分类号_____ 学号 M201676076
学校代码 10487 密级_____

华中科技大学

硕士学位论文

智能监控视频行人检测系统的设计与实现

学位申请人：张　润

学 科 专 业：软件工程

指 导 教 师：管　乐

答 辩 日 期：2018.12.18

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree for the Master of Engineering

Design and Implementation of Intelligent Surveillance Video Pedestrian Detection System

Candidate : Zhang Run

Major : Software Engineering

Supervisor : Guan Yue

Huazhong University of Science & Technology

Wuhan 430074, P.R.China

December, 2018

独创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除文中已经标明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：张润

日期：2019年 1月 5日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，即：学校有权保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权华中科技大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本论文属于 保密□，　　在_____年解密后适用本授权书。
不保密☑。

（请在以上方框内打“√”）

学位论文作者签名：张润　　指导教师签名：

日期：2019年 1月 5日　　日期：2019年 1月 5日

摘 要

近几年人工智能技术发展迅速，其中深度学习技术在各个领域取得了巨大的进展，使得人工智能技术在实际生活中的应用成为了可能。行人检测是计算机视觉领域的一大难点，其应用场景广泛。同时，随着中国城镇化进程的演进，人口流动也越来越频繁，这使得公共安全的压力也与日俱增，为此公安部门和企业等组织推进的天眼工程也使得中国拥有数量庞大的监控摄像头。传统的视频监控系统大多只提供查看、存储、回放等功能，系统功能较为单一，使用过程中无法做到及时预警。

本文通过对市场需求的了解和对技术调研，设计并实现了智能监控视频行人检测系统。本系统针对企业园区环境下监控视频行人检测的需求进行设计与实现。系统采用基于区域的全卷积网络(Region-based Fully Convolutional Network, R-FCN)作为本文的核心算法，针对系统未来实际的应用场景制作数据集，同时针对应用场景对网络细节进行修改以使得模型在处理应用场景数据时具有更好效果，使用 Caffe 深度学习框架训练得到网络模型。但训练得到的模型虽然能够满足系统运行时的精度要求却无法达到实时性要求，因此我们使用 TensorRT 推理引擎对训练得到的模型进行量化压缩得到系统部署使用的推理模型。压缩后的模型将会部署在服务器端，服务器端的 GPU 需要支持 INT8 计算精度。客户端程序使用 Qt5 开发。用户在客户端程序中可以添加自己管理的摄像头，选择对管理的摄像头实时视频流或本地视频文件进行检测，服务器端获取到视频数据后进行检测并将检测结果发送回客户端程序展示给用户。

本文设计实现的智能监控视频行人检测系统可以对获取的视频数据进行行人检测，且精度和实时性均达到要求。用户客户端程序也遵循简洁易用的原则，用户使用时操作方便，逻辑清晰。系统整体满足需求分析和功能设计的要求，经过测试后，系统可以满足实际应用的要求。

关键词：深度学习　　行人检测　　R-FCN　　TensorRT

Abstract

In recent years, artificial intelligence technology has developed rapidly, and deep learning technology has made great progress in various fields, making the application of artificial intelligence technology in real life possible. Pedestrian detection is a major difficulty in the field of computer vision, and its application scenarios are extensive. At the same time, with the evolution of China's urbanization process, population movements have become more frequent, which has led to increasing public security pressures. For this reason, the Tianyan project promoted by public security departments and enterprises has also enabled China to have a large number of surveillance cameras. Most of the traditional video surveillance systems only provide functions such as viewing, storage, and playback. The system functions are relatively simple, and timely warning cannot be achieved during use.

In this paper, through the understanding of market demand and technical research, the intelligent monitoring video pedestrian detection system is designed and implemented. This system is designed and implemented for the needs of monitoring video pedestrian detection in the enterprise campus environment. The system uses the Region-based Fully Convolutional Network (R-FCN) as the core algorithm of this paper, and makes data sets for the actual application scenarios of the system in the future. At the same time, the network details are modified for the application scenarios to make the model. It has a better effect when processing the application scene data, and the network model is trained using the Caffe deep learning framework. However, although the trained model can meet the accuracy requirements of the system operation but can not meet the real-time requirements, we use the TensorRT inference engine to quantify the trained model to obtain the inference model used in the system deployment. The compressed model will be deployed on the server side, and the server-side GPU needs to support INT8 calculation accuracy. The client program is developed using Qt5. The user can add a camera managed by the client in the client program, and select to detect the real-time video stream or the

local video file of the managed camera. After the server obtains the video data, the server detects and sends the detection result back to the client program for display to the user.

The intelligent monitoring video pedestrian detection system designed and implemented in this paper can perform pedestrian detection on the acquired video data, and the accuracy and real-time performance meet the requirements. The user client program also follows the principle of simplicity and ease of use, and the user is convenient to operate and logically clear. The system as a whole meets the requirements of requirements analysis and functional design. After testing, the system can meet the requirements of practical applications.

Key words： Deep learning　Pedestrian detection　R-FCN　TensorRT

目 录

1 绪论

1.1 研究的背景和意义

计算机视觉是一门让计算机能像人一样“看”的科学[1]。计算机视觉的目标是让计算机能处理静态图片或视频，理解图像中所包含的信息。近几年，随着计算机硬件的发展，计算资源得到大幅提升，从而促进了深度学习[2]方法的发展，而以深度学习方法为代表的人工智能技术的发展，使计算机视觉取得了较以往地大幅进步，也使得计算机视觉技术能够落地达到大规模应用成为了可能。

行人检测是计算机视觉这门学科中重要的一个领域。行人检测技术对图片或视频数据进行分析，判断图片或视频中是否存在行人，并标注出检测出的行人位置[3-5]。行人检测的应用前景非常的广泛，特别是在视频监控领域可以发挥出巨大作用，可以继续在检测之后加入跟踪、行为识别等系统，进一步增强视频监控系统的功能。

据咨询公司IHS Market在2016年发布的数据，中国共装有1.76亿个监控摄像头，其中由公安系统掌握的有2000万。“平安城市计划”是中国监控视频行业发展的主要推力，是公安公安部门为了应对城市人口剧增，不安全和不稳定因素的增加而进行的安防系统建设，这一由公安部门推进建设的监控系统又被称为天网工程。公安部门将天网工程摄像头所采集到的视频画面，通过网络传输到监控中心，由专人值守监督，并将一定时间间隔的视频保存起来，方便回放调出[6-8]。

传统的监控系统功能大多由视频采集、网络传输、视频显示、视频存储、视频回调等模块组成，并且由专人值守负责，但是，由于我国警力、安保人员不足，同时，安保压力增大的情况下，往往一个人需要负责很多路的监控视频，但是人的精力和注意力毕竟有限，无法处理太多的监控信息，这就导致了很多信息无法在第一时间获得关注并及时处理，造成了监控系统不能很好的做到预警的功能，且在调查取证时，也需要耗费大量时间。而现在，结合深度学习方法的人工智能技术正在进行大规模应用，可以代替人力去完成类似的繁琐且耗时耗力的任务。因此，在公共安全视频监控系统中应用深度学习技术对监控视频进行分析处理，对于提升监控系

统的效率具有重要意义，而行人检测又是其中至关重要的。而随着天网工程的推进，越来越多的单位、组织都开始建设自己的安防监控系统，因此，针对视频监控场景可用的智能行人检测系统对提升视频监控系统效率，为维护社会治安提供技术支持手段等具有重要意义，并且其后也有着广阔的市场前景。

1.2 国内外研究概况

行人检测在计算机视觉领域中一直都是研究的热点和难点，不管是学术界或工业界都进行了大量的工作，从而也促进了行人检测技术的发展和应用[9-11]。行人检测需要找出图片或视频帧中的行人，并标注出其位置，和人脸检测类似，也是目标检测的典型问题，但由于人体的姿态多样，过于复杂，不同的人体外观差异巨大，并且人体的附着物以及其他物体的遮挡背景变化复杂等问题也非常常见，因此准确的检测出行人目标具有非常大的难度。

传统行人检测方法主要从运动目标和人体特征的角度考虑。基于运动目标的方法思想较为简单，其假设摄像机静止不动，利用背景建模算法提取出运动的前景目标，然后再用分类器对提取出的运动目标进行分类，判断其是否是行人目标，这种方法在特定环境下识别效果获取很好，但检测图像中运动的物体更多情况下不仅只包含行人目标，在真实环境下，由于场景变化的复杂性，检测效果就不够理想[12-14]。

基于人体特征的方法，主要是从人体的形态特征考虑，但是，由于每个人体态、着装以及背景的不同，导致手工设计要选取的特征非常困难，常常带有经验和主观性，影响算法的效果。常见的选取的特征主要包括：颜色特征、边缘特征、纹理特征等[15-19]。在这类方法中，比较著名的且达到非常好的效果的一种算法，是在2005年由Navneet Dalal在IEEE国际计算机视觉与模式识别会议(CVPR)提出的基于方向梯度直方图(HOG[20])特征和支持向量机(SVM[21])分类器的行人检测算法[22]。方向梯度直方图(HOG)是一种边缘特征，这一特征很好的描述了行人的形状、外观信息，比Haar[23]特征更为强大，对光照变化和小量的空间平移不敏感，具有非常好的鲁棒性。基于方向梯度直方图的特征和支持向量机的分类方法在当时取得了非常好的检测效果，也促进了各种基于手动设计特征加分类器的模型的发展。

2012年，由Geoffrey Hinton的学生Alex Krizhevsky设计的AlexNet[24]神经网络在当年的ImageNet Large Scale Visual Recognition Challenge[25]比赛中获得冠军，自此，深度学习技术得到了广泛的关注，并获得了巨大的发展，基于深度学习学到的特征具有很强的层次表达能力和很好的鲁棒性，可以更好的解决一些视觉问题。使用深度学习方法来处理行人检测时，一般来说会将其归类为目标检测的问题。2014年，由加州大学伯克利分校的Ross Girshick等提出的R-CNN[26]目标检测框架首次将卷积神经网络(CNN)应用到目标检测中，取得了当时目标检测领域最好的精度效果。R-CNN的主要思想是，利用选择性搜索[27]对图像中可能存在目标的区域提出候选框，然后使用CNN网络提取出候选框区域的特征，最后使用线性分类器判断是否存在某类物体并使用线性回归器对候选框的位置作出修正。R-CNN在当时虽然取得了非常好的精度效果，但其缺点也非常明显，对选择性搜索提出的每一个候选框都会独立运行CNN获取特征，增大了计算量，也增加了冗余的计算。针对这个缺点，由Kaiming He等提出了SPP-Net[28]。R-CNN之所以需要对每个候选框独立做卷积计算提取特征，是因为卷积后一般还需要对卷积后的特征图做进一步的池化操作并将其输出固定到同样的长度才可以进行之后的分类和回归计算，而每个候选框本身的大小都不可能是相同的，因此SPP-Net提出一种新的空间金字塔池化层(Spatial Pyramid Pooling, SPP)，这样设计的新的池化层，可以将不同长度的特征图池化为相同长度的特征向量，因而只需要对图像做一次卷积计算即可，不再需要对每个候选框进行单独计算，减少了计算量，从而使整体的检测速度获得了提升。受SPP-Net中的空间金字塔池化层的启发，Ross Girshick又提出了Fast R-CNN[29]，在SPP-Net中需要对一个特征图池化为不同尺度的特征，构成了空间金字塔结构特征，但Fast R-CNN采用了空间池化的思想，只需要对最后的特征图做单层的空间池化，称为ROI Pooling层。Fast R-CNN对比与R-CNN还将候选框分类和候选框回归合并成为多任务(multi-task)模型，并使用multi-task loss来训练网络，实现了端到端(end-to-end)的训练方式，使得训练难度大大降低。Fast R-CNN虽然已经取得了非常好的精度和速度，但是其仍然存在瓶颈，问题就在于Fast R-CNN仍然与R-CNN一样采用的是选择性搜索来提出候选框，这种方法通常无法获取质量较高的候选框而且非常耗时，影响了模型整体的速度和精度。

为了解决这个瓶颈问题，Shaoqing Ren等提出了Faster R-CNN[30]，使用区域建议网络(Region Proposal Network, RPN)替代选择性搜索来生成候选框。Faster R-CNN中将RPN放在最后一个卷积层之后，通过训练RPN可以直接得到候选区域，且RPN也是一个卷积网络，这样就进一步减少了计算量提升了检测速度而且得到的候选框更加精确，为后续的分类器也从一定程度上提升了精度。

以R-CNN为代表的一系列基于深度学习的方法的目标检测技术在发展中流程变得越来越精简，精度也越来越高，检测速度也越来越快，是当前目标检测技术领域最主要的一个分支。

1.3 主要工作

本文根据工作内容共分为六章，论文的组织结构为：

第一章为绪论。首先对计算机视觉的发展，特别是在行人检测领域的发展做出了简述，然后提出了设计和实现本系统的背景、目的和意义，介绍了国内外在本领域的相关研究和发展成果。

第二章对本文设计实现的系统相关技术进行了介绍。首先是对目前流行的 学习框架特别是 Caffe 进行了介绍，然后介绍了本文系统的核心算法模型基于区域的全卷积网络(Region-based Fully Convolutional Network, R-FCN)，最后对部署阶段进行模型量化压缩的高性能推理引擎 TensorRT 进行了介绍。

第三章对本系统进行了需求分析。系统的需求分析明确了系统的功能性需求和非功能性需求，功能性需求分析对系统需要实现的功能模块做出了说明，非功能性需求分析对系统的可扩展性、性能、界面设计原则提出要求。

