\frac{1}{2}$ and 2 aspect ratios, the performance drops another 2.1%. Using a variety of default box shapes seems to make the task of predicting boxes easier for the network.

Atrous is faster. As described in Sec. 3, we used the atrous version of a subsampled VGG16, following DeepLab-LargeFOV [17]. If we use the full VGG16, keeping pool5 with $2 \times 2 - s2$ and not subsampling parameters from fc6 and fc7, and add conv5_3 for prediction, the result is about the same while the speed is about 20% slower.

Prediction source layers from:						mAP		# Boxes	
conv4_3	conv7	conv8_2	conv9_2	conv10_2	conv11_2	use boundary boxes?			
✓	✓	✓	✓	✓	✓	Yes	74.3	63.4	8732
✓	✓	✓	✓	✓	✓	No	74.6	63.1	8764
✓	✓	✓	✓	✓			73.8	68.4	8942
✓	✓	✓	✓				70.7	69.2	9864
✓	✓	✓					64.2	64.4	9025
✓	✓						62.4	64.0	8664

Table 3: Effects of using multiple output layers.

Multiple output layers at different resolutions is better. A major contribution of SSD is using default boxes of different scales on different output layers. To measure the advantage gained, we progressively remove layers and compare results. For a fair comparison, every time we remove a layer, we adjust the default box tiling to keep the total number of boxes similar to the original (8732). This is done by stacking more scales of boxes on remaining layers and adjusting scales of boxes if needed. We do not exhaustively optimize the tiling for each setting. Table 3 shows a decrease in accuracy with fewer layers, dropping monotonically from 74.3 to 62.4. When we stack boxes of multiple scales on a layer, many are on the image boundary and need to be handled carefully. We tried the strategy used in Faster R-CNN [2], ignoring boxes which are on the boundary. We observe some interesting trends. For example, it hurts the performance by a large margin if we use very coarse feature maps (e.g. conv11_2 (1×1) or conv10_2 (3×3)). The reason might be that we do not have enough large boxes to cover large objects after the pruning. When we use primarily finer resolution maps, the performance starts increasing again because even after pruning a sufficient number of large boxes remains. If we only use conv7 for prediction, the performance is the worst, reinforcing the message that it is critical to spread boxes of different scales over different layers. Besides, since our predictions do not rely on ROI pooling as in [6], we do not have the *collapsing bins* problem in low-resolution feature maps [23]. The SSD architecture combines predictions from feature maps of various resolutions to achieve comparable accuracy to Faster R-CNN, while using lower resolution input images.

3.3 PASCAL VOC2012

We use the same settings as those used for our basic VOC2007 experiments above, except that we use VOC2012 `trainval` and VOC2007 `trainval` and `test` (21503 images) for training, and test on VOC2012 `test` (10991 images). We train the models with 10^{-3} learning rate for 60k iterations, then 10^{-4} for 20k iterations. Table 4 shows the results of our SSD300 and SSD512⁴ model. We see the same performance trend as we observed on VOC2007 test. Our SSD300 improves accuracy over Fast/Faster R-CNN. By increasing the training and testing image size to 512×512 , we are 4.5% more accurate than Faster R-CNN. Compared to YOLO, SSD is significantly more accurate, likely due to the use of convolutional default boxes from multiple feature maps and our matching strategy during training. When fine-tuned from models trained on COCO, our SSD512 achieves 80.0% mAP, which is 4.1% higher than Faster R-CNN.

Method	data	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast[6]	07++12	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
Faster[2]	07++12	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
Faster[2]	07++12+COCO	75.9	87.4	83.6	76.8	62.9	59.6	81.9	82.0	91.3	54.9	82.6	59.0	89.0	85.5	84.7	84.1	52.2	78.9	65.5	85.4	70.2
YOLO[5]	07++12	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
SSD300	07++12	72.4	85.6	80.1	70.5	57.6	46.2	79.4	76.1	89.2	53.0	77.0	60.8	87.0	83.1	82.3	79.4	45.9	75.9	69.5	81.9	67.5
SSD300	07++12+COCO	77.5	90.2	83.3	76.3	63.0	53.6	83.8	82.8	92.0	59.7	82.7	63.5	89.3	87.6	85.9	84.3	52.6	82.5	74.1	88.4	74.2
SSD512	07++12	74.9	87.4	82.3	75.8	59.0	52.6	81.7	81.5	90.0	55.4	79.0	59.8	88.4	84.3	84.7	83.3	50.2	78.0	66.3	86.3	72.0
SSD512	07++12+COCO	80.0	90.7	86.8	80.5	67.8	60.8	86.3	85.5	93.5	63.2	85.7	64.4	90.9	89.0	88.9	86.8	57.2	85.1	72.8	88.4	75.9

Table 4: **PASCAL VOC2012 `test` detection results.** Fast and Faster R-CNN use images with minimum dimension 600, while the image size for YOLO is 448×448 . data: "07++12": union of VOC2007 `trainval` and `test` and VOC2012 `trainval`. "07++12+COCO": first train on COCO `trainval35k` then fine-tune on 07++12.

3.4 COCO

To further validate the SSD framework, we trained our SSD300 and SSD512 architectures on the COCO dataset. Since objects in COCO tend to be smaller than PASCAL VOC, we use smaller default boxes for all layers. We follow the strategy mentioned in Sec. 2.2, but now our smallest default box has a scale of 0.15 instead of 0.2, and the scale of the default box on conv4_3 is 0.07 (e.g. 21 pixels for a 300×300 image)⁵.

We use the `trainval35k` [24] for training. We first train the model with 10^{-3} learning rate for 160k iterations, and then continue training for 40k iterations with 10^{-4} and 40k iterations with 10^{-5} . Table 5 shows the results on `test-dev2015`. Similar to what we observed on the PASCAL VOC dataset, SSD300 is better than Fast R-CNN in both mAP@0.5 and mAP@[0.5:0.95]. SSD300 has a similar mAP@0.75 as ION [24] and Faster R-CNN [25], but is worse in mAP@0.5. By increasing the image size to 512×512 , our SSD512 is better than Faster R-CNN [25] in both criteria. Interestingly, we observe that SSD512 is 5.3% better in mAP@0.75, but is only 1.2% better in mAP@0.5. We also observe that it has much better AP (4.8%) and AR (4.6%) for large objects, but has relatively less improvement in AP (1.3%) and AR (2.0%) for

⁴ <http://host.robots.ox.ac.uk:8080/leaderboard/displaylb.php?cls=mean&challengeid=11&compid=4>

⁵ For SSD512 model, we add extra conv12_2 for prediction, set s_{min} to 0.1, and 0.04 on conv4_3.

Method	data	Avg. Precision, IoU: 0.5:0.95 0.5 0.75			Avg. Precision, Area: S M L			Avg. Recall, #Dets: 1 10 100			Avg. Recall, Area: S M L		
		S	M	L	1	10	100	S	M	L	S	M	L
Fast [6]	train	19.7	35.9	-	-	-	-	-	-	-	-	-	-
Fast [24]	train	20.5	39.9	19.4	4.1	20.0	35.8	21.3	29.5	30.1	7.3	32.1	52.0
Faster [2]	trainval	21.9	42.7	-	-	-	-	-	-	-	-	-	-
ION [24]	train	23.6	43.2	23.6	6.4	24.1	38.3	23.2	32.7	33.5	10.1	37.7	53.6
Faster [25]	trainval	24.2	45.3	23.5	7.7	26.4	37.1	23.8	34.0	34.6	12.0	38.5	54.4
SSD300	trainval35k	23.2	41.2	23.4	5.3	23.2	39.6	22.5	33.2	35.3	9.6	37.6	56.5
SSD512	trainval35k	26.8	46.5	27.8	9.0	28.9	41.9	24.8	37.5	39.8	14.0	43.5	59.0

Table 5: COCO **test-dev2015** detection results.

small objects. Compared to ION, the improvement in AR for large and small objects is more similar (5.4% vs. 3.9%). We conjecture that Faster R-CNN is more competitive on smaller objects with SSD because it performs two box refinement steps, in both the RPN part and in the Fast R-CNN part. In Fig. 5, we show some detection examples on COCO **test-dev** with the SSD512 model.

3.5 Preliminary ILSVRC results

We applied the same network architecture we used for COCO to the ILSVRC DET dataset [16]. We train a SSD300 model using the ILSVRC2014 DET **train** and **val1** as used in [22]. We first train the model with 10^{-3} learning rate for 320k iterations, and then continue training for 80k iterations with 10^{-4} and 40k iterations with 10^{-5} . We can achieve 43.4 mAP on the **val2** set [22]. Again, it validates that SSD is a general framework for high quality real-time detection.

3.6 Data Augmentation for Small Object Accuracy

Without a follow-up feature resampling step as in Faster R-CNN, the classification task for small objects is relatively hard for SSD, as demonstrated in our analysis (see Fig. 4). The data augmentation strategy described in Sec. 2.2 helps to improve the performance dramatically, especially on small datasets such as PASCAL VOC. The random crops generated by the strategy can be thought of as a "zoom in" operation and can generate many larger training examples. To implement a "zoom out" operation that creates more small training examples, we first randomly place an image on a canvas of $16 \times$ of the original image size filled with mean values before we do any random crop operation. Because we have more training images by introducing this new "expansion" data augmentation trick, we have to double the training iterations. We have seen a consistent increase of 2%-3% mAP across multiple datasets, as shown in Table 6. In specific, Figure 6 shows that the new augmentation trick significantly improves the performance on small objects. This result underscores the importance of the data augmentation strategy for the final model accuracy.

An alternative way of improving SSD is to design a better tiling of default boxes so that its position and scale are better aligned with the receptive field of each position on a feature map. We leave this for future work.



Fig. 5: **Detection examples on COCO test-dev with SSD512 model.** We show detections with scores higher than 0.6. Each color corresponds to an object category.

Method	VOC2007 test		VOC2012 test		COCO test-dev2015 trainval35k		
	07+12 0.5	07+12+COCO 0.5	07++12 0.5	07++12+COCO 0.5	0.5:0.95	0.5	0.75
SSD300	74.3	79.6	72.4	77.5	23.2	41.2	23.4
SSD512	76.8	81.6	74.9	80.0	26.8	46.5	27.8
SSD300*	77.2	81.2	75.8	79.3	25.1	43.1	25.8
SSD512*	79.8	83.2	78.5	82.2	28.8	48.5	30.3

Table 6: **Results on multiple datasets when we add the image expansion data augmentation trick.** SSD300* and SSD512* are the models that are trained with the new data augmentation.

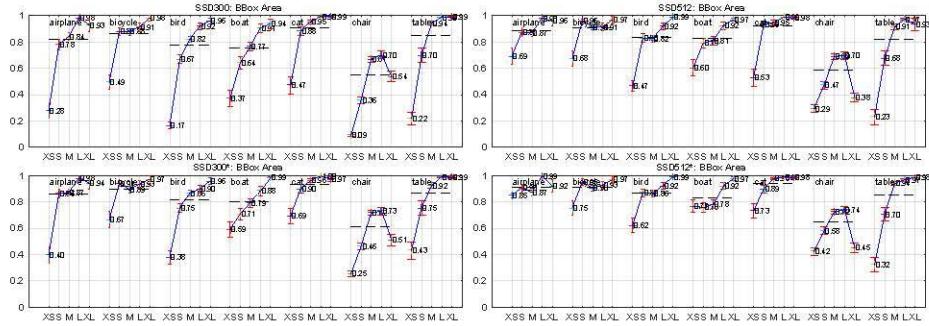


Fig. 6: **Sensitivity and impact of object size with new data augmentation on VOC2007 test set using [21].** The top row shows the effects of BBox Area per category for the original SSD300 and SSD512 model, and the bottom row corresponds to the SSD300* and SSD512* model trained with the new data augmentation trick. It is obvious that the new data augmentation trick helps detecting small objects significantly.

3.7 Inference time

Considering the large number of boxes generated from our method, it is essential to perform non-maximum suppression (nms) efficiently during inference. By using a confidence threshold of 0.01, we can filter out most boxes. We then apply nms with jaccard overlap of 0.45 per class and keep the top 200 detections per image. This step costs about 1.7 msec per image for SSD300 and 20 VOC classes, which is close to the total time (2.4 msec) spent on all newly added layers. We measure the speed with batch size 8 using Titan X and cuDNN v4 with Intel Xeon E5-2667v3@3.20GHz.

Table 7 shows the comparison between SSD, Faster R-CNN[2], and YOLO[5]. Both our SSD300 and SSD512 method outperforms Faster R-CNN in both speed and accuracy. Although Fast YOLO[5] can run at 155 FPS, it has lower accuracy by almost 22% mAP. To the best of our knowledge, SSD300 is the first real-time method to achieve above 70% mAP. Note that about 80% of the forward time is spent on the base network (VGG16 in our case). Therefore, using a faster base network could even further improve the speed, which can possibly make the SSD512 model real-time as well.

4 Related Work

There are two established classes of methods for object detection in images, one based on sliding windows and the other based on region proposal classification. Before the advent of convolutional neural networks, the state of the art for those two approaches – Deformable Part Model (DPM) [26] and Selective Search [1] – had comparable performance. However, after the dramatic improvement brought on by R-CNN [22], which combines selective search region proposals and convolutional network based post-classification, region proposal object detection methods became prevalent.

The original R-CNN approach has been improved in a variety of ways. The first set of approaches improve the quality and speed of post-classification, since it requires

Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	~ 1000 × 600
Fast YOLO	52.7	155	1	98	448 × 448
YOLO (VGG16)	66.4	21	1	98	448 × 448
SSD300	74.3	46	1	8732	300 × 300
SSD512	76.8	19	1	24564	512 × 512
SSD300	74.3	59	8	8732	300 × 300
SSD512	76.8	22	8	24564	512 × 512

Table 7: **Results on Pascal VOC2007 test.** SSD300 is the only real-time detection method that can achieve above 70% mAP. By using a larger input image, SSD512 outperforms all methods on accuracy while maintaining a close to real-time speed.

the classification of thousands of image crops, which is expensive and time-consuming. SPPnet [9] speeds up the original R-CNN approach significantly. It introduces a spatial pyramid pooling layer that is more robust to region size and scale and allows the classification layers to reuse features computed over feature maps generated at several image resolutions. Fast R-CNN [6] extends SPPnet so that it can fine-tune all layers end-to-end by minimizing a loss for both confidences and bounding box regression, which was first introduced in MultiBox [7] for learning objectness.

The second set of approaches improve the quality of proposal generation using deep neural networks. In the most recent works like MultiBox [7,8], the Selective Search region proposals, which are based on low-level image features, are replaced by proposals generated directly from a separate deep neural network. This further improves the detection accuracy but results in a somewhat complex setup, requiring the training of two neural networks with a dependency between them. Faster R-CNN [2] replaces selective search proposals by ones learned from a region proposal network (RPN), and introduces a method to integrate the RPN with Fast R-CNN by alternating between fine-tuning shared convolutional layers and prediction layers for these two networks. This way region proposals are used to pool mid-level features and the final classification step is less expensive. Our SSD is very similar to the region proposal network (RPN) in Faster R-CNN in that we also use a fixed set of (default) boxes for prediction, similar to the anchor boxes in the RPN. But instead of using these to pool features and evaluate another classifier, we simultaneously produce a score for each object category in each box. Thus, our approach avoids the complication of merging RPN with Fast R-CNN and is easier to train, faster, and straightforward to integrate in other tasks.

Another set of methods, which are directly related to our approach, skip the proposal step altogether and predict bounding boxes and confidences for multiple categories directly. OverFeat [4], a deep version of the sliding window method, predicts a bounding box directly from each location of the topmost feature map after knowing the confidences of the underlying object categories. YOLO [5] uses the whole topmost feature map to predict both confidences for multiple categories and bounding boxes (which are shared for these categories). Our SSD method falls in this category because we do not have the proposal step but use the default boxes. However, our approach is more flexible than the existing methods because we can use default boxes of different aspect

ratios on each feature location from multiple feature maps at different scales. If we only use one default box per location from the topmost feature map, our SSD would have similar architecture to OverFeat [4]; if we use the whole topmost feature map and add a fully connected layer for predictions instead of our convolutional predictors, and do not explicitly consider multiple aspect ratios, we can approximately reproduce YOLO [5].

5 Conclusions

This paper introduces SSD, a fast single-shot object detector for multiple categories. A key feature of our model is the use of multi-scale convolutional bounding box outputs attached to multiple feature maps at the top of the network. This representation allows us to efficiently model the space of possible box shapes. We experimentally validate that given appropriate training strategies, a larger number of carefully chosen default bounding boxes results in improved performance. We build SSD models with at least an order of magnitude more box predictions sampling location, scale, and aspect ratio, than existing methods [5, 7]. We demonstrate that given the same VGG-16 base architecture, SSD compares favorably to its state-of-the-art object detector counterparts in terms of both accuracy and speed. Our SSD512 model significantly outperforms the state-of-the-art Faster R-CNN [2] in terms of accuracy on PASCAL VOC and COCO, while being $3\times$ faster. Our real time SSD300 model runs at 59 FPS, which is faster than the current real time YOLO [5] alternative, while producing markedly superior detection accuracy.

Apart from its standalone utility, we believe that our monolithic and relatively simple SSD model provides a useful building block for larger systems that employ an object detection component. A promising future direction is to explore its use as part of a system using recurrent neural networks to detect and track objects in video simultaneously.

6 Acknowledgment

This work was started as an internship project at Google and continued at UNC. We would like to thank Alex Toshev for helpful discussions and are indebted to the Image Understanding and DistBelief teams at Google. We also thank Philip Ammirato and Patrick Poirson for helpful comments. We thank NVIDIA for providing GPUs and acknowledge support from NSF 1452851, 1446631, 1526367, 1533771.

References

1. Uijlings, J.R., van de Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. IJCV (2013)
2. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS. (2015)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016)
4. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networks. In: ICLR. (2014)

5. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR. (2016)
6. Girshick, R.: Fast R-CNN. In: ICCV. (2015)
7. Erhan, D., Szegedy, C., Toshev, A., Anguelov, D.: Scalable object detection using deep neural networks. In: CVPR. (2014)
8. Szegedy, C., Reed, S., Erhan, D., Anguelov, D.: Scalable, high-quality object detection. arXiv preprint arXiv:1412.1441 v3 (2015)
9. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: ECCV. (2014)
10. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR. (2015)
11. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: CVPR. (2015)
12. Liu, W., Rabinovich, A., Berg, A.C.: ParseNet: Looking wider to see better. In: ICLR. (2016)
13. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Object detectors emerge in deep scene cnns. In: ICLR. (2015)
14. Howard, A.G.: Some improvements on deep convolutional neural network based image classification. arXiv preprint arXiv:1312.5402 (2013)
15. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: NIPS. (2015)
16. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: Imagenet large scale visual recognition challenge. IJCV (2015)
17. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. In: ICLR. (2015)
18. Holschneider, M., Kronland-Martinet, R., Morlet, J., Tchamitchian, P.: A real-time algorithm for signal analysis with the help of the wavelet transform. In: Wavelets. Springer (1990) 286–297
19. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: MM. (2014)
20. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: AISTATS. (2010)
21. Hoiem, D., Chodpathumwan, Y., Dai, Q.: Diagnosing error in object detectors. In: ECCV 2012. (2012)
22. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR. (2014)
23. Zhang, L., Lin, L., Liang, X., He, K.: Is faster r-cnn doing well for pedestrian detection. In: ECCV. (2016)
24. Bell, S., Zitnick, C.L., Bala, K., Girshick, R.: Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In: CVPR. (2016)
25. COCO: Common Objects in Context. <http://mscoco.org/dataset/#detections-leaderboard> (2016) [Online; accessed 25-July-2016].
26. Felzenszwalb, P., McAllester, D., Ramanan, D.: A discriminatively trained, multiscale, deformable part model. In: CVPR. (2008)

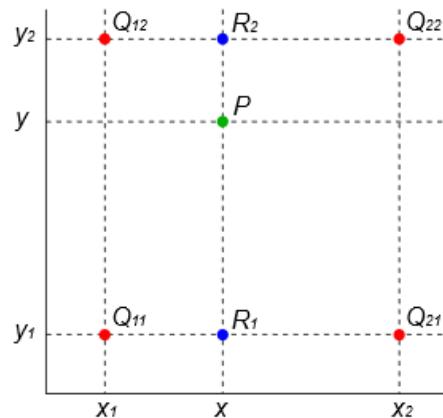
附录三：双线性内插法计算方法：

双线性内插法下图所示。首先设放大的图片中 P 点坐标为 (x, y) ，图像长宽放大倍数分别为 α 与 β 。为了求 P 点的像素值，需要利用到原图的四个最近邻点，这四个点的坐标可通过 P 点坐标除以放大倍数然后取整得到。然后在 x 方向上进行两次线性插值得到 $f(R_1)$ 和 $f(R_2)$ ，如以下公式所示。最后对 P 点在 y 轴上进行一次线性插值算法得到 P 点的像素值 $f(P)$ 。

$$f(R_1) = \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$

$$f(R_2) = \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})$$

$$f(P) = \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2)$$



图： 双线性内插法

附录四：非极大值抑制算法

非极大值抑制算法

输入：候选框集合 $B = \{b_1, \dots, b_N\}$ ，候选框对应的置信度 $S = \{s_1, \dots, s_N\}$ ，阈值 t

1. $D = \{\}$ //输出序列
2. while $B \neq \emptyset$ do
3. $m \leftarrow \text{argmax } S$ // 根据分数排列，得到分数最大的候选框的索引
4. $D \leftarrow D \cup b_m; B \leftarrow B - b_m$ //将 b_m 移除候选序列并加入到输出序列
5. for b_i in B do
6. if $\text{IoU}(b_m, b_i) > t$ then //交并比大于阈值
7. $B \leftarrow B - b_i; S \leftarrow S - s_i$ //将 b_i 移除候选框序列
8. end
9. end
10. end
11. 输出： D, S

附录五：改进版 SSD 模型对 20 种类别的 PR 曲线

