

**本科生毕业论文（设计）**

****

**题 目 基于Yolo和LPRNet的车辆信息识别系统**

**学 院 计算机学院**

**专 业 计算机科学与技术**

**学生姓名 陈家瑞**

**学 号 2020141460290 年级 2020级**

**指导教师 辛卫**

**教务处制表**

**2024年 4 月 18 日**

**基于Yolo和LPRNet的车辆信息识别系统**

专业：计算机科学与技术

学生：陈家瑞 指导老师：辛卫

**[摘要]** 本项目旨在利用神经网络技术和计算机视觉技术实现对车辆的自动识别和信息提取。主要功能包括车辆和车牌检测与识别，以及车辆信息的统一存储。车辆检测与识别利用BIT-Vehicle Dataset数据集训练YOLOv5模型，对车辆视频或图像进行检测，以实现车辆识别。检测到车辆后，将车辆照片截取并保存，方便后续导入车牌检测与识别模型。车牌检测与识别使用CCPD Dataset数据集训练YOLOv5模型，对截取的车辆照片进行车牌检测，并利用LPRNET网络识别车牌号码。车牌颜色和车身颜色直接利用OpenCV通过划分颜色区间进行整体识别。最后，将车牌号码作为车辆信息的唯一标识，存储车辆种类、车身颜色、车牌号码、车牌颜色等信息。为用户提供便利，项目搭建了网页端平台，根据数据库存储的车辆信息搭建展示仓，包括车辆存储数量、网页点击率、车辆识别正确率等数据展示。网页端平台还提供识别图片和视频车辆的功能，用户可直接导入需要识别的图片和视频进行识别，并选择是否将识别的车辆信息存储到系统数据库中。

**[主题词]** 车辆识别 信息存储 神经网络

Vehicle information identification system based on

Yolo and LPRNet

Major: Computer Science and Technology

**Student:** Chen Jiarui **Adviser:** Xin Wei

**[Abstract]** This project aims to use neural network technology and computer vision technology to realize automatic identification and information extraction of vehicles. The main functions include vehicle and license plate detection and recognition, as well as unified storage of vehicle information. Vehicle detection and identification uses the BIT-Vehicle Dataset to train the YOLOv5 model to detect vehicle videos or images to achieve vehicle identification. After the vehicle is detected, the vehicle photo is intercepted and saved to facilitate subsequent import of the license plate detection and recognition model. License plate detection and recognition uses the CCPD Dataset data set to train the YOLOv5 model, detects the license plate on the intercepted vehicle photos, and uses the LPRNET network to identify the license plate number. The license plate color and car body color directly use OpenCV to perform overall recognition by dividing the color interval. Finally, the license plate number is used as the unique identifier of vehicle information, and information such as vehicle type, body color, license plate number, license plate color, etc. is stored. To provide convenience to users, the project has built a web platform and built a display warehouse based on the vehicle information stored in the database, including data display such as the number of vehicles stored, web page click rate, and vehicle identification accuracy. The web platform also provides the function of identifying vehicles in pictures and videos. Users can directly import the pictures and videos that need to be identified for identification, and choose whether to store the identified vehicle information in the system database.

**[Key Words]** Vehicle recognition information storage neural network

**[目录](#_Toc154675103)**

[摘要 Ⅰ](#_Toc23862)

[Abstract Ⅱ](#_Toc23862)

[目录 Ⅲ](#_Toc23862)

[第一章 引言 1](#_Toc23862)

[1.1. 研究背景 1](#_Toc5932)

[1.2. 研究内容 1](#_Toc5932)

[1.2.1. 目标检测 1](#_Toc1895)

[1.2.2. 图片数字识别 1](#_Toc1895)

[1.2.3. 可视化界面的搭建与信息存储 1](#_Toc1895)

[1.3. 国内外研究进展 1](#_Toc5932)

[第二章 背景技术介绍 1](#_Toc23862)

[2.1. YOLOv5 1](#_Toc5932)

[2.2. LPRNET 1](#_Toc5932)

[2.3. NodeJS 1](#_Toc5932)

[2.4. Vue＆SpringBoot 1](#_Toc5932)

[第三章 系统分析与设计 1](#_Toc23862)

[3.1. 系统需求分析与建模 1](#_Toc5932)

[3.1.1. 目标检测模型 1](#_Toc1895)

[3.1.2. 图片数字识别模型 1](#_Toc1895)

[3.1.3. 可视化界面的搭建与信息存储 1](#_Toc1895)

[第四章 系统实现 1](#_Toc23862)

[5.1. 实现环境与工具的简要说明 1](#_Toc5932)

[5.2. 主要程序模块实现 1](#_Toc5932)

[5.2.1. 目标检测模型 1](#_Toc1895)

[5.2.1. 图片数字识别模型 1](#_Toc1895)

[5.2.1. 可视化界面的搭建与信息存储 1](#_Toc1895)

[第五章 实验结果与分析 1](#_Toc23862)

[1.1. 研究背景 1](#_Toc5932)

[1.2. 研究内容 1](#_Toc5932)

[1.2.1. 目标检测模型 1](#_Toc1895)

[1.2.1. 图片数字识别模型 1](#_Toc1895)

[1.2.1. 可视化界面的搭建与信息存储 1](#_Toc1895)

[第六章 讨论 1](#_Toc23862)

[1.1. 研究背景 1](#_Toc5932)

[1.2. 研究内容 1](#_Toc5932)

[1.2.1. 目标检测模型 1](#_Toc1895)

[1.2.1. 图片数字识别模型 1](#_Toc1895)

[1.2.1. 可视化界面的搭建与信息存储 1](#_Toc1895)

[第七章 总结 1](#_Toc23862)

[1.1. 研究背景 1](#_Toc5932)

[1.2. 研究内容 1](#_Toc5932)

[1.2.1. 目标检测模型 1](#_Toc1895)

[1.2.1. 图片数字识别模型 1](#_Toc1895)

[1.2.1. 可视化界面的搭建与信息存储 1](#_Toc1895)

[参考文献 2](#_Toc154675103)

[致谢 2](#_Toc154675103)

[声明 2](#_Toc7507)

[学位论文使用授权书 2](#_Toc9446)

第一章 引言

1.1 研究背景

在智能交通系统中，车辆信息的检测是至关重要的核心任务之一。在复杂多变的自然场景下，迅速准确地检测车辆信息对于智能交通系统至关重要。随着车辆保有量的不断增加，与之相关的交通问题也变得日益严重。因此，实现对道路上车辆的检测和分类势在必行。此外，通过对这项技术的不断优化，可以为无人驾驶技术的发展以及智慧城市的建设提供一定的技术支持。

在此背景下，传统的车辆识别技术显得愈加局限，其在准确性和依赖人工干预方面的不足日益凸显。因此，研发一种高效、准确的车辆识别与信息提取系统已成为解决城市交通管理难题的迫切需求。近年来，深度学习技术的快速发展为车辆识别技术提供了新的契机。基于深度学习的目标检测算法在车辆识别领域取得了巨大进展，尤其是RCNN、YOLO等算法的出现使得自动化车辆识别成为可能。同时，计算机视觉技术的不断进步为车辆信息的提取提供了更多可能性，例如通过颜色、型号等特征进行车辆识别。

然而，尽管深度学习和计算机视觉技术在车辆识别领域取得了巨大进步，但仍然存在一些挑战需要克服。例如，对于复杂多变的场景和交通环境，车辆识别系统仍然面临着识别准确性和鲁棒性的问题，特别是在恶劣天气或光照条件下。此外，对于移动车辆的实时识别和跟踪也是一个挑战，需要系统能够在短时间内对车辆进行准确的检测和分类，以满足实时交通管理的需求。

对于车牌检测来说，我们可以认为它是车辆识别过程中的一个关键环节。车牌检测不仅能够提供车辆的身份信息，还是实施交通法规和管理措施的重要依据。随着技术的进步，车牌检测方法也从基于传统图像处理的算法逐渐转向采用深度学习模型，这些模型能够处理更复杂的图像背景和各种光照条件。目前，使用深度卷积神经网络（CNN）进行车牌识别和检测已显示出极高的效率和准确率。例如，可以通过改进的卷积神经网络来增强特征提取的能力，从而在各种复杂环境下提高识别的准确性。此外，结合区域提议网络（RPN）的方法如Faster-RCNN，可以实现车牌的快速和精确定位。而在定位了车牌之后，我们还需要对车牌的字符进行识别，在这一步骤中，字符识别技术扮演着至关重要的角色。使用深度学习方法如卷积神经网络，可以对车牌上的字符进行精确分割和识别。例如，可以利用长短时记忆网络（LSTM）或其他序列模型处理车牌字符的顺序，从而提高识别的准确性和可靠性。

进一步地，结合车辆检测和车牌识别的系统可以整合为一个统一的框架，通过端到端的训练方式，提升系统的整体性能。这种集成化的方法不仅可以提高处理速度，还可以降低错误识别的可能性。

想要实现端到端的车辆到车牌的识别，还有一个难题亟待解决，就是如何将识别到的车辆信息和车牌信息进行一一对应。如果车牌识别和车辆识别并行进行，会出现车牌和车辆信息无法对应的问题，在视频识别中这个问题尤为重要。在逐帧检测期间，可能一秒会产生检测出很多的车辆和车牌，这些信息如果不加以有效的处理和匹配，很容易导致数据的混乱。

为了确保车辆和车牌的识别准确率，我们需要在检测之后再次验证识别结果是否正确，这是我们必须关注的问题。因此，我们需要选择一个适合的模型，同时设计一套高效的验证机制，以提高车辆和车牌识别系统的准确性。

1.2 研究内容

1.2.1 目标检测模型

目标检测（Object Detection）的任务是找出图像中所有感兴趣的目标（物体），确定它们的类别和位置，是计算机视觉领域的核心问题之一。由于各类物体有不同的外观、形状和姿态，加上成像时光照、遮挡等因素的干扰，目标检测一直是计算机视觉领域最具有挑战性的问题。目标检测模型主要用于在图像或视频中检测和定位物体的位置，并识别物体所属的类别。它在计算机视觉领域广泛应用于诸如智能监控、自动驾驶、人脸识别、医学影像分析等方面。

在本项目中，我们需要定位图像或者视频中所有的车辆位置，所以我们需要选择一个适合的目标检测模型。近年来，随着计算机视觉和深度学习技术的不断发展，目标检测模型已经取得了显著的进步。为了在本项目中有效地定位图像或视频中的所有车辆位置，我们可以考虑使用一些先进的深度学习模型，如YOLO（You Only Look Once）、SSD（Single Shot MultiBox Detector）和Faster R-CNN（Faster Regions with Convolutional Neural Network）。这些模型以其高效率和准确性而闻名，特别适用于实时视频处理和复杂场景下的车辆检测。

在本项目中，作者选择了YOLO（You Only Look Once）算法作为本项目的目标检测算法，YOLO（You Only Look Once）模型的主要优势在于它的极速处理能力和高度整合性，使其在实时目标检测领域表现卓越。与其他模型相比，YOLO将整个目标检测过程视为一个单一的回归问题，从而能在单次前向传递中直接预测出目标的位置和类别。这种方法不仅简化了传统的目标检测流程，避免了先生成候选区域再对其分类的复杂步骤，还大大提高了处理速度。因此，YOLO能够实现极高的帧率，适合应用于需要快速响应的场景。此外，随着版本的迭代，YOLO的精度也得到了显著提升，尤其是在处理小物体和复杂场景时的表现越来越好，使其不仅快速而且准确，满足了广泛的实际应用需求。

为了实现端到端的车辆到车牌的识别，就不能对一个图片和视频同时进行车辆和车牌的目标检测。作者考虑到既然YOLO模型可以识别视频和图片，那么能不能将视频和图片中识别出的车辆单独截取下来，作为车牌识别算法的输入，这样就能够完成端到端的识别，且完成车辆和车牌的统一定位和识别。这种方法通过将车辆检测与车牌识别分开处理，但仍然保持一个统一的工作流程，可以有效地提高整体识别的准确性和效率。

实际操作中，可以首先使用YOLO模型对视频或图片中的车辆进行快速且精确的定位和识别。一旦车辆被检测出，相应的图像区域（即车辆的图像部分）就可以被截取下来并单独处理。这些截取的车辆图像随后被用作车牌识别算法的输入，其中可以应用专门针对车牌特征优化的算法，如基于深度学习的字符识别技术，进一步识别出车牌上的文字和数字。

此种该设计有几个显著优势。首先，由于车辆图像已经被YOLO模型预筛选和定位，车牌识别算法可以在一个更加精确和限定的搜索范围内操作，这大大减少了背景噪声和不相关信息的干扰，提高了识别的精度。其次，这种方法使得处理流程更加模块化，便于优化和维护。每个模块可以独立进行优化和调整，而不会影响到其他部分，这对于应对不同的操作环境和需求尤为有利。

1.2.2 图片数字识别模型

图片数字识别模型主要用于识别图像中的数字，例如手写数字识别、车牌号码识别等。它在各种场景下都有广泛的应用，如自动识别验证码、数字化文档处理、手写数字识别等。数字识别技术的核心通常依赖于深度学习模型，特别是卷积神经网络（CNN）和循环神经网络（RNN），它们能够有效地从复杂背景中提取数字特征并进行准确识别。这些模型通过训练大量带有标记的图像数据，学会识别不同形式和风格的数字，从而能够应对实际应用中各种变化的挑战。

而在本项目中，作者选择了LPRNet这一模型来处理车牌号码识别的任务，因为LPRNet设计用于直接从车牌图像中无需分割即可识别文字，这使得整个识别过程更加高效和准确。LPRNet结合了深度卷积神经网络和循环神经网络的特点，利用CNN层提取图像特征，随后通过RNN层处理这些特征以解析出连续的字符序列，最终通过CTC（Connectionist Temporal Classification）层输出车牌上的完整文本。

这种端到端的方法不仅提升了车牌识别的速度，还因其不需要传统方法中的字符分割环节而降低了错误率。此外，LPRNet能够适应不同国家和地区的车牌格式，包括不同的字体、颜色和布局，这是因为其训练过程中包含了多样化的车牌样本。这种灵活性和鲁棒性使得LPRNet成为广泛使用的车牌识别解决方案之一。

在本项目中，作者将识别到的车牌进行裁剪，将裁剪到的图片导入到LPRNet的网络中，进行进一步的分析和识别。该过程自动执行，无需额外的人工干预，极大地提高了处理效率和减少了错误的可能性。通过这种方式，从视频监控或实时交通流中快速准确地提取车牌信息成为可能。

1.2.3 可视化界面的搭建与信息存储

可视化界面的搭建与信息存储主要用于展示和管理数据、与用户进行交互，并提供友好的用户体验。它在各种应用中都有广泛的应用，例如管理系统、数据分析工具、社交平台等。

为了能够在前端中直接调用后端的python检测模型，作者考虑是否能在前端中直接使用child\_process模块中的spawn子进程调用方法对python检测模型进行直接调用。但是由于ES5和ES6的不兼容关系，作者考虑到能否使用NodeJS独立的搭建一个后台，单独进行对python识别模型的调用。同时在NodeJS后台中开放express网络端口，让前端可以通过网络的socket方式（IP:Port）进行直接调用，同时NodeJs还会将处理到的数据返回到前端，实现一个流畅的数据交互过程。

为了实现这个方案，作者首先需要设置一个NodeJS服务器，这个服务器将使用Express框架来处理HTTP请求，并管理与Python识别模型的交互。通过使用child\_process模块中的spawn方法，NodeJS可以创建一个子进程来运行Python脚本。这种方法允许Python脚本作为一个独立进程运行，同时还可以通过标准输入输出来与NodeJS交换数据。

为了实现信息存储，作者采用了Mysql+Mybatis(Springboot)集成的框架进行后端数据管理。这种集成框架提供了一种高效且易于管理的方式来处理应用程序中的数据存取操作。用户可以在前端界面上选择是否将该识别的车辆数据放入数据库中进行存储，如果选择存储，系统将自动执行数据插入操作。这一过程通过预先配置好的MyBatis映射器实现，存入数据库后，总体数据将会在展示仓界面进行更新操作。