第四章是系统设计部分。根据对系统的需求分析，对系统进行了概要和详细两个层面的设计。

第五章是系统实现和测试。根据对系统的需求和设计，对系统进行了实现，并设计测试用例进行测试工作。

第六章是全文的总结和展望。对本文设计和实现的系统分析不足的地方，并对将来的工作提出意见。

2 相关技术介绍

本章将主要介绍该行人检测系统中使用的相关技术。首先介绍目前主流的几个深度学习框架，并将着重介绍本系统使用的 Caffe 框架。接着将会介绍本系统所采用的主要核心算法 R-FCN，一种基于区域的全卷积目标检测网络。最后将会介绍目前流行的用于实际项目部署时使用的高性能神经网络推理引擎 TensorRT。

2.1 深度学习框架

深度学习的火爆，也使得越来越多的人参与进来，因此有必要开发通用的深度学习框架便于验证、实现、以及在实际项目中使用。由此，越来越多的开发者、研究机构以及商业公司参与进来开了许多非常流行且实用的深度学习框架。

TensorFlow[31]最初由隶属于Google机器智能研究机构的Google大脑小组的研究员和工程师们开发出来，用于机器学习和深度神经网络方面的研究，但其通用性使其也可广泛用于其他计算领域。在 TensorFlow 中，张量(tensor)类型的数据，通过由“结点”(nodes)表示的数学操作以及由“线”(edges)表示的输入/输出关系组成的数据流图(Data Flow Graph)进行计算。TensorFlow 具有高度的灵活性、真正的可移植性、自动求解微分、多语言支持以及性能最优化等特性，并且 Google 已经将其开源。

PyTorch 是由 Facebook 推出的开源深度学习框架，其前身是 Torch。在 PyTorch 中对张量(tensor)进行操作时其风格与 Numpy[32]类似，非常方便易学，PyTorch 也提供了自动求导的能力，还将常用的神经网络层与损失函数优化方法等进行了封装，易于模型训练，支持共享内存的多进程并发。PyTorch 最大的优势是其采用了动态计算图(dynamic computational graph)结构，在处理一些问题时相比于大多数采用静态计算图的深度学习框架可以更加方便高效。

Caffe[33]是由贾扬清在其于加州大学伯克利分校读博士阶段创建的项目，后由伯克利人工智能研究实验室(BAIR)和社区开发者共同贡献完成的开源深度学习框架。Caffe 的设计按照表达能力强、速度快和模块化的思想。Caffe 相比于其他深度学习框架具有很多亮点，主要包括：

（1）模块化：Caffe 的模块化设计思想，允许对新数据格式、网络层和损失函数进行扩展。

（2）表示和实现分离：Caffe 的模型定义是用 Protocol Buffer 语言写进配置文件的。以任意有向无环图的形式，Caffe 支持网络架构。Caffe 会根据网络的需要来正确占用内存。通过一个函数调用，实现 CPU 和 GPU 之间的切换。

（3）测试覆盖：在 Caffe 中，每一个单一的模块都对应一个测试。

（4）Python 和 Matlab 接口：同时提供 Python 和 Matlab 接口

（5）预训练参考模型：针对一些视觉项目，Caffe 提供了一些常见的、经典的模型供使用者参考。

Caffe 的架构主要由几个模块组成：Blobs，Layers，Nets，Solver。Blob 是 Caffe 中数据的包装格式，也是数据、参数、梯度等存储、通信、操作的单位。Blob 以四元组的方式组织数据，按照(Number, Channel, Height, Width)的格式。Layer 是网络的基本单元，Caffe 中包含常见的各种神经网络层，并支持自定义实现的网络层。每一个 Layer 类型定义了三个至关重要的计算：1.初始化网络参数；2.前向传播的实现；3.反向传播的实现；并且对于前向传播和反向传播分别给出 CPU 实现和 GPU 实现。Net 是由 Layers 组成的有向无环图，通过 Layers 直接的连接和集合，搭建的网络结构。Solver 定义了针对网络模型的求解方法，记录网络的训练过程，保存网络模型参数，中断并恢复网络的训练过程，并支持自定义 Solver 从而实现不同的网络求解方式。

2.2 目标检测网络 R-FCN

随着 R-CNN 一系列的网络模型的提出，深度学习使得目标检测技术获得了巨大的发展和提升。Fast R-CNN 借鉴了 SPP-Net 的思想提出的 ROI Pooling 解决了不同尺寸候选框的特征提取问题，从而使得大部分的卷积计算是可以共享的，大大减少了计算量提升了模型的效率。Faster R-CNN 则进一步提出了区域候选网络(RPN)，通过使用共享计算得到的卷积特征，相比与之前的选择性搜索方法可以更加快速的得到建议的候选框。

2.2.1 基于区域的全卷积网络(R-FCN)

在图片分类领域，全卷积网络(Fully Comvolutional Network, FCN)[34]相比与利用全连接层实现特征融合和特征映射并分类的网络能取得更好的精度，例如深度残差神经网络(Deep Residual Neural Network, ResNet[35])。那么，是否可以将目标检测网络也使用全卷积网络来实现？于是，来自微软亚洲研究院视觉计算组的 Jifeng Dai 等人设计了基于区域的全卷积网络(Region-based Fully Convolutional Network, R-FCN)[36]。

现阶段，目标检测网络主要分为基于候选区域建议的方法，如 R-CNN 系列模型，和不基于候选区域建议的方法，如 SSD[37]、YOLO[38]等模型，但目前还是基于候选区域建议的网络效果会更好一些。与 Faster R-CNN 一样，R-FCN 也是基于候选区域建议的两级检测模型。以 Faster R-CNN 为例，对于基于候选区域建议的检测方法，实际上是分成了几个子网络，第一个用来在整张图片上做比较耗时的卷积、池化等操作以提取特征，这些操作与区域无关，是共享计算的，第二个子网络是用来产生候选的框，如区域建议网络(Region Proposal Network, RPN)，第三个子网络用来分类以及进一步对框进行回归修正，这个子网络适合区域有关的，必须对每个区域单独计算，衔接在这个子网络和前面两个子网络中间的就是 ROI Pooling 层。而我们希望将耗时的卷积操作都尽量移到前面共享的子网络上。我们可以发现在 ROI Pooling 层之前的卷积计算都是对所有候选框共享的，而 ROI Pooling 层之后的网络不是所有候选框共享的，主要原因就是因为这一部分的网络的作用是对每个候选框进行分类和回归，不能进行共享计算。问题就在于，ROI Pooling 层之前的网络具有“位置不敏感性”，如果我们将一个卷积网络的所有卷积层都放在共享计算这部分，则后面只剩下全连接层，这样的目标检测网络是位置不敏感的，其检测精度会较低，也会浪费掉分类网络的强大的分类能力。而深度残差网络(ResNet)作者在其论文中为了解决位置不敏感这个缺点，作出了一点让步，将 ROI Pooling 层放置在了卷积层之间，这样 ROI Pooling 层前后都有卷积层，并且之后的卷积层不共享计算，是针对每个 ROI 进行特征提取的，这样设计使得网络获得了位置敏感性，但是却牺牲了模型运行速度。于是 R-FCN 提出了新的位置敏感得分图(position-sensitive score map)和位置感敏感兴趣区域池化层(position-sensitive ROI pooling)以解决模型精度和模型速度之间的矛盾，其思想和具体做法将在下一节中详细描述。

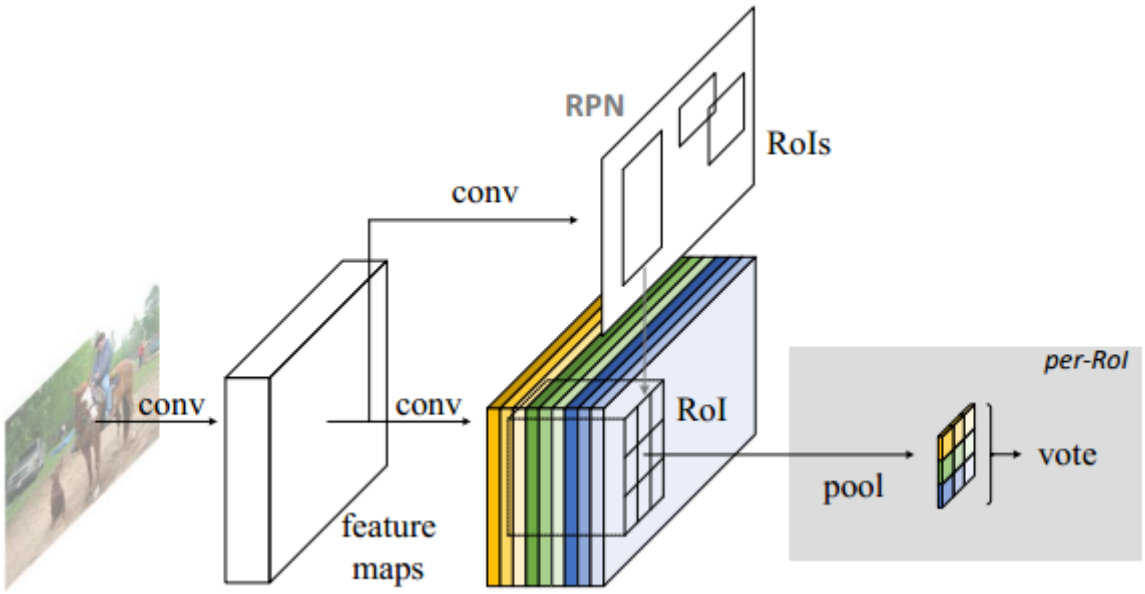

图 2-1 R-FCN 网络结构

R-FCN 的网络结构如图 2-1 所示。我们以一张待检测图片为例，分析 R-FCN 的整个检测步骤：

（1）对待检测图像进行相应的预处理操作。

（2）待检测图像经过预处理之后输入到用于提取特征的骨干网络中，例如深度残差网络 RestNet-101 去除左后的平均池化层和全连接层，保留的部分用于特征提取。

（3）在骨干网络的最后一个卷积层获得的特征图上引出三个分支，第一个分支是将获得特征图继续输入区域建议网络(RPN)以获得感兴趣区域。第二个分支是在特征图上使用卷积层获得一个 K^2*(C+1)维的位置敏感得分图用来分类。第三个分支就是在特征图上使用卷积层获得一个 4*K^2 维的位置敏感得分图，用来给候选框的坐标进行微调。其中，K 为特征图长、宽等分系数，C 为检测的目标类别数。

（4）最后，在 K^2*(C+1)维的位置敏感得分图和 4*K^2 维的位置敏感得分图上分别执行位置敏感感兴趣区域池化操作，获得对应的类别和位置信息。

训练时，由于采用了和 Faster R-CNN 类似的架构，R-FCN 也可以使用端到端的训方式。R-FCN 的损失函数如下公式所示为：

$$L(s, t_{x,y,w,h}) = L_{cls}(S_{c^*}) + \lambda[c^* > 0]L_{reg}(t, t^*) \tag{2-1}$$

$$L_{cls}(s_{c^*}) = -log(s_{c^*}) \tag{2-2}$$

$$L_{reg}(t,t^*) = \sum_{i\in\{x,y,w,h\}} smooth_{L1}(t_i - t_i^*) \quad (2\text{-}3)$$

其中 c*代表区域中的目标类别，c* = 0 意味着是背景，t*代表真实的框，[c* > 0]是一个指示函数当满足条件时为 1 否则为 0，λ 是平衡权重参数，在论文中作者设置为 1。对于每一个 ROI，我们计算其分类的交叉熵(cross-entropy)损失[39]和当其不属于背景是框的回归损失[40]之和。而对于一个 ROI 来说，其是属于目标物体的框或背景是根据其与所有真实框的最大重叠率(intersection-over-union, IoU)来决定，当重叠率大于设定的阈值时，即认为是物体框。

2.2.2 位置敏感特征图与位置敏感感兴趣区域池化层

前面我们提到，R-FCN 将全卷积网络引入到目标检测的网络中，但在此之前，已经有人尝试过利用卷积层代替 Faster R-CNN 中的全连接层，然而检测效果却表现得更差。对于这个问题 R-FCN 的作者认为，对于图像分类任务，一张图片进行一定的平移，模型应该还是给出同样的结果，所以，对于分类任务，我们需要增强模型的平移不变性(translation invariance)，而对于目标检测任务，因为我们是需要标定出目标的位置，如果目标发生了一定的平移，我们希望模型还是能给出正确的坐标，这就要求模型需要有平移变换性(translation variance)，即对位置变化敏感。深度残差网络(ResNet)在其论文中提出的方法是在卷积层之间插入 ROI Pooling 层，这样就使得后面的网络获得了平移变换性，但这样的设计牺牲了模型的速度。为了利用全卷积网络强大的分类能力和计算效率并将平移变换性引入到全卷积网络中，基于区域的全卷积网络(R-FCN)提出了位置敏感得分图(position-sensitive score map)和位置敏感感兴趣区域池化层(position-sensitive ROI pooling)来解决这个问题。

作者在设计位置敏感得分图时认为，如果一个 ROI 含有一个类别 C 的物体，那么我们将该 ROI 划分为 K*K 个区域，分别表示该物体的各个部位，比如假设该 ROI 中含有人这个目标，K 设置为 3，那么就将目标“人”划分为了 9 个子区域，top-center 区域毫无疑问应该是人的头部，而 bottom-center 应该是人的脚部，而将 ROI 划分为 K*K 个区域就是希望这个 ROI 在其中的每一个子区域都应该还有该类别 C 物体的各个部位。当所有的子区域都含有各自对应的物体的相应部位后，分类器才会将该 ROI 判断为该类物体 C。物体的各个部位和 ROI 的这些子区域是“一一映射”的关系。到

这里，我们认为一个 ROI 必须是 K*K 个子区域都含有该物体的相应部位，才能判断该 ROI 属于该物体，如果该物体的很多部位没有出现在相应的子区域中，那么就判断该 ROI 为背景类别。

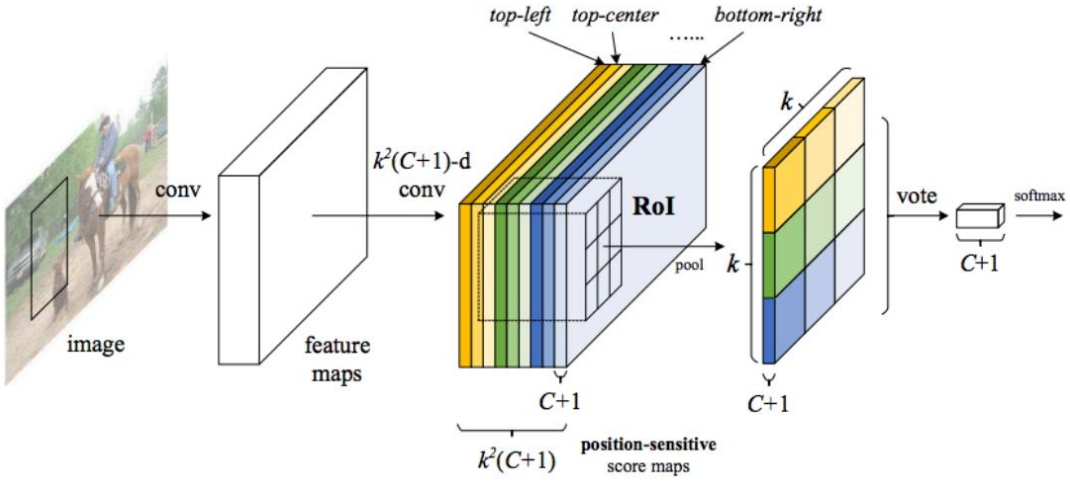

图 2-2 R-FCN 关键思想

到这里，问题就在于网络如何判断一个 ROI 的 K*K 个子区域都含有相应部位？前面我们假设知道每个子区域是否含有物体的相应部位，那么我们就能判断该 ROI 是否属于该物体还是属于背景。那么现在的任务就是判断 ROI 子区域是否含有物体的相应部位。这就是位置敏感得分图(position-sensitve score map)的核心设计思想。R-FCN 会在共享卷积层的最后再接上一层卷积，而该卷积层就是位置敏感得分图，该得分图的含义就是，首先它就是一层卷积层，它的宽和高与共享卷积层一样，即与共享卷积层具有同样的感受野，但是它的通道数为 K^2*(C+1)，如图 2-2 所示。其中 C 表示物体类别数，再加上 1 个背景类别，共(C+1)类，而每个类别都有 K*K 个得分图(score map)。现在我们针对其中的一个类别来说明，假设目标类别是“人”，那么其有 K*K 个得分图，每一个得分图表示原始图像中的哪些位置含有人的某个部位，该得分图会在含有对应的人体的某个部位的位置有高的响应值，也就是说每一个得分图都是用来描述人体的其中一个部位出现在该得分图的何处，而在出现地方就有高响应值。既然是这样，那么我们只要将 ROI 的各个子区域对应到属于人的每一个得分图上然后获取它的响应值就好。但是要注意的是，由于一个得分图都是只属于一个类别的一个部位，所以 ROI 的第 i 个子区域一定要到第 i 张得分图上去寻找对应

区域的响应值，因为 ROI 的第 i 个子区域需要的部位和第 i 张得分图关注的部位是对应的。那么现在该 ROI 的 K*K 个子区域都已经分别在属于人的 K*K 个得分图上找到其响应值了，如果这些响应值都很高，就证明该 ROI 是属于“人”这个类别。

上面我们了解到将 ROI 划分为 K*K 个子区域并在各个类别的得分图上找到其每个子区域的响应值，但是如何找到其响应值？这就需要使用位置敏感的感兴趣区域池化(position-sensitive ROI pooling)操作。通过区域建议网络(RPN)提取出来的感兴趣区域(ROI)包含了坐标和长宽 4 个属性的值，也即是说不同的 ROI 能够对应到得分图的不通位置，而一个 ROI 会分成 K*K 个子区域，每个子区域都对应到得分图上的某一个区域，那么池化操作就是在该子区域对应的得分图上的子区域执行，并且执行的是平均池化。前面提到，第 i 个子区域应该在第 i 个得分图上寻找响应值，那么也就是在第 i 个得分图上的第 i 个子区域对应的位置上进行平均池化操作。由于共有(C+1)个类别，所以每个类别都要进行相同方式的池化操作。图 2-2 已经很明显的说明了池化的方式，对于每个类别，它都有 K*K 个得分图，那么按照上述池化方式，ROI 可以针对该类别获得 K*K 个值，一共有(C+1)个类别，那么一个 ROI 就可以得到 K^2*(C+1)个值，就是图 2-2 中所示的通道数为(C+1)，尺寸为 K*K 的特征图。那么对于每个类别，该类别的 K*K 个值都表示该 ROI 属于该类别的响应值，将这 K*K 个数相加就得到该类别的得分，一共有(C+1)个得分，将这(C+1)个得分用 softmax 函数[41]计算就可以得到每个类别的概率。

上述的位置敏感得分图和位置敏感感兴趣区域池化得到的值是用来分类的，那么同样的，使用相应的操作就可以得到兴趣区域回归的结果。按照位置敏感得分图和位置敏感兴趣区域池化操作的思路，每一个 ROI 都可以得到(C+1)个数作为每个类别的得分，对于边框回归，每个 ROI 还需要坐标和长宽四个属性值作为回归偏移量，仿照分类的思路，还需要一个类似的位置敏感得分图用于回归的得分图。那么，我们可以在 ResNet 的共享卷积层的最后一层上连接一个与分类的位置敏感得分图并行的得分图用于边框的回归修正，将其命名为回归得分图(regression score map)，而该回归得分的维度应当是 4*K^2，然后经过位置敏感的兴趣区域池化操作后，每一个 ROI 就可以得到坐标和长宽四个值作为该 ROI 的修正偏移量。

2.3 高性能神经网络推理引擎 TensorRT

TensorRT[42]是由英伟达(NVIDIA)公司发布的一种高性能神经网络推理(inference)引擎，用于在生产环境中部署深度学习应用程序。最典型的应用是图片分类、分割和目标检测等，可以提供最大的推理吞吐量和效率。TensorRT 是第一款可编程推理加速器，能加速现有和未来的网络架构。TensorRT 需要英伟达的统一计算设备架构(Compute Unified Device Architecture, CUDA) [43]的支持。TensorRT 包含一个为优化生产环境中部署深度学习模型而创建的库，可获取经过训练的神经网络，通常使用 32 位浮点数，并针对降低精度的 16 位浮点数或者 8 位整型数运算来优化这些网络。借助 CUDA 的可编程线，TensorRT 能够加速助推深度神经网络日益多样化、复杂的增长趋势。通过 TensorRT 的大幅加速，服务提供商能够以经济实惠的成本部署这些计算密集型人工智能工作负载。

用深度神经网络解决监督机器学习问题包含两个步骤：第一步是使用 GPU 对海量标签数据进行深度神经网络训练，训练时需要迭代的通过网络进行前向传播和反向传播，最终会生产训练好的模型文件。第二步是推理，即使用训练好的模型对新数据作出预测，仅需通过网络进行前向传播。推理阶段对比与训练阶段，有以下几点不同：

（1）推理阶段网络权值已经固定下来，无反向传播过程，因此可以对计算图进行优化。

（2）推理阶段网络的输入输出大小固定，可以针对内存进行优化。

（3）推理阶段的批尺寸(batch size)要小很多，这是因为推理阶段对延迟(latency)非常敏感，如果批尺寸很大，吞吐可以达到很大，但是就不能做到较低延迟以获得更好的实时性。

（4）推理阶段可以使用低精度技术，训练的时候因为要保证前、后向传播，每次梯度的更新是很微小的，这个时候需要相对较高的精度，一般采用 32 位的浮点型来处理数据，但是在推理的时候，对精度的要求没有这么高，很多研究表明可以用低精度，如 16 位浮点型或 8 位的整型来做推断，并且网络没有特别大的精度损失。低精度计算的好处一方面是可以减少计算量，另一方面是模型需要的空间减少，不

管是权值的存储还是中间值的存储，应用更低的精度，模型大小也会相应减小。

使用 TensorRT 包含构建(build)和部署(deployment)两个阶段。在构建阶段，TensorRT 对网络配置进行优化，并生成一个优化了的 PLAN 文件用于计算深度神经网络的前向传播，这个 PLAN 文件可以序列化存储在内存或磁盘上。部署阶段通常采用长时间运行的服务或用户应用程序的形式，该服务或用户应用程序接受批量输入数据，通过对输入数据执行 PLAN 文件来执行推理，并返回批量输出数据。在部署阶段，TensorRT 以最小化延迟和最大化吞吐量运行优化了的网络。

使用 TensorRT，无需再部署环境中安装并运行训练时所用到的深度学习框架。也是因为 TensorRT 具有的特性使其在应用部署时越来越流行，TensorRT 具有如下高级特征：

（1）插件支持。TensorRT 是支持插件(Plugin)的，即用户可以在某些层 TensorRT 没有支持的情况下，自定义实现需要的特殊层。

（2）低精度支持。低精度指的是前面所说的 16 位浮点型和 8 位整型运算，其中 16 位浮点型主要是英伟达 Pascal 架构的 P100 和 V100 系列 GPU，而 8 位整型主要是 P4 和 P40 系列 GPU。

（3）Python 接口和更多的框架支持。TensorRT 除了提供 C++的接口外，还支持 Python 语言的 API。TensorRT 在做推理的时候是不再需要深度学习框架的，例如使用 Caffe 训练的模型，TensorRT 把模型导入之后是不再需要 Caffe 框架去做推理了，因此在部署应用时，不再需要安装训练阶段使用的深度学习框架，只需要转换后的模型。

2.4 本章小结

本章主要介绍了智能监控视频行人检测系统中用到的相关技术。首先介绍了当前流行的几个深度学习框架，并对本文使用的 Caffe 框架做了重点介绍。然后介绍了本文使用的核心算法模型基于区域的全卷积网络(R-FCN)，并对其中的关键点做出描述。最后，介绍了高性能推理引擎 TensorRT，可以用于在部署阶段对网络模型进行量化压缩，从而加速推理过程，提升模型的运行速度。

3 行人检测系统需求分析

系统的需求分析是一个项目不可或缺的部分。需求分析一般是在项目的早期就开始进行。通过需求分析方法对用户使用本系统的各种需求等进行分析，撰写文档记录，对于系统需要完成的功能进行尽可能详细的描述，对于系统的评估指标等进行明确的定义。

3.1 系统需求分析总体概述

随着平安城市计划的推进，传统的视频监控系统由于功能单一，只能通过专人值守查看监控画面，而由于人手不足等问题，导致大量监控视频无法及时处理信息，从而无法为群体性、公共安全等事件做出及时有效预警，导致社会不稳定因素及人民群众人身及财产安全损失。

本文设计并实现的智能监控视频行人检测系统，可以应用于公共场所或企业园区等场景，并在这些场景下对场景中的行人做出检测，对可能出现的异常情况做出预警，可以大大提升现有视频监控系统的工作效率，提升安全防护效果。系统可以获取网络摄像头的视频流或者录制的本地监控视频文件，对视频流进行处理后，可以检测视频画面中的行人，对行人进行检测和统计后，将处理的视频流和检测的结果信息显示出来，这样值守人员的压力可以大大减轻，只需要重点关注预警区域情况并做出相应措施。

3.1.1 系统用户特征分析

根据本系统的应用场景，我们对本系统使用用户进行分析，主要包括以下几个角色：

（1）系统管理员。系统管理员需要对整个系统的运行状态进行管理，需要了解系统的各个模块，负责对检测模型进行更新替换，对普通用户的账号进行管理。

（2）安保值守人员。安保值守人员是系统的日常使用者，即是系统的普通用户，本系统的主要功能也是针对安保值守人员使用需求进行设计。因而，系统需要提供

便于交互的界面和易于理解的操作逻辑，帮助提高安保值守人员的工作效率，减轻安保值守人员的工作压力。

（3）运维人员。系统运维人员负责系统的部署与日常维护，需要对本系统的运行非常了解。系统也应该便于安装与维护以减少运维人员的压力，保障系统自身的可靠性。

3.1.2 系统用例分析

本系统的主要用户是安保值守人员。当用户在使用本系统时，运行其客户端程序，首先需要进行账户验证，当账户验证通过之后，用户可以选择实时视频流检测模块、本地视频文件检测模块、摄像头信息管理模块、账户管理模块等功能模块进行相应操作。图 3-2 是本系统的用例图。

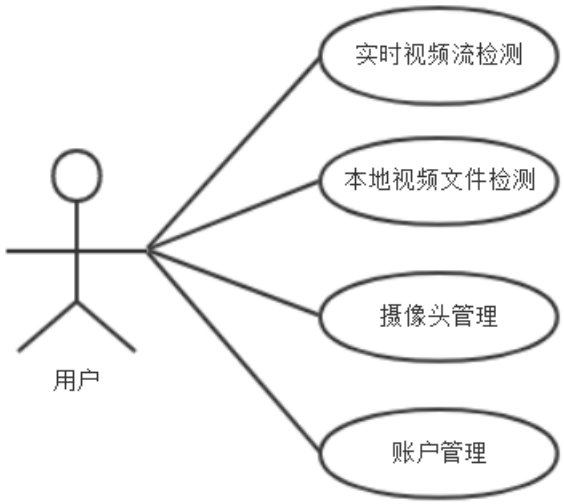

图 3-2　用户用例图

点击进入实时视频流检测模块后，用户可以选择需要进行检测的摄像头，然后点击开始检测之后，系统对选择的实时摄像头视频流进行检测，并将结果返回到客户端程序进行显示，点击停止检测即可停止对当前摄像头视频流的检测。点击进入本地监控视频文件检测模块后，用户可以选择录制好的本地监控视频文件并在文件对话框中选择确认，点击开始检测之后，系统将会对本地视频文件中的行人进行检测，并将结果在客户端程序上显示，点击停止检测即可停止对本地文件的检测。点击进入摄像头管理模块之后，用户可以对摄像头信息进行编辑，包括摄像头的 IP 地

址、用户名、密码、位置等信息。用户可以添加、修改、删除摄像头信息条目。点击进入账户管理模块后，用户可以对自己的账户密码进行修改，系统将在验证原密码和新密码二次确认后通过账户密码的修改，以更新用户密码。

3.2 系统的功能性需求

功能性需求是对针对系统用户的需求而需要实现的功能进行分析，功能性需求是对系统用户的行为进行描述，与系统的功能有关。本系统是基于监控视频流的行人检测系统，用户使用时本系统时首先需要输入账号密码进行验证，账户通过验证后即可进入系统主界面，通过选择不同的模块，可以进入该模块完成相关操作后，系统将实现各种功能。系统的主要功能如下：

（1）实时摄像头视频流检测

实时视频流检测是系统的核心功能的一部分。用户在进入客户端程序主界面之后选择实时摄像头视频流检测进入本模块，在下拉框中选择待检测的摄像头，并点击开始检测按钮开始检测，点击停止检测按钮停止检测。

（2）本地监控视频文件检测

本地监控视频文件检测是系统的核心功能之一。用户在主界面选择了本地监控视频文件检测模块后，可以选择本地的监控视频文件，同样的可以通过点击按钮进行检测和停止检测。

（3）摄像头信息管理

摄像头信息管理是系统的特色功能。系统在进行实时摄像头视频流行人检测之前，需要获取摄像头的相关信息从而获取实时的视频画面。如果每次进行检测时，都需要用户输入摄像头信息的话，对于用户来说这个操作就过于繁琐，且费事费力还容易出现输入信息错误的情况，影响用户体验，因此，本系统应该提供摄像头管理功能。在本模块，用户可以添加、修改、删除摄像头信息，方便使用时选取，无需每次输入相关信息，提升了用户的效率和用户体验。

（4）账户管理

账户管理是系统的基础功能。本着保护隐私和视频数据安全的考虑，我们需要

对使用系统的用户进行管理。同时，用户在使用系统时也为了安全考量需要对自己的账户密码进行定期修改维护等。

（5）结果展示

结果展示模块是系统的功能之一。系统的检测一般是在服务器端完成，而且，由于检测功能模块的不同，包含实时摄像头视频流检测和本地监控视频文件检测，我们需要对检测的结果进行处理后发送到用户的客户端程序中直观的显示出来。

3.3 系统的非功能性需求

非功能性需求是需求的一个重要组成部分，它影响了系统的架构设计，需要开发人员重点关注。本文的智能监控视频行人检测系统的非功能性需求主要如下：

（1）高可扩展性。本系统设计需求为运行在服务器端的行人检测模块，为了满足将来的场景需要，还应该能够在不同的设备上运行。且行人检测的结果还可以在以后的业务场景中根据需求添加不同的模块，增强系统的智能性，提高系统的工作效率。

（2）性能需求。本系统接收实时监控视频流，因此系统要在保证检测精度的情况下，提高系统的运行速度，以达到系统的实时性目标。由于行人检测的应用场景，模型对于多目标检测和环境变化的抗干扰能力也需要达到鲁棒性。同时，对于部署来说，模型还应该进行压缩，从而满足在不损失过多精度的情况下，减小模型所占用的磁盘和内存空间。

（3）界面简洁。对于安保值守人员来说，其主要需要关注监控视频信息，因此，在进行界面设计时应该保持简洁的原则，在保证功能完善的前提下，使主要信息处在突出位置，不存在过多的选项或显示干扰。

3.4 本章小结

本章的工作是对本系统进行需求分析，首先对系统的用户特征进行描述并进行用例分析，然后对系统的功能性需求和非功能性需求做出了描述。

4 行人检测系统的设计

系统的设计是一个项目的关键，是系统实现的前提。通过对系统需求进行分析之后，对系统进行完整的设计，是系统实现时的指南，良好的系统设计可以提升系统的开发效率增加系统的健壮性。本章首先对系统的总体结构进行设计，然后将会对系统的各个功能模块进行概要和详细两个层面的设计。

4.1 系统总体结构设计

本系统是在传统监控视频系统的基础之上，通过使用深度学习技术，检测监控视频流中的行人。系统首先需要有针对应用场景的训练后的模型，然后利用训练好的模型部署后进行监控视频的检测。

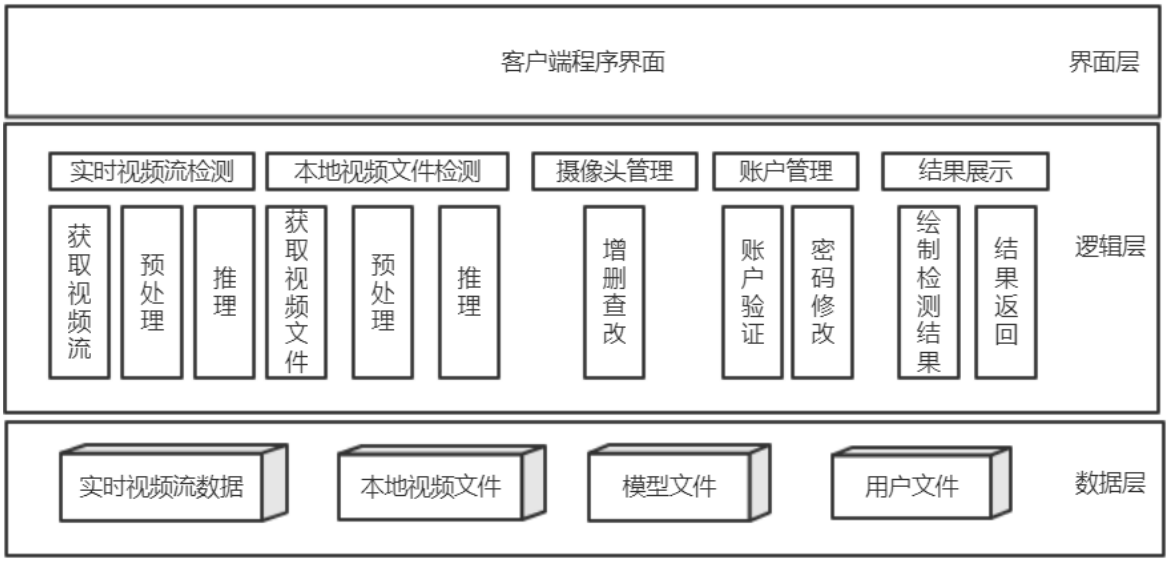

图 4-1 总体系统结构设计图

训练应用于场景的模型，首先需要获取应用场景的监控视频数据，然后制作数据集后在训练服务器上训练模型。将训练好的模型通过 TensorRT 压缩之后部署在服务器上，将用户选择的摄像头实时视频流或本地视频文件进行检测后得到检测结果并发送到用户机器上客户端程序显示出来。图 4-1 所示是本文设计实现的行人检测系统的整体架构图。

（1）界面层

界面层是用户与系统发生交互的系统层次。主要由客户端程序的界面进行交互，包括不同功能模块的视图、按钮、对话框、检测结果的展示等。界面层是用户对系统功能的最直观的感受，是系统在给用户提供功能时向用户进行展示和信息交互的工具，设计一个简洁但功能完善且漂亮美观的界面可以提升用户对系统的认知度和使用体验。

（2）逻辑层

逻辑层是系统功能实现的核心层次。用户在界面层中进行的各种操作以及系统需要完成的各项功能，都是在逻辑层中具体实现的。本文设计实现的是一个智能监控视频行人检测系统，实现较好精度并在实际部署中能达到较好检测速度的网络模型，以及相应的功能模块是本系统的关键。

（3）数据层

数据层是系统读取和存储数据的系统层次，包括部署的网络模型文件的读取，检测数据的读取，摄像头信息的读写，账户信息的读写。

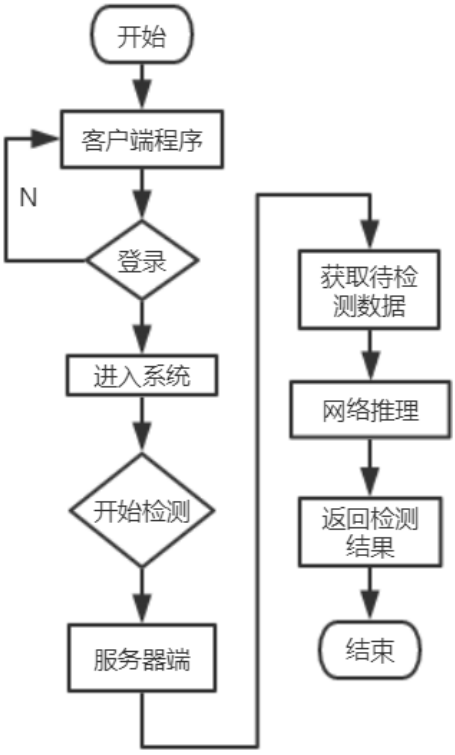

图 4-2 系统流程图

图 4-2 所示是系统的操作流程。用户运行客户端程序后，首先需要输入账号密码进行验证，验证通过后即可进入主界面，然后可以选择本地视频文件或摄像头视频流，点击开始检测，向服务器端发送检测请求，服务器端在收到请求后，即可获取本地视频或目标摄像头的监控视频流，将其输入到 TensorRT 压缩之后的基于区域的全卷积网络(R-FCN)模型开始检测，并将检测的结果发送到用户端客户端程序显示。

4.2 系统概要设计

概要设计是对系统划分其模块，同时还需要对每个模块的层次结构进行描述。根据上一章的需求分析和上一小节的总体设计，本系统主要包括网络摄像头实时视频流检测模块、本地视频检测模块、摄像头信息管理模块、账户管理模块、结果展示模块。

图 4-3 是系统的总体功能结构图。

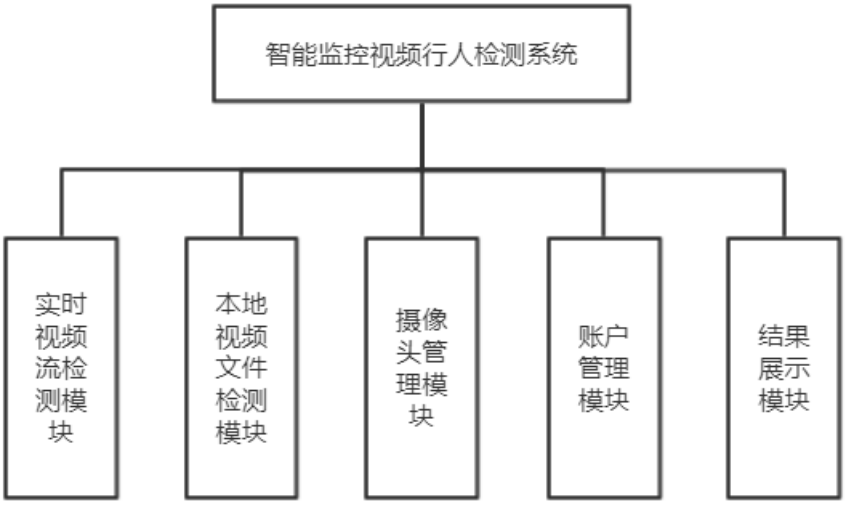

图 4-3　系统总体功能结构图

（1）实时视频流检测模块

本文设计实现的智能监控视频行人检测系统主要的场景是对网络摄像头的实时监控视频流检测行人并将检测结果发送到用户客户端程序。用户可以选择网络摄像头，指定检测模块的输入数据源，然后点击开始检测按钮，即可发送指令至服务器，服务器将获取摄像头实时视频流进行检测。图 4-4 所示是实时视频流检测模块的结构图，包含选择摄像头、开始检测、停止检测三个子功能。

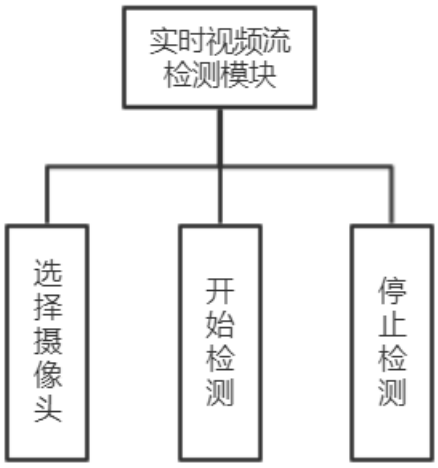

图 4-4　实时视频流检测模块结构图

选择摄像头：用户可以从添加的摄像头列表中选取一个摄像头，其已经包含了摄像头的 IP 地址、账户名、密码等信息，这样用户在使用时可以很方便的选择要进行检测的监控视频流而无需每次输入摄像头的 IP 地址等信息。

开始检测功能：用户选择好摄像头之后，点击开始检测按钮，系统将会获取摄像头的实时视频流开始检测。

停止检测功能：当用户不再需要对选择的摄像头视频流进行检测时，可以点击停止检测按钮，停止对当前选择摄像头视频流的检测。

（2）本地视频检测模块

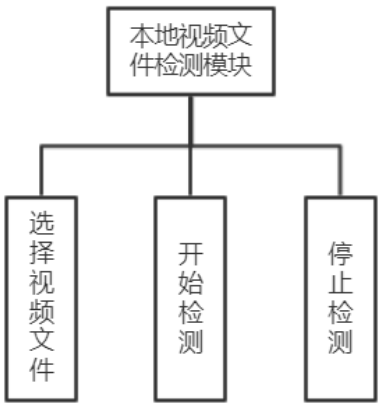

图 4-5　本地检测模块结构图

本地视频检测模块是本系统提供的检测功能之一，用户可以对录制的监控视频进行检测，用户选择本地的视频文件之后，点击开始检测，服务器就会获取本地视

频并开始检测。本地视频检测模块主要包括本地视频文件选择、开始检测、停止检测三个功能。本地视频检测模块的结构图如图 4-5 所示。

本地视频文件选择：用户点击选择本地视频文件，客户端程序会通过会话窗口供用户选择本地的视频文件。

开始检测功能：用户选择了本地视频文件之后，点击开始检测按钮，服务端将会读取本地视频文件，并开始进行检测。

停止检测功能：当用户不再需要对当前选择的视频文件进行检测时，可以点击停止检测按钮，系统将会停止对视频文件进行检测。

（3）摄像头管理模块

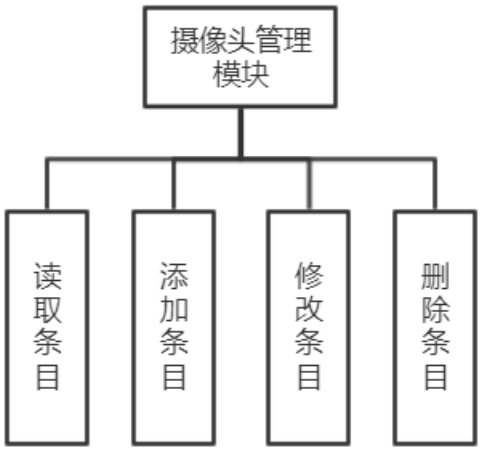

图 4-6　摄像头管理模块结构图

如图 4-6 所示是摄像头管理模块的结构图。摄像头管理模块是为了方便用户使用系统快捷地进行行人检测而进行设计的。用户可以在摄像头管理模块添加、修改、删除摄像头信息，包含 IP 地址、用户名、密码、位置等信息。这样，用户在实际使用时不需要每次繁琐地填写摄像头信息，极大地提升了用户的体验。摄像头管理模块主要包括读取、添加、修改、删除四个子功能。

用户初次使用系统时，可以进入摄像头管理模块，添加需要进行检测的摄像头信息，包含 IP 地址、用户名、密码、位置等信息。并在系统以后的使用过程中，可以对存储的摄像头记录进行修改、删除等操作。

（4）账户管理模块。

账户信息管理是一个系统的基础功能之一。在本系统中，用户可以在登录时，

系统将会对用户输入的账户密码进行验证，验证通过进入系统后，用户可以通过账户管理模块对自己当前使用的账号信息进行修改。账户管理模块主要包括账户验证、密码修改、校验等功能。账户管理模块的结构如图 4-7 所示。

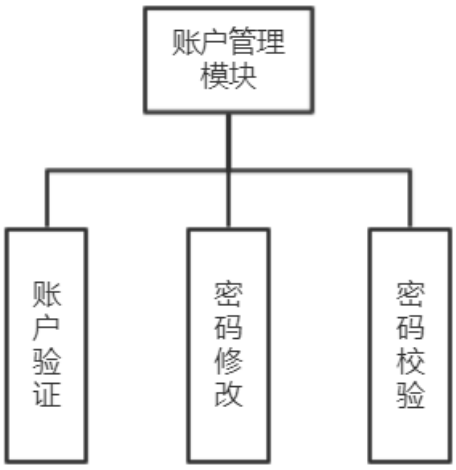

图 4-7　账户管理模块结构图

用户在登录系统时，系统将会对用户名和密码进行验证。验证通过后，用户可以在账户管理模块对自己的账号密码进行更新。

更新密码时，首先要对用户输入的原密码进行验证，用户还需要输入新密码以及对新密码的再次确认，当新密码的两次验证通过后，用户即可成功使用新密码替换旧密码。

（5）结果展示模块。

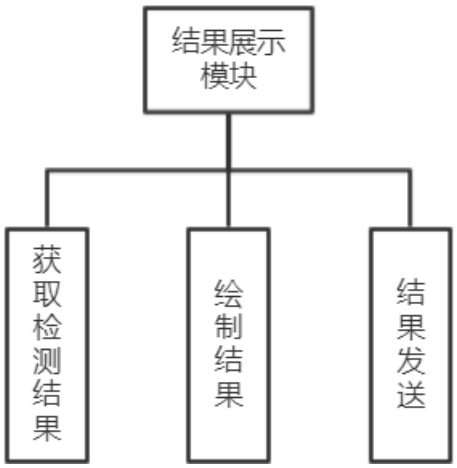

图 4-8　结果展示模块结构图

结果展示模块是系统在对用户进行的检测任务，获取实时的检测结果后，在用户的客户端程序上进行展示的功能。针对系统进行检测的数据的不同，结果展示模块将对实时的摄像头视频流和本地视频监控文件分别进行处理。结果展示模块的结构图如图 4-8 所示。

4.3 系统详细设计

在本节中，我们将对上一节概要设计划分的模块做出详细设计，为了方便理解，我们将对每个模块画出其流程图。

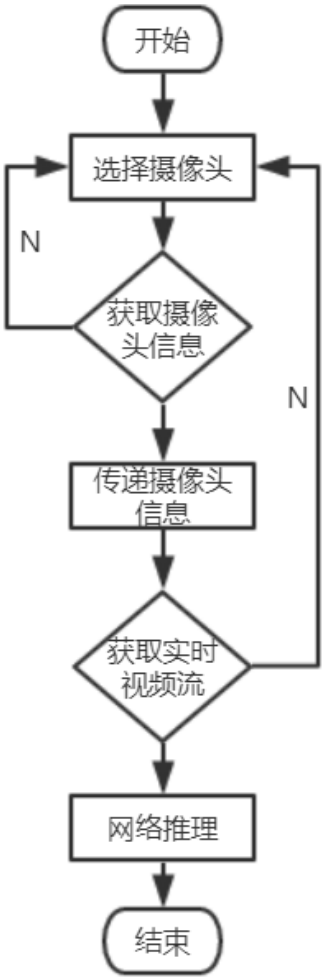

图 4-10　实时视频流检测流程图

实时视频流检测模块。图 4-10 所示的是实时视频流检测模块的流程图。用户通过了账户验证进入程序主界面后，默认视图为实时视频流检测模块。在实时视频流

检测模块中，用户可以选择需要进行检测的网络摄像头，点击开始检测按钮，程序将会把用户选择的摄像头 IP、用户名、密码等连接信息发送到服务器上，服务器端接受到客户端发送的摄像头信息后，就会开始读取摄像头的实时视频流，并输入到部署的模型中进行检测，如果用户需要对当前摄像头视频流停止检测，点击界面中的停止检测功能即可停止。

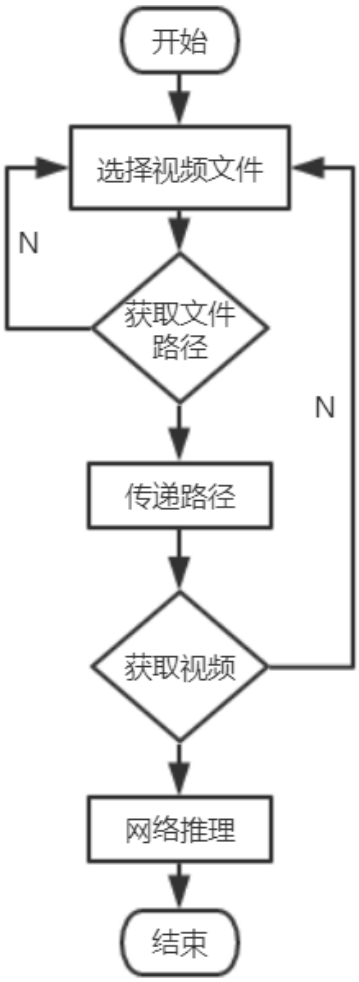

图 4-11　本地检测流程图

本地监控视频文件检测模块。图 4-11 所示是本地视频文件检测模块的流程图。用户进入客户端程序后，可以点击进入本地视频文件检测模块。点击文本框程序将显示文件对话框，用户可以选择本地视频文件并确认选择后，点击开始检测，客户端程序将会把本地视频文件的路径发送到服务器端，服务器端在接收到客户端发送的数据路径等数据之后，将会从本地获取视频数据，输入到模型中进行检测，当用

户不再需要对本地视频文件进行检测时，可以点击界面上的停止检测按钮，即可停止检测功能。

摄像头管理模块。图 4-12 所示是摄像头管理模块的流程图。摄像头管理模块是为了方便用户使用本系统，不必需要在每次对实时视频流进行检测时输入摄像头的 IP 地址、用户名、密码等连接信息。用户在进入客户端程序主界面后，可以选择进入摄像头管理模块，用户可以查看已经添加的摄像头信息，并可以点击添加按钮添加一个摄像头信息条目，包括 IP 地址、用户名、密码、安装位置信息。用户也可以点击一条展示出来的存储的摄像头信息，选中这条记录之后点击修改按钮可以对该条存储的摄像头信息进行修改，点击删除按钮可以删除不再需要存储的摄像头信息。用户完成相应操作之后，客户端程序将操作指令及数据信息发送到服务器端，服务器端将会完成摄像头数据的添加、修改、删除操作。

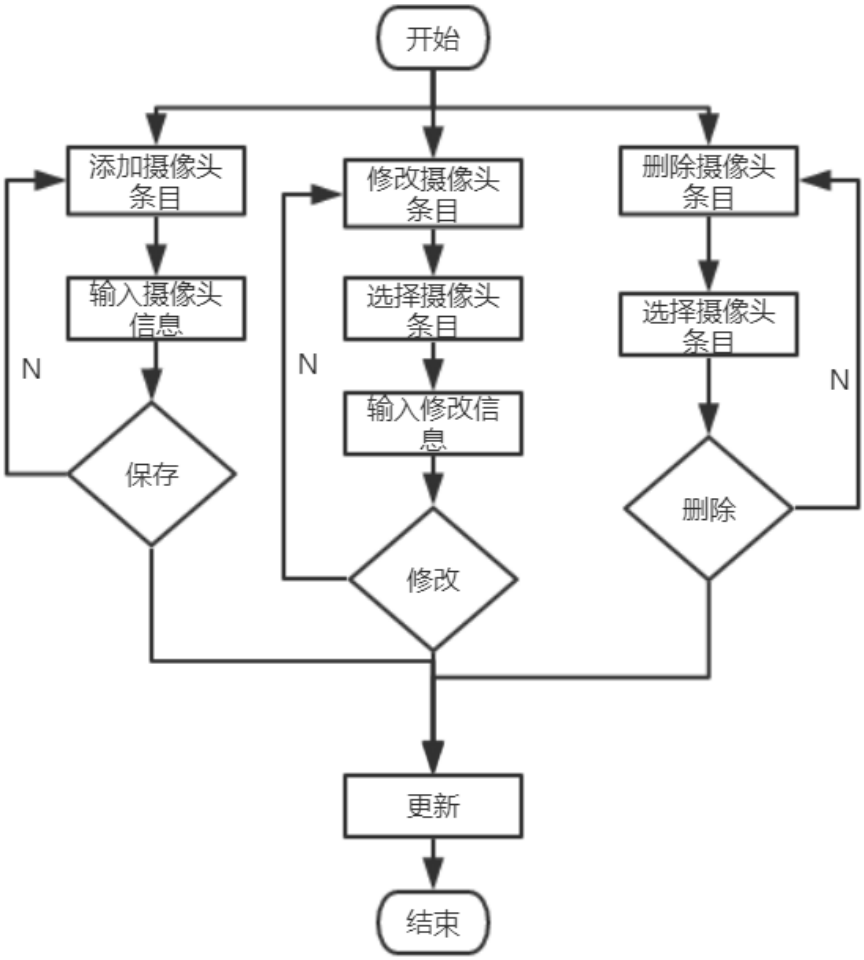

图4-12　摄像头管理模块流程图

账户管理模块。图 4-13 所示是账户管理模块的流程图。账户管理是一个系统的基础功能。本系统中用户运行客户端程序，首先需要输入用户名和密码进行验证，通过验证后用户进入客户端程序主界面可以选择进入账户管理模块。输入原密码，并两次输入新密码后，用户点击修改按钮，客户端程序首先会验证两次输入的密码是否一致，如果一致，会将原密码与保存的数据进行对比，对比通过后，会将用户输入的新密码替换掉原来的就密码。

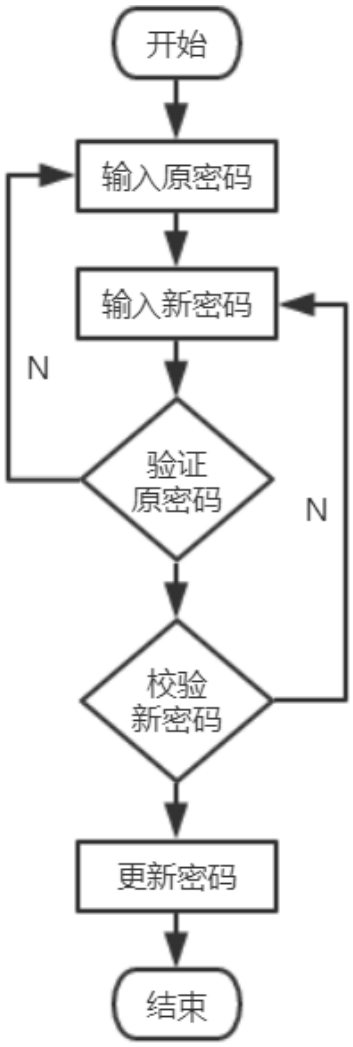

图4-14　账户管理模块流程图

结果展示模块。图 4-15 所示是结果展示模块的流程图。对于两种不同类型的待检测数据，为了方便结果的展示，服务器端会在检测模型检测完成后，将检测的结果直接绘制在输入的视频上，并通过网络返回到客户端相应界面完成展示。

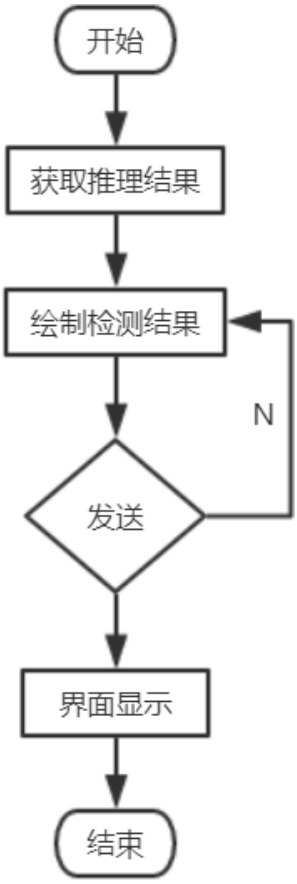

图4-15　结果展示模块流程图

4.4　系统数据结构设计

根据系统的需求分析与功能设计，本系统还需要保存用户在使用过程中需要存储的信息，方便用户进行使用。从系统的详细设计中我们可以得出，我们需要对用户保存的摄像头信息和用户账户信息进行存储。

表 4-1　摄像头信息表

字段	类型	描述
Camera_IP	STRING	摄像头 IP 地址
Camera_Username	STRING	摄像头连接用户名
Camera_Password	STRING	摄像头连接密码
Camera_Position	STRING	摄像头安装位置
User_ID	STRING	摄像头管理者 ID

针对摄像头管理模块的功能设计，我们需要存储摄像头的 IP 地址、用户名、密码、位置、管理用户 ID 信息，表 4-1 是摄像头信息数据结构设计。

表 4-2　用户信息表

字段	类型	描述
User_ID	STRING	用户的 ID，唯一标识符
Username	STRING	用户名
Password	STRING	密码
flag	bit	0 为普通用户，1 为管理员

对于账户管理模块的功能设计，我们需要存储用户 ID、用户名、密码、用户类型信息，表 4-2 是用户信息数据结构设计。

4.5　本章小结

本章在需求分析的基础上，对系统的总体架构进行了设计，并对系统将要实现的功能模块做出概要和详细的设计，同时还对系统中需要存储的信息数据结构进行设计。

5 行人检测系统的实现与测试

5.1 系统开发运行环境

本系统的服务端程序是在 CentOS 开发的，服务端程序和 TensorRT 压缩模块采用 C++语言实现，模型训练部分使用 Python 语言实现，客户端程序使用 Qt 框架使用 C++语言实现。深度神经网络模型的训练是在公司服务器上进行训练的。具体的开发环境如表 5-1 所示。

表 5-1 系统开发运行环境表

硬件参数	CPU：Intel Xeon E5-2697a V4 @ 2.6GHz x2 内存：128 GB GPU：NVIDIA TITAN X x 2 GPU 显存：12GB x 2
开发环境	操作系统：CentOS 7.5 CUDA 版本：8.0 Caffe 版本：Microsoft-version IDE：PyCharm2018.02、CLion 2018.02、Qt 5.6.1

5.2 系统功能实现

搭建好系统的开发运行环境后，根据对系统的需求分析和系统设计，我们需要对系统进行最终实现，本节将对系统开发过程中的模块进行描述，并对系统实现的主要功能模块的界面做出展示。

5.2.1 行人检测模型训练

本文设计实现的智能监控视频行人检测系统的核心就在于使用基于区域的全卷积网络(R-FCN)的目标检测网络训练得到应用于企业园区场景下的行人检测模型。系

统的关键就在于检测模型的训练，本系统最终展现给用户的关键也在于此，模型的精度对系统最终的呈现的效果影响最大。与此同时在系统实际交付运行后，随着环境的不断变化，需要对检测模型定期进行迭代改进和更新，以保证系统能随着环境的变化保持良好的检测精度。

本系统使用的核心算法是基于区域的全卷积网络(R-FCN)，已在第二章中对其进行了详细介绍，本小节我们将会对具体训练过程做出描述。

（1）数据集制作

我们想要系统在目标场景下获得好的效果，就必须要利用目标场景的数据对模型进行训练，这样模型会学习到在特定场景下的特征，更新模型中的权重。并且，深度学习的特点就是数据量越多效果就可以越好，本系统针对的是企业园区监控视频进行行人检测，通过对真实场景中录制的监控视频文件进行视频帧的获取后，经过挑选，得到 30000 张场景图片，使用数据增强方法后共 150000 张。

获取了真实场景图片之后，我们需要对图片进行标注，把图片中的待检测目标也即是行人的位置和标签信息记录下来。为了方便后续的工作，我们使用 LabelImg 工具将数据集制作成与 Pascal VOC2007 数据集[44]相同的格式。数据集包含三个文件夹 Annotations、ImageSets、JPEGImages。JPEGImages 文件夹中存放这所有的图片数据，文件都是 jpg 格式。Annotations 文件夹中存放的是每张图片的标注信息，图片的标注信息都保存在与图片同名的.xml 文件中，xml 文件中将会保存图片中每一个目标的类别、目标框的左上角坐标和右下角坐标。ImageSets 文件夹下的 Main 文件夹中保存了四个.txt 文件，分别是 train.txt、test.txt、trainval.txt、val.txt。test.txt 中保存的是测试所用的所有样本的文件名，train.txt 中保存的是训练所用的样本文件名，val.txt 中保存的是验证所用的样本文件名，trainval.txt 中保存的训练和验证样本，是 train.txt 和 val.txt 的总和。

（2）修改网络结构配置文件

基于区域的全卷积网络(R-FCN)的骨干网络有很多可以选择，例如 VGG16[45]、ZFNet[46]、ResNet 等。从精度和速度两个方面考量，本系统选用的是 ResNet50 的骨

干网络，采用端到端的训练方式，训练过程中使用在线难例挖掘(Online Hard Example Mining, OHEM)[47]等技巧，网络结构配置文件是以.prototxt 为后缀的文件。

本系统设计的目标是识别场景中的行人，而 R-FCN 原始模型是在 Pascal VOC2007 数据集上进行训练的，包含有 20 个物体类别和背景共 21 个类别，而我们的模型只包含人体和背景 2 个类别，就需要对网络结构配置文件做出相应修改。这里我们主要针对与类别相关的网络层的输出维度，主要是输入数据层的类别数目和网络末端输出的类别位置敏感得分图的维度和类别预测的维度。为了使得网络模型对于实际的应用场景具有更好的效果，我们还需要对特定网络层进行修改和调整。

其次，R-FCN 是基于区域建议的目标检测网络，原始网络会在区域建议子网络(RPN)中以特征图中的锚点(anchors)为中心生成 9 个不同尺度和长宽比例的建议框，因此，我们可以针对自己的数据集的特点，设置生成的建议框的尺度和长宽比例，这就会产生数目可能不再是 9 个的建议框，我们也需要对网络结构配置文件中相应层的输出维度进行修改，主要是区域建议网络(RPN)中与边框生成相关的层。在我们的网络中，针对实际场景的只检测行人的特点，对实际场景数据集中的行人框的尺度和长宽比进行聚类，得出共 12 种不同的中心点，并修改生成锚点框中的参数与网络层的输出长度。

（3）修改训练脚本

在进行训练之前，我们还需要对数据进行预处理，我们需要将图片解析为 imdb 类型的数据，将标注信息解析为 roidb 类型的数据，然后才可以将数据输入到网络模型中。

使用 Caffe 训练模型时，还需要设置一些环境参数，包括指定 GPU 设备、骨干网络类型、网络结构配置文件路径、预训练模型的权重文件路径、数据集路径、训练迭代次数、网络参数配置文件路径、日志和训练中间结果保存路径等。为了方便经常性的训练工作，我们将其写成 shell 脚本，在每一次训练之前修改相应的环境参数即可。

（4）修改训练参数配置文件

模型的参数影响着最终训练出来的模型精度，基于区域的全卷积网络也不例外。在 R-FCN 训练时，我们可以对这些模型参数进行设置。所有可设置的参数都写在 config.py 的 Python 脚本中，训练时将会读取这个脚本，但是为了每次训练时方便设置，我们将常用的一些参数写入 yml 配置文件中，每次训练可以直接修改 yml 文件即可，对于 yml 文件中缺少的参数，我们可以随时添加。yml 文件中常用的参数如图 5-1 所示。

```
EXP_DIR: rfcn_end2end_ohem
TRAIN:
  HAS_RPN: True
  IMS_PER_BATCH: 1
  BBOX_NORMALIZE_TARGETS_PRECOMPUTED: True
  RPN_POSITIVE_OVERLAP: 0.7
  RPN_NORMALIZE_TARGETS: True
  RPN_BATCHSIZE: 256
  PROPOSAL_METHOD: gt
  BG_THRESH_LO: 0.0
  BATCH_SIZE: -1
  AGNOSTIC: True
  SNAPSHOT_ITERS: 10000
  RPN_PRE_NMS_TOP_N: 6000
  RPN_POST_NMS_TOP_N: 300
TEST:
  HAS_RPN: True
  AGNOSTIC: True
```

图 5-1　yml 文件中常用的参数

（5）训练

在前面的工作都准备好之后，我们就可以对模型进行训练。只需要在终端下进入 py-R-FCN 工程目录下运行编写的训练脚本 rfcn_end2end_ohem.sh 即可。

（6）测试

模型的测试过程与训练过程类似。首先要准备测试阶段的数据，然后需要测试阶段的网络结构配置文件，测试阶段没有反向传播，因此不再需要 loss 层，之后需要修改测试脚本中的环境参数和测试阶段的网络参数，然后运行测试脚本即可。经过测试，我们训练得到的模型平均 IOU 为 79.8%，召回率为 93.2%。

5.2.2 行人检测模型压缩

通过上一小节的描述，我们得到了使用 Caffe 深度学习框架训练出来的网络模型权重文件。但在实际部署时，我们并不直接利用 Caffe 框架来进行模型的推理。一方面，由于 Caffe 的安装过程繁琐依赖包太多，不利于环境部署和维护，但主要的是，在实际应用的推理阶段我们需要使用 TensorRT 来对模型进行优化加速，使其可以减小模型大小以及达到更快的运行速度。

TensorRT 使用时分为两个阶段：第一个阶段是构建(build)阶段，构建阶段构建网络定义，执行优化，并生成推理引擎；第二个是执行(execution)阶段，执行阶段使用推理阶段生成的推理引擎运行推理任务即可。

本系统是使用基于区域的全卷积网络(R-FCN)的行人检测系统，在使用 TensorRT 构建推理引擎时，需要准备部署阶段的 Caffe 网络结构配置文件和保存的网络权重 caffemodel 文件。部署阶段的 Caffe 网络结构配置文件和训练阶段的 Caffe 网络结构配置文件相比，只需要修改输入层和删除 loss 层即可。此外，由于 R-FCN 中的位置敏感感兴趣区域池化层(position-sensitive ROI pooling)不在 TensorRT 支持的常用层之中，属于自定义层，需要继承 IPlugin 抽象类并进行实现。然后在 caffeToGIEModel 函数中将训练好的 Caffe 模型解析其网络结构，指定推理阶段的计算精度为 INT8，确定输出的张量(tensor)，构建推理引擎，并将推理引擎序列化(serialize)保存起来。

在执行阶段，TensorRT 首先需要将保存的推理引擎文件进行解析反序列化，然后执行 doInference 函数对每一帧图片进行计算获得检测结果。通过测试，经过压缩后的模型在执行阶段设置 batch 为 8 张时达到了 33FPS，满足实时性要求。

5.2.3 实时视频流检测模块的实现

实时视频流检测的所要功能就是对获取的摄像头实时视频流进行检测。用户进入模块后，首先需要选取需要检测的摄像头，然后点击开始检测，系统就会获取视频流数据，进行预处理之后即可开始检测，点击停止检测之后系统就会停止对当前选择的摄像头视频流进行检测。图 5-2 所示为实时视频流检测模块的交互界面。

实时视频流检测功能的实现。用户进入实时视频流检测模块时，程序会读取保存摄像头信息的 camera.xml 文件，并以下拉列表框的形式展示出来供用户选择。用

户选择摄像头之后点击开始检测按钮后，按钮的 clicked 信号将被触发，并将在槽函数中获取下拉列表框当前选择的摄像头信息，将其发送到服务器端，服务器端收到请求后将会调用 stream_detect 函数，这个函数将会获取收到的摄像头信息，并执行检测脚本，脚本将会运行检测代码，首先使用 OpenCV 获取指定的摄像头视频流，然后将视频帧输入到经过 TensorRT 压缩的 R-FCN 模型中，经过计算获得检测结果，然后将其绘制在视频帧中发送到客户端程序显示。点击停止检测按钮，按钮的 clicked 信号将被触发，程序将会向服务器端发送停止检测的指令，服务器端接收到后，将会把对应的摄像头检测模型停止运行。

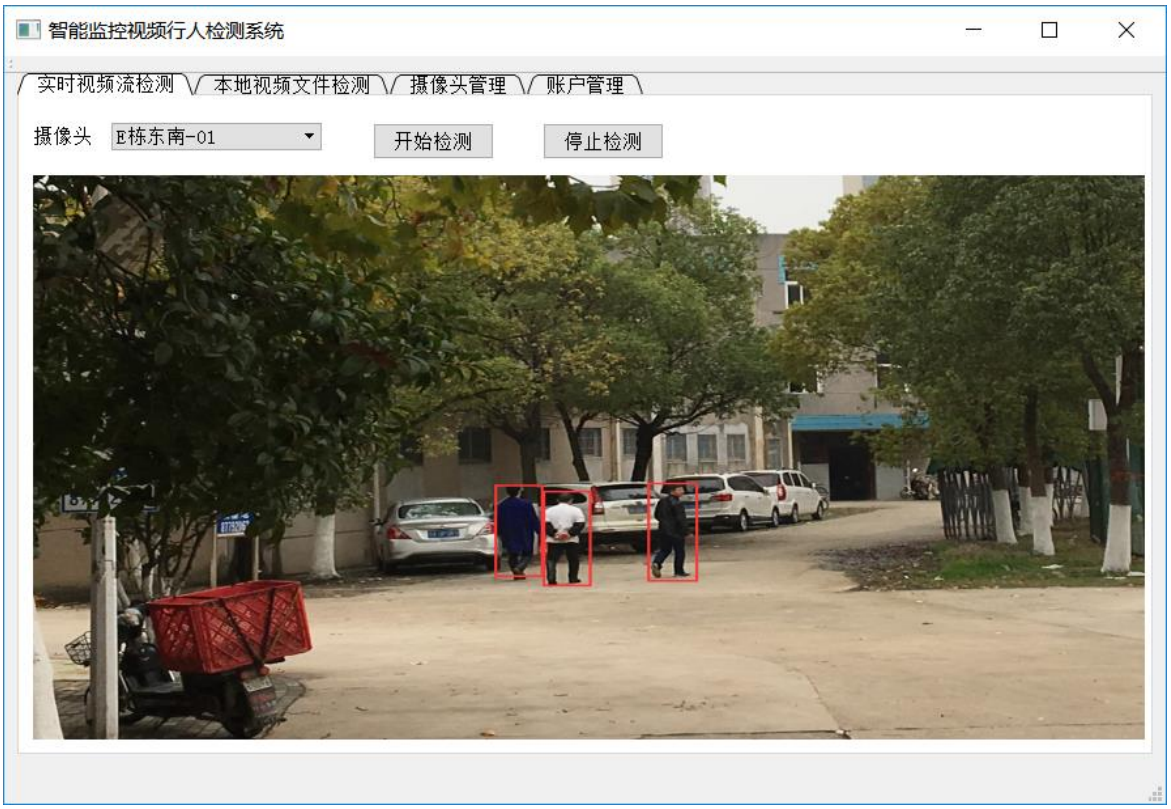

图 5-2　实时视频流检测界面

5.2.4　本地视频文件检测模块的实现

本地视频文件检测除了数据来源不同，其他均与实时视频流检测模块的流程类似。用户点击选择文件按钮，程序会弹出文件对话框，用户选择待检测文件后，点击开始检测按钮，客户端程序就会获取本地视频文件路径，并将视频数据发送到服务器端，服务器端对其进行预处理后输入到模型中进行检测。点击停止检测即可停

止对当前文件的检测。图 5-3 所示是本地视频文件检测模块的交互界面。

进入本地视频文件检测模块后，用户点击选择文件按钮触发 clicked 信号，槽函数将会创建一个文件对话框，用户可以在文件对话框中选择待检测视频文件之后点击确定，然后点击开始检测按钮，触发按钮的 clicked 信号，然后槽函数将会选择文件路径和检测指令发送到服务器端，服务器端在收到请求后将会调用 file_detect 函数，这个函数将会获取文件路径，之后的流程与实时视频流检测模块类似，首先会使用 OpenCV 获取视频数据，然后将视频帧输入到模型中进行计算得到行人的位置信息，最后将检测结果绘制在视频帧中发送到客户端中显示。同样，当用户不再需要对当前文件进行检测时，可以点击停止检测按钮，触发 clicked 信号，在槽函数中将会发送指令到服务器端，服务器在接收到指令后，会停止对当前视频的获取、处理与检测操作。

图 5-3　本地视频文件检测界面

5.2.5　摄像头管理模块的实现

摄像头管理模块是为了用户使用系统时更加方便高效。用户进入摄像头管理模块，可以看到记录过得摄像头信息，并可以通过点击添加按钮添加新的摄像头信息

条目，也可以通过点选已经存在的条目然后点击修改按钮进行修改或点击删除按钮删除此条记录。图 5-4 所示是摄像头管理模块的交互界面。

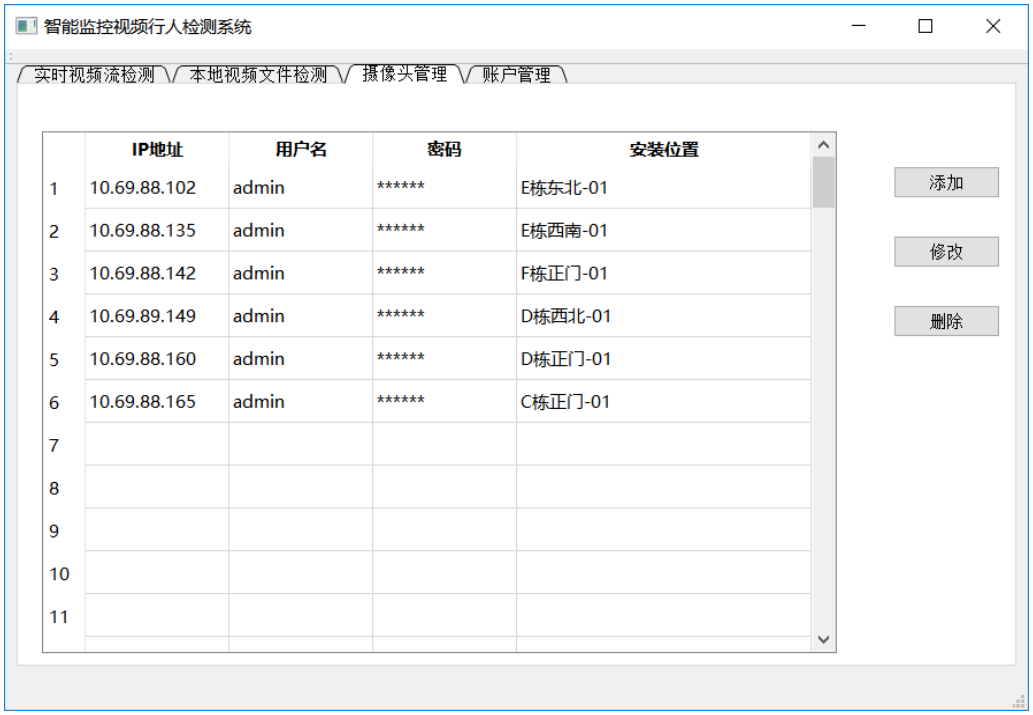

图 5-4　摄像头管理模块界面

用户进入摄像头管理模块时，程序会先读取保存摄像头信息的 camera.xml 文件，解析之后将其显示在数据表格部件中。用户点击添加按钮，触发其 clicked 信号，槽函数中将会创建一个输入对话框，用户在输入对话框中输入新的摄像头信息，然后点击确认程序将会把新的摄像头信息写入 camera.xml 文件中并更新数据表格部件的内容。用户也可以对已经存在的条目进行修改，首先点击选中数据表格部件中的需要修改的条目，然后点击修改按钮，触发按钮的 clicked 信号，在槽函数中，我们将获取这个条目的信息，并创建一个输入对话框，将读取的信息显示在输入对话框中，用户修改相应属性，然后点击确认按钮，程序将会对 camera.xml 文件中相应的条目进行更新并更新数据表格部件的内容。用户点击选中需要删除的摄像头条目，然后点击删除按钮，触发 clicked 信号，槽函数中将会创建一个消息对话框提醒用户是否确认删除，用户点击确认之后，槽函数获取消息对话框返回的结果并删除 camera.xml 文件中的相应条目并更新数据表格部件的内容。

5.2.6 账户管理模块的实现

账户管理模块是为了方便用户对自己的账户密码进行维护而设计实现的。用户在本模块中可以输入原密码和新的密码进行匹配，通过之后将会对密码进行替换。图 5-5 所示的是账户管理模块的交互界面。

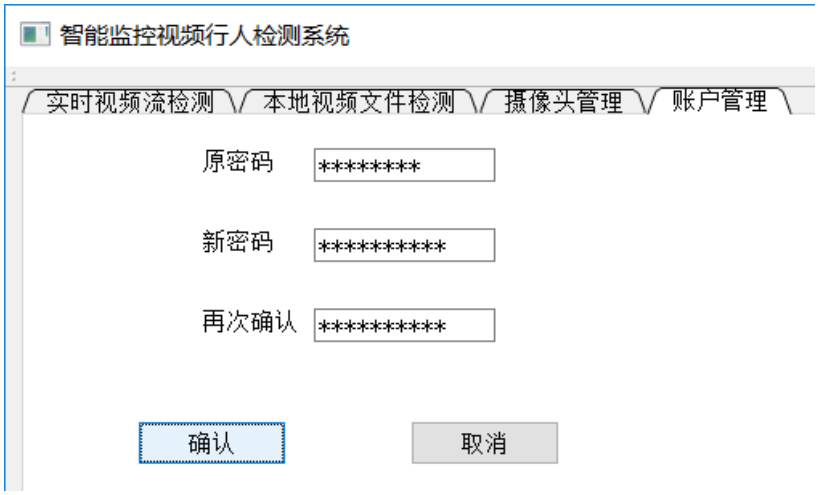

图 5-5 账户管理模块界面

用户进入账户管理模块之后，可以看到三个密码输入框，“原密码”、“新密码”、“再次确认”，依次输入后，点击确认按钮后，触发 clicked 信号，槽函数将会将原密码和保存用户账户信息的 user.xml 文件中对应用户的密码进行比对，验证通过后，将会对两次输入的新密码进行比对，通过后会将新的密码更新到 user.xml 文件中，若两项依次验证时有一项没有通过，将会返回相应的提示信息。

5.3 系统测试

测试工作是一个系统项目整个开发过程中不可或缺的部分。在对系统进行尽可能完备的测试工作，可以保证系统开发质量，对发现的问题可以及时的改正，而不必在交付之后出现太多问题。

5.3.1 测试目的和范围

对系统的测试工作是为了发现系统的错误而进行的，经过测试发现系统中的错

误后，通过分析错误产生的原因，可以帮助我们发现当前的软件过程的缺陷，以便改进。

同时，通过分析也能帮助我们设计出有针对性的检测方法，改善测试的有效性。即使在测试过程中没有发现错误，测试工作也是有价值的。本文设计实现的是智能监控视频行人检测系统，对系统进行测试，需要根据系统的需求来制定对应的测试计划，并且测试工作需要伴随着整个开发过程，制定的测试计划也需要针对不同的测试阶段而进行设计，从而保证最终交付的系统的质量。

本文设计实现的智能监控视频行人检测系统进行检测时，我们需要针对需求定义中的各个功能模块设计良好的测试用例并得出测试结果，测试计划还需要对测试阶段的需要资源、测试完成的进度、测试需要达到的目标做出良好的规划。

5.3.2 测试环境

由于英伟达 GPU 的产品策略，我们使用的训练服务器中的 GPU 并不支持低精度(INT8)计算，因此最终的 TensorRT 模型将会在英伟达 Tesla P4 GPU 上运行，测试阶段的环境如表 5-2 所示。

表 5-2　测试环境表

硬件参数	CPU：Intel Xeon E5-2600 V4 @ 2.6GHz x2 内存：64 GB GPU：NVIDIA Tesla P4 GPU 显存：8GB
部署环境	操作系统：CentOS 7.5 CUDA 版本：8.0 TensorRT 版本：3.0.4

5.3.3 系统功能测试及结果

为了验证本文设计实现的智能监控视频行人检测系统的各个功能模块，以及系统的非功能性需求的度量，我们针对性的设计了测试用例，如表 5-3 所示。

表 5-3　系统功能测试用例

编号	功能测试内容	测试步骤	预期结果	测试结果
1	实时视频流检测功能	1. 登录系统 2. 进入程序主界面 3. 进入实时视频流检测模块 4. 选择摄像头 5. 点击开始检测 6. 点击停止检测	客户端程序正确进行账户验证，读取保存的摄像头信息并显示，用户从下拉列表框中选择摄像头信息后，点击开始检测按钮服务器端将正确接收指令及摄像头信息并开始检测，同时将检测结果显示在模块界面上。点击停止检测按钮，服务器端将停止对视频流进行检测。	通过验证
2	本地视频文件检测功能	1. 登录系统 2. 进入程序主界面 3. 进入本地视频文件检测模块 4. 点击选择文件按钮选择文件 5. 点击开始检测 6. 点击停止检测	客户端程序正确验证账户，点击选择文件，弹出文件对话框，用户选择文件后，点击开始检测，服务器端可以正确获取指令和视频文件数据，并开始检测。	通过验证
3	摄像头管理功能	1. 登录系统 2. 进入程序主界面 3. 进入摄像头管理模块 4. 点击添加按钮 5. 选中一条记录点击修改按钮 6. 选中一条记录点击删除按钮	用户进入摄像头管理模块，读取保存摄像头信息的 xml 文件并显示，点击添加按钮，弹出输入对话框，用户输入完摄像头信息后点击确认，将条目写入 xml 文件并更新数据表格部件。选中一条记录点击修改按钮，弹出输入对话框，用户修改信息后点击确认，将修改内容更新到 xml 文件中并更新数据表格部件。选中一条记录点击删除按钮，弹出提醒消息框，用户点击确认后，删除 xml 文件中的记录更新数据表格部件内容。	通过验证

续表 5-3　系统功能测试用例及结果

编号	功能测试内容	测试步骤	预期结果	测试结果
4	账户管理功能	1. 登录系统 2. 进入程序主界面 3. 进入账户管理模块 4. 输入原密码 5. 输入新密码 6. 再次输入新密码	系统正确验证账户，进入账户管理模块后，用户分别输入原密码、新密码、再次确认新密码后点击确认按钮，服务器端将接收原密码并进行验证，通过后客户端程序对两次输入的新密码进行输入是否一致验证，通过后服务器端接收新密码并替换原密码。	通过验证
5	可扩展性	1.将检测核心算法模块部署在其他类型的设备，如嵌入式设备中	检测核心算法模块能运行在其他类型设备中运行	未通过
6	实时性	1. 登录系统 2. 进入程序主界面 3. 进入实时视频流检测模块 4. 选择摄像头 5. 点击开始检测	服务器端检测时能达到每秒 25 帧及以上，在客户端程序界面上播放的检测结果画面不会出现卡顿	通过验证
7	界面友好	1. 登录系统 2. 进入程序主界面 3. 进入实时视频流检测模块 4. 进入本地视频文件检测模块 5. 进入摄像头管理模块 6. 进入账户管理模块	用户进入每个功能模块之后能方便快捷的进行相应操作完成相关功能。	通过验证

5.4 本章小结

本章在前文需求分析和系统设计的基础上，对系统的实现做出了描述，针对性的设计了测试用例进行系统测试。在系统实现部分中，详细描述了系统的每个功能模块实现方式和实现过程，展示了各个模块的使用界面。在系统测试部分，我们介绍了测试环境，设计了测试用例并记录了测试结果。

6 总结与展望

在本章，我们将对论文所做的工作进行总结，阐述系统所设计并实现的功能，并将对系统将来可以做出改进的地方提出了展望。

6.1 全文总结

深度学习近几年的发展，促进了人工智能技术的进步，带来的是计算机视觉等领域水平的提升，但是算法研究不应该仅仅是存在于理论之中，更重要的是在实际的应用中发挥其强大的作用。本文从行人检测技术的发展开始，对深度学习框架特别是 Caffe 进行了介绍，也对本文采用的核心的行人检测算法基于区域的全卷积网络(R-FCN)进行了讲解，并介绍了在模型量化压缩领域非常流行的 TensorRT 推理引擎。

本文设计实现的智能监控视频行人检测系统的整个过程经过系统需求分析，系统的总体及功能模块设计，最后根据设计进行检测模型训练、模型量化压缩、功能模块实现，并根据设计的测试计划对系统进行了测试工作。

本文设计实现的智能监控视频行人检测系统主要实现了以下成果：

（1）实现了获取网络摄像头实时视频流并进行行人检测的功能，并将检测结果进行了展示。

（2）实现了对本地监控视频文件的行人检测功能，并将检测结果进行展示。

（3）实现了对摄像头信息的管理，方便了系统的日常使用。

（4）实现了 R-FCN 网络结构，针对应用场景对模型做出改进，使用 Caffe 训练得到符合实际应用时的精度要求的网络模型。

（5）实现了对训练得到 R-FCN 模型使用 TensorRT 进行量化压缩，加速了推理速度，使得模型在实际使用阶段能够达到实时性。

本文设计的智能监控视频行人检测系统按照软件工程方法进行开发，整体上完成了系统要满足的需求和设计中要实现的功能，并设计测试用例测试了本系统，系统达到了实际应用的需要。

6.2 展望

本文设计实现的智能监控视频行人检测系统使用基于区域的全卷积网络(R-FCN)模型达到了很好的精度和实时性，但是目前系统仍然是对一路视频进行的检测，而对于一个监控系统来说，往往不止一路视频，我们需要考虑在对多路监控视频进行检测时如何进行设计和优化，使得所需要的计算资源不至于成倍增长。另外，虽然R-FCN 已经取得了很好的精度，但是我们也仍然需要定期训练更新模型以使模型的精度能随监控视频的环境变化而保持其良好的表现，因此，我们也需要持续的考虑对模型的改进和优化。

本文设计实现的系统还存在功能模块划分不够清晰等问题，需要对模块的设计实现等做出改进。系统功能仍然可以增强，添加更多的特性，使得整个系统更具有泛用性。此外，随着 docker 等应用容器引擎的流行，我们也应该考虑在开发和部署的过程之中应用此项技术，特别是在部署阶段，类似的技术可以大大简化部署工作流程，提高部署效率，也可以提高系统的可维护性，这在后续是一个方向。

致 谢

转眼间，两年多的研究生生活将要谢幕，届时，我也将走出校园，迈入社会中。在软件学院学习和生活的这两年多的时间里，无论是学习还是生活都过得很充实，没有辜负自己当年的选择。

首先，需要感谢我的导师管乐老师。从我研究生入学开始，管老师指导了我研究和学习的方向，管老师深厚的学识和极强的开发能力令我敬佩，这将是我终身学习的目标。同时管老师平等对待学生的方式，与学生交流时的耐心，都使我受益良多。在这篇论文选题和写作之前，管老师给我提出了非常多的指导和建议，这才促使了这篇论文的完成。

其次，要感谢 18 届的三位学长，毫无保留的将自己的学习和生活经验与我们分享，使我能少走一些弯路。另外还要感谢实验室的小伙伴，与你们共处的日子既充实又快乐，这将是我人生中宝贵的记忆。

最后，要感谢我的家人，你们的爱与支持，是我人生前进的动力。

参考文献

[1] Richard Szeliski. 计算机视觉—算法与应用. 北京: 清华大出版社, 2012: 1-5

[2] 余凯, 贾磊, 陈雨强, 徐伟. 深度学习的昨天, 今天和明天. 计算机研究与展, 2013, 50(9): 1799-1804

[3] Xu Yan-wu, Cao Xian-bin, and QiaoHong. Survey on the latest development of pedestrian detection system and its key technologies expectation. Acta Electronica Sinica, 2008, 36(5): 368-376

[4] 张旭东. 行人检测技术研究: [硕士学位论文]. 成都: 电子科技大学图书馆, 2011

[5] 陆堃. 基于图像的行人检测方法研究: [硕士学位论文]. 上海: 上海交通大学图书馆, 2009

[6] 董向华, 杨勇. 基于网络的视频监控系统的设计与实现. 通信技术, 2013(2): 64-66

[7] 宋磊, 黄祥林, 沈兰荪. 视频监控系统概述. 测控技术, 2003, 22(5): 33-35

[8] 谌飞超. 远程视频监控系统的设计与实现. 工业控制计算机, 2016, 29(1): 107-108

[9] 许言午, 曹先彬, 乔红. 行人检测系统研究新进展及关键 技术展望. 电子学报, 2008, 36(5): 368-376

[10] 苏松志. 行人检测若干关键技术研究: [博士学位论文]. 厦门: 厦门大学图书馆, 2011

[11] P. Viola, M. J. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. IJCV, 2005, 63: 153–161

[12] P. Dollar, C. Wojek, B. Schiele, et al. Pedestrian detection: an evaluation of the state of the art. IEEE Transactions on PatternAnalysis andMachine Intelligence, 2012, 34(4): 743-761

[13] D. Geronimo, A. M. Lopez and A. D. Sappa, et al. Survey of pedestrian detection for advanced driverassistance systems. IEEE Transactionson Pattern Analysis and

Machine Intelligence, 2010, 32(7): 1239-1258

[14] 贾慧星, 章毓晋. 车辆辅助驾驶系统中基于计算机视觉的行人检测研究综述. 自动化学报, 2007, 33(1): 84-90

[15] 龚声蓉, 刘纯平, 王强. 数字图像处理与分析. 北京: 清华大学出版社, 2006: 1-50

[16] Ojala T, Pietikainen M, Harwood D. A comparative study of texture measures with classification based on feature distributions. Pattern Recognition, 1996, 29(1): 51-59

[17] Ojala T, Pietikainen M, Maenpaa T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans PAMI, 2002, 24(7): 971–987

[18] David G. Lowe. Object recognition from local scale-invariant features. International Conference on Computer Vision, Corfu, Greece, 1999(9): 1150-1157

[19] David G. Lowe. Distinctive Image Features from Scale- Invariant Keypoints. International Journal of Computer Vision, 2004, 60(2): 91-110

[20] William T. Freeman, Michal Roth. Orientation Histograms for Hand Gesture Recognition, Tech. Rep. TR94-03, Mitsubishi Electric Research Laboratories, Cambridge, MA, 1994

[21] Cortes, Corinna; Vapnik, Vladimir N.. Support-vector networks. Machine Learning, 1995, 20(3): 273–297

[22] Dalal N, Triggs B. Histograms of oriented gradients for human detection. Computer Vision and Pattern Recognition, 2005: 886-893

[23] Papageorgiou, Oren and Poggio. A general framework for object detection. International Conference on Computer Vision, 1998

[24] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hintion. ImageNet Classification with Deep Convolutional Neural Networks. Conference and Workshop on Neural Information Processing Systems, 2012

[25] Olga Russakovsky, Jia Deng, Hao Su, et al. ImageNet Large Scale Visual Recognition Challenge. arXiv preprint arXiv: 1409. 0575, 2014

[26] R. Girshick, J. Donahue, T. Darrell, J. Malik. Rich Feature Hierarchies for

Accurate Object Detection and Semantic Segmentation. IEEE Conference on Computer Vision and Pattern Recognition, 2014

[27] Jasper R. R. Uijlings, Koen E. A. van de Sande, Theo Gevers, et al. Selective Search for Object Recognition. International Journal of Computer Vision, 2013, 104(2): 154-171

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. European Conference on Computer Vision(ECCV), 2014

[29] Ross Girshick. Fast R-CNN. IEEE International Conference on Computer Vision(ICCV), 2015

[30] Shaoqing Ren, Kaiming He, Ross Girshick, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Neural Information Processing Systems(NIPS), 2015

[31] MarMartín Abadi, Paul Barham, Jianmin Chen, et al. TensorFlow: A System for large-scale machine learning. arXiv preprint arXiv: 1605. 08695, 2016

[32] S. van der Walt, S. C. Colbert and G. Varoquaux. The NumPy Array: A Structure for Efficient Numerical Computation. Computing in Science Engineering, 2011, 13(2): 22-30

[33] Yangqing Jia, Evan Shelhamer, Jeff Donahue, et al. Caffe: Convolutional Architecture for Fast Feature Embedding. arXiv preprint arXiv: 1408. 5093, 2014

[34] E. Shelhamer, J. Long and T. Darrell. Fully Convolutional Networks for Semantic Segmentation, 2014

[35] IEEE Conference on Computer Vision and Pattern Recognition(CVPR), 2015

[36] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al. Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition(CVPR), 2016

[37] Jifeng Dai, Yi Li, Kaiming He, et al. R-FCN: Object Detection via Region-based Fully Convolutional Networks. Conference on Neural Information Processing Systems(NIPS), 2016

[38] Liu W, Anguelov D, Erhan D, et al. SSD: Single shot multibox detector Proc of European Conference on Computer Vision: Springer International Publishing,

2016: 21-37

[39] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. arXiv preprint arXiv: 1506. 02640, 2015

[40] de Boer, P-T. Kroese, D. Mannor, et al. A Tutorial on the Cross-Entropy Method. Annals of operations research, 2005, 134(1): 19-67

[41] David A. Freedman. Statistical Models: Theory and Practice. