分类号：TP391　　密　　级：公开

U D C：　　单位代码：10142

沈阳工业大学

硕士学位论文

基于深度学习的道路交通标志识别算法研究

学　　号：202121033

作　　者：胡宝江

专业学位类别：电子信息

领　　域：控制工程

论 文 类 型：工程设计与研究

2024 年 9 月 2 日

沈阳工业大学硕士学位论文

基于深度学习的道路交通标志识别算法研究

Road Traffic Sign Recognition Algorithms through Deep Learning

作　　者： 胡宝江　**单位：** 沈阳工业大学

指 导 教 师： 陈丽 副教授　**单位：** 沈阳工业大学

协助指导教师： 李庆鑫 副研究员　**单位：** 中国科学院沈阳自动化研究所

论文答辩日期：2024 年 **8** 月 **30** 日

学位授予单位：沈 阳 工 业 大 学

摘要

随着自动驾驶技术的快速发展，对道路交通标志的准确识别成为保障行车安全的关键技术之一。近年来，人工智能，尤其是深度学习技术的蓬勃发展对交通标志识别研究带来巨大推动，以卷积神经网络为主的深度学习方法已经在交通标志检测、目标识别等问题中展现出巨大潜力，然而现有方法受限于复杂的背景和光照条件变化，难以达到理想的识别效果。针对交通标志识别精度低和速度慢的问题，本文以交通标志为研究对象，设计了一种改进的 YOLOv8（You Only Look Once version 8）模型，在数据集构建、模型改进设计、轻量化剪枝及嵌入式部署测试等方面进行了研究，主要工作内容如下：

针对交通标志数据集数据量少且各类别数据不均衡导致模型识别精度低的问题，基于改进深度卷积对抗网络（Deep Convolutional Generative Adversarial Networks, DCGAN）网络扩充交通标志数据集，并通过几何变换、颜色变换、噪声添加等数据增强方法进行后处理，为交通标志识别模型训练提供数据基础。

针对复杂场景下小且模糊的交通标志目标识别精度低的问题，基于改进 YOLOv8 网络设计了交通标志识别模型。首先为提高交通标志识别模型的泛化能力改进缩放指数型线性单元激活函数（Scaled Exponential Linear Unit, SELU），其次为加强交通标志特征融合与多尺度目标的检测识别引入渐近特征金字塔网络，最后为提升网络的识别精度构建双级路由注意力机制，对识别结果贡献率高的区域进行特征提取。实验结果表明，在保持实时性的基础上改进后的模型识别平均精度得到了提升，同时相较于主流的交通标志识别算法也有优势。

针对交通标志识别模型速度慢且对嵌入式平台算力要求高的问题，基于 Smooth-L1 正则化函数建立稀疏训练模型，对模型进行轻量化剪枝以减少模型体积，将剪枝后的最终改进模型部署在 RK3588 核心板测试。实验结果表明，在精度基本保持不变的情况下模型参数量降低了 66.51%，每秒帧数提高了 43.8%，达到 27.9 帧/s，满足实时性要求。

关键词：交通标志识别，深度学习，YOLOv8 模型，模型剪枝

Abstract

In the realm of autonomous driving, the precise recognition of road traffic signs is paramount for ensuring safety. In recent years, the booming development of artificial intelligence, especially deep learning technology, has brought tremendous impetus to research on traffic sign recognition. Deep learning methods mainly based on convolutional neural networks have shown great potential in traffic sign detection, target recognition, and other problems. However, existing methods are limited by complex background and lighting conditions, making it difficult to achieve ideal recognition results. In response to the problems of low accuracy and slow speed in traffic sign recognition, this thesis takes traffic signs as the research object and designs an improved YOLOv8 model. This thesis focuses on dataset construction, model improvement design, lightweight pruning, and embedded deployment testing. The main work contents are as follows:

Addressing the issue of low model recognition accuracy due to the small size and imbalanced data distribution in traffic sign datasets, an enhanced deep convolutional generative adversarial networks, deep convolutional generative adversarial networks is employed to augment the traffic sign dataset. Subsequently, post-processing techniques such as geometric transformations, color transformations, and noise addition are applied to provide a data foundation for training traffic sign recognition models.

Addressing the issue of low recognition accuracy for small and blurry traffic sign targets in complex scenarios, a traffic sign recognition model has been designed based on an improved YOLOv8 model. Firstly, the scaled exponential linear unit activation function is modified to enhance the generalization capability of the traffic sign recognition model. Secondly, the asymptotic feature pyramid network is improved to strengthen the fusion of traffic sign features and the detection of multi-scale targets. Then, bi-level routing attention is constructed to extract features from regions with high contribution rates to the recognition results. The experimental results show that the improved model achieves an increase in mAP while maintaining real-time performance, demonstrating its advantage over mainstream traffic sign recognition algorithms.

Addressing the issues of slow speed and high computational requirements for embedded platforms in traffic sign recognition models, a sparse training model is established based on

the Smooth-L1 regularization function. The model is then lightened and pruned to reduce its size, and the final improved pruned model is deployed on an RK3588 core board for testing. Experimental results indicate that, with minimal changes in accuracy, the model's parameter count is reduced by 66.51%, and the frames per second are increased by 43.8% to 27.9 frames/s, meeting real-time requirements.

Key Words: Traffic sign recognition, Deep learning, YOLOv8 model, Model pruning

目　录

第 1 章 绪论

1.1 研究背景与意义

在 21 世纪，伴随着计算机视觉、神经网络、人工智能及机器学习等诸多领域的迅猛发展，自动驾驶技术已经取得了引人注目的成就。这些领域的技术突破为自动驾驶汽车的研发与实际应用提供了坚实的技术基础[1]。

随着消费者对出行安全、舒适和便捷性的需求不断提升，自动驾驶汽车作为一种新型的出行方式，正逐渐成为市场关注的焦点。特别是在公共交通、物流配送、出租车等领域，自动驾驶汽车具有广阔的应用前景。各国政府纷纷出台政策支持自动驾驶技术的发展，包括制定相关法规、建立测试场地、提供资金支持等。

随着自动驾驶技术的逐步成熟和市场需求的增长，交通标志识别作为自动驾驶系统的重要组成部分，其研究与应用也愈发受到关注。道路交通环境的复杂性和多变性对交通标志识别技术提出了更高的要求与挑战。交通标志识别技术需要能够应对各种天气条件、路况变化以及交通标志的多样性，确保驾驶的安全与稳定。要实现这一目标，交通标志识别技术必须克服多项技术挑战[2]。

首先，由于交通环境的复杂性，交通标志可能在多种不同的条件下出现。例如，它们可能在不同的视角和光照条件下被拍摄，这就要求识别算法必须具备高度的适应性和鲁棒性。其次，交通标志通常位于复杂的背景中，如树木、建筑物、车辆等，这些都可能干扰标志的识别。因此，算法需要具备强大的抗干扰能力，以准确地区分出标志与背景。同时，考虑到驾驶安全的重要性，交通标志识别系统对实时性的要求也非常高。这意味着算法不仅要准确，还要快速，以便在行驶过程中及时提供必要的信息。最后，由于交通标志的尺寸可能很小，特别是在远距离或图像质量不佳的情况下，识别算法需要具备处理这些细节模糊的能力，以确保识别的准确性和可靠性。

综上所述，为确保车辆在道路上行驶的安全性，设计出一套低资源占用且高识别准确率的交通标志识别模型对于保障驾驶者和行人的安全具有重要意义。

1.2 交通标志识别国内外研究现状

传统目标识别算法包含图像预处理、区域选择、候选区域特征提取和分类等环节，虽然表现出较高的识别准确度，但由于特征设计需要依赖于研究人员的经验和对应用领域的了解，因此传统算法在复杂环境的交通标志识别领域中表现出泛化能力差、准确率低等问题[3]。

近年来，深度学习在交通标志识别领域取得了显著进展，其算法逐渐展现出卓越的性能[4]，它们在鲁棒性和泛化能力上明显优于传统方式，能够自主完成目标特征识别，无需人为设计特征，对研究人员的主观经验依赖较小。

目前主流的目标检测算法可以归结为两大类。一是单阶段检测算法，比如 YOLO（You Only Look Once）和单次多盒探测器（Single Shot MultiBox Detector, SSD）系列，它们的特点是一步到位，直接从图像中识别并定位目标[5]。另一类则是二阶段检测算法，区域卷积神经网络（Region-Convolutional Neural Networks, R-CNN）系列就是其代表，这类算法会先进行一次区域提议，然后再进行细致的目标检测和分类。这两类算法各有优势，单阶段算法通常速度更快，而二阶段算法在精确度上往往更胜一筹。

单阶段的目标检测算法将目标检测问题转化为一个回归问题，可以直接将待检图片输入网络，输出层可以得到待检图片中目标的边界框和类别信息[6]。虽然这类算法的检测精度相较于基于候选区域的算法有所降低，但它们在实时性方面能满足自动驾驶领域的检测速度要求。虽然基于候选区域的二阶段目标检测算法在精度上表现卓越，但其在实时性方面的表现却无法满足自动驾驶领域的需求[7]。相对而言，单阶段目标检测算法不仅能在保持一定的检测精度的同时，还能确保实时性要求得到满足，因此更适合应用于自动驾驶领域的检测任务[8]。

交通标志识别主要包括两个主要部分：一是通过从实际交通场景中检测出交通标志；二是进一步对检测出的交通标志进行识别和分类[9]。自 20 世纪 70 年代以来，交通标志识别领域一直备受研究者关注。其中，Smith[10]的研究为该领域奠定了重要基础，深入探讨了交通标志识别的技术难点与挑战。随后的几十年中，该技术取得了显著的进步。

在交通标志识别的技术演进中，多种方法被陆续提出并不断优化。例如，2010 年梁文昭等人[11]通过集成反向传播神经网络，实现了对 11 种交通标志的准确识别，展现了小波矩特征在仿射不变性方面的优势。随后的研究中，神经网络、深度学习等方法被广泛应用。杨守建等人[12]在 2011 年利用离散型霍普菲尔德神经网络提升了交通标志在各种条件下的识别率。到 2012 年，Smirnov 等人[13]发现深度卷积神经网络在图像分类中的出色表现，并对比了不同正则化技术的性能。

进入 2010 年代中后期，更多的研究聚焦于提升交通标志识别的准确性和鲁棒性。蔡自兴等人[14]于 2013 年提出了一种结合形状特征与双树复小波变换的识别系统，显著提高了识别的准确性。随后的几年中，研究者们不断探索神经网络结构的优化，如 Karis 等人[15]在 2014 年提出的有监督前馈-反向传播神经网络，以及 Aghdam 等人[16]在 2015

年提出的最优卷积神经网络。

近年来，随着深度学习技术的不断发展，基于卷积神经网络的交通标志识别方法逐渐成为主流。例如，在 2016 年，钟晓明等人[17]创新性地提出了一种交通标志识别方法，该方法以快速区域卷积神经网络（Faster Region-based Convolutional Neural Networks, Faster R-CNN）为基础，有效提升了在复杂背景下交通标志识别的准确率。随后的研究中，Ren 等人[18]引入了区域生成网络，张邯等人[19]通过改进深度卷积神经网络结构提升了识别精度，而 Kong 等人[20]则提出了一种轻型的级联卷积神经网络算法用于交通标志识别。

YOLO 模型，由华盛顿大学的 Joseph Redmon 和 Ali Farhadi 共同研发，是一种广泛应用于物体检测和图像分割的模型。自 2015 年问世以来，YOLO 模型便以其卓越的处理速度和精准的检测结果而备受瞩目[21]。2016 年发布的 YOLOv2 通过纳入批量归一化、锚框和维度集群改进了原始模型。

2018 年推出的 YOLOv3 使用更高效的骨干网络、多锚和空间金字塔池进一步增强了模型的性能[22]。YOLOv3 采用了逻辑函数（Sigmoid）来替代传统的归一化指数函数（Softmax）激活函数，以此构建多标签分类器，同时汲取了残差网络的精髓，构建了更深层次的 Darknet-53 网络结构，进而显著提高了模型的检测精度[23]。此外，为了有效应对小目标检测的难题，YOLOv3 还巧妙地引入了金字塔网络，实现了多尺度的预测功能。

在 2020 年，YOLOv4 应运而生，由于 YOLOv1-v3 的原作者停止了 YOLO 框架的更新工作，Alexey Bochkovskiy 便肩负起了继续发展和优化 YOLO 的使命。与前代版本相比，YOLOv4 不仅融合了目标检测领域的多种有效技巧和即插即用模块，还创新性地引入了 Mosaic 数据增强技术、新型无锚检测头以及新的损失函数，从而显著提升了模型的性能[24]。

YOLOv5 进一步提高了模型的性能，并增加了超参数优化、集成实验跟踪和自动导出为常用导出格式等新功能。相比较于 YOLOv4，YOLOv5 在模型结构上进行了优化，使用更小的模型减少了计算资源和存储空间的消耗，同时保证了高精度的检测性能[25]。这使得 YOLOv5 在运行速度上比 YOLOv4 更快，可以处理更高分辨率的图像。此外，YOLOv5 还通过优化网络设计和使用更高效的模型设计，实现了更快的推理速度。YOLOv5 在边界框的预测上使用了广义交并比损失函数（Generalized Intersection over Union Loss, GIOU_Loss）以及加权非极大值抑制等新的方法，这些方法可以更好地处理目标检测中的边界框回归和重叠目标检测的问题，提高检测精度。虽然 YOLOv5

在很多方面都表现出色，但在处理小物体检测、光照和角度变化等方面仍有一定的挑战。YOLOv6 于 2022 年由美团推出并开源，目前已用于该公司的许多自主配送机器人。YOLOv7 不仅扩展了其应用范围，譬如涵盖了诸如 COCO 关键点数据集上的姿势估计等额外任务，还创新性地提出了一种针对辅助头的训练方法[26]。该方法的核心在于，由于 YOLOv7 的辅助头仅仅只在训练阶段被使用，因此通过适度提升训练阶段的计算成本，可以显著提升模型的检测精度，同时确保在推理阶段不增加任何时间开销。

得益于 YOLO 模型的飞速发展，最近的几年里，针对 YOLO 系列算法的改进成了一个新的研究热点。张传伟等人[27]在 2020 年针对 YOLOv2 算法在小尺寸交通标志检测上的不足提出了改进方法。随后，徐迎春[28]在 2021 年通过修改参考框尺寸和增强特征提取能力等方式改进了 YOLOv3 算法。到了 2022 年，周钰如等人[29]提出了基于 YOLOv5 算法的交通标志识别系统，进一步提升了检测的精度和速度。最近，彭瑾等人[30]在 2023 年通过引入注意力机制和跨层连接结构等方式，对 YOLOv5s 网络模型进行了改进，进一步提高了交通标志识别的精度。

以上研究论证了通过目标检测识别算法在自动驾驶领域中识别道路交通标志的可行性，但仍存在一些问题。目前的单阶段交通标志识别算法的精度尚未达到预期，对于背景复杂、小面积交通标志以及交通标志残缺、极端天气影响等情况的处理能力较弱，影响了整体的识别精度与速度。而且现有的交通标志数据集存在实例分布不均匀的问题，导致算法泛化能力差，对于实例少的类别识别不准确。因此，自动驾驶系统领域中提高交通标志识别的准确性和鲁棒性的交通标志识别算法研究具有重要意义。

1.3 研究内容和论文结构

综上所述，现有的道路交通标志识别算法存在精度低、实时性弱等问题，导致识别效果不佳。因此，本文以道路交通标志作为研究对象，提出一种基于改进 YOLOv8 模型的交通标志识别算法，课题主要研究并分析了以下几个方面问题：

（1）交通标志数据匮乏

目前开源的交通标志数据集数据量不足并且各类别实例数不均衡，因此基于改进深度卷积生成对抗网络（Deep Convolutional Generative Adversarial Networks, DCGAN）扩增数据集，且对扩增后的实例通过多种数据增强方式进行后处理，为交通标志识别模型提供更好的数据基础。

（2）交通标志图像识别精度低

复杂的交通标志形状、多变的光照条件以及恶劣天气状况下的交通标志识别精度

低，因此设计交通标志识别模型 YOLOv8-improve，该模型设计了激活函数、渐近特征金字塔网络以及双级路由注意力机制来提高模型识别精度并降低参数量以及计算量。

（3）识别模型实时性要求及嵌入式实现

在实际道路场景中设备无法满足要求比较高的计算能力，难以在识别速度较快的情况下保持高检测精度，因此需要对模型进行轻量化剪枝，降低模型的体积。

针对以上几方面问题，本文设计了交通标志识别算法，并通过实验进行数据分析，验证提出策略的有效性。首先对当下自动驾驶技术的重要性进行说明，进而引出交通标志识别的研究背景与意义、国内外研究现状，并介绍了经典的目标检测与识别算法。针对 YOLOv8 在交通标志识别领域内存在的精度不佳、实时性不佳等问题进行改进，整体框架如图 1.1 所示：

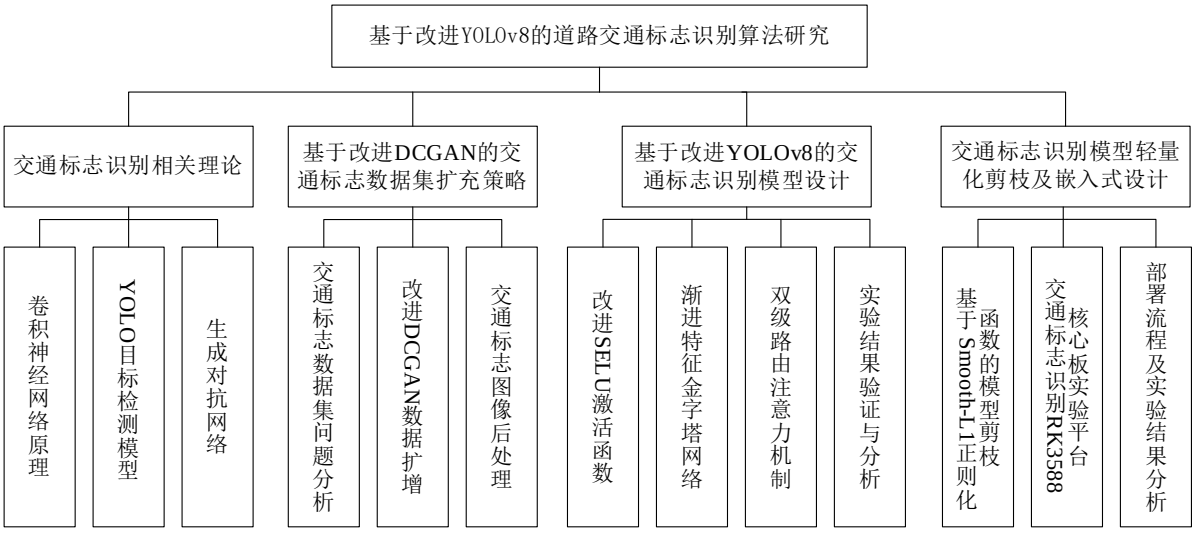


图 1.1 论文框架

Fig. 1.1 Thesis framework

第 1 章绪论。主要叙述了目前发展背景下交通标志识别技术的研究背景与意义，介绍了交通标志识别技术的发展历程与国内外学者的当前的研究进展以及交通标志识别领域内存在的难题，提出了主要研究内容并论述工作。

第 2 章交通标志识别相关理论。本章首先介绍道路场景下的交通标志识别流程以及用于设计交通标志识别系统的卷积神经网络和 YOLO 系列算法的基础知识，然后介绍了生成对抗网络原理，为后面章节中针对目前交通标志识别模型存在问题的改进策略提供了理论依据。

第 3 章基于改进 DCGAN 的交通标志数据集扩充策略。设计了一种基于改进

DCGAN 的交通标志数据集扩充策略，并通过多种数据增强方式等对图像进一步处理，增加数据量并解决实例不均的问题，为训练交通标志识别模型提供数据基础。

第 4 章基于改进 YOLOv8 的交通标志识别模型设计。针对现有交通标志识别模型精度低的问题设计基于改进 YOLOv8 的交通标志识别模型。该模型首先对 YOLOv8 的激活函数进行改进，同时为模型设计渐近特征金字塔网络，最后构建双级路由注意力机制实现对交通标志的准确识别，并设计对比实验验证模型性能。

第 5 章交通标志识别模型轻量化剪枝及嵌入式部署。针对交通标志识别模型速度差的问题，基于 Smooth-L1 正则化函数设计模型轻量化剪枝方法减轻模型体积，将训练后的模型剪枝并进行嵌入式部署实验测试，结果表明识别效果和识别速度满足实际需求，验证了改进策略的有效性。

第 6 章结论。对研究内容进行概括性总结，讨论了算法的设计与改进策略，并对下一步研究方向进行了展望。

第 2 章 交通标志识别相关理论

本章内容首先介绍了道路场景下的交通标志识别流程以及卷积神经网络的组成及其相关基础理论，其次阐述了 YOLO 模型的思想和模型架构，最后介绍了生成对抗网络原理，为后续交通标志识别数据集的建立及模型针对识别精度低及速度差的问题做出的改进设计提供理论依据。

2.1 道路场景下的交通标志

交通标志是用文字或符号传递引导、限制、警告或指示信息的道路设施，又称道路标志、道路交通标志。交通标志一般具有安全、醒目、清晰、明亮的特点，是实施交通管理，保证道路交通安全、顺畅的重要措施。通过设立不同的交通标志，可以对交通流量进行调节，有效疏导交通，提高道路通行能力。同时，交通标志还能提醒驾驶人和行人注意危险，并传递道路方向、地点和距离等信息，为导航服务提供重要支持。

交通标志的范例设计均遵循特定的形状模板（如圆形、正三角形等）以及标准的色彩规范（如黄色、红色、蓝色）。然而，在实际应用中，交通标志的检测与识别系统所面临的挑战远不止于此。系统采集的图像通常来源于复杂的室外交通场景，这些图像因多种因素而呈现多样化特点。首先，车辆的颠簸和高速行驶可能导致图像产生运动模糊；其次，光照条件的变化直接影响图像的清晰度与质量。再者，随着车辆的行进，图像采集设备与交通标志之间的角度不断变化，可能导致图像中的交通标志形状发生畸变。此外，交通标志可能会因外部因素如破坏、残缺或污损而难以辨认；同时，当多个交通标志位置接近时，它们在图像中可能相互遮挡或重叠，进一步增加了识别和检测的难度。因车辆快速行驶导致识别模糊的道路交通标志图如图 2.1a 所示，光照条件较差导致亮度低的道路交通标志图如图 2.1b 所示。

a 模糊交通标志　　b 弱光条件下的交通标志

图 2.1 复杂环境下的道路交通标志

Fig. 2.1 Road traffic signs in complex environments

目前主流的基于深度学习的交通标志识别过程如图 2.2 所示，首先通过摄像头采集交通标志图像信息，然后通过神经网络提取感兴趣区域，检测交通标志区域，之后提取交通标志特征信息，最后分类识别交通标志。

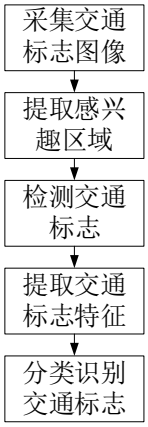


图 2.2 交通标志识别过程

Fig. 2.2 Traffic sign recognition process

目前主流用于交通标志模型训练的中国交通标志检测数据集 2021（China Comprehensive Traffic Sign Detection Benchmark 2021, CCTSDB2021）是由长沙理工大学综合交通运输大数据智能处理湖南省重点实验室的张建明教授团队制作而成。CCTSDB2021 数据集在之前发布的 CCTSDB2017 的基础上，新增了 4000 多张真实交通场景图像及相应标注，并替换了许多原本易于检测的图像以更好地适应复杂多变的检测环境。在结构上，CCTSDB2021 数据集中的训练集和正样本测试集总共有 17856 幅图像，其中训练集图像有 16356 张和正样本测试集图像有 1500 张。该数据集中训练集与测试集的图片数量比例近似为 10:1，部分包含交通标志的训练图片如图 2.3 所示。

图 2.3 CCTSDB 示例图

Fig. 2.3 CCTSDB example diagram

2.2 卷积神经网络原理

目前主流的交通标志识别方法是基于卷积神经网络（Convolutional Neural Networks, CNN），其凭借卷积层、池化层及全连接层等核心组件的协同工作，能够高效执行分类任务，尤其擅长处理大规模的图像数据。CNN 具备自动提取图像特征的能力，这些特征涵盖了形状、颜色以及纹理等多个方面，对于精确识别交通标志而言至关重要。即便面临不同的光照条件、天气状况及拍摄角度，CNN 依然能够保持出色的识别性能。因此，在研究交通标志识别时，深入了解和充分利用 CNN 的相关知识显得尤为重要，有助于提高识别的准确性并提升效率[31]。

CNN 具有很强的适应性，可以处理各种复杂的图像任务。在图像识别、目标检测等领域，CNN 表现卓越，已经广泛应用于自动驾驶等场景，成为人工智能领域的重要技术之一[32]。通过 CNN 能够识别图像中的各种细节，包括颜色、形状、纹理等，并能够将复杂的图像信息转化为计算机可以理解的形式，进而进行智能分析和决策。

2.2.1 卷积神经网络的基本架构

卷积神经网络是一种特殊的神经网络，专为处理具有网格结构的数据设计，如图像数据。其独特之处在于能够利用输入数据的局部相关性，通过卷积操作提取出多层次的特征表示[33]，卷积神经网络结构图如图 2.4 所示。

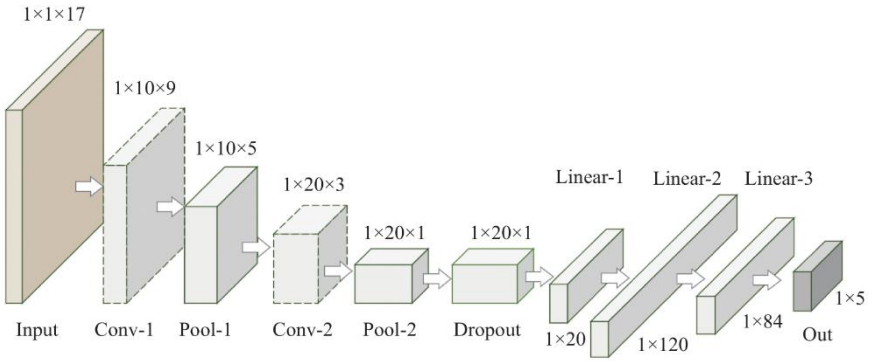


图 2.4 卷积神经网络结构图

Fig. 2.4 CNN structure diagram

卷积神经网络的起始是输入层（Input），它接收原始数据，如图像、音频等[34]。在图像处理中，输入层代表图片像素矩阵，在 RGB 色彩模式下，图像的深度为 3，分别对应红、绿、蓝三个颜色通道。

卷积层（Conv）是 CNN 的核心，负责从输入数据中提取特征。卷积层中的每个神经元都只与输入数据的一个局部区域相连，这种局部连接的方式能够有效减少网络参

数，提高计算效率，卷积层的结构如图 2.5 所示。

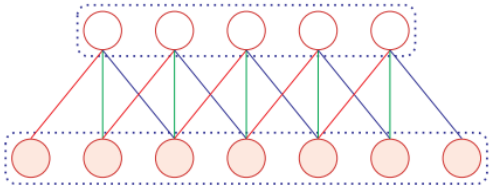


图 2.5 卷积层

Fig. 2.5 Convolutional layer

卷积层通过一组可学习的滤波器（或称为卷积核）对输入数据进行卷积操作。每个滤波器在输入数据上滑动，进行点积运算，从而生成一个二维的激活图。这个激活图反映了滤波器对输入数据的响应，即提取到的特征。不同的滤波器可以提取到不同的特征，如边缘、纹理等。

卷积层的输出称为特征图，其通道数与使用的滤波器数量相同。每个通道代表了一种特定的特征，这些特征在后续层中会被进一步组合和抽象。

在卷积层之后，通常会引入激活函数，如线性整流函数（Rectified Linear Unit, ReLU）函数或 Sigmoid 函数[35]。激活函数的作用是为网络引入非线性因素，使得网络能够拟合更复杂的模式[36]。

卷积层的结构包含输入特征映射组、输出特征映射组以及卷积核[37]。

输入特征映射组以三维张量 $x \in R^{M \times N \times D}$ 的形式存在，每一个切片矩阵 $X^d \in R^{M \times N}$ 代表一个输入特征映射。此三维张量的标准形态是（高度 M，宽度 N，深度 D），D 在此处指的是输入特征映射的数量，或者说通道数。例如，在处理彩色图像时，常见到 D 等于 3，这恰好对应于红、绿、蓝这三个颜色通道。

输出特征映射组以三维张量 $y \in R^{M' \times N' \times P}$ 的形式存在，每一个切片矩阵 $Y^p \in R^{M' \times N'}$ 代表一个输出特征映射。这些输出特征映射的生成，依赖于卷积核对输入特征映射进行的卷积操作。输出特征映射组的一般形态为（高度 M'，宽度 N'，深度 P），其中的 P 代表了输出特征映射的数量或通道数，其数值通常由卷积核的数量决定。

卷积核以四维张量 $W \in R^{U \times V \times P \times D}$ 的形式存在，每一个切片矩阵 $W^{p,d} \in R^{U \times V}$ 中的高度 U 和宽度 V 定义了二维卷积核的大小[38]。每一个二维切片实质上就是一个卷积滤波器，负责在输入特征映射上进行卷积运算，以生成相应的输出特征映射。

输出映射 Y^p 是由卷积核 $W^{p,1}, W^{p,2}, \cdots, W^{p,D}$ 对输入特征映 $X^1, X^2, \cdots, X^D$ 应用卷积核进行卷积操作，随后，将这些卷积结果进行求和，并与一个标量偏置 b^p 相加从而得

到卷积层的净输入 Z^p，最终，通过非线性激活函数 $f(x)$ 的处理得到输出特征映射 Y^p：

$$Z^p = W^p \otimes X + b^p = \sum_{d=1}^{D} W^{p,d} \otimes X^d + b^p \tag{2.1}$$

$$Y^p = f(Z^p) \tag{2.2}$$

式（2.1）及式（2.2）中，$W^p \in R^{U \times V \times D}$ 表示三维卷积核，$f(x)$ 表示非线性激活函数。

池化层（Pool）在卷积层之后使用，其功能是对特征图实施下采样，以达到缩减数据空间维度、减轻计算负担的目的，同时确保关键特征信息的留存。两种广泛采用的池化方法是最大池化和平均池化。在最大池化中，输出为各个池化区域内的最大值；而平均池化，则是通过算出池化区域内所有数值的平均值来获得输出。

最大池化是指对于一个区域 $R_{m,n}^d$，选取该区域内全部神经元的最大活跃值代表整体表现，如式（2.3）所示：

$$y_{m,n}^d = \max_{i \in R_{m,n}^d} x_i \tag{2.3}$$

式（2.3）中，x_i 代表区域 R_k^d 内每个神经元的活跃值。

平均池化在执行时，网络会在输入的特征图上滑动一个固定大小的窗口，并计算窗口内所有元素的平均值，然后用这个平均值作为输出特征图的一个元素，这个过程会在整个输入特征图上重复进行，从而得到一个新的、尺寸更小的输出特征图，如式（2.4）所示：

$$y_{m,n}^d = \frac{1}{|R_{m,n}^d|} \sum_{i \in R_{m,n}^d} x_i \tag{2.4}$$

针对每个输入特征映射 X^d 的 $M' \times N'$ 个区域实施子采样操作，从而获得池化层相对应的输出映射，如式（2.5）所示：

$$Y^d = \{y_{m,n}^d\}, 1 \le m \le M', 1 \le n \le N' \tag{2.5}$$

卷积层和池化层之后，通常会引入全连接层（Linear）。这一层之所以被称为“全连接”，是因为其每一个输入节点都与输出节点通过权重相连。全连接层的核心功能在于，它能将前置层（例如卷积层、池化层）所提取的特征进行融合，进而形成更为高级的特征表达，为后续的分类或回归任务提供便利。

输出层（Out）是 CNN 的最后一层，根据任务的不同，输出层可能采用不同的形式。在分类任务中，输出层通常使用 Softmax 函数，将全连接层的输出转化为类别概率分布。在回归任务中，输出层则可能直接输出预测值[39]。

卷积神经网络由多个组件构成，这些组件之间相互配合，完成了从原始数据到高

级特征的有效转换，并最终实现了分类或回归的任务目标。卷积运算是广泛应用于信号处理、图像处理及机器学习等多个领域的一种计算方法。其主要用于图像识别、语音识别以及构建神经网络等场景[40]。在执行卷积运算时，需经过两个核心步骤：第一步，将一个输入数据（如图像）与一个或多个滤波器（亦称卷积核或过滤器）进行配对；第二步，通过名为“卷积”的处理方式来加工这些已配对的数据。所谓“卷积”，即描述的是两个函数在特定维度上的加权“叠合”效应。卷积的函数定义如式（2.6）所示：

$$s(t)=\int_{-\infty}^{\infty} f(a)*g(t-a)\mathrm{d}a \tag{2.6}$$

式（2.6）中，函数 $f(a)$ 和函数 $g(a)$ 是卷积对象，a 为积分变量，星号“$*$”表示卷积，简记为：

$$s(t)=f(t)*g(t) \tag{2.7}$$

一维卷积操作在信号处理和深度学习领域被广泛应用，特别是对于一维数据序列的处理，例如时间序列分析和文本数据处理。该操作的核心在于利用一个可滑动的窗口，也被称为卷积核或滤波器，在一维数据上逐步移动[41]。在每个窗口位置，内部的数据会与卷积核进行点乘运算，并将结果求和，最终得到卷积的输出。这一过程能够有效地提取输入数据的局部特性，一维卷积过程如式（2.8）所示：

$$\begin{aligned} y_t &= 1\times x_t + 1/2\times x_{t-1} + 1/4\times x_{t-2} \\ &= w_1\times x_t + w_2\times x_{t-1} + w_3\times x_{t-2} \\ &= \sum_{k=1}^{3} w_k x_{t-k+1} \end{aligned} \tag{2.8}$$

将 $w_1, w_2, \cdots w_t$ 这些元素称为滤波器，亦称卷积核。设定滤波器的长度为 K，设定其与一个信号序列 $x_1, x_2, \cdots x_t$ 的卷积如式（2.9）所示：

$$y_t=\sum_{k=1}^{K} w_k x_{t-k+1} \tag{2.9}$$

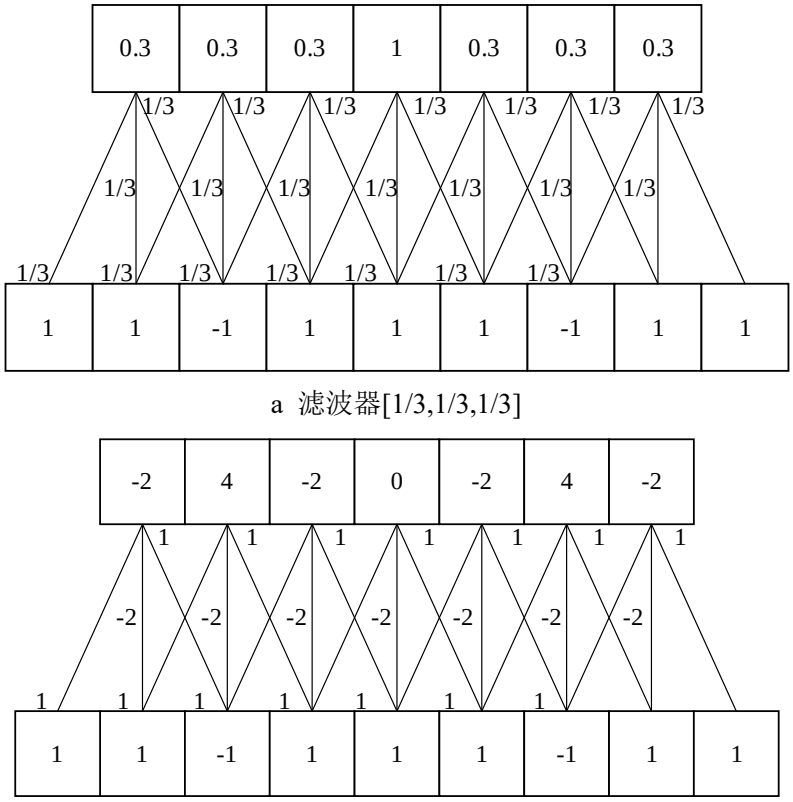
为了明确卷积的输出，设定输出 y_t 的下标 t 从 K 起始，基于此给出信号序列 x 与滤波器 w 的卷积，其定义如式（2.10）所示：

$$y=w*x \tag{2.10}$$

式（2.10）中，“$*$”表示卷积运算操作。设计各异的滤波器能够捕捉信号序列的不同特性。比如，若设定滤波 w 为$[1/K, \cdots, 1/K]$时，即等同于对信号序列执行窗口大小为 K 的简单移动平均处理。若选择滤波器 w 为$[1,-2,1]$时，则可以近似地模拟信号序列的二阶微分效应，如式（2.11）所示：

$$x''(t)=x(t+1)+x(t-1)-2x(t) \tag{2.11}$$

滤波器[1/3,1/3,1/3]以及滤波器[1,−2,1]的一维卷积示例图如图 2.6 所示。



a 滤波器[1/3,1/3,1/3]

b 滤波器[1,−2,1]

图 2.6 一维卷积示例

Fig. 2.6 Example of one-dimensional convolution

这两种滤波器各自捕捉了输入序列的不同特性。滤波器 $w=[1/3,1/3,1/3]$ 更侧重于感知信号序列中的低频成分，而滤波器 $w=[1,-2,1]$ 则对信号序列中的高频信息更为敏感，从而实现对其的检测。

与一维卷积所处理的一维数据序列不同，二维卷积处理的是二维数据，尤其是图像数据。在二维卷积过程中，有一个称为“卷积核”或“滤波器”的小矩阵，它从图像的左上角开始，作为移动窗口在整个图像上进行滑动[42]。在每个位置上，卷积核会与所覆盖的图像区域进行对应元素的乘法运算，并将这些乘积进行累加，得出该位置在输出图像中的对应值。这一过程会在整个图像范围内重复进行，最终生成一个全新的二维矩阵，即是输出图像。对于图像 $X \in R^{M\times N}$ 和滤波器 $W \in R^{U\times V}$ 来说，通常滤波器的维度 $U\times V$ 远小于图像的维度 $M\times N$，其卷积如式（2.12）所示：

$$y_{ij}=\sum_{u=1}^{U}\sum_{v=1}^{V}W_{uv}X_{i-u+1,j-v+1} \tag{2.12}$$

输入信息 X 和滤波器 W 的二维卷积定义如式（2.13）所示：

$$Y = W * X \tag{2.13}$$

式（2.13）中，星号“*”代表了二维卷积运算操作，二维卷积示例图如图 2.7 所示。

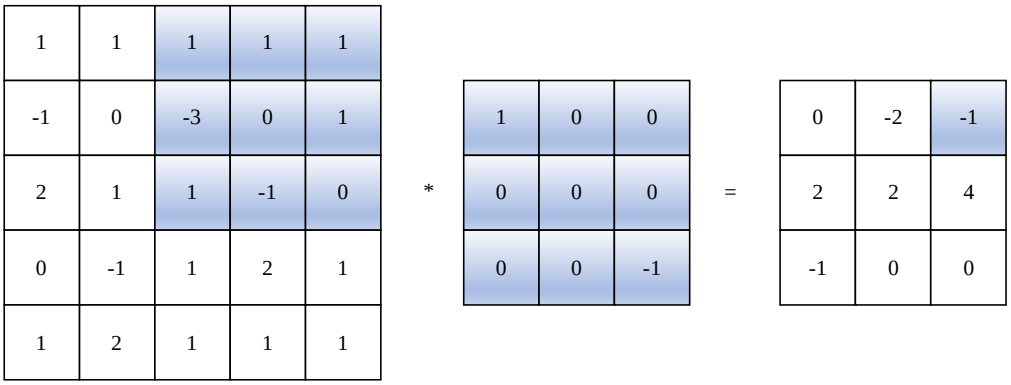


图 2.7 二维卷积示例图

Fig. 2.7 Example of 2D convolution

2.2.2 典型卷积神经网络模型

(1)VGGNet

视觉几何组网络（Visual Geometry Group Network, VGGNet）是由牛津大学计算机视觉组提出的深度卷积神经网络模型，其特点是层数较深且参数规模较大。其在图像分类任务上取得了优秀的性能，并对后续的深度学习研究起到了重要的推动作用[45]。VGGNet 网络结构主要由卷积层和全连接层构成，VGGNet 的具体结构如图 2.8 所示。

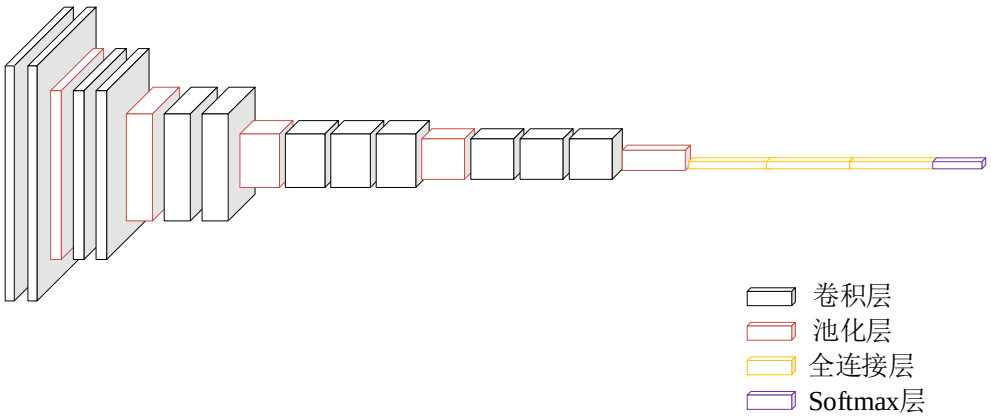


图 2.8 VGGNet 网络结构图

Fig. 2.8 VGGNet network structure diagram

其核心思想是将多个较小尺寸的卷积核串联起来替代一个较大尺寸的卷积核，从而减少模型参数量[46]。

输入层：接受输入图像的像素值。

卷积层：通过多次卷积操作提取图像的特征。VGGNet 使用了连续的卷积层堆叠，每个卷积层都使用3×3的卷积核，并采用 ReLU 激活函数进行非线性映射。

池化层：用于降低特征图的空间维度，减少计算量。VGGNet 使用了最大池化操作，采用2×2的池化窗口进行子采样。

全连接层：将卷积层输出的特征图展开为一维向量，通过多个全连接层进行分类。VGGNet 的全连接层设置了几个较大的隐藏层，最后连接一个 Softmax 输出层，得到分类结果。

VGGNet 采用了较深的网络结构，其网络深度可达 16 层或 19 层。这种深度结构使得 VGGNet 能够更好地捕获图像的细节和高层次特征。

(2)MobileNet

移动神经网络（MobileNet）作为谷歌团队于 2017 年推出的创新之作，特别针对移动端及嵌入式设备设计，旨在打造轻量级的卷积神经网络[47]。相较于传统的 CNN 模型，MobileNet 在维持较高准确率的同时，显著减少了参数数量及运算复杂度，从而实现了更为高效的计算与存储资源利用。这一特性使得 MobileNet 在资源受限的环境中表现尤为出色，为实际应用提供了更加灵活和高效的解决方案。

深度可分离卷积是 MobileNet 模型的核心，它代表一种因式化卷积形式，能将常规卷积分解为深度卷积与点卷积两个步骤，类似于将乘法进行因式分解[48]。

假设输入卷积核大小为 D_K，输出卷积核大小为 D_F，若输入与输出的尺寸维持不变，一个标准卷积流程会把输入层从 $D_K \times D_K \times M$ 转化为输出层 $D_F \times D_F \times N$，这里的输入输出特征图的尺寸是 $D_F \times D_F$，输入通道数为 M，输出通道数为 N，标准卷积示意图如图 2.9 所示。

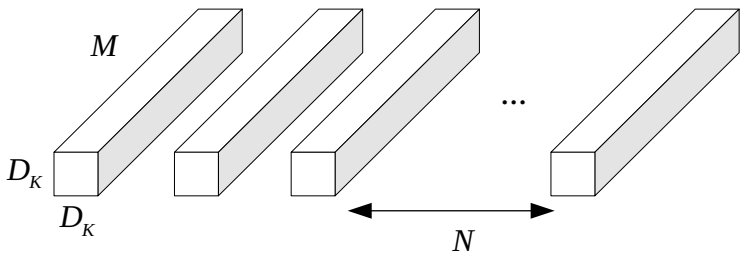


图 2.9 标准卷积

Fig. 2.9 Standard convolutions

深度可分离卷积计算过程可拆解为两个阶段。首先在深度卷积阶段 M 个尺寸为 $D_K \times D_K$ 的矩阵会进行 $D_F \times D_F$ 次移动；其次在逐点卷积阶段，N 个尺寸为 $1\times1\times M$ 的卷

积核会进行 $D_F \times D_F$ 次移动，深度卷积示意图和逐点卷积示意图分别如图 2.10（a）及 2.10（b）所示。

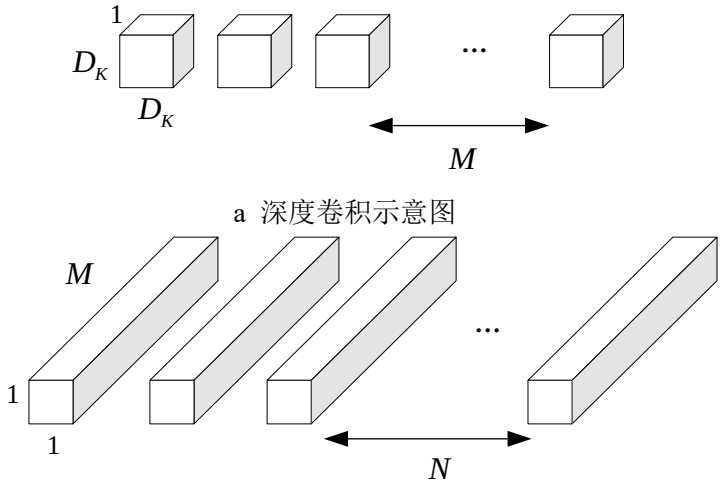


图 2.10 深度卷积和逐点卷积示意图

Fig. 2.10 Diagram of depthwise convolution and pointwise convolution

MobileNet 深度可分离卷积计算过程如图 2.11 所示，经过深度卷积和逐点卷积两个步骤，实现了在减少参数数量的同时提取特征的目的。

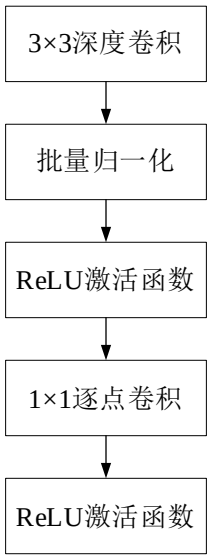


图 2.11 MobileNet 深度可分离卷积示意图

Fig. 2.11 Diagram of MobileNet depth-separable convolution

综合深度卷积和逐点卷积两个阶段，可以得出总共的卷积移动次数，从而求得深度可分离卷积的总计算量为 $D_K \times D_K \times M \times D_F \times D_F + 1 \times 1 \times M \times N \times D_F \times D_F$，深度可分离

卷积与标准卷积在计算量上的比例如式（2.14）所示：

$$\frac{D_K \times D_K \times M \times D_F \times D_F + M \times N \times D_F \times D_F}{D_K \times D_K \times D_F \times D_F \times M \times N} = \frac{1}{N} + \frac{1}{D_k^2} \tag{2.14}$$

通常情况下 N 的取值非常大，前一项的影响可以忽略。在 MobileNet 模型中卷积核的尺寸 D_K 值被设定为3×3的尺寸。将这一值代入公式进行计算，可以发现深度可分离卷积相较于标准卷积在计算量上大约能够降低 87.5%。深度可分离卷积的设计使得 MobileNet 模型在保持性能的同时，大大减少了计算复杂度，并且提高了运算效率。

(3)ResNet

残差网络（Residual Network, ResNet）是深度学习领域中的一个重要突破，它极大地推动了卷积神经网络的发展，特别是在解决深度网络的训练问题方面。

ResNet 的核心思想是引入残差学习。传统的卷积神经网络在训练时，每一层都试图直接拟合一个潜在的恒等映射函数，但随着网络深度的增加，这个任务变得越来越困难，而残差学习则试图让网络学习这种恒等映射的残差。通过这种方式，ResNet 能够更容易地学习到恒等映射，并使得深层网络的训练成为可能。

在深度网络中某个非线性单元 $f(x,\theta)$ 能够尽可能的接近目标函数 $h(x)$，假设将目标函数拆分为恒等函数 $g_1(x)=x$ 和残差函数 $g_2(x)=h(x)-x$ 两部分，其中恒等函数表示输入与输出完全相同的部分，而残差函数则表示目标函数与恒等函数之间的差异，如式（2.15）所示：

$$h(x) = x + (h(x) - x) \tag{2.15}$$

等宽卷积是一种卷积操作，其特点在于输入特征图与输出特征图的通道数保持一致，从而维持了通道维度上的大小相等。

在等宽卷积中，卷积核的数量精确地对应于输入特征图的通道数，确保每个卷积核都能在对应的输入通道上进行滑动操作，这样的设计有助于有效提取输入特征图在空间上的局部特征。

典型的残差单元结构示例如图 2.12 所示，图中 ReLU 代表激活函数，等宽卷积主要作用是保持输入和输出特征图的尺寸相同，以便将残差连接应用于相同尺寸的特征图。

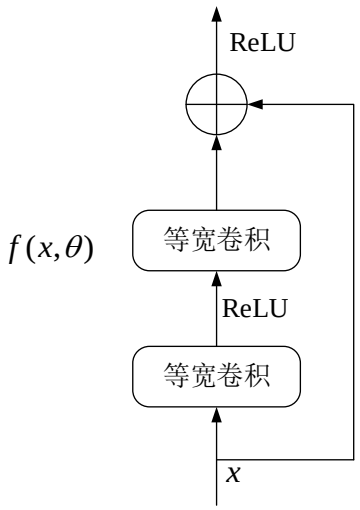


图 2.12 残差单元结构

Fig. 2.12 Residual unit structure

2.3 YOLO 目标检测模型

YOLO 系列目标检测模型在交通标志识别方面相对二阶段检测算法具有显著的优势，包括速度快、精度高、可解释性强、适用性广和泛化能力强等。

YOLOv8 是在 2023 年 1 月 10 日开源的基于 YOLOv5 的下一个重大更新版本，由 YOLOv5 的作者 ultralytics 公司更新。YOLOv8 支持全方位的视觉 AI 任务，包括检测、分割、姿态估计、跟踪和分类，继承了 YOLO 系列一贯的实时目标检测优势，并在网络结构、特征提取、损失函数等方面进行了优化和创新，其网络结构采用了 Darknet 框架。整个网络由多个卷积层、池化层和全连接层组成，形成了一个深度卷积神经网络。

YOLOv8 的网络结构可以分为以下几个部分：

主干网络（Backbone）部分组合了 CBS 模块（Conv BatchNorm SiLU）、通道到像素特征融合模块（Channel-to-Pixel Feature Fusion Module, C2f）以及快速空间金字塔池化模块（Spatial Pyramid Pooling Fast, SPPF）。在输入端，运用了自适应图片缩放技术，以灵活调整输入尺寸。同时，采用了数据增强策略，有效增强了模型的鲁棒性。CBS 模块集成了卷积操作、批归一化以及 SiLU 激活函数，这一组合不仅提升了模型的稳定性，还加速了收敛过程，有效避免了梯度消失的问题。相较于 YOLOv5 采用的 C3 模块，C2f 模块引入了跳层连接和额外的分裂（Split）操作，丰富了模型的梯度流。而在 SPPF 模块中，通过结合池化和卷积操作，实现了特征的融合，能够自适应地融合各种尺度的特征信息，进而增强了模型的特征提取能力。

颈部网络（Neck）部分主要负责处理从主干网络提取的特征。通过跨层连接机制，

实现了自顶向下与自下向上的特征融合，使得特征信息得以更充分的利用。

头部网络（Head）部分，YOLOv8 采用了解耦头结构，将检测任务与分类任务分离。通过根据分类和回归的分数加权得到的分数来确定正负样本，有效提升了模型的性能表现，YOLOv8 网络结构图如图 2.13 所示。

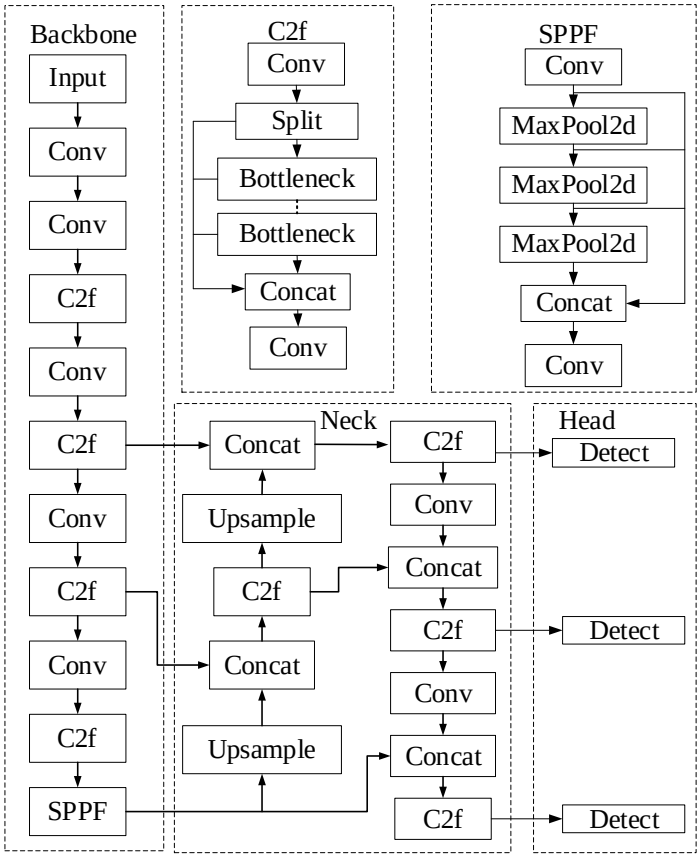


图 2.13 YOLOv8 模型结构图

Fig. 2.13 YOLOv8 model structure

在特征提取方面，YOLOv8 采用了多种技术来提高特征的表达能力和鲁棒性。

首先，通过深度可分离卷积和点卷积的组合，在减少了网络的参数量和计算量的同时也保持了特征的表达能力。其次，引入了注意力机制，使得网络能够自适应地关注到图像中的重要区域和特征通道，提高了特征的判别性和鲁棒性。

在特征融合领域，YOLOv8 展示了一种先进的多尺度特征融合技术。该技术通过有效地融合与传递来自不同尺度的特征，使得网络得以全面捕捉各种尺度的目标信息。

YOLOv8 还运用了上采样与下采样操作，实现了特征在不同尺度间的顺畅融合。此外，它还创新性地整合了跨层连接的理念，将浅层次与深层次的特征紧密结合，进而提升了特征的多样性与表达力。

在损失函数方面，YOLOv8 采用了多种损失函数的组合来优化模型的性能。首先，对于目标框的回归任务，采用了均方误差损失或交叉熵损失等常见的损失函数来衡量预测框与真实框之间的差异。其次，对于目标的分类任务，采用了多类别交叉熵损失来衡量预测类别与真实类别之间的差异。此外，还引入了正则化项来防止模型过拟合和提高泛化能力。在优化方面，YOLOv8 采用了随机梯度下降和优化算法来更新网络参数。此外，还采用了数据增强和批量归一化等技术来提高模型的鲁棒性和泛化能力。

2.4 生成对抗网络

生成对抗网络（Generative Adversarial Networks, GAN）是一种无监督的深度学习模型，其基本思想是通过框架中至少两个模块：生成器 G 和判别器 D 的互相博弈学习来产生输出[49]。在 GAN 中，生成器 G 的目标是捕捉样本数据的分布，并生成新的数据，这些数据在理想情况下应该与真实数据无法区分。而判别器 D 则是一个二分类器，其目标是判断一个样本是来自于真实数据还是由生成模型产生的，GAN 网络结构示意图如图 2.14 所示。

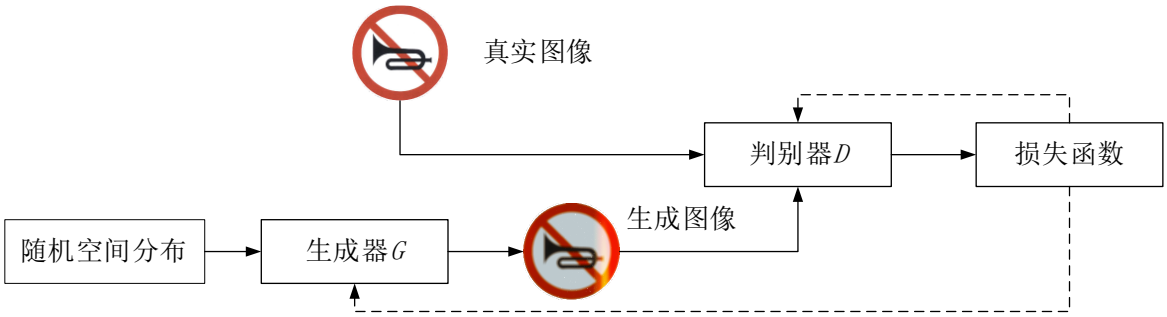


图 2.14 GAN 网络结构图

Fig. 2.14 GAN model structure

在生成器的运作过程中，首先将随机分布 z 引入生成器 G。随后，生成器将这些随机分布转换为图像。通过深入学习和掌握真实样本的分布特性，生成器不断调整其内部参数，以期能够产生更为逼真的新样本。从这个角度来看，生成器 G 的主要目标在于减少生成图像 $G(z)$ 与真实图像分布之间的差异性，即最小化两者之间的距离。

判别器的原理是基于一个二分类网络，用于对输入的样本进行分类，并输出一个介于 0 到 1 之间的分数 $D(x)$，这个分数表示该样本是真实样本的概率。$D(x)$ 越接近于 1，表示判别器认为该样本越有可能是真实样本；$D(x)$ 越接近于 0，表示判别器认为该样本越有可能是由生成器生成的假样本。通过这种方式，判别器在 GAN 中起到了至关

重要的作用，它不仅能够指导生成器的训练方向，还能够确保生成的样本与真实样本在分布上尽可能接近。

生成器G与判别器D的博弈价值函数如式（2.16）所示：

$$\min_{G}\max_{D}V(D,G)=E_{x\sim P_{data}}[\log D(x)]+E_{z\sim P_z}[\log(1-D(G(z)))] \tag{2.16}$$

式（2.16）中，x表示真实数据，z表示网络随机噪声，$D(x)$表示输入数据的判断结果，当判断输入来源为真实数据时结果为 1，判断来源为生成器生成数据时结果为 0，P_{data}和P_z表示真实数据和生成数据所服从的分布。

2.5 本章小结

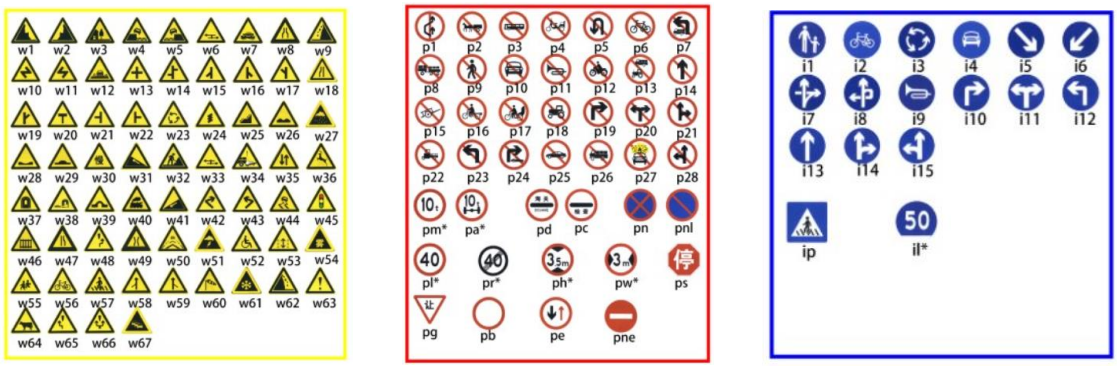
本章首先介绍了道路场景下的交通标志识别流程以及卷积神经网络的基础知识，包括其工作原理、核心组件以及在图像处理中的应用。其次介绍典型的卷积神经网络模型相关思想，如 VGGNet、MobileNet 和 ResNet。然后介绍了 YOLO 目标检测模型的发展历程与网络结构，解释了以 YOLO 模型为基础设计交通标志识别模型的优势。最后介绍了生成对抗网络的原理。通过介绍交通标志识别算法相关理论，为后续针对交通标志识别模型现存问题提出的改进策略提供了理论基础。

第 3 章 基于改进 DCGAN 的交通标志数据集扩充策略

为解决现有算法对于交通标志识别存在的精度低和速度差的问题，首先需要建立合适的交通标志数据集。数据集是神经网络训练的关键，提供了丰富的信息供模型学习，有助于优化参数并防止过拟合。但是现有的交通标志数据集存在整体数据量不足以及各类别实例数不均匀的现象，导致对实例数少的类别识别性能较差。针对以上问题，在原始数据集的基础上设计基于改进深度卷积生成对抗网络（Deep Convolutional Generative Adversarial Networks, DCGAN）的扩充策略，再通过几何变换、颜色变换、噪声添加等多种数据增强方法进行后处理，为模型训练提供了数据基础。

3.1 交通标志数据集问题分析

清华-腾讯 100K 数据集（Tsinghua-Tencent 100K, TT100K）是由清华大学提出的一个高分辨率的交通标志数据集。为了构建这一数据集，研究者选择了中国五个不同城市的 10 个区域，并使得数据集具有更广的地理和场景多样性。对 TT100K 数据集中存在交通标志的图片进行筛选后，包含共 9176 张图像，其中包括 6105 张用于训练的图片和 3071 张用于测试的图片，TT100K 数据集中警告标志、禁止标志以及指示标志三大类别分别如图 3.1（a）、图 3.1（b）及图 3.1（c）所示。



a 警告标志　　b 禁止标志　　c 指示标志

图 3.1 TT100K 数据集三大类别

Fig. 3.1 Three major categories of TT100K dataset

然而，TT100K 数据集各类别之间的实例数量存在着显著的不均匀现象，对于交通标志识别领域的研究存在障碍，统计后数据集类别-实例数柱状图如图 3.2 所示。

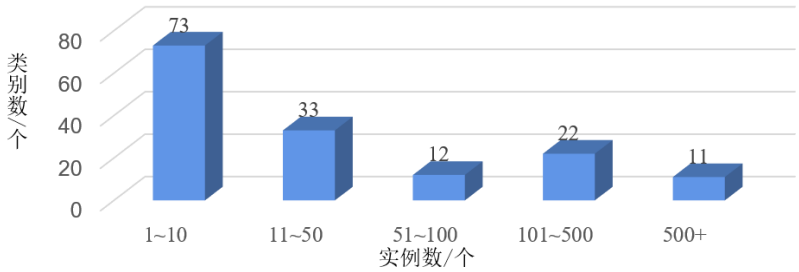


图 3.2 TT100K 类别-实例数柱状图

Fig. 3.2 TT100K category-instance histogram

从图 3.2 中可以看出，在 TT100K 数据集中，实例数在 1~10 范围内的类别数占了很大比重，有 73 类。将统计扩大到实例数 1~50 范围，这些样本比较少的的类别占了整个数据集大部分比重。当训练数据中的某一类别样本数量远多于其他类别时，模型在训练过程中可能会过度关注数量多的类别，反而忽视数量少的类别。这一现象导致模型在预测时更倾向于将结果判定为数量多的类别，对于数量少的类别样本，模型的预测性能会较差。

针对以上问题，在原始 TT100K 数据集的基础上筛选出包含实例数在 50 个以上的 45 类作为基础数据。对于筛选后的数据集，通过设计基于改进 DCGAN 扩充交通标志数据集的方法，再通过多种数据增强方法对生成的图像进行后处理。

3.2 基于改进 DCGAN 网络数据扩增

DCGAN 是基于卷积神经网络架构的一种 GAN 模型，将卷积神经网络应用到 GAN 的生成网络和鉴别网络中，增强了 GAN 的稳定性和生成图像的逼真度和多样性。

通过 DCGAN 网络，可以为交通标志识别提供更多的高质量样本。通过优化判别器的设计，DCGAN 网络可以实现无监督特征学习，使生成器更准确地提取交通标志特征，提高生成交通标志图像的质量。本文设计的改进 DCGAN 网络结构图如图 3.3 所示，图中 z 表示随机噪声，方块代表卷积层。

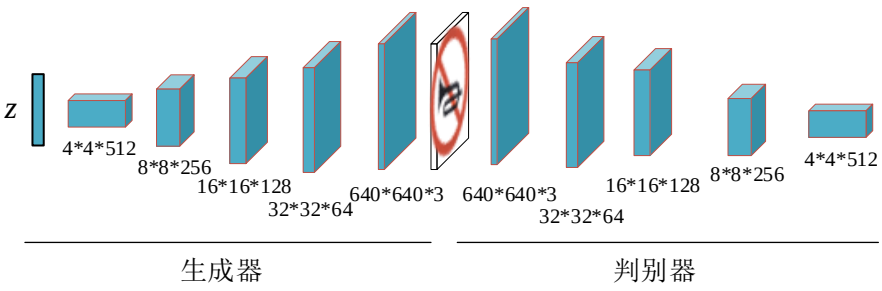


图 3.3 改进 DCGAN 网络结构图

Fig. 3.3 Improved DCGAN network structure diagram

改进后的 DCGAN 网络由生成器G和判别器D构成，其中生成器负责生成交通标志图像，通过输入一个随机噪声z来生成一个图像$G(z)$；判别器D负责判断图像是否为真实的，它的输入图像记为x，输出$D(x)$表示x为真实图像的概率。判别器的工作原理是判别输入图像数据的来源，即判断图像数据是源于真实数据分布，还是来自于由生成器G所产生的模拟数据分布。在整体训练过程中，生成器的目标是生成可以媲美真实图像的模拟图像$G(z)$，当判别器无法区分真伪时，便得到了一个用于扩充交通标志数据集的生样本图像，DCGAN 网络示意图如图 3.4 所示。

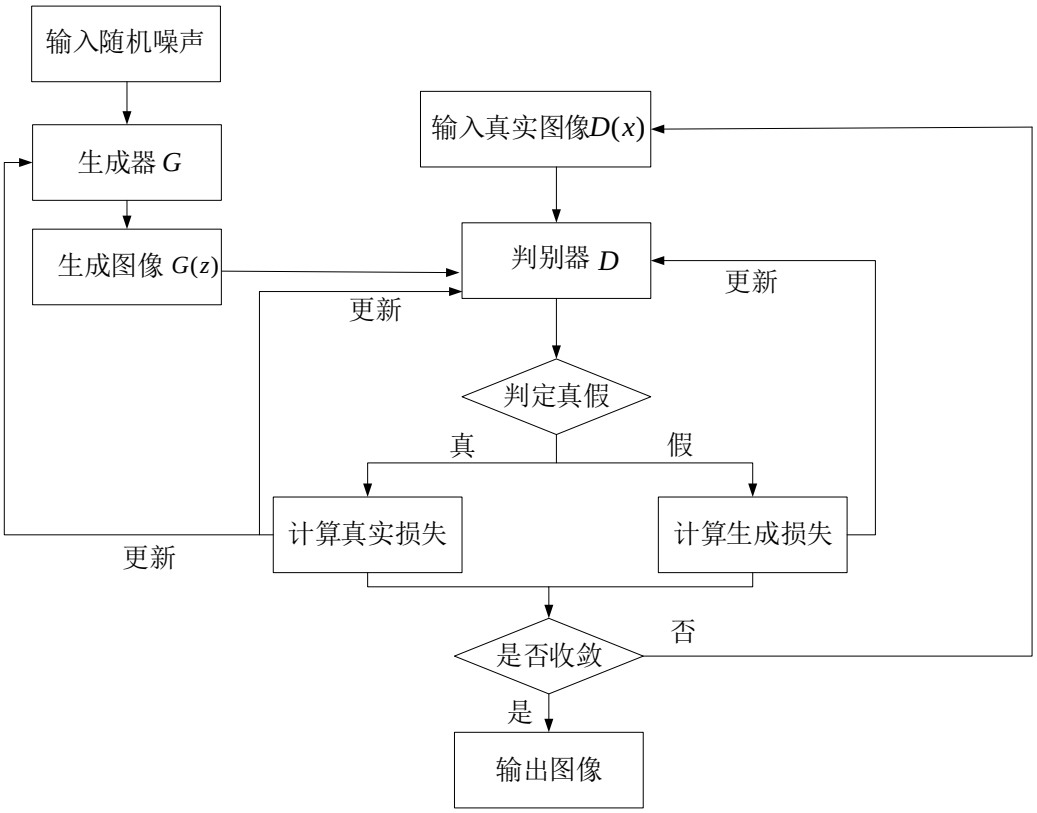


图 3.4 DCGAN 网络示意图

Fig. 3.4 DCGAN network schematic diagram

在 DCGAN 网络的生成器G中，采用跨步卷积替代了传统的确定性空间池化函数，使其能够自主学习空间下采样，来构建出更深层的网络结构。同时，在输入层选用 ReLU 激活函数，在输出层选用式（3.1）所示的 Softplus 激活函数来替代原本采用的 tanh 函数，以加速对交通标志饱和颜色空间的学习。具体流程为将符合正态分布的随机噪声z输入到生成器G中，通过全连接层将其重塑为 3D 向量，随后串联多个转置卷积层，最终输出$640\times640\times3$大小的图像。

$$softplus(x)=log(1+e^{x}) \tag{3.1}$$

式（3.1）中，x 表示输入变量。

判别器 D 的构建则采用 LeakyReLU 作为激活函数，通过卷积运算逐步进行下采样操作，并在每一层级中获取交通标志图像的特征信息，并在输出层应用 sigmoid 函数进行二分类预测，LeakyReLU 及 sigmoid 函数如式（3.2）及式（3.3）所示：

$$LeakyReLU(x)=\begin{cases}x, & if \quad x>0\\ \alpha x, & if \quad x\le 0\end{cases} \tag{3.2}$$

$$sigmiod(x)=\frac{1}{1+e^{-x}} \tag{3.3}$$

式（3.2）及式（3.3）中，x 表示输入变量。

DCGAN 网络的损失函数如式（3.4）所示：

$$L=\frac{1}{m}\sum_{i=1}^{m}[ln(1-D(G(z^{(i)})))]+ln(D(x^{(i)}))+ln(1-D(G(z^{(i)}))) \tag{3.4}$$

式（3.4）中，m 表示样本个数。

3.3 交通标志数据集图像后处理

通过改进 DCGAN 网络生成图片后，结合几何变换、颜色变换、噪声添加多种数据增强方式对其进行后处理，可以在保持图像基本特征的同时，加强交通标志识别网络的泛化性，交通标志图像后处理流程如图 3.5 所示。

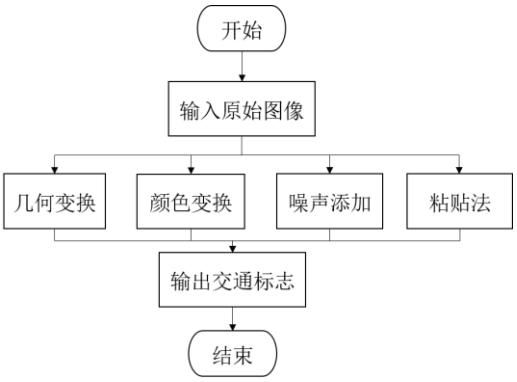


图 3.5 交通标志图像后处理流程图

Fig. 3.5 Flowchart of post-processing of traffic sign images

几何变换作为数据增强的一种常用方法，涵盖了诸如旋转、翻转、缩放、裁剪等多种操作，这些方法可以改变图像的空间布局，而不改变内容。通过将图像旋转一定的角度，或者将图像水平或垂直翻转，来生成新的样本。

几何变换处理后的交通标志图示例如图 3.6 所示，可以看出，变换后的图像角度发生了改变，这种方式对于提高交通标志识别模型对空间变换的鲁棒性非常有帮助。

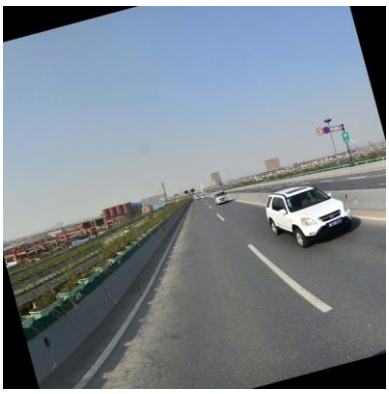


图 3.6 几何变换后的示例图

Fig. 3.6 Example diagram after geometric transformation

颜色调整作为一种常见的数据增强策略，涵盖了诸如调整亮度、对比度和饱和度等多种操作，可以有效地改变图像的色彩分布，同时保持图像的形状与结构不受影响。例如，通过增加或减少图像的亮度，或调整图像的对比度和饱和度，生成新的样本。

颜色变换处理后的交通标志图示例如图 3.7 所示，可以看出，变换后的图像增加了亮度，经过颜色变换的交通标志图像对于提高模型对光照变化和颜色变化的适应性非常有帮助。

图 3.7 颜色变换后的示例图

Fig. 3.7 Example image after color change

噪声添加是一种通过向图像中添加随机噪声来生成新样本的方法。可以选择不同的噪声类型和强度来生成新的样本，常用的噪声类型包括高斯噪声、椒盐噪声等。高斯噪声是一种常见的噪声类型，其分布遵循正态分布。通过向图像中添加高斯噪声，可以模拟图像在传输或捕获过程中可能遇到的随机干扰。椒盐噪声是一种在图像中随机出现的黑白点噪声。这种噪声通常由图像传感器或传输错误引起，通过添加椒盐噪声，可

以测试模型对这类突发干扰的鲁棒性。根据交通标志识别任务需求，选择以上几种噪声类型添加进原数据集图像，对数据集进行扩充。噪声添加处理后的交通标志图示例如图 3.8 所示，可以看出，噪声添加后的图像增加了噪点，通过这种方法可以模拟实际场景中的噪声干扰，提高模型对噪声的抗干扰能力。

图 3.8 噪声添加后的示例图

Fig. 3.8 Example diagram after noise addition

粘贴法主要是指通过复制和粘贴数据集中的样本，增加数据集的规模。这种方法在处理小目标或类别不均衡问题时特别有效。在交通标志数据集中选择一个或多个小目标，然后在图像中的随机位置复制并粘贴这些目标。粘贴法处理后的交通标志图示例如图 3.9 所示，可以看出，粘贴后不同尺寸的交通标志目标随机分布在不同位置，这样可以增加小目标在数据集中的出现频率，有助于模型更好地学习这些目标的特征。

图 3.9 粘贴法处理后的示例图

Fig. 3.9 Example diagram after pasting method

将扩充后的交通标志图片通过 Python 环境中的 labelimg 工具标注，并选择 YOLO 格式，保存为 txt 文件，通过 labelimg 工具可以对扩充后的数据集打上标签，便于下一步模型训练。同时，由于开源的 TT100K 数据集标注文件为 COCO 格式的 json 文件，

因此需要先将原始标注文件转换为 xml 文件，再转换为 YOLO 格式的 txt 文件使用。

3.4 结果分析

通过扩充策略，最终得到 11584 张包含交通标志的道路图片，在其中划分 8901 张图片为训练集、2683 张图片为测试集，划分后的训练集与测试集图片数量的比例近似为三比一，扩充后的 TT100K 类别-实例数柱状图如图 3.10 所示。

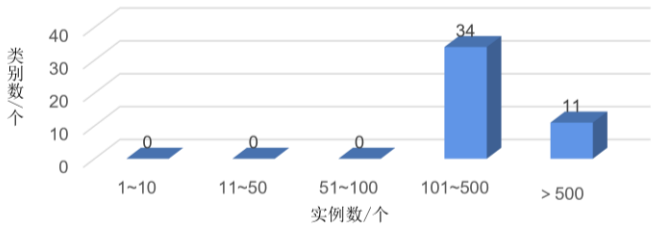


图 3.10 扩充后的 TT100K 类别-实例数柱状图

Fig. 3.10 Expanded TT100K category-instance histogram

从图 3.10 中可以看出，对筛选后的 45 类交通标志图像扩充后，实例数在 1~100 范围内的类别为 0，实例不均匀问题得到改善，扩充前后数据集数据对比如表 3.1 所示：

表 3.1 扩充前后数据集数据对比

Tab. 3.1 Comparison of data sets before and after expansion

	训练集实例	测试集实例	总实例	实例数 1~100 类别	实例数 100+类别
扩充前	4585	2683	7268	34	11
扩充后	8901	2683	11584	0	45

表 3.1 中可以看出，扩充后的数据集数据量增加了 59.34%，且不存在实例数偏少的类别，有助于模型泛化性的提升。

3.5 本章小结

本章针对现有交通标志数据集存在的整体数据量不足以及各类别实例数不均匀的问题，在原始数据集的基础上设计基于改进 DCGAN 的交通标志数据集扩充策略，再通过几何变换、颜色变换、噪声添加等数据增强方式对生成的图像进行后处理，扩充后的交通标志数据集数据量增加了 59.34%，且不存在实例数偏少的类别，为交通标志识别模型训练提供更好的数据基础。

第 4 章 基于改进 YOLOv8 的交通标志识别模型设计

现有的交通标志识别算法在面对背景复杂、小面积交通标志以及交通标志残缺、极端天气影响等情况的处理能力较弱，导致整体识别精度低、速度差。因此，本章首先为提高交通标志识别模型的泛化能力改进缩放指数型线性单元激活函数（Scaled Exponential Linear Unit, SELU），其次为加强交通标志特征融合与多尺度目标的检测识别改进渐近特征金字塔网络，最后为提升网络的识别精度构建双级路由注意力机制，对识别结果贡献率高的区域进行特征提取。实验证明，本文设计模型在速度保持不变的情况下精度有所提升。

4.1 改进 SELU 激活函数

在交通标志识别任务中，自然场景下的交通标志的图像质量容易受到不同时段的天气、光照等条件变化的影响，导致图像质量存在较大的变化，一些交通标志可能会被树木或其他障碍物遮挡，导致部分信息丢失。同时，由于摄像头拍摄角度的不确定性，会导致拍出的交通标志形状、大小等有一定的不同或失真，使得模型识别精度低。

为了应对这些问题，本文设计使用 SELU 激活函数，使得交通标志识别模型能更有效地适应这些变化，从而显著提升其泛化能力。这种泛化能力的提升可以进一步增强模型在实际应用中应对复杂场景的能力，从而提高交通标志识别的准确性。

在深度学习中，激活函数起着至关重要的作用，通过非线性激活函数的使用可以更快地解释复杂数据，使得网络能够学习和逼近复杂的函数[50]。YOLOv8 在网络结构中采用了 SiLU（Sigmoid Gated Linear Unit）激活函数[51]，其数学表达式如式（4.1）所示：

$$f(x) = x^{*} sigmoid(\beta x) \tag{4.1}$$

式（4.1）中，$sigmoid(x)$ 是经典的激活函数，β 是一个可调整的参数，通常设为 1 或固定值。当 $\beta = 1$ 时，函数简化为：

$$f(x) = x / (1 + e^{-x}) \tag{4.2}$$

SiLU 激活函数结合了 sigmoid 函数和 ReLU 函数的优点，具有平滑、非单调的特性。在输入值接近零时，SiLU 函数具有较高的梯度，有助于缓解梯度消失问题。然而，当输入值非常大或非常小时，SiLU 函数可能会出现梯度饱和的问题，导致模型训练困难，因此拟对 YOLOv8 所采用的 SiLU 做出改进，以便在交通标志识别领域中实现更

好的效果。

相比于 SiLU，SELU 激活函数通过引入缩放因子和指数线性单元，使得函数在输入值较大时仍能保持良好的梯度特性[52]。这种改进有助于进一步提高模型的训练效率和性能表现。SELU 激活函数可以有效增强模型的泛化能力。通过结合缩放因子和指数线性单元，SELU 能够在数据通过网络层传播时保持更为稳定的分布，从而有助于控制梯度消失和梯度爆炸的问题。

SELU 激活函数数学表达式为：

$$f(\alpha,x)=\lambda\begin{cases}\alpha(e^{x}-1) & for\ x<0\\ x & for\ x\geq 0\end{cases} \tag{4.3}$$

式（4.3）中，x 代表输入值，$f(\alpha,x)$ 代表经过激活函数的输出值，λ 和 α 是两个常数，通常设置为 $\lambda=1.0507$ 和 $\alpha=1.67326$，SELU 激活函数的曲线图像如图 4.1 所示。

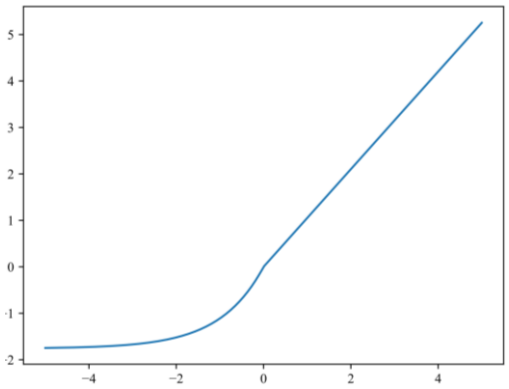


图 4.1 SELU 激活函数曲线

Fig. 4.1 Curve of SELU activation function

在交通标志识别任务中，本文设计采用的 SELU 激活函数相比 SiLU 可以显著提升模型训练速度和稳定性。SELU 激活函数在输入值较大时仍能保持较高的梯度，这有助于加速模型的收敛速度。在交通标志识别任务中，模型需要快速学习到图像中的关键特征以进行准确的识别。使用 SELU 激活函数，模型能够更快地捕捉到关键特征，提升训练效率。SELU 激活函数独特的梯度性质能够有效缓解训练过程中可能出现的梯度消失或爆炸问题，增强模型的稳定性，这种增强的稳定性赋予模型在处理错综复杂的交通标志场景时更强的抗干扰能力。

将 YOLOv8 的激活函数从 SiLU 改进为 SELU 后，模型在交通标志识别任务中的性能表现将会有所提升。SELU 激活函数的优秀梯度特性有助于模型更准确地捕捉到图像中的特征信息，从而提高识别的准确性。此外，SELU 激活函数不仅增强了模型的抗

干扰能力，还有效地减轻了过拟合的风险，进而提升了模型的整体性能。SELU 激活函数的设计考虑了计算效率和资源占用方面的平衡。相比一些复杂的激活函数，SELU 具有相对较低的计算复杂度。在实际应用中，这种性能的提升可以显著提高交通标志识别的准确性和效率。

在交通标志识别任务中，使用 SELU 激活函数的 YOLOv8 模型能够更好地适应不同硬件和计算资源的需求，实现高效且准确的识别。这种对硬件和计算资源的良好适应性也有助于模型在实际应用中更好地部署和推广。

4.2 交通标志小目标识别改进策略

交通标志识别领域内，对于小目标的检测识别面临多个难点。由于小目标在图像中所占的像素点数量较少，其覆盖的图像面积也相对较小。因此，能够为目标检测提供有效信息的特征也相对有限，这导致检测器难以从有限的特征中提取出足够的信息进行准确的目标识别和定位。同时由于交通标志目标的尺寸较小，更容易受到图像噪声的影响；目标与周围环境的对比度可能较低，难以从复杂的背景中区分出来。

原始 YOLOv8 实验所使用的数据集主要包含大型目标，因此在面对远距离的交通标志小目标以及模糊目标时，其检测与识别的效果并不理想，因此在模型中引入渐近特征金字塔网络以及双级路由注意力机制对 YOLOv8 进行改进。

4.2.1 渐近特征金字塔网络

在传统的交通标志识别算法中，由于图像中物体大小存在不确定性，会导致单一尺度的特征提取丢失详细信息。因此，物体检测模型通常会引入特征金字塔网络（Feature Pyramid Network, FPN）架构解决尺度变化问题，FPN 结构如图 4.2 所示。

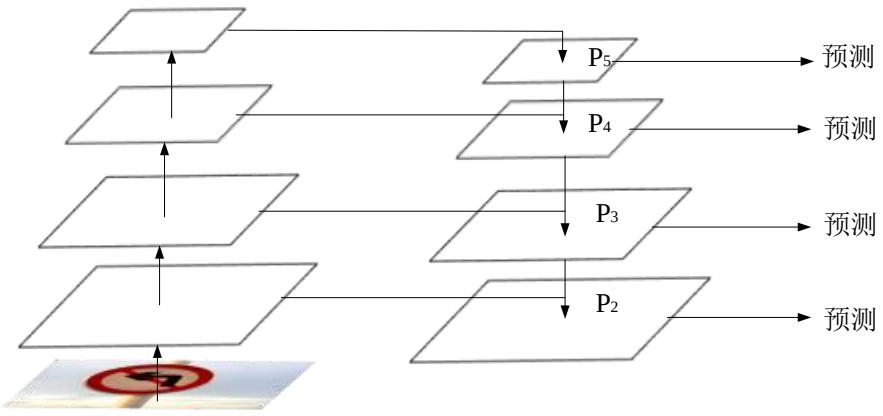


图 4.2 特征金字塔网络

Fig. 4.2 Feature pyramid networks

图 4.2 中，P2、P3、P4、P5 分别表示通过不同尺度的特征融合而得到的特征层，这些不同尺度的特征层可以用于后续的目标检测任务中，以捕捉不同尺度的目标。

特征金字塔网络结构的核心思想是通过对最底层的特征进行上采样处理，再将这些特征与原始底层特征相融合，进而获取具备高分辨率与强语义信息的特征，从而达到提升特征提取效果的目的。这一过程有助于增强模型对图像中不同尺度目标的识别能力，提高整体性能[53]。

在 FPN 的基础上，YOLOv8 所采用的路径聚合特征金字塔网络（Path Aggregation Feature Pyramid Networks, PAFPN）为特征金字塔网络增加了一条自下而上的路径，可以将低层的特征与顶层特征融合，弥补了 FPN 的不足，PAFPN 结构如图 4.3 所示，图中 N2、N3、N4、N5 表示与 P2、P3、P4、P5 对应的自下而上新生成的特征图。

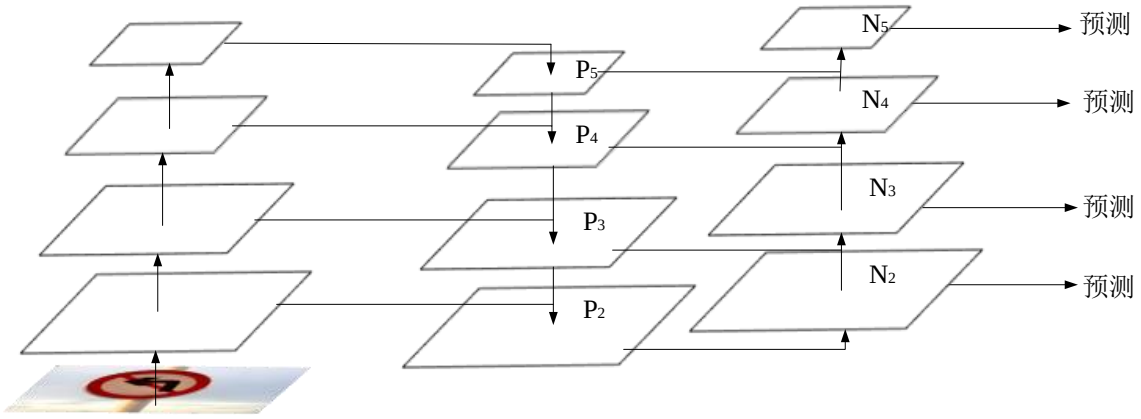


图 4.3 路径聚合特征金字塔网络

Fig. 4.3 Path aggregation feature pyramid networks

FPN 架构中高层次特征位于金字塔顶端，需跨越多个中间尺度以进行传递。这些特征在与底部低层次特征融合之前，还需与中间尺度的特征进行一系列的交互，但此过程可能会引发高层次特征中语义信息的遗失或退化。与 FPN 相似，PAFPN 在进行自下而上的传播时，低层次特征的细节信息在传递与交互过程中亦有可能受损或退化。

为此，本文在 YOLOv8 的原 FPN 的基础上改进为渐近特征金字塔网络（Asymptotic Feature Pyramid Network, AFPN），针对交通标志识别过程中特征丢失或退化导致的识别精度低的问题，做出优化。AFPN 在进行特征融合之前，会首先通过骨干网络来抽取多层次的特征。这样的处理方式有助于更好地提取图像信息，为后续的交通标志检测识别提供更丰富的特征表示，AFPN 的架构示意图如图 4.4 所示。

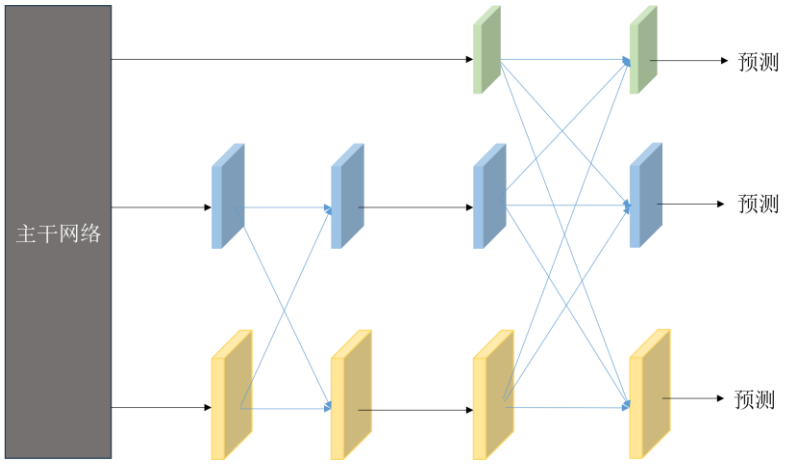


图 4.4 渐近特征金字塔网络

Fig. 4.4 Asymptotic feature pyramid network

图 4.4 中，AFPN 从骨干网络的每个特征层中提取最后一层的特征，生成一组不同尺度的特征集合，表示为{C2,C3,C4,C5}。为了进行特征融合，首先将低级特征 C3 输入到特征金字塔网络中，随后加入 C4，在最后加入高级特征 C5。经过特征融合步骤后，生成一组多尺度特征{P2,P3,P4,P5}。最终的多尺度特征集合为{P2,P3,P4,P5}，对应的特征层具有不同的空间分辨率，分别对应的特征跨度是{4,8,16,32}个像素。

传统的 YOLOv8 所采用的骨干网络在特征提取过程中只采用从下往上的特征融合过程，非相邻层次特征之间的语义差距大于相邻层次特征间的语义差距，尤其是底部和顶部特征，这直接导致了非相邻层次特征的融合效果不尽如人意。

改进后的 AFPN 交通标志特征提取在自下而上的过程中，首先融合低层次与高层次的特征信息，然后再将融合后的特征信息与顶层特征相结合，在这个过程中渐近地整合了底层、高层以及顶层的特征信息，以确保特征信息的全面性与准确性。这样的处理方式不仅提升了特征的表达能力，还有助于后续的交通标志目标检测以及识别任务的性能提升。由于 AFPN 的渐近式架构特点，不同层次的特征在融合过程中，其语义信息会逐步趋于一致，从而有效缓解非相邻分层特征之间的语义差距大于相邻分层特征之间的语义差距导致的融合效果不佳的问题。通过渐近融合，C3 和 C4 之间的语义差距得以缩小。同时，由于 C4 与 C5 是相邻层次特征，它们在融合过程中也能更好地相互协调，进而减小 C3 与 C5 之间的语义差距。为了调整特征维度，从而优化其后的特征融合操作，运用了步长为 1 的1×1卷积和双线性插值法，对特征信息进行了上采样的处理。同时，在下采样方面，针对下采样的需求，根据预设的下采样率，灵活选择不同大小的卷积核与步长。当需要实现 2 倍的下采样时，采用步长为 2 的2×2卷积；对于 4 倍的下采样，则运用4×4卷积。这样不仅确保了维度的对齐，还为后续的特征融

合工作创造便利条件。特征融合操作后，使用四个类似 ResNet 的由两个3×3卷积构成的残差单元来实现特征学习的操作[54]，ResNet 浅层网络残差单元如图 4.5 所示。

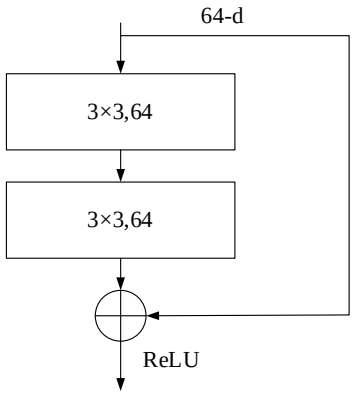


图 4.5 ResNet 网络残差单元

Fig. 4.5 ResNet network residual unit

特征融合模块通常被整合到已有的固定拓扑特征金字塔中，以增强其特征。也有一些研究用于增强特征金字塔的上采样模块[55]。

自适应特征融合模块（Adaptively Spatial Feature Fusion, ASFF）为不同层次的特征添加权重，以便在不同层次的特征之间可能存在矛盾信息的情况下对它们进行有效融合。

通过使用 ASFF 对交通标志图像的低层、高层、顶层三个不同层次的特征进行自适应渐近空间融合，ASFF 结构如图 4.6 所示，图中水平、向下及向上的箭头分别表示水平连接、下采样及上采样。

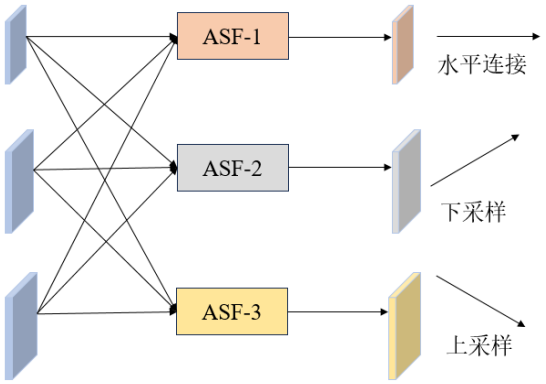


图 4.6 自适应空间融合操作

Fig. 4.6 Adaptive spatial fusion operation

假设 $x_{ij}^{n\to l}$ 为从第 n 层到第 l 层的在 (i, j) 位置的特征向量。通过多层次特征信息的自

适应特征融合，得到由 $x_{ij}^{1\to l}, x_{ij}^{2\to l}$ 以及 $x_{ij}^{3\to l}$ 所表示的结果特征向量 y_{ij}^{l} 如式（4.4）所示：

$$y_{ij}^{l}=\alpha_{ij}^{l}\cdot x_{ij}^{1\to l}+\beta_{ij}^{l}\cdot x_{ij}^{2\to l}+\gamma_{ij}^{l}\cdot x_{ij}^{3\to l} \tag{4.4}$$

式（4.4）中，α_{ij}^{l}、β_{ij}^{l} 以及 γ_{ij}^{l} 表示在第 l 层的三个等级特征的空间权重，服从约束条件：

$$\alpha_{ij}^{l}+\beta_{ij}^{l}+\gamma_{ij}^{l}=1 \tag{4.5}$$

考虑到在 AFPN 的每个阶段中融合特征数量的差异，针对每个阶段实施特定数量的自适应空间融合模块。通过设计 AFPN，可以加强特征的融合与多尺度目标的检测识别，从而提高对交通标志识别的准确性。

4.2.2 双级路由注意力机制

在传统的交通标志识别领域中，随着实际应用场景的复杂性和多样性不断增加，模型如何更好地处理不同尺度、不同光照条件下的交通标志十分关键。虽然 YOLOv8 已经在交通标志识别任务上取得了不错的表现，但仍然存在一些挑战性问题，如小目标的检测识别、光照条件不佳目标以及遮挡条件下的交通标志的识别等[56]。

因此针对交通标志小型且模糊的特点，本文提出双级路由注意力机制，直接建立长距离依赖关系模型。交通标志识别领域中普遍所采用的传统注意力机制通常需要计算输入序列中所有元素之间的相关性，这会导致计算量大，时间复杂度高，尤其是在处理长序列时[57]。

双级路由注意力机制的基本思路是首先在较为宽泛的区域层级中进行筛选，以剔除大量不相关的键值对，进而仅保留少数具有潜力的路由区域。紧接着，在这些经过筛选的路由区域的合集之上，实施更为精细化的 token-to-token 注意力机制。通过这种双级过滤与聚焦的方式，该机制能够更有效地处理信息，提升模型的性能，双级注意力机制如图 4.7 所示。

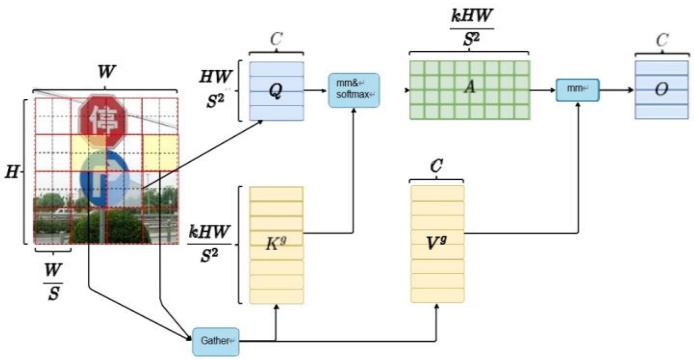


图 4.7 双级路由注意力机制

Fig. 4.7 Bi-level routing attention

给定一个二维输入特征映射图 $X \in R^{H \times W \times C}$，首先将特征图分解为 $S \times S$ 个不重叠的区域，分解后的每个区域内包含 HW / S^2 个特征向量，使得：

$$X^r \in R^{S^2 \times \frac{HW}{S^2} \times C} \tag{4.6}$$

接下来使用线性投影导出 $Q, K, V \in R^{S^2 \times \frac{HW}{s^2} \times C}$：

$$Q = X^r W^q, K = X^r W^k, V = X^r W^v \tag{4.7}$$

式（4.7）中，$W^q, W^k, W^v \in R^{C \times C}$ 分别为 Q, K, V 的投影权重。

通过构建有向图，采用区域间路由策略，明确了哪些区域应为给定区域提供注意力。在实现过程中，首先针对 Q 和 K 分别实施区域均值运算操作，进而生成区域级别的 $Q^r, K^r \in R^{S^2 \times C}$。接下来，将 Q^r 与 K^r 的转置执行矩阵乘法，由此获得反映区域间关联度的邻接矩阵 $A^r \in R^{S^2 \times S^2}$ 如式（4.8）所示：

$$A^r = Q^r \left(K^r\right)^T \tag{4.8}$$

邻接矩阵 A^r 中的元素用来衡量两个区域之间的语义相关性。在这之后的核心步骤是通过保留每个区域的前 k 个连接来修剪亲和图。通过利用逐行取前 k 个操作符，推导出一个路由索引矩阵 $I^r \in N^{S^2 \times k}$：

$$I^r = topkIndex(A^r) \tag{4.9}$$

由此可得，路由索引矩阵 I^r 的第 i 行包含与第 i 个区域最相关的 k 个区域的索引信息，区域 i 的每个 token 将关注从由 $I^r_{(i,1)}, I^r_{(i,2)}, \ldots, I^r_{(i,k)}$ 指定的 k 个路由区域的并集中所汇集的所有键值对。不过，由于这些路由区域可能广泛分布在特征图上，因此如何有效地实现这一步是一个挑战。

目前的 GPU 更倾向于连续的内存访问，它们一次加载多个连续字节。因此，在继续执行注意力机制之前，为了确保更高效的内存操作和更快的计算速度，优先考虑聚集键和值张量，如式（4.10）所示：

$$K^g = gather(K, I^r), V^g = gather(V, I^r) \tag{4.10}$$

式（4.10）中，$K^g, V^g \in R^{S^2 \times \frac{kHW}{s^2} \times C}$ 代表得到的聚集键以及值张量，通过它们，得以实现细粒度的 token-to-token 的注意力机制，如式（4.11）所示：

$$O = Attention(Q, K^g, V^g) + LCE(V) \tag{4.11}$$

式（4.11）中，LCE 表示卷积核为 5 的深度可分离卷积。

双级路由注意力机制首先对粗糙区域进行过滤，剔除大量无关的键值对，以减少

冗余信息，接着仅对一小部分关键路由区域应用注意力机制。相较于直接在全局范围应用自注意力机制，本文设计的双级机制处理显著降低了计算量，并且有利于交通标志识别的实时性。

4.3 实验环境及性能评价指标

实验采用了经过扩充后的 TT100K 数据集以及 CCTSDB 数据集对交通标志识别模型进行训练。实验软件环境：操作系统为 Windows 11，Python:3.9.17，深度学习框架 Pytorch2.0.1，YOLOv8.0.114；硬件环境：CPU 为 Intel Core i5 12400，GPU 为 NVIDIA GTX 3070，运行内存为 32GB，硬盘为 1TB。

训练过程中对网络设定的部分超参数如表 4.1 所示：

表 4.1 实验选取的超参数

Tab. 4.1 Experimental selection of hyperparameters

训练参数	参数值
训练轮次 epochs	100
输入图片尺寸 imgsz	64
初始学习率 lr0	0.01
学习率衰减因子 lrf	0.01
动量值 momentum	0.937
权重衰减 weight_decay	0.0005
预热期轮次 warmup_epochs	3.0
预热期动量 warmup_momentum	0.8
预热偏差 warmup_bias_lr	0.1
每轮样本数 batch-size	16
边界框回归损失 box	7.5
损失增益 Cls	0.5
损失增益 Dfl	1.5
标签平滑 label_smoothing	0.0
验证 Val	True
标准批次大小 nbs	64
数据增强 mosaic	1.0

YOLOv8 算法的性能评估涵盖多个维度，包括：准确率、精确率、召回率、F1 分数、交并比（Intersection over Union, IoU）、平均精度（Average Precision, AP）、多类别平均精度（Mean Average Precision, mAP）、漏检率、虚警率，以及前向传播时间、每秒处理帧数、模型层数、参数数量和浮点运算次数。

准确率反映了模型正确预测的目标与总预测数量的比例。精确率则揭示了模型预

测为正样本的实例中真正正样本的比重，这体现了模型预测的准确性。而召回率展示了实际为正样本的实例中，被模型正确预测出来的比例，这衡量了模型找出真正正样本的能力。为了综合精确率和召回率的表现，引入了 F1 分数，它是精确率和召回率的调和平均数，为模型提供了一个统一的性能度量标准。IoU 是一个评估模型检测框与实际目标框重叠程度的指标，通过计算检测框与实际框的交集面积与并集面积之比来衡量目标检测的准确性。平均精度表示精确率-召回率曲线（Precision-Recall, PR）下的面积，它全面考虑了不同阈值下的精确率和召回率。对于多类别目标检测任务，mAP 则是一个重要的性能度量指标，它计算了所有类别 AP 的平均值，以此全面评估模型在多类别目标检测上的表现。此外，漏检率反映了模型未能检测到的目标占所有实际正样本的比例，而虚警率则表示模型错误地将负样本预测为正样本的比例。

在模型效率方面，前向传播时间衡量了模型进行一次预测所需的时间；每秒帧数则描述了模型在单位时间内能够处理的图像帧数。模型的层数，包括卷积层、池化层、全连接层等，反映了模型的复杂度。参数数量表示模型中需要学习的参数总数，这直接影响模型的规模和存储空间需求，同时也影响训练和推理的计算复杂度。

浮点运算次数是衡量模型计算复杂度的一个关键指标，它表示模型进行一次前向传播所需的浮点运算次数，数值越高意味着模型在推理过程中所需的计算量越大。因此，在设计和选择模型时，需要综合考虑 GFLOPs 以及其他性能指标，以确保模型能够在有限的计算资源下实现高效的推理。

这些性能指标有助于全面评估 YOLOv8 在目标检测任务中的性能，对于交通标志识别的应用场景，主要关注层数（Layers）、参数量（Parameters）、浮点运算次数（GFLOPs）、精确率（Precision）、召回率（Recall）、每秒帧数（FPS）、IoU 阈值为 0.5 时的平均精度指标（mAP50），通过表 4.2 所示的混淆矩阵来计算精确率、召回率和 mAP 等指标。

表 4.2 混淆矩阵

Tab. 4.2 Confusion matrix

样本真实情况	正样本	负样本
正样本	TP	FN
负样本	FP	TN

具体计算公式如下所示：

$$Precision = \frac{TP}{TP + FP} \tag{4.12}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.13}$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{4.14}$$

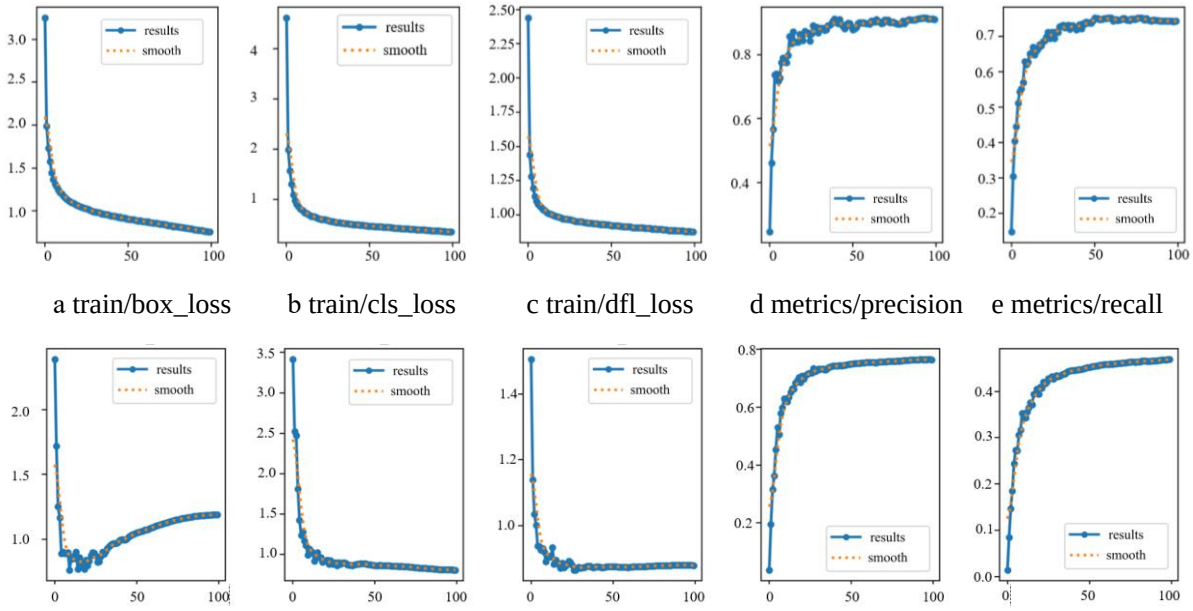
式（4.12）中，真正例（TP）指的是被正确预测为正样本的数量，而假正例（FP）则表示实际为负样本但被错误预测为正样本的数量。另一方面，假负例（FN）则代表实际为正样本却被错误预测为负样本的数量，真负例（TN）则代表被正确预测为负样本的数量。关于平均精确度（AP）的计算是通过将某一类别在给定集合中的所有精确率求和，然后除以包含该类别目标的图像总数来获得的。

4.4 实验结果验证与分析

设计对各部分改进措施与原 YOLOv8 算法分别做对比实验，最终将各改进模块整合为最终算法与原算法详细对比。

4.4.1 改进激活函数后的算法性能对比

使用 SELU 激活函数替代 YOLOv8 所采用的 SiLU 激活函数，将改进后的网络称为 YOLOv8-A。对原网络以及改进后的网络在实验环境下使用 CCTSDB2021 数据集分别进行一百轮训练，得到的训练结果分别如图 4.8 及图 4.9 所示。

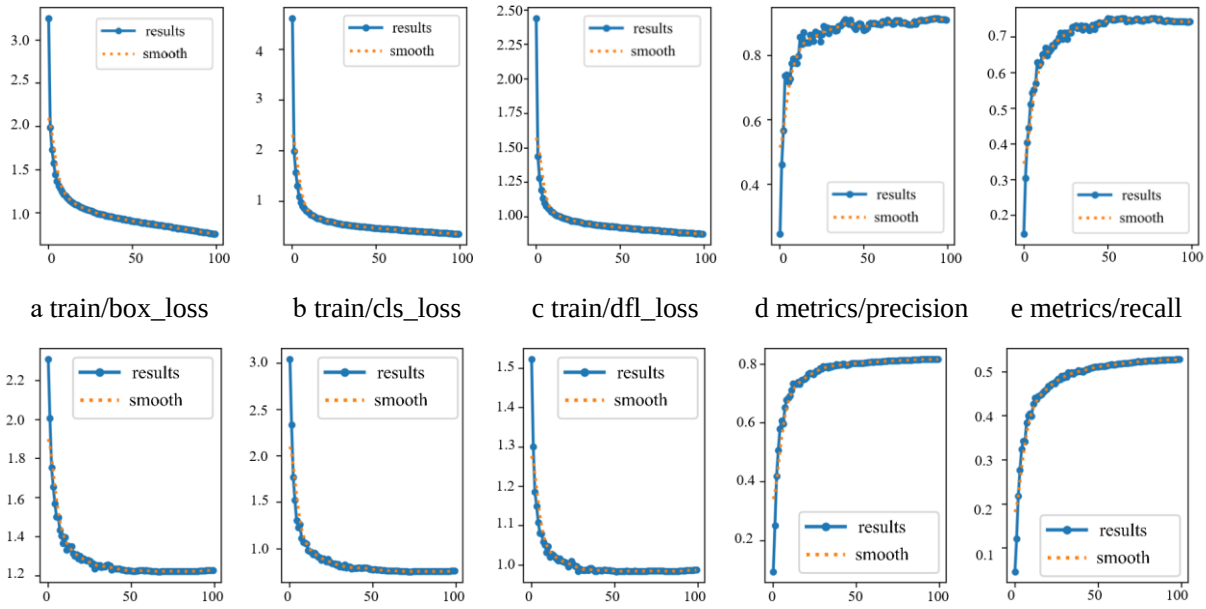


a train/box_loss　b train/cls_loss　c train/dfl_loss　d metrics/precision　e metrics/recall

f val/box_loss　g val/cal_loss　h val/dfl_loss　i mAP50　j mAP50-95

图 4.8 原 YOLOv8 训练结果

Fig. 4.8 YOLOv8 training results



a train/box_loss　b train/cls_loss　c train/dfl_loss　d metrics/precision　e metrics/recall

f val/box_loss　g val/cal_loss　h val/dfl_loss　i mAP50　j mAP50-95

图 4.9 YOLOv8-A 训练结果

Fig. 4.9 YOLOv8-A training results

图 4.8 及图 4.9 中，横坐标为训练轮数，train/box_loss 曲线代表训练集的边界框回归损失，表示预测边界框与真实边界框之间的差异或误差；train/cls_loss 曲线代表分类损失，表示模型的预测结果与真实情况的接近程度；train/dfl_loss 曲线代表训练集的 DFL（Distribution Focal Loss）损失函数曲线，表示模型在预测物体边界框时误差的变化情况；metrics/precision 曲线代表精确率曲线，表示分类器预测为正例的样本中，实际为正例的比例；metrics/recall 代表召回率曲线，表示模型成功找出的正例样本占所有正例样本的比例；val/box_loss 曲线代表验证集的边界框回归损失，表示预测边界框与真实边界框之间的差异程度；val/cls_loss 曲线代表验证集的分类损失，表示验证集上的分类性能；val dfl_loss 曲线代表验证集的 DFL 损失函数曲线，mAP50 代表平均精度变化曲线，表示模型的平均精度，mAP50-95 曲线代表在 0.95 阈值下的平均精度变化曲线。

对图 4.8 以及图 4.9 选取五个关键的性能指标——准确率（P）、召回率（R）、平均精度均值在 IoU 阈值为 0.5 时的表现（mAP50）、模型参数数量（parameters）以及每秒处理的帧数（FPS），来全面对比和评估改进前后的算法性能。这些指标能够从多个角度反映算法在分类准确性、检测精度、模型复杂度以及实时处理能力等方面的表现。对比数据如表 4.3 所示：

表 4.3 改进 SELU 后训练结果对比

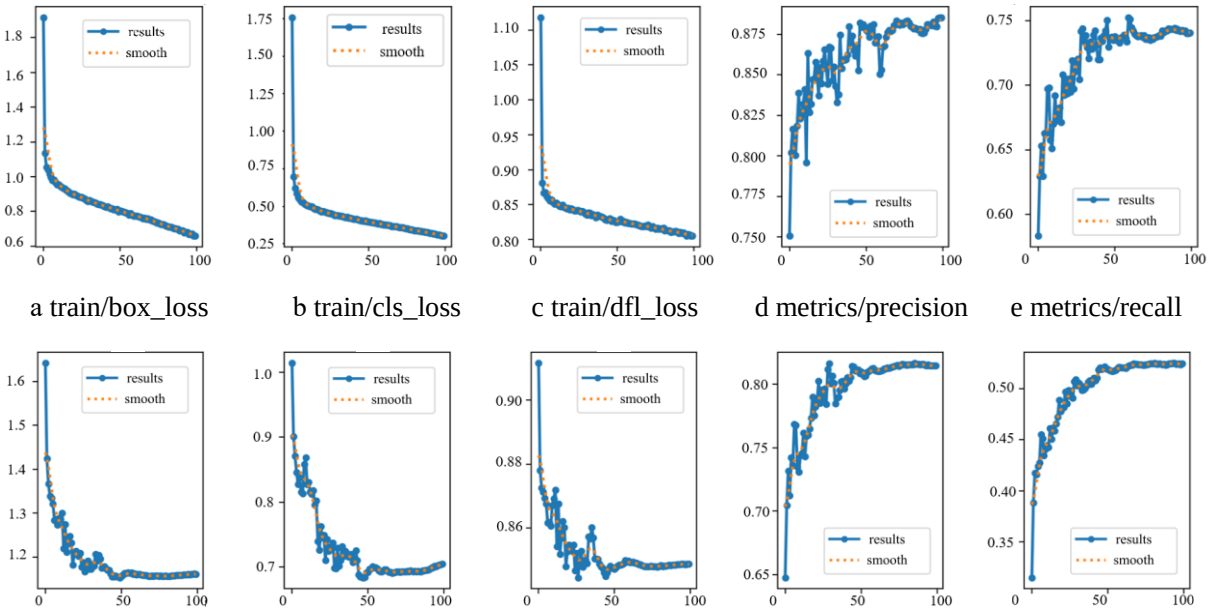
Tab. 4.3 Comparison of training results after improving SELU

	P	R	mAP50	parameters	FPS(帧/s)
YOLOv8	0.846	0.705	0.763	11126745	322.58
YOLOv8-A	0.853	0.727	0.786	11126745	333.33

可以看出改进激活函数后的 YOLOv8-A 算法对比原 YOLOv8 算法在多项指标上均有提升，P 值提升了约 0.7%，R 值提升了约 2.2%，mAP50 也有约 2.3%的提升，证明改进措施是有效的。

4.4.2 加入渐近特征金字塔网络后的算法性能对比

在 YOLOv8 的基础上引入 AFPN（渐近特征金字塔网络），针对交通标志识别过程中特征丢失或退化问题，做出一定优化，得到的算法命名为 YOLOv8-B，训练结果如图 4.10 所示。



a train/box_loss　b train/cls_loss　c train/dfl_loss　d metrics/precision　e metrics/recall

f val/box_loss　g val/cal_loss　h val/dfl_loss　i mAP50　j mAP50-95

图 4.10 YOLOv8-B 训练结果

Fig. 4.10 YOLOv8-B training results

与原 YOLOv8 训练结果图进行对比分析，选取 P、R、mAP50、parameters 、FPS 五项指标对比改进前后的算法性能如表 4.4 所示：

表 4.4 引入 AFPN 后训练结果对比

Tab. 4.4 Comparison of training results after introducing AFPN

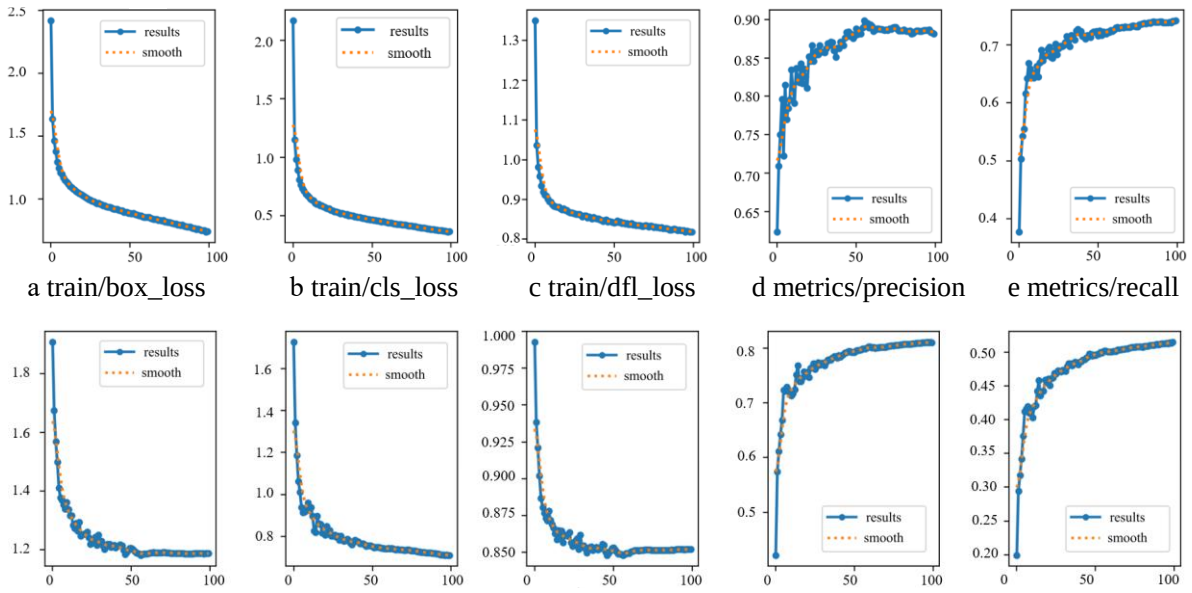
	P	R	mAP50	parameters	FPS(帧/s)
YOLOv8	0.846	0.705	0.763	11126745	322.58
YOLOv8-B	0.885	0.740	0.814	12192505	256.41

从表 4.4 中可以得出，加入渐近特征金字塔网络后的算法 P 值、R 值、mAP50 均有一定提升，其中 P 值提升了约 3.9%，R 值提升了约 3.5%，mAP50 提升了约 5.1%，但是识别速度降低了一些，FPS 值下降了约 66 帧/s。

FPS 值下降是由于渐近特征金字塔网络涉及多尺度特征融合过程要求交通标志识别模型处理更丰富的特征图，并实现不同尺度间的信息互通。这种多尺度的特征融合机制加深了模型的深度，拓展了其宽度，因此提升了模型的复杂度。更复杂的模型在进行推理时往往需要更多的计算资源和时间，这不可避免地会导致每秒帧数的下降。

4.4.3 添加双级路由注意力机制后的算法性能对比

加入渐近特征金字塔网络后识别速度不可避免地得到了一定损失，为解决这一问题，保证模型的实时性，添加双级路由注意力机制。通过动态分配图像块到上下层路由器，双级路由注意力机制可以减少不必要的计算量。模型只需要处理与当前任务最相关的图像块，从而提高了整体的计算效率。在原 YOLOv8 基础上添加双级路由注意力机制后的算法命名为 YOLOv8-C，训练结果如图 4.11 所示。



a train/box_loss　b train/cls_loss　c train/dfl_loss　d metrics/precision　e metrics/recall

f val/box_loss　g val/cal_loss　h val/dfl_loss　i mAP50　j mAP50-95

图 4.11 YOLOv8-C 训练结果

Fig. 4.11 YOLOv8-C training results

将图 4.11 与原 YOLOv8 训练结果图 4.8 进行对比分析，选取 P、R、mAP50、parameters、FPS 五项指标对比改进前后的算法性能如表 4.5 所示：

表 4.5 添加双级路由注意力机制训练前后结果对比

Tab. 4.5 Comparison of training results after introducing bi-level routing attention

	P	R	mAP50	parameters	FPS(帧/s)
YOLOv8	0.846	0.705	0.763	11126745	322.58
YOLOv8-C	0.873	0.746	0.811	8339369	384.62

可以得出，通过动态分配图像块到上下层路由器，添加双级路由注意力机制后模型参数量相比较原 YOLOv8 模型下降了约 25.1%，FPS 提升了约 62.1 帧/s，并且 P 值、R 值与 mAP50 均有一定提升。

4.4.4 改进 YOLOv8-improve 算法实验分析

结合改进 SELU 激活函数、加入渐近特征金字塔网络、添加双级路由注意力机制这几项改进策略，将整体改进后的算法命名为 YOLOv8-improve，选取 CCTSDB2021 数据集进行 100 轮训练，训练结果如图 4.12 所示。

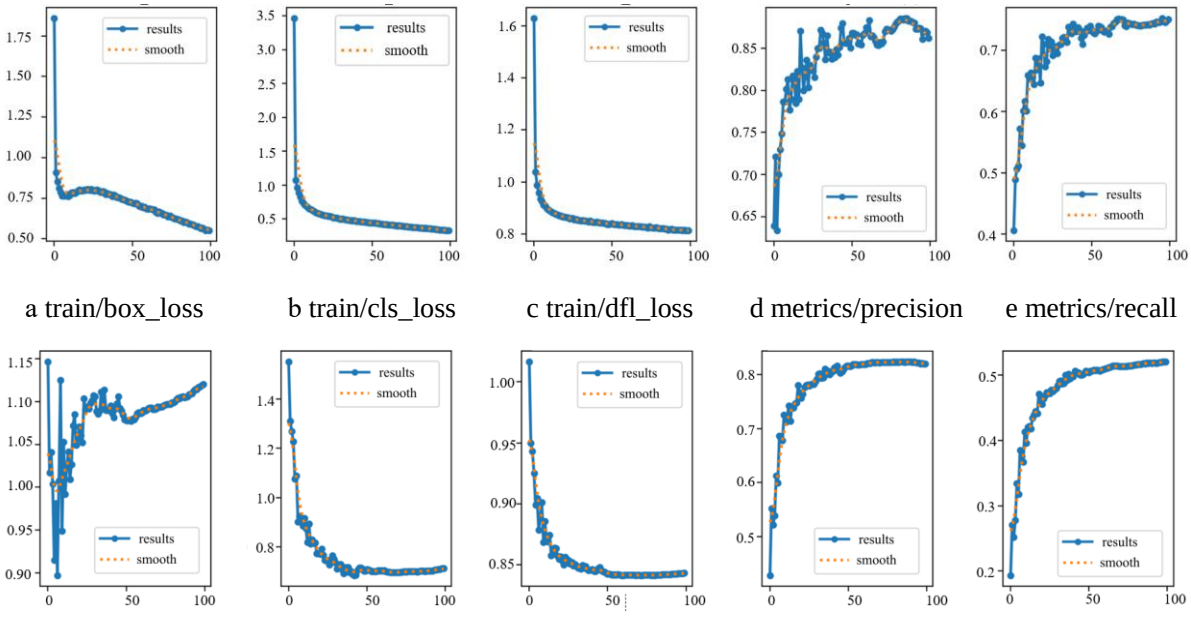


图 4.12 YOLOv8-improve 训练结果

Fig. 4.12 YOLOv8-improve training results

与原 YOLOv8 训练结果图 4.8 进行对比分析，选取 P、R、mAP50、parameters、

FPS 五项指标对比改进前后的算法性能如表 4.6 所示：

表 4.6 YOLO-improve 与原 YOLOv8 训练结果对比

Tab. 4.6 Comparison of YOLO-improve and YOLOv8 training results

	P	R	mAP50	parameters	FPS(帧/s)
YOLOv8	0.846	0.705	0.763	11126745	322.58
YOLOv8-improve	0.878	0.748	0.829	9918406	333.33

经过深入分析和对比实验，可以清晰地观察到，在经过一百轮的训练之后，改进后的 YOLOv8-improve 算法在多个关键性能指标上均取得了显著的提升。在精确度方面，该算法相较于原始版本提高了大约 3.7%，这意味着模型在识别目标时能够更准确地做出判断，减少了误检的情况。在召回率上，YOLOv8-improve 也展现了约 4.3%的提升。召回率的提高意味着模型能够更全面地识别出图像中的目标，降低了漏检的风险，这对于交通标志识别这种需要高准确度的场景来说至关重要。在 IoU 阈值为 0.5 的条件下，改进后的算法在 mAP 上提升了约 6.6%。mAP 是衡量目标识别算法性能的重要指标之一，其提升表明模型在不同类别和不同难度的目标上都能够取得更好的识别效果。

YOLOv8-improve 算法在提升识别效果的同时，还成功地降低了模型的参数量，减少了约 11%。参数量的减少有助于模型的存储和传输，在一定程度上减少过拟合风险，提高模型泛化能力，同时该算法在 FPS 方面增加了约 11 帧/s，确保了实时性。改进后的 YOLOv8-improve 与原模型经过 100 轮训练的识别精度变化曲线如图 4.13 所示。

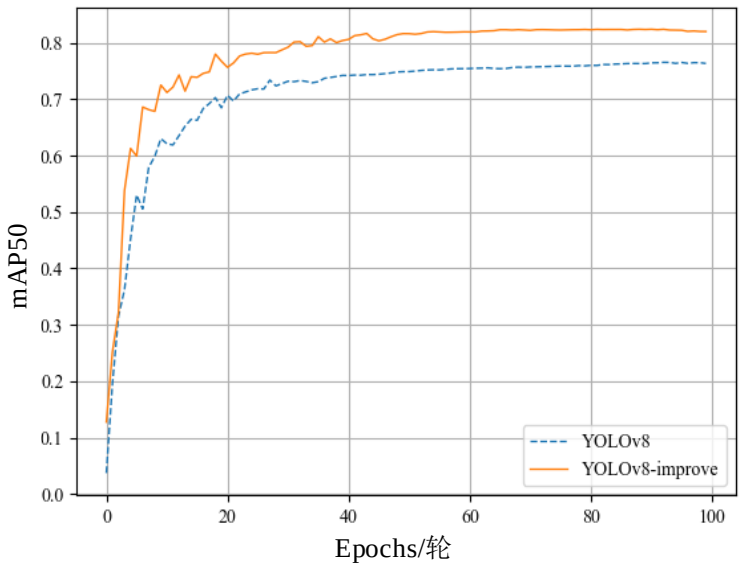


图 4.13 YOLOv8-improve 与原 YOLOv8mAP 对比曲线

Fig. 4.13 Comparison curve of mAP between YOLOv8 improve and YOLOv8

图 4.13 中，橙色实线代表 YOLOv8-improve 的 mAP 曲线，蓝色虚线代表原 YOLOv8 的 mAP 曲线。横坐标代表训练轮次，纵坐标代表 IoU 阈值为 0.5 下算法平均精度，可以看出，改进后的模型精度相较于原模型有所提升。YOLOv8-improve 算法在保持模型轻量化的同时，全面提升识别效果和性能表现，为实际应用提供了更为可靠和高效的目标检测与识别解决方案，改进前后的两种算法模型识别效果对比如图 4.14 所示。

a 改进前两个场景识别效果　　b 改进后两个场景识别效果

图 4.14 交通标志识别效果

Fig. 4.14 Traffic-sign recognition effect

在图 4.14 中，a 图中的两张示意图为改进前的原 YOLOv8 模型对于两个场景中交通标志的识别结果，可以看出改进前的模型识别到了一个交通标志；b 图中的两张示意图为改进后的 YOLOv8-improve 模型对于同样的两个场景中识别结果，可以看出改进后的模型识别到了三个交通标志。可以看出改进后的 YOLOv8-improve 模型比原 YOLOv8 模型多识别了两个目标，对于漏检的情况有所改善，在精度上的效果有所提高。改进后的 YOLOv8-improve 模型 PR 曲线如图 4.15 所示。

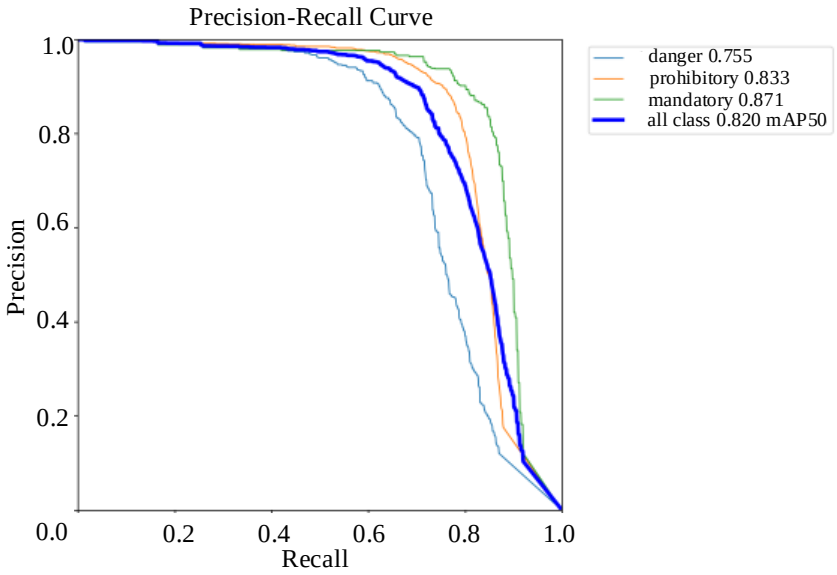


图 4.15 YOLOv8-improve 模型 PR 曲线

Fig. 4.15 PR curve of YOLOv8-improve model

图 4.15 中，蓝色、橙色及绿色细线分别代表 danger、prohibitory、mandatory 三类交通标志的 PR 曲线，蓝色粗线代表所有分类整体的 PR 曲线。

通过观察 PR 曲线的形状和变化趋势，可以评估改进后的 YOLOv8-improve 模型在不同召回率水平下的精确率表现。PR 曲线在召回率较低时具有较高的精确率，说明模型在准确识别正样本方面表现较好；PR 曲线在召回率较高时仍能保持较高的精确率，说明模型在准确识别正样本的同时具有较低的误报率。

为了更好地验证改进算法的有效性，设计在扩充后的 TT-100K 数据集上与 HOG+SVM、Faster RCNN、SSD300、YOLOv3、YOLOv5、原 YOLOv8 这几个主流算法做对比实验，结果如表 4.7 所示：

表 4.7 与主流算法训练结果对比

Tab. 4.7 Comparison of mainstream algorithm training results

算法名称	Parameters（M）	P	R	mAP50
HOG+SVM				0.632
Faster RCNN	179	0.762	0.316	0.609
SSD300	100.3	0.659	0.508	0.638
YOLOv3	235.1	0.698	0.568	0.665
YOLOv5	14.2	0.774	0.612	0.731
YOLOv8	11.1	0.783	0.646	0.757
YOLOv8-improve	9.9	0.831	0.733	0.819

由表 4.7 可以看出，改进后算法的 mAP 达到了 81.9%，高于其他主流的单阶段检测算法，对比改进前的原 YOLOv8 算法也提高了 6.2%。改进后的 YOLOv8-improve 算法在精确率、召回率相较于原 YOLOv8 算法也分别提高了 4.8%、8.7%，实验数据表明，改进后的算法优于目前主流的交通标志识别算法，具有一定的价值。

4.5 本章小结

针对交通标志识别模型存在的精度低的问题基于 YOLOv8 模型提出了改进 SELU 激活函数、引入渐近特征金字塔网络、双级路由注意力机制等改进策略，对各部分改进后的训练模型分别对比原 YOLOv8 模型进行了实验和结果分析，验证了改进的有效性。实验表明，改进后的 YOLOv8-improve 模型在 CCTSDB 数据集上测试 mAP 提升了约 6.6%，在扩增后的 TT100K 数据集上提升了约 6.2%，同时相较于主流的交通标志识别算法也有优势。

第 5 章 交通标志识别模型轻量化剪枝及嵌入式部署

为了提升交通标志识别模型的性能，前文已经设计改进策略解决现有模型识别精度低的问题，并通过实验证明改进后的模型精度有所提高。在实际的交通标志识别应用中，设备通常部署在车辆内部，这就要求识别算法能在移动设备上流畅运行。鉴于移动端嵌入式设备的硬件运算能力受限且成本有限，对算法的内存占用和运行时间提出了更为严格的要求。即在确保检测精度的同时，还需兼顾实时性能。因此，算法需要在保持高精度的同时经过一定程度的轻量化处理，以减少其内存占用和运行时间，进而使移动端设备能够实时进行交通标志目标的识别。本章将在前文设计模型 YOLOv8-improve 的基础上，对模型进行剪枝轻量化并在 RK3588 核心板上进行性能测试。

5.1 基于 Smooth-L1 正则化函数的模型剪枝

由于 CNN 在应对复杂的交通标志识别任务时，通常需要构建结构复杂、较深的网络，导致模型占用内存空间大，计算量随之剧增，不仅影响了交通标志识别的实时性能，还提高了对硬件条件的要求，使得实际应用变得困难重重。为此，本文设计了一种网络模型剪枝的方法来压缩模型的体积，以优化其性能，便于后续的嵌入式部署设计。

当前，大部分交通标志识别网络在卷积层之后都普遍引入了批规范化（Batch Normalization, BN）层，从而形成了“卷积层-BN 层-激活函数”的基础结构单元，是卷积神经网络的重要组成部分。BN 层的核心作用在于促进模型的快速收敛，并有效预防梯度消失现象的发生，从而确保网络的稳定性和性能[58]。

在神经网络的每一层，BN 层都会接收输入的数据，并对其进行标准化处理[59]。首先，它会计算当前 mini-batch 数据的均值和方差，然后将每个特征的数值减去该均值并除以该方差，使得每个特征的数值分布在一个相对稳定的范围内。这样，无论输入的数据分布如何变化，BN 层都能将其转换为均值为 0、方差为 1 的标准正态分布。然后，BN 层引入了两个可学习的参数：γ 和 β。这两个参数用于调整归一化后的特征值的范围和偏移量，从而恢复模型的表达能力，BN 层处理过程如式（5.1）至式（5.4）所示：

$$\mu_{\mathrm{i}} = \frac{1}{NHW}\sum_{n=1}^{N}\sum_{h=1}^{H}\sum_{w=1}^{W} x_{nhw} \tag{5.1}$$

$$\sigma_i^2 = \frac{1}{NHW}\sum_{n=1}^{N}\sum_{h=1}^{H}\sum_{w=1}^{W}(x_{nhw} - \mu_{\mathrm{i}})^2 \tag{5.2}$$

$$\hat{x}_{nhw} = \frac{x_{nhw} - \mu_{\mathrm{i}}}{\sqrt{\sigma_i^2 + \varepsilon}} \tag{5.3}$$

$$y_{nhw} = \gamma_i \bullet \hat{x}_{nhw} + \beta_i \tag{5.4}$$

式（5.1）至式（5.4）中，μ 和 σ 分别表示卷积层通道 i 输出特征图的均值和方差，N 表示一个批次样本个数，H 和 W 分别表示卷积层通道 i 输出特征图的高和宽，γ_i 和 β_i 表示通道 i 对应可训练参数，$\hat{x}_{nhw}$ 表示归一化特征图，y_{nhw} 表示平移缩放之后的特征图。

在常规交通标志识别模型训练过程中，接近零的权重参数 γ 在模型中相对较少[60]。训练完成后，模型的 BN 层参数大致遵循正态分布，其中中间值占比最高。因此，直接使用单一阈值来区分通道的重要性并不理想，这导致直接剪枝后难以获得高压缩率的模型。理想的情况是大部分权重参数 γ 分布在“0”附近，使得这些参数对应的通道对后续网络结构的影响极小。为了实现这一点，本文设计在原始损失函数中加入正则项，用以约束复杂度较高的参数。在交通标志识别领域普遍采用的正则化函数包括 $L1$ 正则化函数，也被叫作 Lasso 回归；以及 $L2$ 正则化函数，也被叫做 Ridge 回归。$L1$ 正则化函数以及 $L2$ 正则化函数的表达式分别如式（5.5）及式（5.6）所示：

$$L1 = |x| \tag{5.5}$$

$$L2 = x^2 \tag{5.6}$$

Smooth-L1 函数结合了 $L1$ 和 $L2$ 损失函数的优点。当预测值与真实值之间的差值较大时，它表现得像 $L1$ 损失函数，即采用绝对值误差；而当差值较小时，它表现得像 $L2$ 损失函数，即采用平方误差。Smooth-L1 正则化函数表达式如式（5.7）所示：

$$Smooth_{L_1}(x)\begin{cases} 0.5x^2 & |x|<1 \\ |x|-0.5 & |x|\geq 1 \end{cases} \tag{5.7}$$

三种正则化函数曲线对比图如图 5.1 所示。

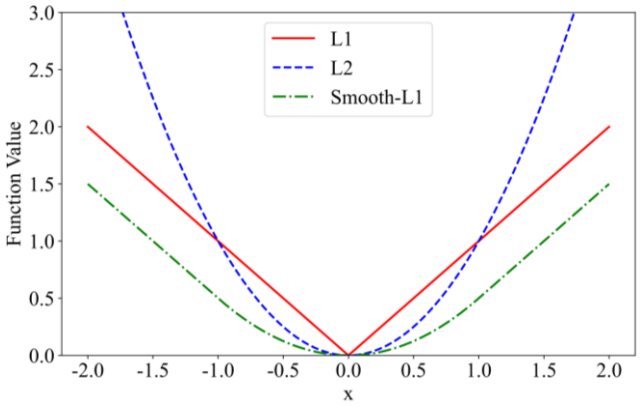


图 5.1 L1 L2 和 Smooth-L1 正则化函数曲线

Fig. 5.1 Regularization function curve of L1 L2 and Smooth-L1

由图 5.1 可以看出，这种设计使得 Smooth-L1 损失函数在“0”附近梯度平滑，而远离“0”的时候梯度稳定对离群点相对鲁棒，不像 $L2$ 损失函数那样对离群点非常敏感，同时又能保持对误差较小的样本的平滑处理，在改进后 YOLOv8-improve 模型基础上加入 Smooth-L1 正则化函数来促进模型权重的稀疏性，可规避在非光滑点处运用次梯度方法求解极值，有利于对模型进行轻量化剪枝操作，降低模型的复杂度。

对权重参数 γ 采用 Smooth-L1 正则化函数后，稀疏训练的损失函数设计如式（5.8）所示：

$$Loss_{total} = Loss_{YOLOv8-improve} + \lambda \sum_{\gamma \in \Gamma} Smooth_{L_1}(\gamma) \tag{5.8}$$

式（5.8）中，$Loss_{YOLOv8-improve}$ 是原 YOLOv8-improve 模型的损失函数，$\lambda \sum_{\gamma \in \Gamma} Smooth_{L_1}(\gamma)$ 是 γ 的正则项，λ 是手动设定的常数，是正则化函数 Smooth-L1 的惩罚项，作为原始损失函数和引入正则项之间的平衡参数。经过稀疏训练后，可以获得更多权重参数 γ 接近于零的模型。

通道剪枝作为一种精细化的剪枝方法，主要聚焦于卷积层的特定通道。在神经网络的基本构成中，“卷积-BN-激活”结构扮演着核心角色，而 BN 层的参数则能够广泛应用于众多网络结构中。

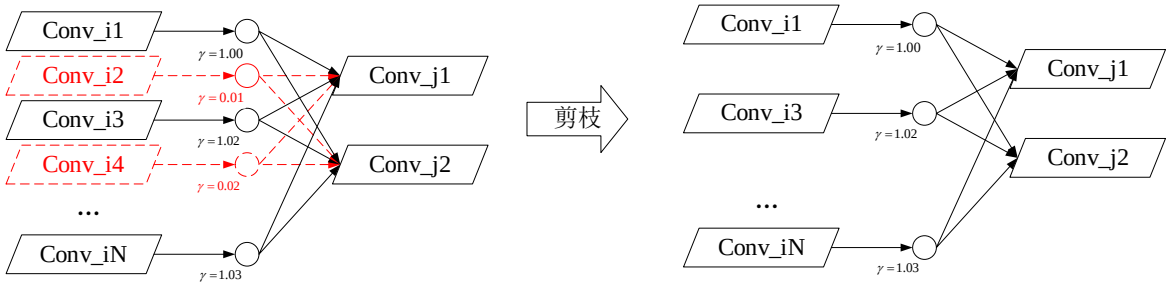


图 5.2 通道剪枝示意图

Fig. 5.2 Schematic diagram of channel pruning

如图 5.2 所示，通过实施剪枝策略能够有效地减少整体网络的参数数量和权重冗余，从而实现网络的轻量化效果。在训练出适当的稀疏模型后，大部分 BN 层的权重参数 γ 会趋近于“0”。为了进一步优化网络结构，为卷积层通道设定一个阈值，当 γ 参数小于该阈值时，对应的通道会被剪除。

通道剪枝的实际阈值通过局部阈值和全局阈值共同决定，将 BN 层中的所有权重参数 γ 按 $\gamma_1, \gamma_2, ..., \gamma_n$ 进行排序，n 为该层的通道数，定义如式（5.9）所示：

$$\eta = TP / TC \times 100\% \tag{5.9}$$

式（5.9）中，*TP* 代表剪枝通道数量，*TC* 代表稀疏模型通道数量。

$\hat{\gamma}$ 是全局值，该阈值通过剪枝率乘以排序结果得到。同时，在每一个卷积层中给定一个局部值 π 以防止过度剪枝，π 是所有 BN 层中最大 γ 值中的最小值，最终剪枝掉比例因子 γ 小于 π 和 $\hat{\gamma}$ 的通道，模型通道剪枝过程如图 5.3 所示。

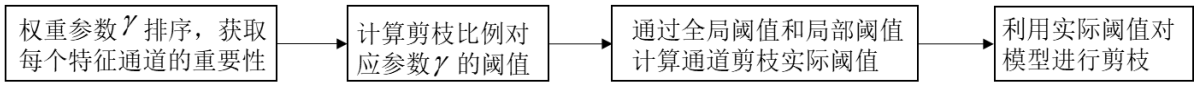


图 5.3 通道剪枝过程

Fig. 5.3 Channel pruning process

模型剪枝后结构更加紧凑、体积更小、模型参数与计算量大幅度减少，在实际推理时所需内存也会减少，更节省硬件资源，便于后续嵌入式部署设计。

5.2 实验平台与部署流程

嵌入式部署所采用的 RK3588 核心板是一款基于瑞芯微 RK3588 芯片设计的高性能计算模块。RK3588 是瑞芯微推出的新一代旗舰高端处理器，采用 8nm 工艺设计，搭载四核 A76 和四核 A55 的八核 CPU，内置 6T 计算能力的 NPU，能够能效平衡的同时处理复杂的计算任务。在内存和存储方面，BPI-RK3588 核心板支持高达 8GB 的内存和 32GB 的 eMMC 存储，提供了充足的数据存储空间。

使用改进后的 YOLOv8-improve 模型算法进行 100 轮训练并得到了权重文件，由于 RK3588 只支持.rknn 文件，不支持 YOLO 模型训练出的.pt 文件，所以需要将.pt 文件先转换成.onnx 文件再转换为.rknn 文件。首先安装必要的 onnx 库再通过 YOLOv8 官方提供的 export.py 文件将 pt 格式转换为 onnx 格式。在得到 onnx 格式文件后，从 Rockchip 的 GitHub 仓库下载 RKNN Toolkit 的源代码，并在 anaconda 的 rknn 环境下安装 RKNN Toolkit 的依赖包。使用 RKNN Toolkit 的 API 加载 ONNX 模型文件。创建一个 RKNN 对象并调用加载模型的函数，根据 YOLOv8 模型的结构配置输入和输出的节点，包括指定输入数据的形状、数据类型和归一化参数等。使用 RKNN Toolkit 的 API 构建 RKNN 模型，将加载的 ONNX 模型转换为 RKNN 的中间表示形式。将构建好的 RKNN 模型导出为一个.rknn 格式文件，换后的文件可以搭载 RK3588 芯片的硬件设备上加载和运行，Pytorch 模型转化为 rknn 模型的具体步骤如图 5.4 所示。

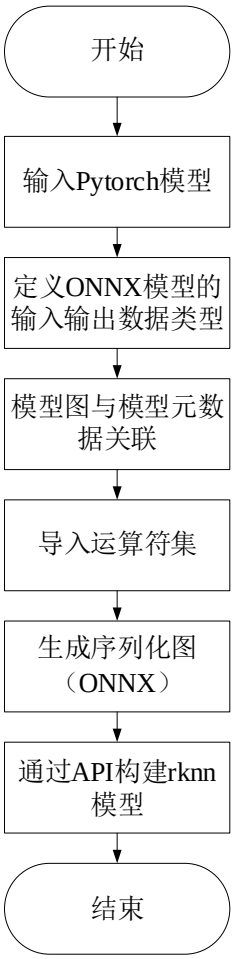


图 5.4 Pytorch 模型转换 rknn 步骤

Fig. 5.4 Pytorch model conversion rknn process

文件格式转换的界面如图 5.5 所示，导入 ONNX 模型并在目标框架中加载 ONNX 模型的定义和参数。

```
D RKNN: [08:52:55.594] 86  ConvExSwish model.23.cv2.conv.bias        INT32     (128)         | 0x0017ab00 0x0017af00 0x00000400
D RKNN: [08:52:55.594] 87  ConvExSwish model.23.m.0.cv1.conv.weight  INT8      (128,128,1,1) | 0x0018b700 0x0018f700 0x00004000
D RKNN: [08:52:55.594] 87  ConvExSwish model.23.m.0.cv1.conv.bias    INT32     (128)         | 0x0018f700 0x0018fb00 0x00000400
D RKNN: [08:52:55.594] 88  ConvExSwish model.23.m.0.cv2.conv.weight  INT8      (128,128,3,3) | 0x0018fb00 0x001b3b00 0x00024000
D RKNN: [08:52:55.594] 88  ConvExSwish model.23.m.0.cv2.conv.bias    INT32     (128)         | 0x001b3b00 0x001b3f00 0x00000400
D RKNN: [08:52:55.594] 90  ConvExSwish model.23.cv3.conv.weight      INT8      (256,256,1,1) | 0x0017af00 0x0018af00 0x00010000
D RKNN: [08:52:55.594] 90  ConvExSwish model.23.cv3.conv.bias        INT32     (256)         | 0x0018af00 0x0018b700 0x00000800
D RKNN: [08:52:55.594] 91  ConvSigmoid model.24.m.2.weight           INT8      (255,256,1,1) | 0x001c0f00 0x001d0f00 0x00010000
D RKNN: [08:52:55.594] 91  ConvSigmoid model.24.m.2.bias             INT32     (255)         | 0x001d0f00 0x001d1700 0x00000800
D RKNN: [08:52:55.594] ------------------------------------------------------------------------+------------------------------
D RKNN: [08:52:55.596] ----------------------------------------
D RKNN: [08:52:55.596] Total Internal Memory Size: 7600KB
D RKNN: [08:52:55.596] Total Weight Memory Size: 1866.5KB
D RKNN: [08:52:55.596] ----------------------------------------
D RKNN: [08:52:55.596] <<<<<<<< end: rknn::RKNNMemStatisticsPass
I rknn building done.
done
--> Export rknn model
done
--> Init runtime environment
W init_runtime: Target is None, use simulator!
done
```

图 5.5 模型 rknn 格式转换

Fig. 5.5 Model rknn format conversion

在部署模型之前，需要在 RK3588 上安装操作系统并配置嵌入式部署所需的开发环境。在准备好所需的环境后将 rknn 文件放入 rknpu 指定文件夹中使用其写好的 c++ 文件编译得到一个可执行程序，再将该可执行程序移植到开发板端。

交通标志识别系统整体流程如图 5.6 所示，首先导入训练好的交通标志识别模型，然后根据是否实时识别选择实时模式或本地模式。

在实时模式下，开启摄像头，实时识别交通标志图像并显示分类结果；在本地模式下，从 SD 卡读取交通标志图像或者视频，用改进后的模型进行识别分类，完成检测过程。

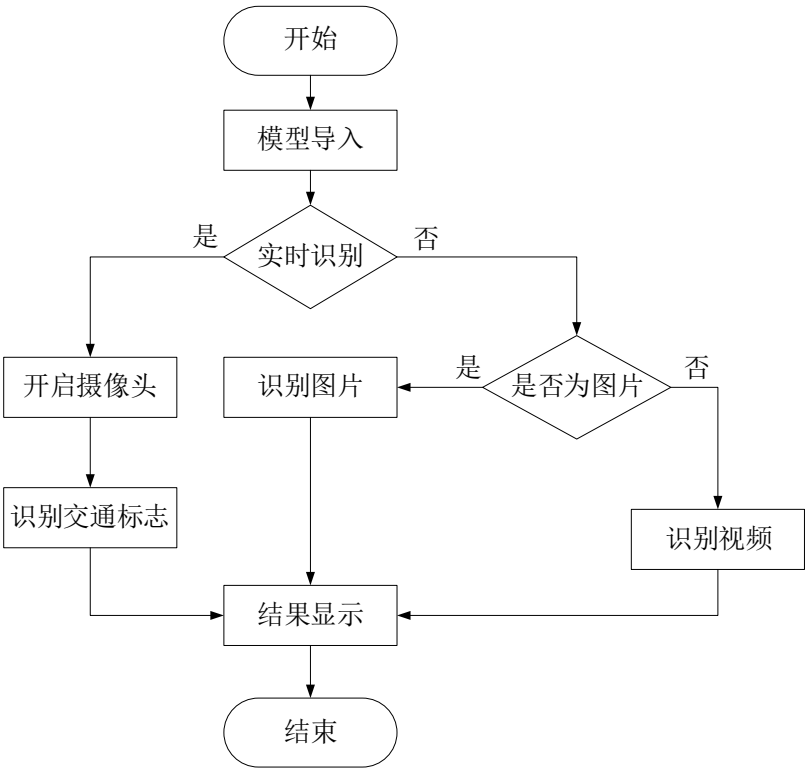


图 5.6 交通标志识别系统流程图

Fig. 5.6 Flowchart of traffic sign recognition system

识别结果如图 5.7 所示，从图中可以看出左侧为模型推理过程界面，右侧为识别结果。改进后的 YOLOv8-improve 模型识别到画面中的交通标志并用检测框圈住，准确分类为禁止标志。

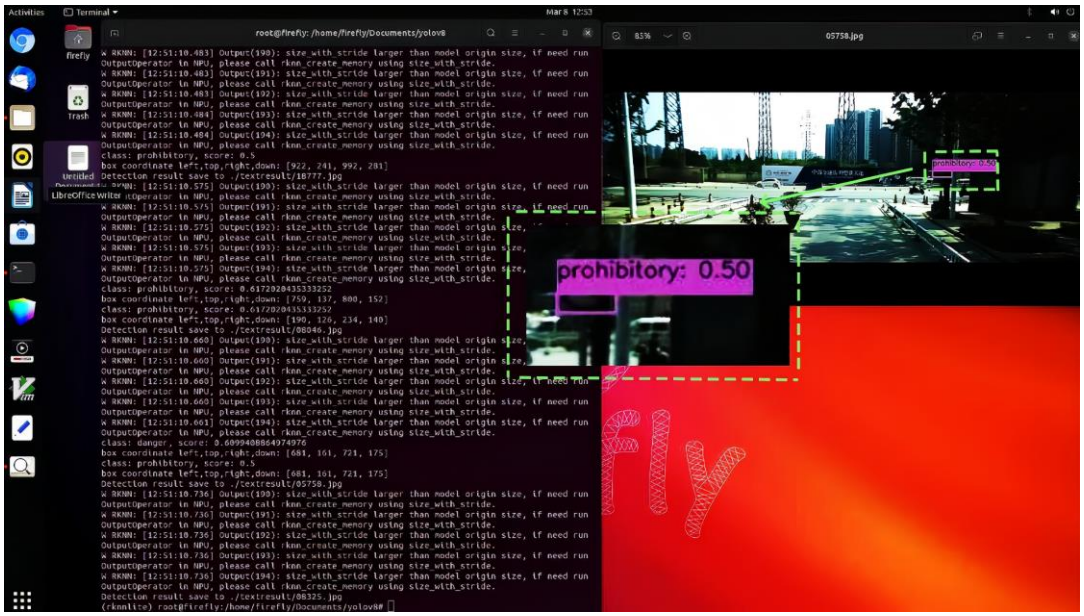


图 5.7 交通标志分类识别结果

Fig. 5.7 Classification result of traffic-sign

5.3 实验结果验证与分析

为确定最优剪枝率η以及惩罚项λ，分别采用惩罚项为 0.001、0.005、0.01 的稀疏模型进行剪枝实验，同时对这些模型分别设置了 20%、40%和 60%的剪枝率，以探究不同参数对模型性能的影响，实验结果如表 5.1 所示：

表 5.1 稀疏惩罚项与剪枝率η对 mAP0.5 的影响

Tab. 5.1 Effect of sparse penalty terms and pruning rate on mAP0.5

稀疏惩罚项	剪枝率η(%)	实际剪枝率(%)	剪枝前 mAP0.5(%)	剪枝后 mAP0.5(%)
λ=0	0	0	82.9	82.9
	20	18.6	82.7	82.7
λ=0.001	40	37.4	82.7	82.5
	60	56.1	82.7	82.4
	20	18.5	73.6	73.3
λ=0.005	40	37.4	73.5	73.2
	60	55.9	73.5	72.9
	20	18.5	35.5	34.9
λ=0.01	40	37.4	35.5	34.8
	60	55.9	35.4	34.8

表 5.1 中，实际剪枝率是在给定剪枝率下，加上局部阈值的影响后实际对模型的剪枝率，略小于设定剪枝率。可以看出，设计稀疏惩罚项λ越大的时候，模型在稀疏化剪

枝之后的 mAP0.5 值损失越大，这是由于稀疏训练使得 BN 层的γ值变化剧烈，稀疏模型出现了欠拟合的情况，导致 mAP0.5 值大幅降低。可以看出当惩罚项λ值取 0.001 时，剪枝后的模型精度基本没有损失，因此采用$\lambda=0.001$作为最佳参数。

在确定惩罚项之后，还需要对剪枝率进行实验，得到精度损失较小的轻量化模型。通过设置$\lambda=0.001$作为惩罚项后，将剪枝率η分别设置为从 10%、20%递增到 95%，训练结果如表 5.2 所示：

表 5.2 剪枝率η对 mAP0.5 的影响

Tab. 5.2 Effect of pruning rate on mAP0.5

剪枝率η(%)	实际剪枝率(%)	剪枝前 mAP0.5(%)	剪枝后 mAP0.5(%)
0	0	82.9	82.9
10	9.5	82.7	82.7
20	18.6	82.7	82.7
30	28.3	82.7	82.7
40	37.4	82.7	82.7
50	46.8	82.7	82.7
60	56.1	82.7	82.7
70	65.6	82.7	82.7
80	73.9	82.7	82.5
85	79.3	82.7	81.4
90	84.1	82.7	22.1
95	88.6	82.7	0

从表 5.2 中可以看出，不同剪枝率的影响下 mAP0.5 值会降低，并且当剪枝率η设置为 80 后，剪枝后的模型 mAP0.5 值大幅度下降并逐渐降至 0，这是因为剪枝掉的部分数据在稀疏训练中起到一定作用，因此，将剪枝率η设置为 80%可以实现最佳轻量化效果。

下一步检测模型实时识别性能，首先，需要通过 RKNN SDK 提供的 API 来初始化摄像头，调用相应的函数来打开摄像头设备，并设置其参数，如分辨率、帧率等。在初始化过程中，还需要确保摄像头正常工作，并能够捕获到实时的视频流。

接下来，加载预先转换好的 RKNN 模型。加载模型时，需要指定模型文件的路径，并进行必要的配置以确保模型与硬件平台的兼容性。在一个循环中，从摄像头捕获帧，将这些帧作为输入传递给 YOLOv8-improve 模型，然后获取并处理模型的输出。

为了进行实时检测识别，需要准备输入和输出缓冲区。输入缓冲区负责存放从摄像头捕获的视频帧，这些帧将作为推理数据传递给模型。而输出缓冲区则用于存储模

型的推理结果，包括检测到的对象的类别、置信度和边界框等信息。

在一个持续运行的循环中，将不断地从摄像头捕获视频帧。每捕获一帧，将帧数据转换为模型所需的格式并传递给 YOLOv8-improve 模型进行推理。等待模型完成推理，并获取其输出。调试好后将摄像头打开，对交通标志图片进行实时测试，识别效果如图 5.8 所示。

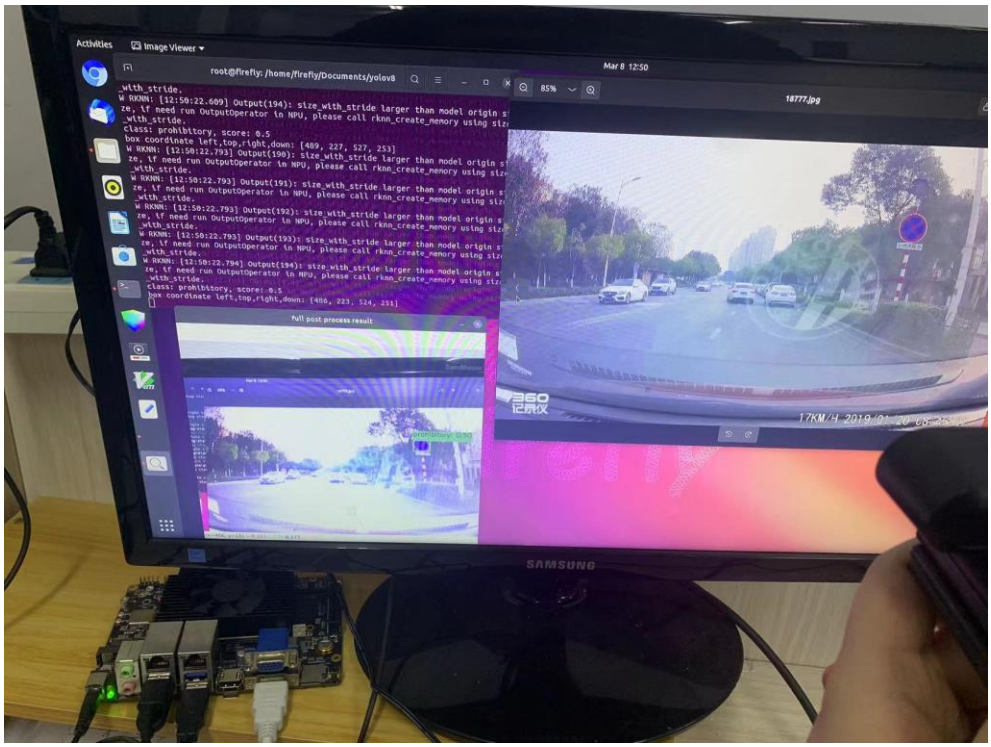


图 5.8 实时分类识别结果

Fig. 5.8 Real-time classification result

图 5.8 中所示右上角图片为待识别交通标志示例图，左下角所示图片为系统识别后的分类结果可以看出模型能实时对交通标志图片进行识别，达到预期要求。

基于 Smooth-L1 正则化函数的模型剪枝前后的 RK3588 核心板训练结果对比如表 5.3 所示：

表 5.3 基于 Smooth-L1 正则化函数的模型剪枝前后训练结果对比

Tab. 5.3 Comparison of training results after model pruning based on Smooth-L1 regularization function

	mAP50	parameters	FPS（帧/s）
YOLOv8-improve	0.829	9918406	19.4
YOLOv8-Smooth-L1	0.825	3321456	27.9

从表 5.3 中可以看出，剪枝后的模型在保持精度损失不大的情况下参数量降低了66.51%，FPS 提高了 43.8%，达到了 27.9 帧/s，提高了识别速度。

5.4 本章小结

为满足实际应用需求，基于 Smooth-L1 正则化函数设计模型剪枝方法，设计实验验证剪枝方法的有效性。将剪枝后训练的模型文件转换格式后部署在 RK3588 核心板上进行实验测试，并调用摄像头进行实时识别。实验结果表明，模型参数量降低了66.51%，实时识别帧数可达 27.9 帧/s，满足实时性要求。

第 6 章 结论

6.1 总结

本文结合了深度学习技术，以交通标志作为研究对象，针对现有算法在交通标志识别领域内应用存在的精度低及速度差的问题进行改进，设计基于改进 YOLOv8 的交通标志识别算法并对模型轻量化剪枝后部署在嵌入式系统上测试，主要围绕课题背景相关的数据集建立及扩充、交通标志识别模型设计、轻量化剪枝以及嵌入式部署等方面展开研究。

数据集建立及扩充方面，针对现有交通标志数据集数据量少且各类别数据不均衡导致模型识别精度低的问题，筛选出实例数大于 50 的 45 个类别作为基础，基于改进 DCGAN 网络扩充交通标志数据集，并通过几何变换、颜色变换、噪声添加等数据增强方法进行后处理，扩充后的数据集数据量增加了 59.34%，且不存在实例数偏少的类别，为交通标志识别模型训练提供了更好的数据基础，有助于模型泛化性的提升。

交通标志识别模型设计方面，针对交通标志识别模型在恶劣条件下精度低、实时性差等问题，基于 YOLOv8 模型改进了 SELU 激活函数，提出一种渐近特征金字塔网络的同时添加双级路由注意力机制，不仅提升了模型对交通标志的识别精度，还有效降低了模型的参数量与运算量。实验证明，改进后的算法对比原 YOLOv8 算法在保持实时性基本不变的情况下多项指标上均有提升，在 CCTSDB 数据集上测试 mAP 提升了约 6.6%，在扩增后的 TT100K 数据集上提升了约 6.2%，同时相较于主流的交通标志识别算法也有优势。

轻量化剪枝及嵌入式部署方面，针对交通标志识别模型占用内存空间庞大、对硬件条件要求高导致的难以在车辆内部署的问题，基于 Smooth-L1 正则化函数设计了一种模型剪枝方法，有效地减少整体网络的参数数量和权重冗余，从而实现网络的轻量化效果。选择 RK3588 核心板作为部署平台，实验证明，剪枝后的模型在保持精度基本不损失的情况下参数量降低了 66.51%，FPS 提高了 43.8%，达到了 27.9 帧/s，满足实时性要求。

综上所述，改进后的交通标志识别算法提升了识别精度与实时性，同时采用模型剪枝技术实现轻量化，使得算法在保持高精度与实时性的同时，大幅减少内存占用并提高计算效率，便于在嵌入式系统等资源受限环境中高效部署，具有较高的实际应用价值。

6.2 展望

以 YOLOv8 算法作为基础，改进后的算法在交通标志识别领域内有比较理想的实验效果，但是由于研究时间有限、个人能力存在不足等问题，仍有一些可以优化的问题，下一步的工作可以从以下几个方面展开：

目前的工作主要集中在基于视觉的交通标志识别，然而在实际应用中，其他传感器如雷达、激光雷达和超声波等也可以提供有用的信息。后续工作可以探索如何有效地融合这些多模态数据，以提高交通标志识别的准确性和鲁棒性。

当前的模型主要在特定的数据集上进行训练和测试，为了增强模型的泛化能力，可以探索如何在不同的数据集之间进行迁移学习。

参考文献

[1] 夏溦．自动驾驶汽车技术最新发展[J]．新经济导刊，2015，1(7)：24-27．

[2] 李文举，张干，崔柳，等. 基于改进的 YOLO 算法的交通标志识别[J]. 计算机仿真，2023，40(1)：149-155．

[3] Smith J, Zhang L. A traditional approach to object recognition with image preprocessing, region selection, feature extraction, and classification[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(9): 1750-1763.

[4] 冯爱棋，吴小俊，徐天阳．融合注意力机制和上下文信息的实时交通标志检测算法[J]．计算机科学与探索，2023，17(11)：2676-2688．

[5] Redmon J, Divvala S, Girshick R, et al. You only look once: unified, real-time object detection[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas: IEEE, 2016: 779-788.

[6] 汪宋，费树岷．SSD 目标检测算法的研究与改进[J]．工业控制计算机，2019，32(04)：103-105．

[7] 梁正友，耿经邦，孙宇. 基于改进 SSD 模型的交通标志检测算法[J]. 现代计算机，2021，27(32)：54-58+84．

[8] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus: IEEE, 2014: 580-587.

[9] Lin Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2017, 66(99): 2999-3007.

[10] Smith R. Techniques for traffic sign recognition[J]. Image and Vision Computing, 1975, 2(3): 123-134.

[11] 梁文昭，蔡念，郭文婷，等．交通标志识别新方法[C]．中国图象图形学学会．第十五届全国图象图形学学术会议论文集．广州，2010：225-228.

[12] 杨守建，陈恳．基于 Hopfield 神经网络的交通标志识别[J]．计算机工程与科学，2011，33(08)：132-137．

[13] Smirnov A, Timoshenko M, Andrianov N. Comparison of regularization methods for imageNet classification with deep convolutional neural networks[J]. AASRI Procedia, 2014, 6(1): 89-94.

[14] Cai Z, Gu M. Traffic sign recognition algorithm based on shape signature and dual-tree complex wavelet transform[J]. Journal of Central South University, 2013, 20(2): 433-439.

[15] Karis M, Ali M, Safei J. Hidden nodes of neural network: Useful application in traffic sign recognition[C]. Proc of the IEEE International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA), Kuala Lumpur: IEEE, 2014: 1-4.

[16] Aghdam H, Heravi J, Puig D. Toward an optimal convolutional neural network for traffic sign recognition[C]. Eighth International Conference on Machine Vision (ICMV 2015), Sousse: SPIE, 2015: 108-112.

[17] 钟晓明，余贵珍，马亚龙，等．基于快速区域卷积神经网络的交通标志识别算法研究[C]．2016 中国汽车工程学会年会，上海：中国汽车工程学会，2016：2033-2036．

[18] 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, 2016, 39(6): 1137-1149.

[19] 张邯，罗晓曙，袁荣尚．基于优化的卷积神经网络在交通标志识别中的应用[J]．现代电子技术，2018，41(21)：132-136．

[20] Kong S, Park J, Lee S, et al. Lightweight traffic sign recognition algorithm based on cascaded CNN[C].2019 19th International Conference on Control, Automation and Systems (ICCAS), Singapore: IEEE, 2019: 506-509.

[21] 刘学平，李玓乾，刘励，等．自适应边缘优化的改进 YOLOV3 目标识别算法[J]．微电子学与计算机，2019，36(7)：59-64．

[22] Martinez-Alpiste I, Golcarenarenji G, Wang Q, et al. A dynamic discarding technique to increase speed and preserve accuracy for YOLOv3[J]. Neural Computing and Applications, 2021, 33(16): 9961-9973.

[23] 杜婷婷，钟国韵，江金懋，等．基于 Darknet23 和特征融合的交通标志检测方法[J]．电子技术应用，2023，49(1)：14-19．

[24] Yang Z, Zheng Y, Shao J, et al. Improved YOLOv4 based on dilated coordinate attention for object detection[J]. Multi-media Tools and Applications, 2024, 83(19): 56261-56273.

[25] 杨晓玲，江伟欣，袁浩然．基于 yolov5 的交通标志识别检测[J]．信息技术与信息化，2021(4)：28-30．

[26] Baghbanbashi M, Raji M, Ghavami B. Quantizing YOLOv7: A comprehensive study[C]. 2023 28th

International Computer Conference, Computer Society of Iran (CSICC), Tehran: IEEE, 2023: 1-5.

[27] 张传伟，李妞妞，岳向阳，等．基于改进 YOLOv2 算法的交通标志检测[J]．计算机系统应用，2020，29(6)：155-162．

[28] 徐迎春．基于 YOLOv3 改进的交通标志识别算法[J]．数字技术与应用，2021，39(01)：108-111+116．

[29] 周钰如，厉丹，肖辰禹，等．基于 YOLOv5 的交通标志识别系统[J]．电脑知识与技术：学术版，2022，18(19)：97-99．

[30] 彭瑾，桑正霄，李木易．一种基于 YOLOv5s 的交通标志检测算法[J]．自动化技术与应用，2023，42(9)：53-57．

[31] 蔡军，邱会然，谭静，等．多尺度上下文融合的交通标志识别算法研究[J]．无线电工程，2022，52(1)：114-120．

[32] Krizhevsky A, Sutskever I, Hinton E. ImageNet classification with deep convolutional neural networks[J]. Communications of the ACM, 2017, 60(6): 84-90.

[33] 田晟，宋霖．基于 CNN 和 Bagging 集成的交通标志识别[J]．广西师范大学学报，2022，40(4)：35-46．

[34] Kannojia P, Jaiswal G. Effects of varying resolution on performance of CNN based image classification: An experimental study[J]. International Journal of Computational Science and Engineering, 2018, 6(9): 451-456.

[35] Nair V, Hinton G. Rectified linear units improve restricted boltzmann machines[C]. Proceedings of the 27th International Conference on Machine Learning (ICML-10), Madison: IEEE, 2010: 807-814.

[36] Alkhouly A, Mohammed A, Hefny A. Improving the performance of deep neural networks using two proposed activation functions[J]. IEEE Access, 2021, 9: 82249-82271.

[37] 周飞燕，金林鹏，董军．卷积神经网络研究综述[J]．计算机学报，2017，40(6)：1229-1251．

[38] 高强，靳其兵，程勇．基于卷积神经网络探讨深度学习算法与应用[J]．电脑知识与技术，2015，11(13)：169-170．

[39] Chorowski J, Zurada J M. Learning understandable neural networks with nonnegative weight constraints[J]. IEEE Transactions on Neural Networks and Learning Systems, 2014, 26(1): 62-69.

[40] 包志强，赵志超，王宇霆．快速的卷积神经网络算法及应用[J]．计算机工程与设计，2020，41(8)：2213-2217．

[41] Mohapatra K, Mohanty N. Detection of atrial fibrillation from cardiac signal using convolutional neural network[J]. International Journal of Innovative Computing and Applications, 2022, 13(3): 172-179.

[42] Anantrasirichai N, Biggs J, Albino F, et al. The application of convolutional neural networks to detect slow sustained deformation in InSAR time series[J]. Geophysical Research Letters, 2019, 46(21): 11850-11858.

[43] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[44] Girshick R, Donahue J, Darrell T, et al. Region-based convolutional networks for accurate object detection and segmentation[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 38(1): 142-158.

[45] Van N, Postma E. Learning scale-variant and scale-invariant features for deep image classification[J]. Pattern Recognition, 2017, 61: 583-592.

[46] 顾曦龙，宫宁生，胡乾生．基于 YOLOv3 与改进 VGGNet 的车辆多标签实时识别算法[J]．计算机科学，2022，49(S2)：542-548．

[47] Howard G, Zhu M, Chen B, et al. MobileNets: Efficient convolutional neural networks for mobile vision applications[C]. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Hawaii: IEEE, 2017: 36-42.

[48] 朱宁可，葛青，王翰文，等. 基于 Yolov5-MGC 的实时交通标志检测[J]. 激光与光电子学进展，2024，61(12)：113-116．

[49] Goodfellow J, Pouget J, Mirza M, et al. Generative adversarial networks[C]. Proceedings of the 27th International Conference on Neural Information Processing Systems, Cambridge: MIT Press, 2014: 2672-2680.

[50] Li W, Yu Z. A lightweight convolutional neural network flame detection algorithm[C]. 2021 IEEE 11th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing: IEEE, 2021: 83-86.

[51] Real E, Aggarwal A, Huang Y, et al. Regularized evolution for image classifier architecture search[C]. Proceedings of the Aaai Conference on Artificial Intelligence, Hawaii: AAAI, 2019: 4780-4789.

[52] Shao J, Hu K, Wang C, et al. Is normalization indispensable for training deep neural network[J]. Advances in Neural Information Processing Systems, 2020, 33: 13434-13444.

[53] Lin Y, Dollar P, Girshick R, et al. Feature pyramid networks for object detection[C]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu: IEEE, 2017: 936-944.

[54] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas: IEEE, 2016: 770-778.

[55] Qiu M, Huang L, Tang H. ASFF-YOLOv5: Multielement detection method for road traffic in UAV images based on multiscale feature fusion[J]. Remote Sensing, 2022, 14(14): 3498-3507.

[56] 晏文靖，戴建华．基于改进 YOLOv3 的交通标志识别[J]．武汉工程职业技术学院学报，2023，35(1)：31-35．

[57] 李翔宇，王倩影. 基于改进 YOLOv5 的复杂背景下交通标志识别研究[J]. 现代信息科技，2023，7(10)：30-32+36．

[58] 何源宏，姜晶菲，许金伟．注意力机制量化剪枝优化方法[J]．国防科技大学学报，2024，46(1)：113-120．

[59] 宋允飞．基于模型剪枝的深度神经网络分级授权方法的实现[J]．现代信息科技，2024，8(8)：128-132+137．

[60] 肖志良，汪丽娟，郑雁予．基于模型剪枝的物联网识别方法研究[J]．产品可靠性报告，2024，(3)：102-104．