1.3 国内外研究现状

在2001年，Viola 和 Jones 在CVPR上发表了经典的《Rapid Object Detection using a Boosted Cascade of Simple Features》和《Robust Real-Time Face Detection》，提出了Viola-Jones检测器。VJ框架在AdaBoost算法的基础上，使用Haar-like小波特征和积分图技术来进行人脸检测。在深度学习出现以前，工业界的方案都是基于VJ算法。但VJ算法仍存在一些问题：Haar-like特征是一种相对简单的特征，其稳定性较低；弱分类器采用简单的决策树，容易过拟合。因此，该算法对于解决正面的 人脸效果好，对于人脸的遮挡，姿态，表情等特殊且复杂的情况，处理效果不理想；基于VJ-cascade的分类器设计，进入下一个分类器后，之前的信息都丢弃了，分类器评价一个样本不会基于样本在之前步骤中的表现，导致分类器的鲁棒性差。

基于手工特征提取的传统目标检测算法进展缓慢，且性能较差。然而，自2012年卷积神经网络（Convolutional Neural Networks, CNNs）的兴起之后，目标检测领域迎来了巨大的变革。基于CNN的目标检测算法主要分为两个技术路线：Anchor-Based和Anchor-Free方法。这两种方法的主要区别在于是否利用锚点（Anchor）来进行训练和预测；其中，Anchor-based方法可以进一步分为一阶段和二阶段检测算法。通常，二阶段目标检测算法在精度上优于一阶段方法，但后者在速度上更具优势。近年来，Anchor-Free算法也在不断地完善和发展。

在Anchor-based的一阶段方法中，它们直接在提取的特征基础上预测物体的分类和位置，无需生成RP。这类方法的任务流程简化为：特征提取 --> 分类/定位回归。常见的一阶段目标检测算法包括OverFeat、YOLO系列、SSD和RetinaNet等。

而在Anchor-based的二阶段方法中，首先生成区域建议（region proposal，简称RP），这些RP是可能包含目标对象的预选框。随后，通过卷积神经网络对这些RP进行样本分类和定位回归。二阶段目标检测算法的任务流程大致为：特征提取 --> 生成RP --> 分类/定位回归。其中常见的二阶段目标检测算法包括R-CNN、SPP-Net、Fast R-CNN、Faster R-CNN和R-FCN等。

第二章 背景技术介绍

2.1 YOLOv5

YOLOv5（You Only Look Once version 5）是一种目标检测算法，其原理基于YOLO系列的基本思想，但在架构和细节上进行了改进和优化。与传统的目标检测算法不同，YOLOv5采用了单次检测的方法，即将整个图像作为输入，并在一次前向传递中直接预测边界框及其对应的类别和置信度。为了提高模型的适应性，YOLOv5使用了锚框（Anchor Boxes），这些是事先定义的边界框，用于在图像中提取目标，从而更好地适应不同大小和形状的目标。另外，YOLOv5采用了CSPDarknet53作为其主干网络，具有Cross Stage Partial连接（CSP连接），能够有效地提取图像特征。在特征提取和检测方面，YOLOv5将特征图传递给一系列卷积层和池化层，转换为边界框及其相应的类别概率。它还使用了一种称为"cross-stage partial connections"的连接方式，促进特征的传递和信息流动，提高了检测性能。损失函数方面，YOLOv5结合了目标位置的回归损失、目标类别的分类损失以及目标置信度的置信度损失，通过最小化这些损失函数来学习准确地检测和定位目标。训练过程包括数据加载、网络前向传播、损失计算和反向传播等步骤，一旦训练完成，模型就可以用于推理，即在新的图像上进行目标检测。

2.2 LPRNET

LPRNet（License Plate Recognition Net）是一种用于车牌识别的神经网络模型，主要用于自动识别车牌上的字符以及车牌的区域。它采用基于深度学习的端到端模型，直接从原始图像中提取特征并输出车牌上的字符序列，消除了传统方法中复杂的手工特征提取和后续处理步骤。利用卷积神经网络（CNN）来提取图像中的特征，帮助网络理解车牌上的字符以及它们的空间位置关系，其深度和宽度有助于模型学习复杂的图像模式。LPRNet的主要任务是从车牌图像中提取特征，并识别其中的字符，通过多个卷积层和池化层逐渐提取和压缩图像的特征，最后通过全连接层和Softmax层将特征映射到字符类别，并输出字符序列。为了识别不同尺度的字符和车牌，LPRNet可能会使用多尺度的特征图，经过不同层级的特征提取和池化后进行融合，获得更全面的特征表示。训练过程通常采用端到端的方式进行，在大量标记的车牌图像上进行训练，使网络学习到有效的特征提取和字符识别策略。在字符识别后可能需要进行一些后处理步骤，如字符序列的过滤和纠正，以确保识别的准确性和一致性。综上所述，LPRNet通过深度学习方法，利用卷积神经网络提取车牌图像的特征，并识别其中的字符序列，从而实现车牌识别的功能。

2.3 NodeJS

Node.js是一个基于Chrome V8引擎的JavaScript运行环境，主要用于构建高性能、可扩展的网络应用程序。它采用事件驱动的编程模型，基于事件循环机制，通过注册回调函数来处理各种事件，如I/O操作完成、HTTP请求等。通过单线程模型处理请求，所有的I/O操作都是异步的，不会阻塞主线程，同时利用底层的libuv库实现了非阻塞I/O，充分利用多核CPU。事件循环是实现异步非阻塞I/O的关键机制，负责监听事件队列中的事件，并调度执行相应的回调函数，使得Node.js能够高效地处理大量的并发请求。Node.js采用了CommonJS模块规范，允许开发者将代码模块化，提供了一系列内置模块和支持第三方模块的安装和使用。通过回调函数和Promise等机制来实现异步编程，编写非阻塞的代码，提高了程序的性能和吞吐量。借助Chrome V8引擎的优势，Node.js具有出色的性能表现，将JavaScript代码编译成本地机器码执行，采用了即时编译和垃圾回收等技术来优化性能。Node.js的原理基于事件驱动、单线程、非阻塞I/O等特性，通过事件循环和异步编程实现高效的网络应用程序开发。

2.4 Vue＆SpringBoot

Vue.js是一款流行的JavaScript前端框架，用于构建交互式的Web界面。它提供了响应式数据绑定机制，通过将数据和DOM进行绑定，当数据发生变化时，DOM会自动更新，使开发者更方便地管理和维护应用程序的状态。另外，Vue.js支持组件化开发，允许将页面拆分为独立的组件，每个组件负责特定的功能，提高了代码的可复用性和可维护性。它还利用虚拟DOM技术提高页面渲染的性能，通过将页面的状态保存在内存中的虚拟DOM树中，与实际DOM进行比较并进行更新。Vue.js提供了一系列的生命周期钩子函数，允许开发者在组件的不同阶段执行特定的操作，如数据初始化、DOM渲染、销毁等。

Spring Boot是一款用于构建Java后端应用程序的框架，基于Spring框架，简化了Spring应用程序的配置和开发过程。它采用依赖注入的方式管理应用程序中的组件和服务，通过将依赖关系定义在配置文件中或通过注解来实现，更灵活地管理应用程序的组件。Spring Boot提供了自动配置功能，根据应用程序的依赖关系和配置文件，自动配置应用程序的各种组件，减少了开发者的配置工作量。它集成了多种嵌入式Web服务器（如Tomcat、Jetty等），使开发者可以将应用程序打包成一个独立的可执行JAR文件，并运行在内置的Web服务器上。此外，Spring Boot还集成了多种ORM框架（如Hibernate、MyBatis等），方便与数据库进行交互，实现持久化操作，并提供了丰富的支持来构建RESTful风格的API，包括对HTTP请求的映射、参数验证、异常处理等功能。Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

第三章 系统分析与设计

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

第四章 系统分析与设计

4.1 实现环境与工具的简要说明

4.1.1 硬件环境

表4.1 硬件环境

|  |  |  |  |
| --- | --- | --- | --- |
| CPU | RAM | GPU | OS |
| Intel(R) Core(TM) i7-12700H 2.30GHz | 16.0 GB | NVIDIA RTX 3060 6G | Windows 11 |

4.1.2 软件环境

表4.2 软件环境

|  |  |
| --- | --- |
| Python | 3.8.18 |
| Torch | 1.9.0+cu111 |
| Torchaudio | 0.9.0 |
| Torchvision | 0.10.0+cu111 |
| Numpy | 1.20.3 |
| NodeJS | 18.13.0 |
| Java | 8 |
| SpringBoot | 2.7.6 |
| Mybatis | 3.5.7 |
| Mysql | 5.7.28 |

## 4.2主要程序模块实现

4.2.1 车辆检测模型训练

①数据集收集和标注

为训练该模型，作者选择的是BITVehicle车辆检测数据集。BITVehicle车辆检测数据集：总共9850张图片，标注了6个类别：公共汽车(Bus)、微型客车(Microbus)、小型货车(Minivan)、轿车(Sedan)、SUV(SUV) 和卡车(Truck)。该数据集包括了两个部分，第一部分是高速上的车辆图片（格式为.jpg），第二部分是车辆图片对应的类别和方框的定位信息（格式为.xml）。



图4.3 数据集截图

在后续实验中发现如果按照该数据集进行车辆检测的训练，训练出来的模型对于高速车辆的检测来说正确率较高，但对于普通街区，复杂地形的车辆检测的正确率偏低。所以尽管模型在高速车辆检测方面表现出色，但在应对复杂街区环境中的车辆检测时存在局限性。这是由于训练数据集主要包含高速行驶的车辆图像，而缺乏足够的复杂环境下的车辆图像，导致模型未能学习到处理复杂环境所需的特征。

为了提高模型在复杂地形下的表现，作者还使用了标注工具LabelImg对部分街区的汽车图片（如图4.3所示）采用YOLO标签格式进行手动标注。作者为了同时适应BITVehicle车辆检测数据集。也标注了6个类别：公共汽车(Bus)、微型客车(Microbus)、小型货车(Minivan)、轿车(Sedan)、SUV(SUV) 和卡车(Truck)（如图4.4所示）。



图4.3 复杂街区数据集截图

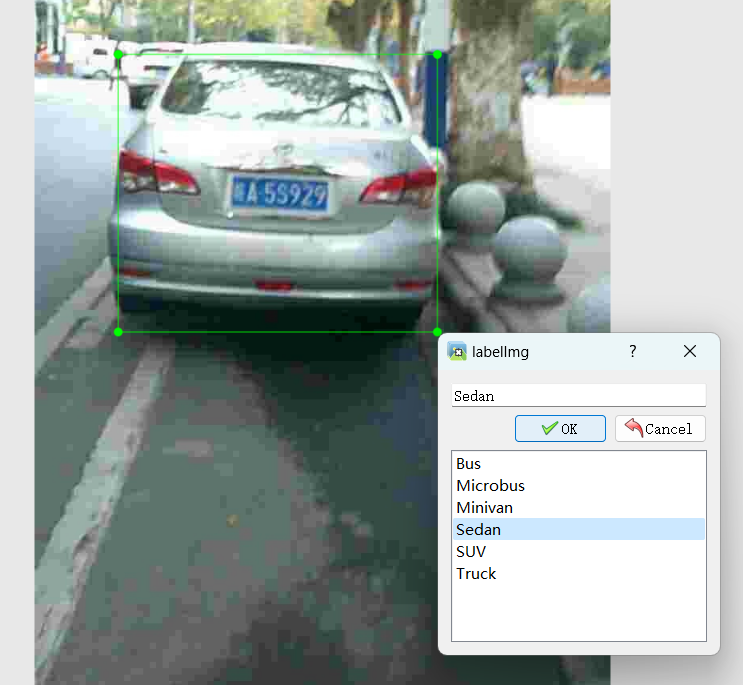


图4.4 LabelImg标注界面

②模型训练

将所有的xml格式的标注文件通过python程序转换成用于检测的txt格式后，再将大约10000张图片按照 4:1 的比例分为训练集和测试集。最终的数据集结构如下所示：其中，VOC2024文件夹保存了分割前的原始图片和标注文件；images文件夹存放了车辆图片，并且按照训练集（train）和测试集（val）进行了划分；labels文件夹存放了对应的车辆图片的txt格式标注文件，同样按照训练集（train）和测试集（val）划分。

├─images

│ ├─train

│ └─val

├─labels

│ ├─train

│ └─val

└─VOC2024

├─Annotations

├─JPEGImages

└─YOLOLabels

在配置完剩余的配置文件后，选择yolov5s.pt的预训练模型，选择epochs的次数为50次，键入以下命令让yolov5模型开始训练。python .\train.py --data .\data\car.yaml --cfg .\models\car.yaml --weights .\weights\yolov5s.pt --batch-size 8 --epochs 50

③训练结果

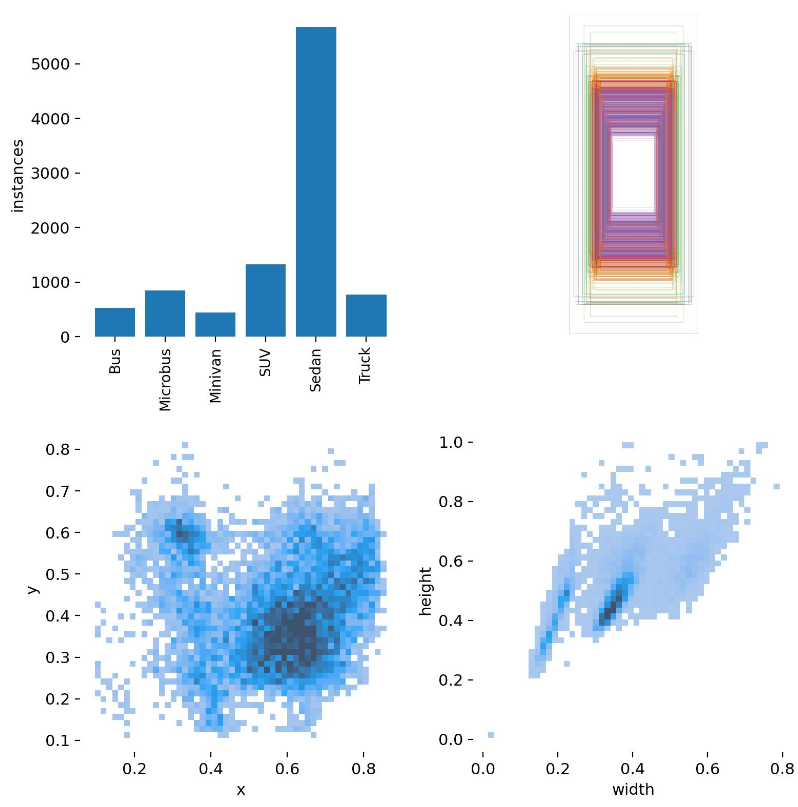


图4.5 标签分布概览



图4.6 混淆矩阵

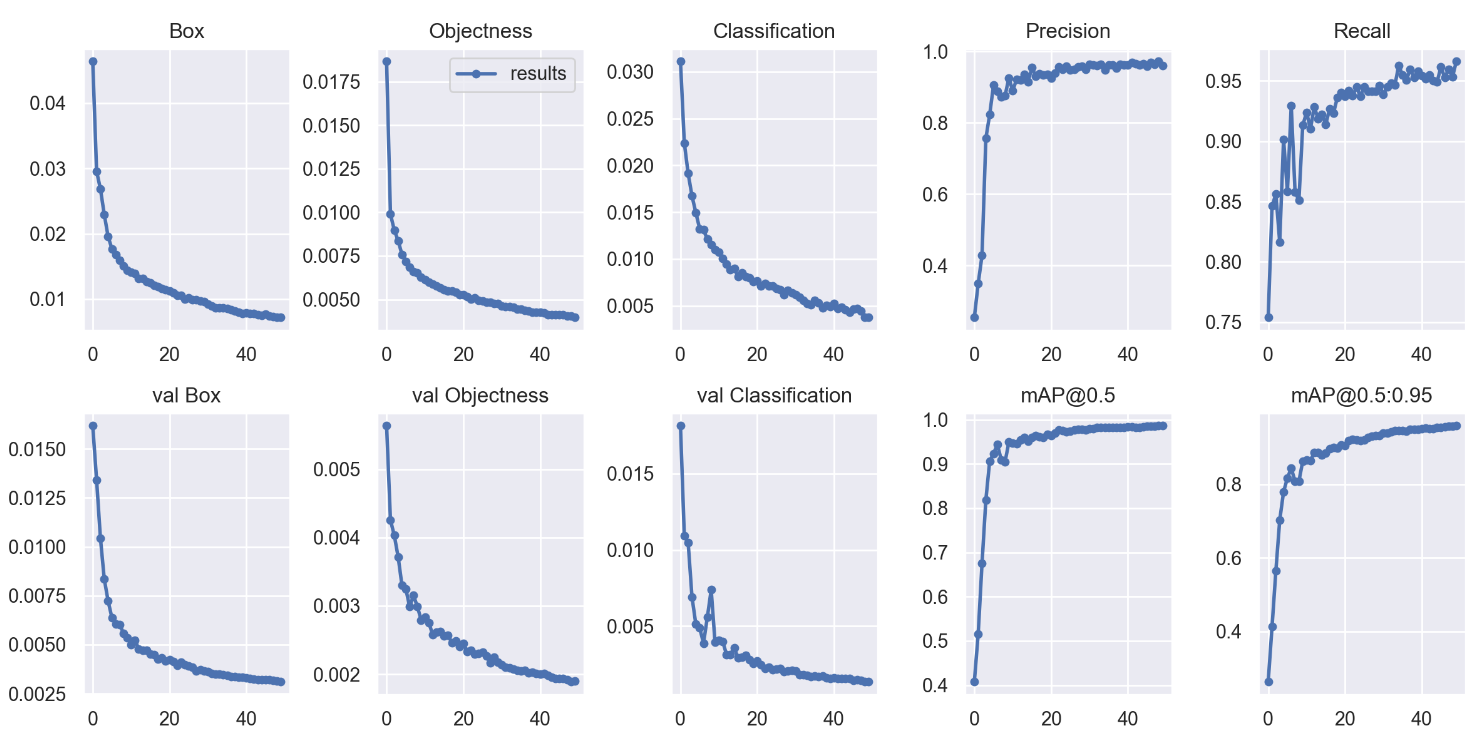


图4.7 训练过程中的损失，召回率，精准率和平均精度

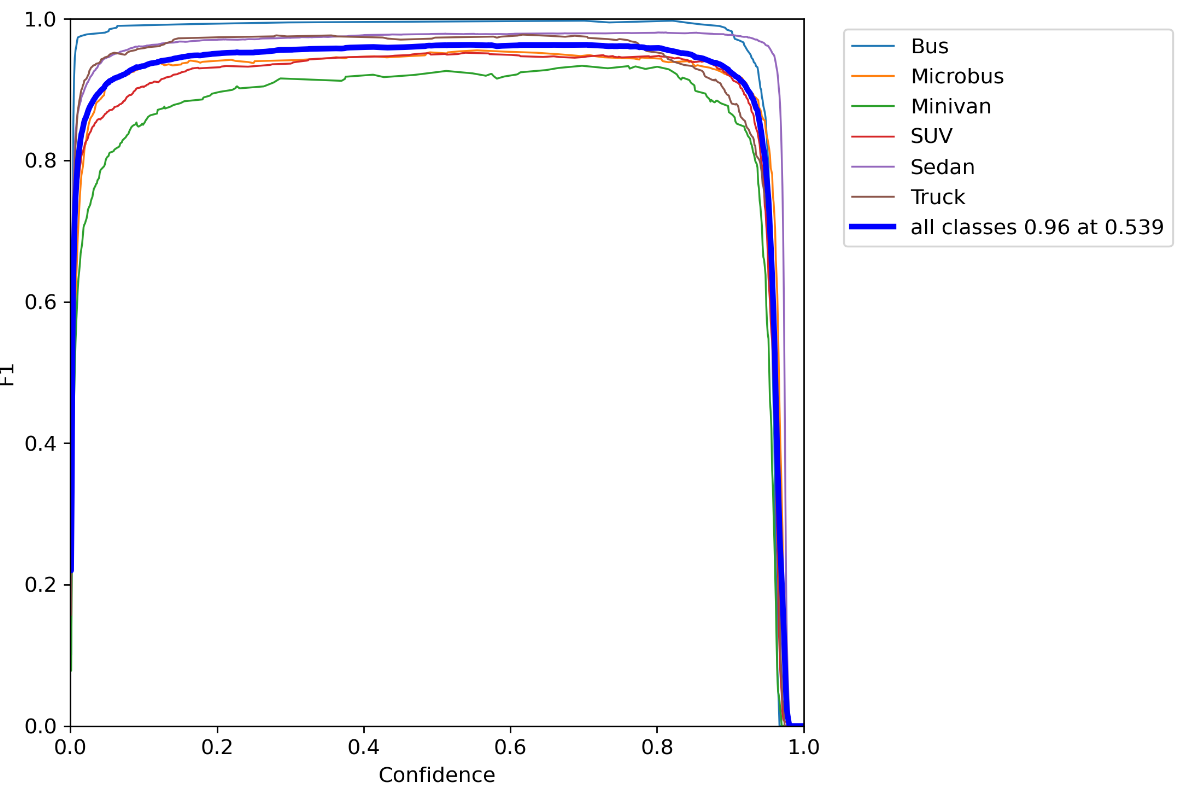


图4.8 模型F1曲线

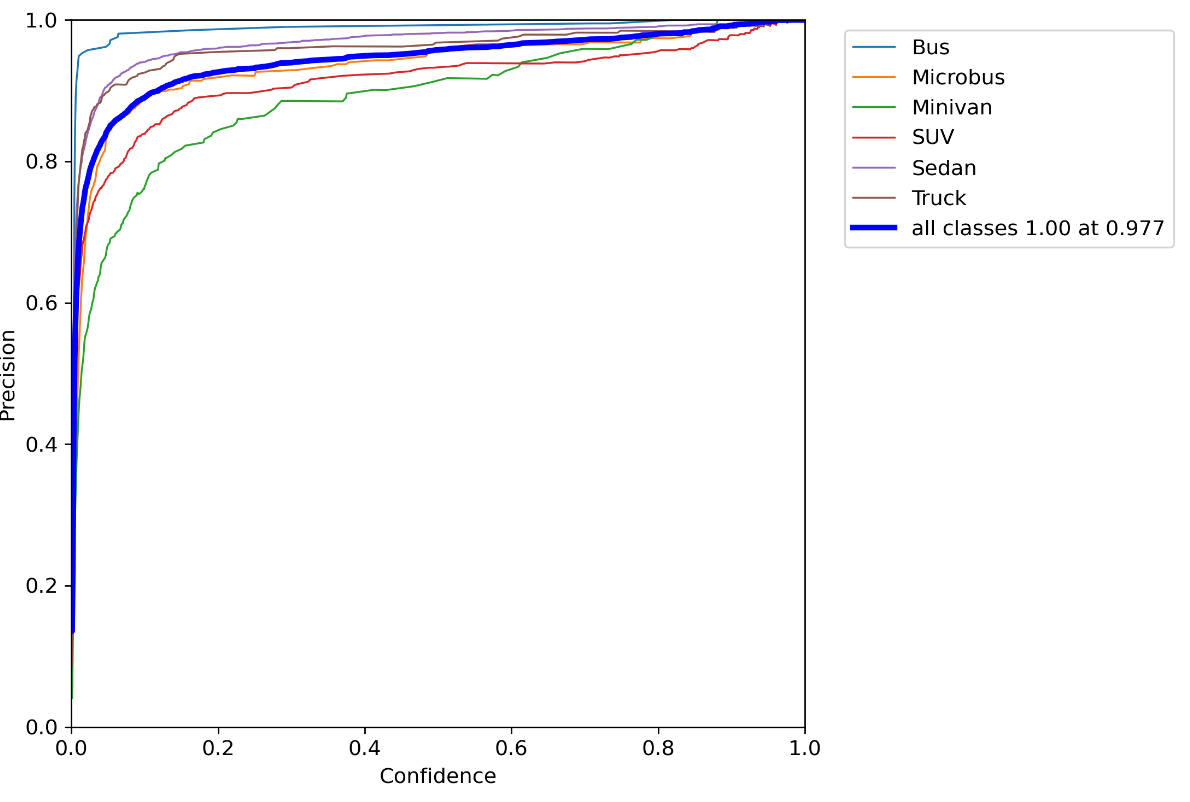


图4.9 模型P曲线

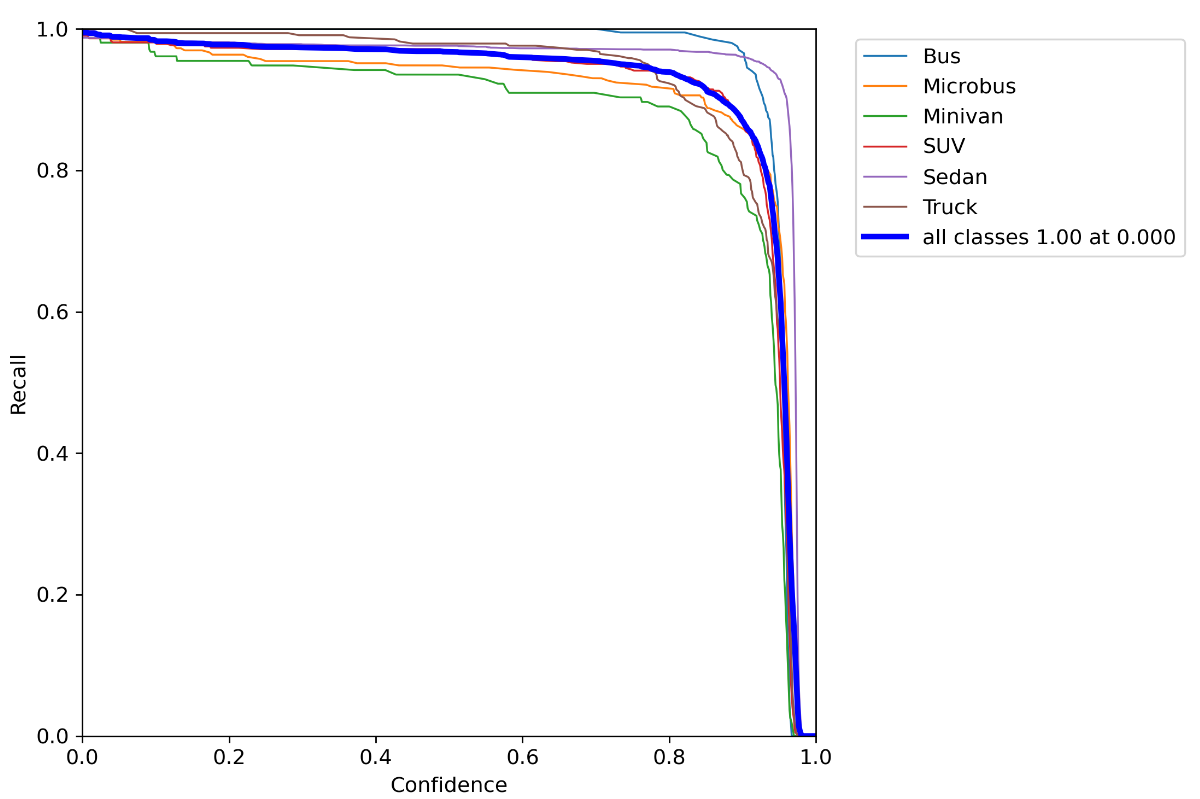


图4.10 模型R曲线

④训练结果分析

混淆矩阵展示了实际类别与模型预测类别之间的关系。从这个矩阵中可以观察到，大多数预测结果集中在对角线上，这表明模型在多数情况下能够正确地进行分类。同时，从训练过程中的损失、召回率、精确率以及平均精度的图表中可以看出，随着训练epoch数的增加，模型的性能整体呈现出改善的趋势。而模型训练的F1曲线，P曲线，R曲线，进一步验证了模型随训练进度的持续优化。F1曲线，代表精确率与召回率的平衡，随着训练的深入，其值逐渐增加，显示出模型在维持错误识别最低的同时提高正确识别的能力。P曲线，即精确率曲线，随着模型对各类别判断的准确性提高，其上升趋势明显。R曲线，表示召回率，亦随着模型对正样本的捕捉能力增强而提升。

4.2.2 车牌检测模型训练

①数据集收集和标注

为训练该模型，作者选择的是BITVehicle车辆检测数据集。BITVehicle车辆检测数据集：总共9850张图片，标注了6个类别：公共汽车(Bus)、微型客车(Microbus)、小型货车(Minivan)、轿车(Sedan)、SUV(SUV) 和卡车(Truck)。该数据集包括了两个部分，第一部分是高速上的车辆图片（格式为.jpg），第二部分是车辆图片对应的类别和方框的定位信息（格式为.xml）。



图4.3 数据集截图

在后续实验中发现如果按照该数据集进行车辆检测的训练，训练出来的模型对于高速车辆的检测来说正确率较高，但对于普通街区，复杂地形的车辆检测的正确率偏低。所以尽管模型在高速车辆检测方面表现出色，但在应对复杂街区环境中的车辆检测时存在局限性。这是由于训练数据集主要包含高速行驶的车辆图像，而缺乏足够的复杂环境下的车辆图像，导致模型未能学习到处理复杂环境所需的特征。

为了提高模型在复杂地形下的表现，作者还使用了标注工具LabelImg对部分街区的汽车图片采用YOLO标签格式进行手动标注。作者为了同时适应BITVehicle车辆检测数据集。也标注了6个类别：公共汽车(Bus)、微型客车(Microbus)、小型货车(Minivan)、轿车(Sedan)、SUV(SUV) 和卡车(Truck)（如图4.3所示）。

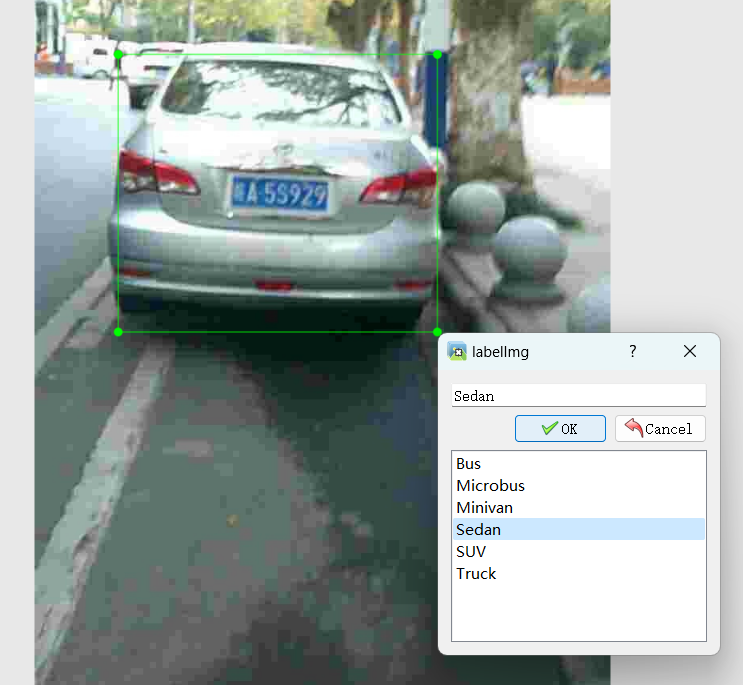


图4.3 LabelImg标注界面

②模型训练

将所有的xml格式的标注文件通过python程序转换成用于检测的txt格式后，再将大约10000张图片按照 4:1 的比例分为训练集和测试集。最终的数据集结构如下所示：其中，VOC2024文件夹保存了分割前的原始图片和标注文件；images文件夹存放了车辆图片，并且按照训练集（train）和测试集（val）进行了划分；labels文件夹存放了对应的车辆图片的txt格式标注文件，同样按照训练集（train）和测试集（val）划分。

├─images

│ ├─train

│ └─val

├─labels

│ ├─train

│ └─val

└─VOC2024

├─Annotations

├─JPEGImages

└─YOLOLabels

在配置完剩余的配置文件后，选择yolov5s.pt的预训练模型，选择epochs的次数为50次，键入以下命令让yolov5模型开始训练。python .\train.py --data .\data\plate.yaml --cfg .\models\plate.yaml --weights .\weights\yolov5s.pt --batch-size 8 --epochs 50

③训练结果

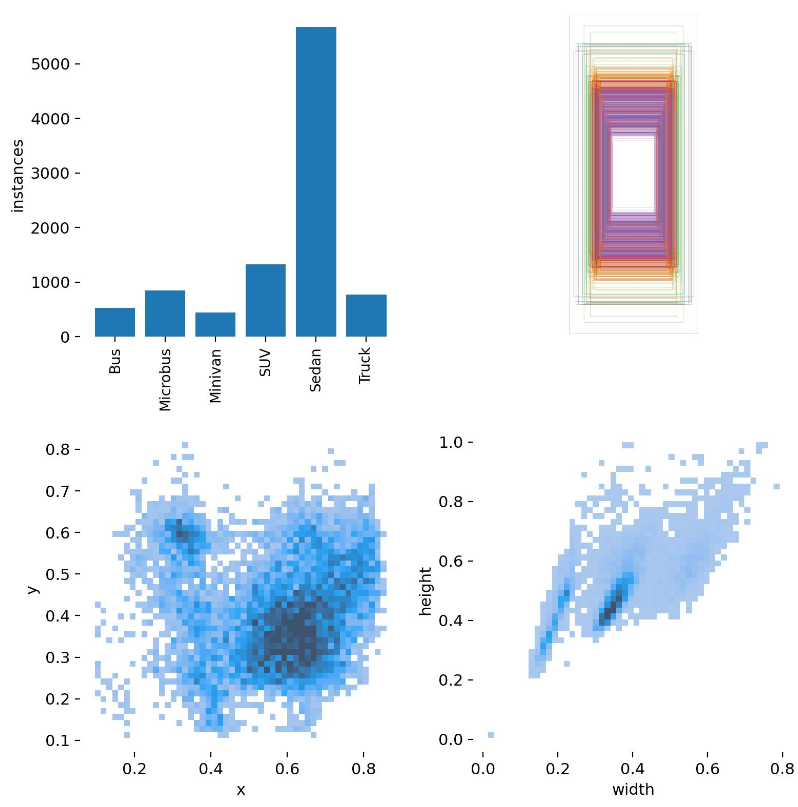


图4.4 标签分布概览



图4.5 混淆矩阵

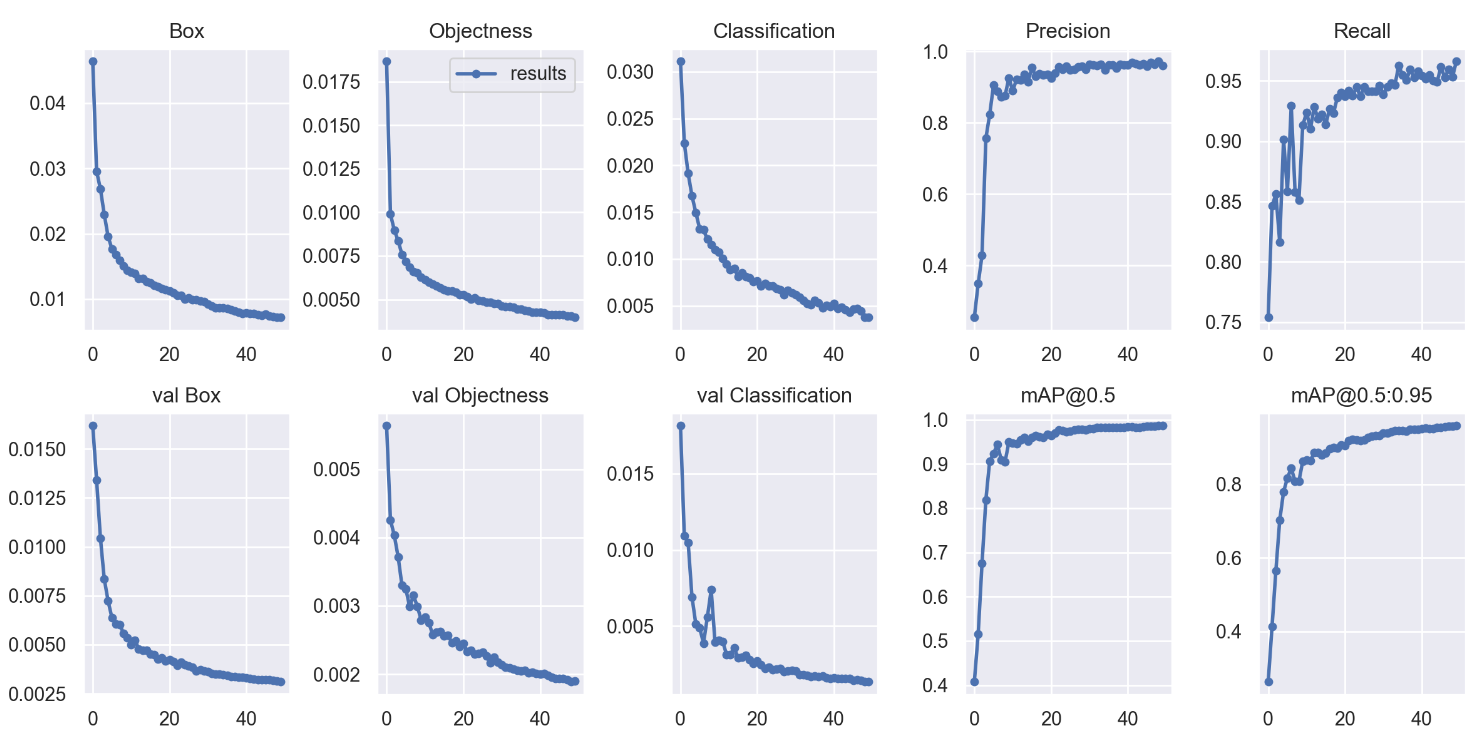


图4.6 训练过程中的损失，召回率，精准率和平均精度

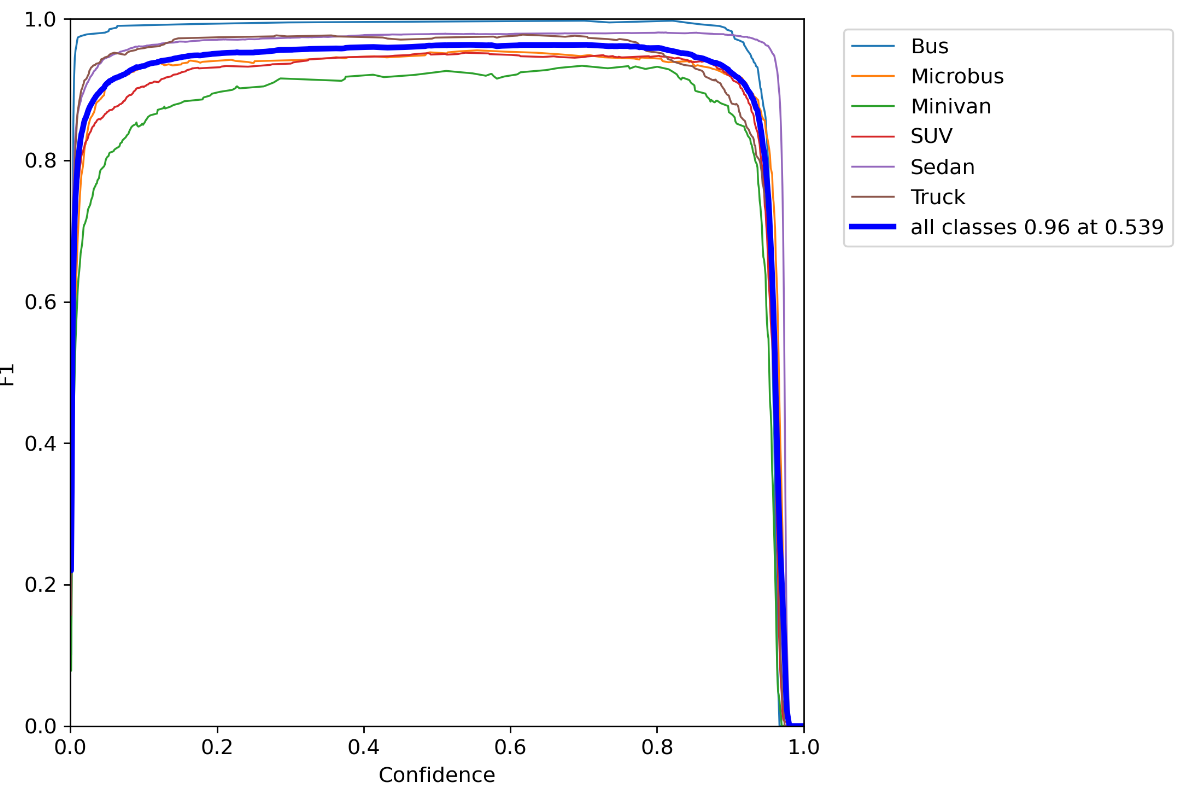


图4.7 模型F1曲线

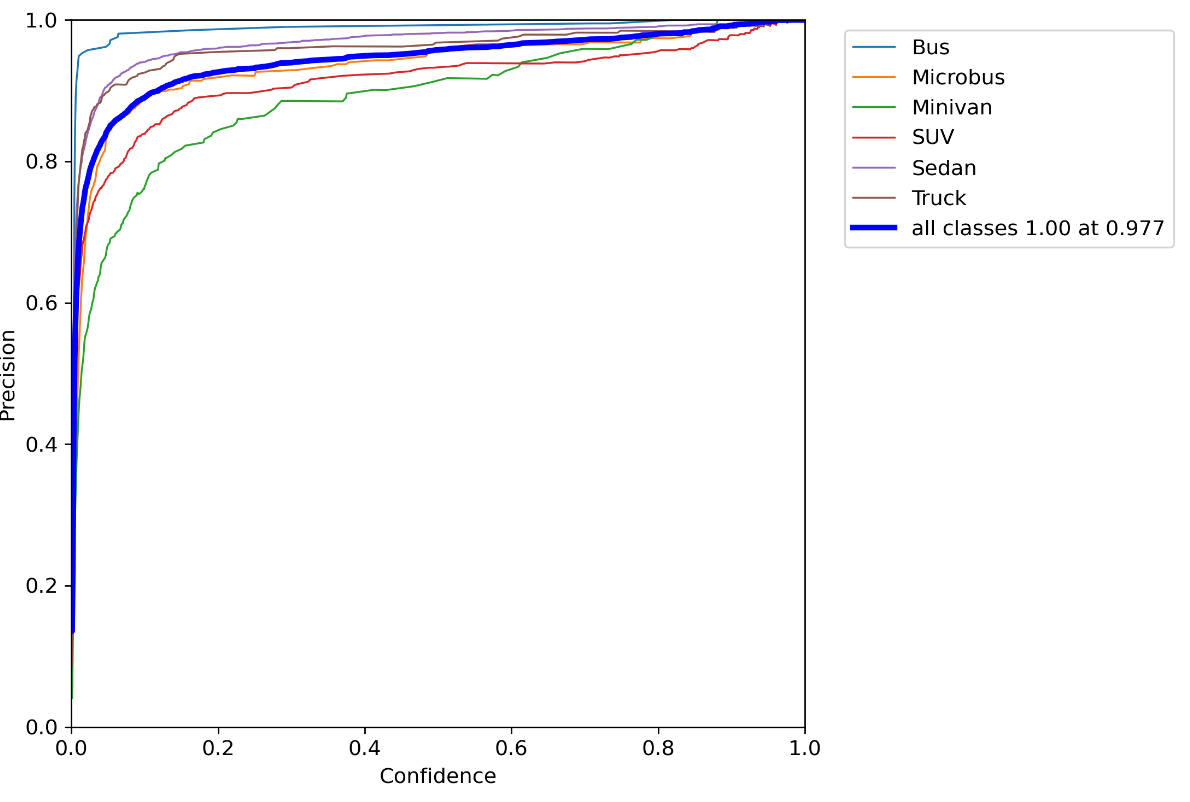


图4.8 模型P曲线

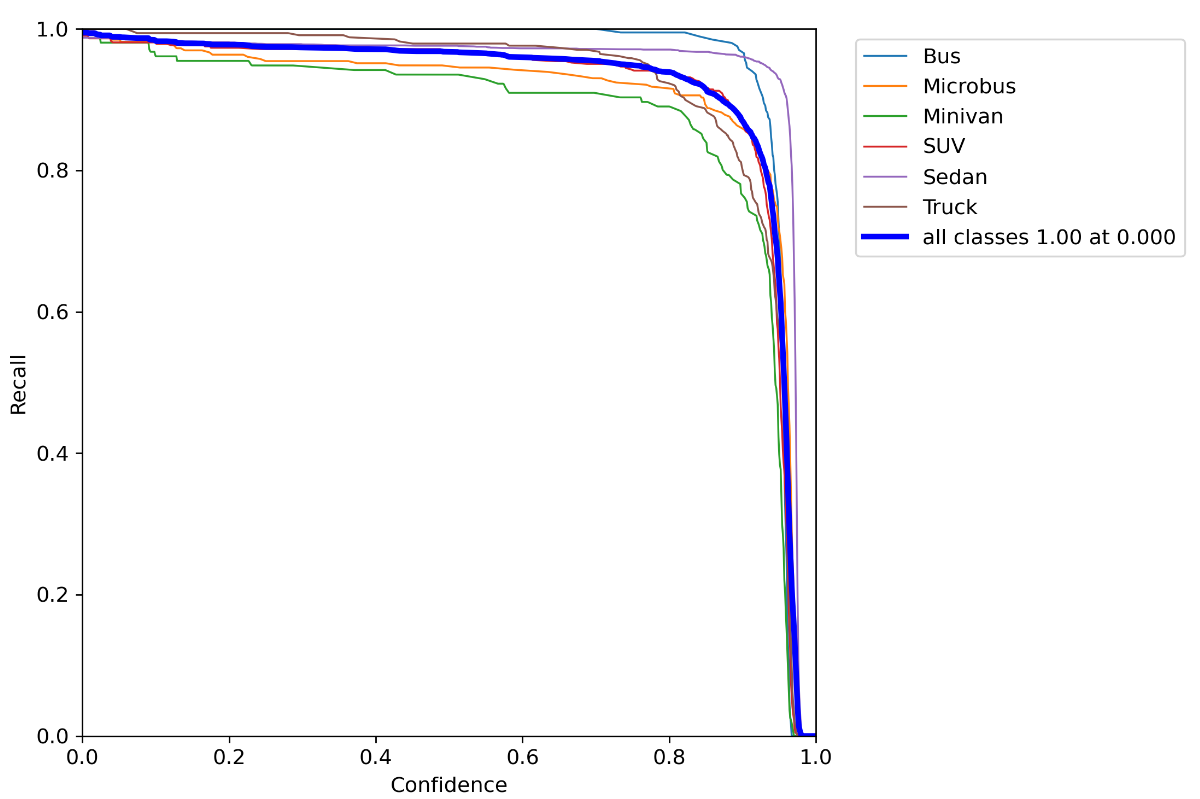


图4.9 模型R曲线

④训练结果分析

混淆矩阵展示了实际类别与模型预测类别之间的关系。从这个矩阵中可以观察到，大多数预测结果集中在对角线上，这表明模型在多数情况下能够正确地进行分类。同时，从训练过程中的损失、召回率、精确率以及平均精度的图表中可以看出，随着训练epoch数的增加，模型的性能整体呈现出改善的趋势。而模型训练的F1曲线，P曲线，R曲线，进一步验证了模型随训练进度的持续优化。F1曲线，代表精确率与召回率的平衡，随着训练的深入，其值逐渐增加，显示出模型在维持错误识别最低的同时提高正确识别的能力。P曲线，即精确率曲线，随着模型对各类别判断的准确性提高，其上升趋势明显。R曲线，表示召回率，亦随着模型对正样本的捕捉能力增强而提升。

4.2.3 车辆检测模型训练

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

、精确率以及平均精度的图表中可以看出，随着训练epoch数的增加，模型的性能整体呈现出改善的趋势。而模型训练的F1曲线，P曲线，R曲线，进一步验证了模型随训练进度的持续优化。F1曲线，代表精确率与召回率的平衡，随着训练的深入，其值逐渐增加，显示出模型在维持错误识别最低的同时提高正确识

4.2.3 车辆检测模型训练

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

4.3 实现环境与工具的简要说明

第五章 讨论

## 4.2主要程序模块实现

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

4.3 实现环境与工具的简要说明

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

Vue.js用于构建交互式的前端界面，提供了响应式数据绑定和组件化开发等特性；而Spring Boot用于构建Java后端应用程序，提供了依赖注入、自动配置、嵌入式Web服务器等特性。

将Vue.js和Spring Boot结合使用可以构建出功能丰富、性能优越的全栈Web应用程序。

参考文献

致谢

声 明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得四川大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

本学位论文成果是本人在四川大学读书期间在导师指导下取得的，论文成果归四川大学所有，特此声明。

作者签名： 导师签名：

年 月 日

学位论文使用授权书

本学位论文作者完全了解四川大学有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或相关机构送交论文的原件、复印件和电子版，允许论文被查阅和借阅。本人授权四川大学将本学位论文的全部或部分内容编入有关数据库进行信息技术服务，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文，并用于学术活动。

（涉密学位论文在解密后适用于本授权书）

作者签名： 导师签名：

年 月 日

（注意：最终电子版论文终稿该页所有签名用电子签名或打印出来纸质版签字后规整扫描后再插入，最终稿不能提交无签名的页面，否则全国论文抽检将无法通过！！）