第 27 卷 第 3 期
2021 年 6 月
上海大学学报（自然科学版）
JOURNAL OF SHANGHAI UNIVERSITY (NATURAL SCIENCE EDITION)
Vol. 27 No. 3
June 2021

DOI: 10.12066/j.issn.1007-2861.2279

基于改进 YOLOv3 算法的水下小目标分类与识别

邵慧翔, 曾 丹
(上海大学 通信与信息工程学院, 上海 200444)

摘要: 针对声纳图像中小目标检测识别率低、虚警率高的问题, 提出一种改进的YOLOv3算法. 改进的YOLOv3网络在原始YOLOv3的基础上进行优化, 改变网络的层级连接, 融合更浅层的特征与深层特征, 形成新的更大尺度的检测层, 提高了网络对水下小目标检测的能力; 同时, 使用线性缩放的K-means聚类算法优化计算先验框个数和宽高比, 提高了先验框与ground truth box之间的匹配度, 较原始YOLOv3算法均值平均精度提高了7%. 实验结果表明, 所提出的改进YOLOv3算法能够有效分类与识别小目标且有更高的准确率和更低的虚警率, 同时保持了原始YOLOv3算法的实时性.

关键词: YOLOv3; 小目标检测; 深度学习
中图分类号: TB 567 **文献标志码:** A **文章编号:** 1007-2861(2021)03-0481-11

Classification and recognition of underwater small targets based on improved YOLOv3 algorithm

SHAO Huixiang, ZENG Dan
(School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China)

Abstract: This study proposes an improved YOLOv3 algorithm designed to address the twin issues of low detection and recognition rate and high false alarm rate with respect to the detection of small targets by sonar. The improved YOLOv3 network is optimised on the basis of the original YOLOv3 algorithm, with the hierarchical connection of the network changed and the features of the shallow and deep layers fused to form a new larger-scale detection layer. Concurrently, the linear scaling K-means clustering algorithm is used to optimise the calculation of the number of a priori boxes and the aspect ratio, thereby improving the correlation between the a priori and ground truth boxes. These modifications improve the average accuracy of the YOLOv3 algorithm by 7%. Experimental results show that the proposed improvements to the YOLOv3 algorithm result in the effective identification of small targets with higher accuracy and lower false alarm rate, while maintaining the real-time processing capabilities of the YOLOv3 algorithm.

收稿日期: 2020-09-28
通信作者: 曾 丹(1982—), 女, 教授, 博士, 研究方向为计算机视觉与模式识别. E-mail: dzeng@shu.edu.cn
引用格式: 邵慧翔, 曾丹. 基于改进YOLOv3算法的水下小目标分类与识别 [J]. 上海大学学报(自然科学版), 2021, 27(3): 481-491.
Citation format: SHAO H X, ZENG D. Classification and recognition of underwater small targets based on improved YOLOv3 algorithm [J]. Journal of Shanghai University (Natural Science Edition), 2021, 27(3): 481-491.

Key words: YOLOv3; small target detection; deep learning

水下目标识别一直是水声领域的研究热点. 水下目标识别技术应用广泛, 在军事领域可用于侦察跟踪、布雷探雷和海上救援等, 在民用领域可用于海底管道铺设、鱼群监测、海底状况考察、海底设施的维护与监测等. 针对声纳图像识别技术, 国外研究方向主要有3个: 一是通过大量的测绘与试验, 收集海洋环境数据信息和人类在海上的活动信息; 二是通过大量试验测量海洋中声纳信号的传播特性; 三是研究主动声纳工作的特性, 根据目标噪声或回波的波形音调、节奏分布特征进行人为识别. 基于传统机器学习方法进行水下目标识别的研究已有很多, 例如Sherin等[1]采用语言信号在不同频率范围的分布(梅尔频率倒谱系数(Mel frequency cepstrum coefficient, MFCC))特征, 对每帧特征向量利用K-means算法进行聚类, 构造声纳图像纹理特征, 作为支持向量机(support vector machine, SVM)的训练集来训练二分类模型, 泛化误差为9%. 在深度神经网络出现之前, 早期的传统声纳图像目标检测方法耗时且精度不高, 随着基于深度学习的区域卷积神经网络(region-convolutional neural network, R-CNN)方法[2]提出后, 目标检测的性能有了质的飞跃. 在目标检测领域, 主要有两类方法: 一类是以R-CNN为代表的二阶段(two-stage)检测算法, 使用区域候选网络(region proposal network, RPN)产生候选区域, 然后通过神经网络对候选区域进行分类及定位, 这种方法的准确度较高, 但检测速度稍慢; 另一类是以YOLO(you only look once)[3]为代表的单阶段(one-stage)检测算法, 直接回归得出目标区域, 再通过神经网络进行分类, one-stage检测算法不需要RPN阶段, 所以检测速度较快, 但检测精度较低.

已有的检测算法中以深度学习的目标检测算法最为“优秀”, 不过单发多框检测器(single shot multibox detection, SSD)[4]、R-CNN系列的网络复杂度过高, 即便使用运算速度非常快的GPU, 推断速度仍然很慢. 而YOLO系列的网络解决了复杂度过高的问题, 在主流GPU上运行速度达到60帧/s以上, 满足实时的需求. 本工作采用改进的YOLOv3[5]算法进行水下小目标的分类与识别, 使用改进的聚类方法对ground truth进行聚类, 即随机确定k个聚类中心, 再进行聚类计算; 同时调整YOLOv3的网络连接结构, 提高了算法对小目标的检测能力; 最后, 使用水雷声纳图像数据集进行验证. 实验结果表明, 改进的YOLOv3算法实现了端到端的水下小目标检测, 具有较高的准确度和较好的实时性.

1 YOLOv3算法原理

1.1 检测原理

当声纳图像中的目标雷很小, 所占像素面积约小于30×30时, 称这种目标雷为小目标. 由于神经网络不断下采样, 自然会造成小目标的丢失, 一般在检测器关于物体大小的平均精确率报告中, 小目标的平均精确率是最低的.

相比原始YOLO系列算法, YOLOv3算法在小目标检测方面的精度有显著提升. 在目标位置的预测上使用预设的锚点框(anchor boxes), 采用特征金字塔[6]布局网络, 使用3个不同尺度的特征图, 通过拼接和卷积运算, 融合浅层特征与深层特征. YOLOv3网络分别对提取出的3个不同尺度的特征图进行检测, 提升了算法对不同大小目标的检测性能. 如图1所示, 对于一张输入图像, YOLOv3算法将输入图像分为$S \times S$个网格, 网格的宽为c_x, 高为c_y, 在每一个网格中预测B个边界框(bounding boxes), 输出是边界框中心点相对于当前网格左上角坐标的偏移量(t_x, t_y, t_w, t_h), 然后用如下公式计算出边界框相对于特征图的位置和大

小 (b_x, b_y, b_w, b_h), 对应图 1 中的蓝色实线,

$$b_x = \sigma(t_x) + c_x, \quad b_y = \sigma(t_y) + c_y,$$
$$b_w = p_w e^{t_w}, \quad b_h = p_h e^{t_h}. \tag{1}$$

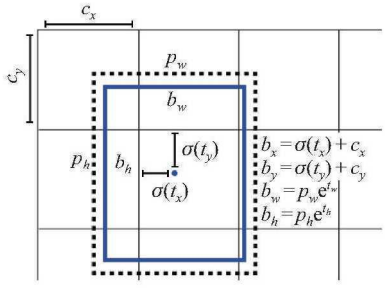

图 1　预测框

Fig. 1　Prediction box

1.2　候选框的确定

图像中总共有 $S \times S$ 个网格, 每个网格预测出 B 个边界框, 每个边界框对应 5 个预测参数: 4 个坐标和 1 个置信度. 参与损失函数计算的值包括预测框的中心点位置和宽高, 预测框的置信度指的是预测框与标记框之间的交并比 (intersection-over-union, IOU)[7], YOLOv3 中设置 IOU>0.7 的预测框为正例, IOU<0.3 的预测框为负例, 忽略 0.3~0.7 之间的预测框, 即只是用被判定为正例和负例的预测框信息计算损失函数.

1.3　激活函数

改进激活函数也可以提高小目标的识别能力, 激活函数的选取标准应是使更多的浅层神经元被激活, 并且赋予不同的权重. YOLOv3 中使用 Leaky ReLU[8] 作为激活函数,

$$\text{Leaky ReLU} = \begin{cases} x_i, & x_i \geqslant 0, \\ \dfrac{x_i}{a_i}, & x_i < 0, \end{cases} \tag{2}$$

式中: a_i 是 $(1, +\infty)$ 区间内的一个值. 可知, ReLU[9] 将输入为负值都设为 0, 但是在 Leaky ReLU 中是在输入为负值时添加一个非零斜率. 如图 2 所示, 横坐标为网络输入, 纵坐标为网络输出, 解决了 ReLU 中在输入值为负值时, 输出始终为 0, 一阶导数也始终为 0, 导致神经元不能正常更新参数的问题. 由于在负值处引入了很小的坡度, 使得导数不为 0, 因此允许进行很慢的梯度学习[10].

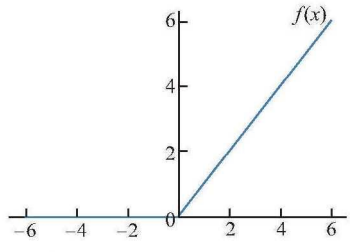

图 2　Leaky ReLU 函数图像

Fig. 2　Leaky ReLU function graph

1.4 损失函数

在模型优化过程中, 通过调整学习率对模型的优化起到的作用很小[11], 而使用更优的损失函数对于模型的优化有很大的作用. YOLOv3 中将 loss 分为 3 个部分: 回归损失 $l_{\rm box}$ 是通过边界框坐标计算 loss, 对应式 (3); 置信度损失 $l_{\rm cls}$ 是置信度带来的 loss, 对应式 (4); 分类损失 $l_{\rm obj}$ 是类别带来的 loss, 对应式 (5).

$$\begin{aligned} l_{\rm box} =&\lambda_{\rm coord}\sum_{i=0}^{S\times S}\sum_{j=0}^{B}I_{ij}^{\rm obj}(2-w_i\times h_i)((x_i-\widehat{x}_i)^2+(y_i-\widehat{y}_i)^2)\\ &+\lambda_{\rm coord}\sum_{i=0}^{S\times S}\sum_{j=0}^{B}I_{ij}^{\rm obj}(2-w_i\times h_j)((w_i-\widehat{w}_i)^2+(h_i-\widehat{h}_i)^2), \end{aligned} \tag{3}$$

$$\begin{aligned} l_{\rm cls} =&\lambda_{\rm obj}\sum_{i=0}^{S\times S}\sum_{j=0}^{B}I_{ij}^{\rm obj}(\widehat{C}_i\log C_i+(1-\widehat{C}_i)\log(1-C_i))\\ &-\lambda_{\rm noobj}\sum_{i=0}^{S\times S}\sum_{j=0}^{B}I_{ij}^{\rm noobj}(\widehat{C}_i\log C_i+(1-\widehat{C}_i)\log(1-C_i)), \end{aligned} \tag{4}$$

$$l_{\rm obj}=\sum_{i=0}^{S\times S}\sum_{j=0}^{B}I_{ij}^{\rm noobj}\sum_{c\in{\rm classes}}(\widehat{p}_i(c)\log p_i(c)+(1-\widehat{p}_i)\log(1-p_i(c))), \tag{5}$$

式 (3)~(5) 中: S 为图像的划分系数; B 为每个网格预测的边界框个数; C 为类别数; p 为类别概率; x_i, y_i, w_i, h_i 为网格中边界框中心点处的横纵坐标及宽高; λ 为权重系数.

从 YOLOv3 中的损失函数来看, 并不是网络的所有输出都需要计算 loss, 当物体中心落入 cell 中, 就需计算分类损失和置信度损失, 预测出的边界框与 ground truth 的交并比较大的需要计算回归损失. 而回归损失会乘以 $(2-w_i\times h_i)$ 的比例系数, 以提高对小目标的识别能力, 其中 w_i 和 h_i 分别是 ground truth 的宽和高.

2 YOLOv3 算法的改进

2.1 改进的 K-means 聚类算法

不同数据中目标的尺度大小及纵横比不同, 所以选择合适的锚框尺度对于算法识别能力有很大影响. 那么, 给定若干个尺度大小的数据, 如何将其进行归类呢? 这里的“类”是指具有相似特征的数据的集合. 聚类算法就是将整个数据集划分为多个类, 使得最后每个类内数据相似, 类间数据差异尽可能大. K-means[12] 是一种简单且经典的基于距离的聚类算法, 也称为 k 均值聚类算法, 一般采用距离作为相似性评价指标, 即认为两个对象的距离越近, 那么二者的相似度就越大. 给定一个数据集 $\{x_1,x_2,\cdots,x_n\}$, 其中每个数据都是一个 d 维度数据点, k 均值聚类就是将这 n 个数据划分到 k 个集合中, 并使得组内平方和最小. 换句话说, K-means 聚类的目标就是找到使式 (6) 满足的聚类 S_i,

$$\arg\min_{S}\sum_{i=1}^{k}\sum_{x\in S_i}\|x-\mu_i\|^2, \tag{6}$$

式中: μ_i 是 S_i 中所有点的均值.

K-means 的实现是多次迭代的结果, 算法分为如下 4 步.

(1) 随机产生 k 个位置, 作为 k 个聚类中心;

(2) 对数据集中的每个点, 计算其与聚类中心的距离, 离哪个聚类中心近, 就划分到聚类中心所属的集合;

(3) 将数据归好集合后, 重新计算每个集合的聚类中心;

(4) 若重新计算出的聚类中心与原始的聚类中心之间的距离小于某个阈值, 表示趋于稳定, 就可以认为聚类已经达到期望的结果, 算法终止. 否则返回步骤 (2), 继续迭代.

在 YOLOv3 中, Anchors 参数需要人为设定, 通过 K-means 聚类得到给定数据集先验框的宽高, 然后将聚类结果按照大小顺序分配给 3 个 feature map 作为先验框, 使用 IOU 作为聚类的距离函数, 聚类的目的是使 anchor boxes 和临近的 ground truth 有更大的 IOU 值,

$$d(\text{box}, \text{centroid}) = 1 - \text{IOU}(\text{box}, \text{centroid}), \tag{7}$$

可见距离越小越好, 但 IOU 越大越好, 所以使用 1—IOU 就可以保证满足条件.

针对 YOLOv3 的网络结构, 采用特征金字塔网络 (feature pyramid networks, FPN) 进行多尺度特征预测, 使用小尺度的 feature map 检测大目标, 大尺度的 feature map 检测小目标, 但是考虑到水下小目标数据集中的目标尺度大小是比较集中的, 所以没有充分发挥出 YOLOv3 多尺度的价值, 而且在实验过程中发现 ground truth 比大多数聚类得到的锚框尺度要大.

本工作针对以上问题进行了特定改动, 即将 K-means 聚类后的结果进行线性缩放, 将锚框的尺度进行拉伸 (见图 3), 简单来说就是将锚框的尺度乘以一定的系数, 使大的尺度更大, 小的尺度更小,

$$\begin{aligned} x_1' &= \alpha x_1, \\ x_9' &= \beta x_9, \\ x_i' &= \frac{(x_i - x_1)}{(x_9 - x_1)}(x_9' - x_1') + x_1', \\ y_i' &= x_i' \frac{y_i}{x_i}, \end{aligned} \tag{8}$$

式中: 设定 $\alpha = 0.5$; $\beta = 3$.

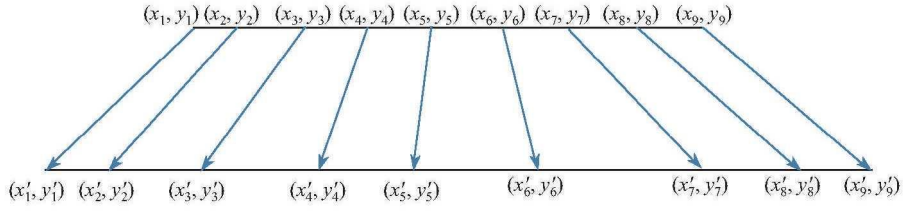

图 3　线性缩放 K-means 聚类算法

Fig. 3　Linear scaling K-means clustering algorithm

本工作中的缩放方式是一种线性缩放, 并不会改变真值框中心点的位置, 即缩放方式是对真值框中心点进行长宽的同步拉伸. 但这种缩放方式也有一定的局限性, 当目标尺寸接近训练图尺寸时, 无使用的必要性. 由于自建数据集中目标较小 (远小于训练图尺寸), 人为框出的真值框并不是紧贴目标, 所以将小尺寸真值框进行缩小或者大尺寸真值框进行放大, 并不会造成真值框框不全目标. 并且本工作以原始 YOLOv3 为基础进行多尺度训练, 而对真值框进行线性缩放能够使真值框的尺度更加符合多尺度, 有益于选框的多样性.

选定 $K = 1 \sim 14$ (坐标轴 x 对应 1~14) 分别对数据集中的 ground truth 框的尺度大小进行聚类计算, 得出 K 与 Avg IOU 之间的关系如图 4 所示. 随着 K 值的增大, 目标函数的变化趋于平稳, 从数据上可以分析出, 当 K 等于 12 时, Avg IOU 的值达到最大.

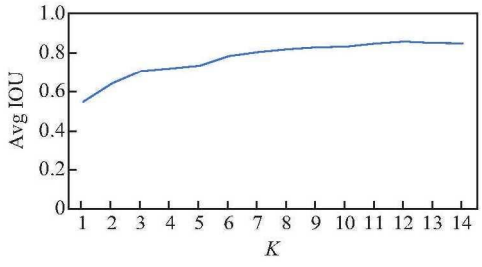

图 4 K-means 聚类分析结果

Fig. 4 K-means clustering analysis results

2.2 网络结构改进

YOLOv3 中图像特征提取采用 Darknet-53 的网络结构 (见表 1), 含有 53 个卷积层, 并借鉴了残差网络[13], 用 1×1 和 3×3 大小的卷积核构成残差块, 在一些层之间设置快捷链路

表 1 Darknet-53 网络结构

Table 1 Darknet-53 network structure

网络层	卷积核	尺寸	输出
Convolutional	32	3×3	256×256
Convolutional	64	$3 \times 3/2$	128×128
Convolutional	32	1×1	—
Convolutional	64	3×3	—
Residual	—	—	128×128
Convolutional	128	$3 \times 3/2$	64×64
Convolutional	64	1×1	—
Convolutional	128	3×3	—
Residual	—	—	64×64
Convolutional	256	$3 \times 3/2$	32×32
Convolutional	128	1×1	—
Convolutional	256	3×3	—
Residual	—	—	32×32
Convolutional	512	$3 \times 3/2$	16×16
Convolutional	256	1×1	—
Convolutional	512	3×3	—
Residual	—	—	16×16
Convolutional	1 024	$3 \times 3/2$	8×8
Convolutional	512	1×1	—
Convolutional	1 024	3×3	—
Residual			8×8

(shortcut connections), 避免了网络层数增加后梯度消失的问题. 每个卷积层之后使用 Batch Normalization[14] 和 Leaky ReLU, 提高了网络的泛化性及推断速度. Darknet-53 为全卷积结构, 去掉了所有的最大池化 (MaxPooling) 层, 使用步长为 2 的卷积层代替, 避免使用池化后小目标特征丢失. 以自建数据集为例, 使用原始的 YOLOv3 网络进行检测, 输入图片的大小为 512×512, 最终划分的网格大小分别为 16×16, 32×32, 64×64, 通过 3 个检测层检测后, 能够得到的预测框数量有 5 376 个. 原始 YOLOv3 中使用表 1 中的 32×32, 16×16, 8×8 大小特征层, 在两两之间上采样后拼接, 对小、中、大尺度目标进行预测.

使用上采样的原因是, 越深层级中的细节越容易丢失, 甚至丢失小目标的语义, 所以上采样的操作对小目标是“友好”的. 但是直接使用浅层的特征进行预测往往不可行, 因为浅层的语义信息不够丰富, 所以对深层的特征图进行上采样后与浅层的特征图拼接成一个更大的特征图更有利于预测.

与原始 YOLOv3 相比, 本工作的改进算法利用了更浅层的特征图与深层特征图上采样后进行拼接, 获得了更大的特征图, 并且使用了更浅的层去学习小目标的特征. 调整后的网络如图 5 所示, 其中 DBL 模块、Res_unit、resn 模块均与原始 YOLOv3 保持一致, 图中虚线部分即为改动部分, 新的大尺寸特征图更有利于小目标检测. 以自建数据集为例, 使用本工作算法进行实验, 输入图片的大小为 512×512, 最终划分的网格大小分别为 32×32, 64×64, 128×128, 通过 3 个检测层检测后, 能够得到的预测框数量有 21 504 个, 预测框数量是原始 YOLOv3 的 4 倍, 预测框数量的增加提高了模型的检测能力.

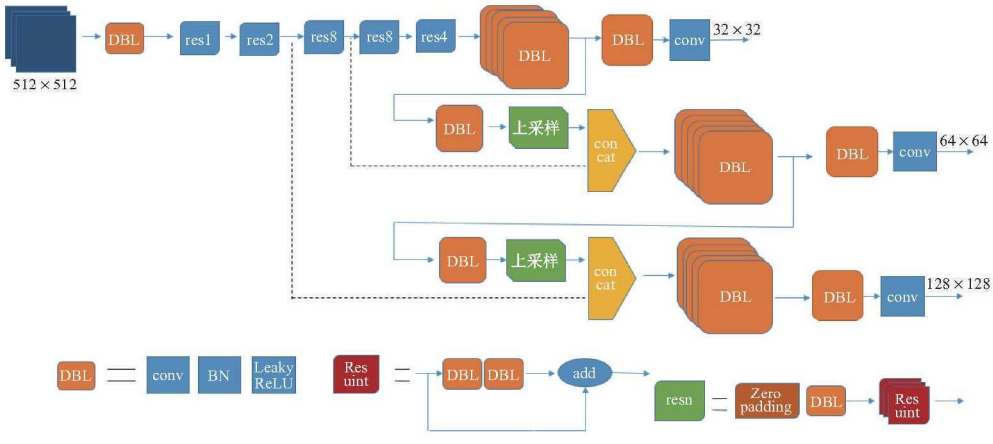

图 5　改进的 YOLOv3 网络结构

Fig. 5　Improved YOLOv3 network structure

3　实验与分析

3.1　数据集和实验平台

实验数据库使用的是实验室自建的水下目标数据集, 图像为声纳采集的水下目标, 其中目标尺度基本属于小目标, 且目标尺度分布跨度大, 数据量较少, 容易产生过拟合, 所以使用数据扩增方式扩大了数据集规模, 包括随机生成类似目标形状的数据、最近邻放大目标[15]、线性增加像素值、目标梯度图[16]、左右翻转、旋转一定角度等操作. 自建仿真数据集约 1 812 张图像, 其中 60% 作为训练, 20% 作为验证, 20% 作为测试使用.

采用批梯度下降的方式训练网络, 批大小设置为 4, 初始学习率设置为 0.001, 动量设置

为 0.9, 权重衰减设置为 0.000 5, 训练时对图像进行饱和度调整、曝光量调整和色调调整等操作, 提高模型的泛化性能.

实验基于 Anaconda3 平台, 通过 Python 3.6 完成, 所有实验均在 Intel Core i7-9750H CPU 和 NVDIA GeForce RTX 2070, 16 GB 内存的 Windows 平台上完成, 深度学习框架为 Pytorch 1.4.0.

3.2 实验结果分析

由于自建数据集 131 和 138 只有一类目标 (雷), 因此采用精确率 P (precision) 和召回率 R (recall) 作为模型的评价标准[17],

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} = \frac{\mathrm{TP}}{n}, \tag{9}$$

$$R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = \frac{\mathrm{TP}}{n'}, \tag{10}$$

式中: TP 为网络预测出的真目标数; FN 为未能成功预测出的真目标数; TP+FP (n) 为预测的目标中是正类目标的数目; TP+FN (n') 为所有完全满足标注的目标的数目. 精确率计算公式 (9) 表示预测出的准确目标数占总的正类预测数的比例; 召回率计算公式 (10) 表示预测出的准确目标数占实际目标数的比例, 以交并比大于阈值 0.5 为真目标.

3.2.1 合适的 anchor 个数 K 对网络的影响

使用原始 YOLOv3 算法和线性缩放 K-means 聚类算法进行实验, 验证聚类中心个数对网络的影响. 实验选取 3 个不同的聚类中心个数, 分别为 9, 12, 15. 单纯比较聚类算法的准确率在不平衡的样本中并不准确, 所以使用自建数据集 131 和 138 进行模型的训练与测试, 表 2 为不同 anchor 个数 K 下网络的性能对比. 选择更多的 anchor 数来增加检测的密度, 有利于召回率的提升, 在 anchor 个数从 9 增加为 12, 15 的过程中, 单类别目标的平均精确率从 0.96 下降至 0.95, 0.88, 平均精确率下降明显, 所以一味增加 anchor 的个数并不可行.

表 2 不同 anchor 个数 K 下模型的性能

Table 2 Performance of the model under different anchor numbers K

K	mAP	AP	Recall
9	0.77	0.96	0.76
12	0.77	0.95	0.78
15	0.77	0.88	0.78

3.2.2 验证改进的聚类算法的优异性

分别采用原始 YOLOv3 和改进聚类方式的 YOLOv3 对自建数据集 131 和 138 中的目标进行单类别检测, 结果如表 3 所示. 可见, 使用线性缩放的 YOLOv3 算法对小目标检测的平均精确率为 0.96, 与未使用聚类的 YOLOv3 算法相比提高了 3 个百分点, 与使用了聚类的 YOLOv3 算法相比提高了 5 个百分点, 且召回率均有一定的提升.

表 3　聚类方式不同的YOLOv3算法目标检测结果对比

Table 3　Comparison of target detection results of YOLOv3 algorithm with different clustering methods

聚类方式	mAP	AP	Recall
—	0.77	0.93	0.774
K-means	0.77	0.91	0.768
线性缩放 K-means	0.77	0.96	0.775

3.2.3　验证改进网络连接结构对小目标检测的优异性

分析了改进的YOLOv3网络和原始YOLOv3网络对自建小目标数据集131的检测效果, 从表4中可以看出, 改进的YOLOv3算法在自建小目标数据集131上的各项性能有明显的提升. 与原始YOLOv3算法相比, 改进的YOLOv3算法的均值平均精确率(mAP)提升了7个百分点, 召回率(Recall)提升了2个百分点, 能够有效检测出场景中的小目标, 并且在一定程度上避免了对小目标的错检漏检.

表 4　不同YOLOv3算法目标检测结果的对比

Table 4　Comparison of target detection results of different YOLOv3 algorithms

检测算法	mAP	AP	Recall
YOLOv3	0.74	0.65	0.84
改进的YOLOv3	0.81	0.82	0.86

3.2.4　验证改进YOLOv3算法的综合性能

使用规模更大的新一批次的自建数据集, 其中包含4类目标(雷、管线、基阵、潜标), 进行模型综合性能的分析. 表5是对改进YOLOv3的综合性能进行分析, 其中新增的APmine为雷的平均精确率, APpipeline为管线平均精确率, AParray为基阵平均精确率, APbuoy为潜标平均精确率, Params为模型总参数量. 从表中可以看出, 对于小目标(mine), 改进YOLOv3算法的平均精确率为0.95(提升了9个百分点); 对于管线, 改进YOLOv3算法的平均精确率为0.94; 对于基阵, 改进YOLOv3算法的平均精确率为0.98; 对于潜标, 改进YOLOv3算法的平均精确率为1; 相比原始YOLOv3算法, 改进YOLOv3算法的均值平均精确率为0.97(提升了2个百分点), 召回率为0.98(提升了2个百分点). 并且, 改进YOLOv3模型的参数量为62.6×10^{12}, 能够保持原有算法的实时性.

表 5　改进YOLOv3算法的综合性能分析

Table 5　Comprehensive performance analysis of improved YOLOv3 algorithm

检测算法	mAP	APmine	APpipeline	AParray	APbuoy	Recall	Params
YOLOv3	0.95	0.86	0.95	0.98	1.0	0.96	61.5×10^{12}
改进的YOLOv3	0.97	0.95	0.94	0.98	1.0	0.98	62.6×10^{12}

4　结 束 语

针对水下声纳小目标检测问题, 本工作提出一种改进的YOLOv3算法, 首先使用线性缩放

的 K-means 算法代替传统 K-means 算法对先验框尺度进行聚类, 接着简单改动 YOLOv3 算法的多尺度连接方式, 从而获得了一种快速、准确的适合水下声纳小目标的检测算法. 应用改进 YOLOv3 算法对自建数据集进行目标分类与识别, 验证了基于深度学习进行水下目标检测任务的可行性. 实验结果表明, 本工作所提出的改进 YOLOv3 算法能够实现对小目标的有效检测, 且针对所有类别的检测精度均达到 90% 以上, 召回率高达 98%, 相比原始 YOLOv3 算法在检测小目标上性能明显提升.

参考文献:

[1] Shierin B M, Supriya M H. SOS based selection and parameter optimization for underwater target classification [C]//IEEE OCEANS 2016 MTS. 2016: 1-4.

[2] Ren S, He K, Girshick R, et al. Faster R-CNN: towards real-time object detection with region proposal networks [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(6): 1137-1149.

[3] Redmon J, Divvala S, Girshick R, et al. You only look once: unified, real-time object detection [C]// IEEE Conference on Computer Vision and Pattern Recognition. 2016.

[4] Liu W, Anguelov D, Erhan D, et al. SSD: single shot multibox detector [C]// European Conference on Computer Vision. 2016.

[5] Redmon J, Farhadi A. YOLOv3: an incremental improvement [C]// International Conference on Computer Vision and Pattern Recognition. 2017.

[6] Li Z, Xu X, Shen F, et al. CR-FPN: channel relation feature pyramid network for object detection [J]. Wireless Networks, 2020, DOI: 10.1007/s11276-020-02391-3.

[7] Li J L, Luo S J, Zhu Z Q, et al. 3D IoU-Net: IoU guided 3D object detector for point clouds [EB/OL]. (2020-04-10)[2020-09-20]. https://arxiv.org/abs/2004.04962.

[8] Li C L, Ravanbakhsh S, Poczos B. Annealing Gaussian into ReLU: a new sampling strategy for Leaky-ReLU RBM [EB/OL]. (2016-11-11)[2020-09-20]. https://arxiv.org/abs/1611.03879.

[9] Xu L, Choy C S, Li Y W. Deep sparse rectifier neural networks for speech denoising [C]// 2016 IEEE International Workshop on Acoustic Signal Enhancement. 2016.

[10] Rathi D, Jain S, Indu S. Underwater fish species classification using convolutional neural network and deep learning [C]// 2017 Ninth International Conference on Advances in Pattern Recognition (ICAPR). 2017.

[11] Smith L N. Cyclical learning rates for training neural networks [C]//2017 IEEE Winter Conference on Applications of Computer Vision. 2017.

[12] Zhang F. Modular configuration of service elements based on the improved K-means algorithm [J]. Expert Systems, 2019, 36(5): e12344.

[13] Szegedy C, Ioffe S, Vanhoucke V, et al. Inception-v4, Inception-ResNet and the impact of residual connections on learning [C]//International Conference on Learning Representations. 2016.

[14] Santurkar S, Tsipras D, Ilyas A, et al. How does batch normalization help optimization? [C]//32nd Conference on Neural Information Processing Systems. 2018.

[15] 宋刚, 杜宏伟, 王平, 等. 纹理细节保持的图像插值算法 [J]. 计算机科学, 2019(增刊 1): 169-176.
Song G, Du H W, Wang P, et al. Image interpolation algorithm for texture detail preservation [J]. Computer Science, 2019(Suppl.1): 169-176.

[16] 秦玺淳. 数字图像处理技术 [J]. 数字通信世界, 2017(12): 174, 207.
Qin X C. Digital image processing technology [J]. Digital Communication World, 2017(12): 174, 207.

[17] 赵永强, 饶元, 董世鹏, 等. 深度学习目标检测方法综述 [J]. 中国图象图形学报, 2020, 25(4): 5-30.
Zhao Y Q, Rao Y, Dong S P, et al. Overview of deep learning target detection methods [J]. Journal of Image and Graphics, 2020, 25(4): 5-30.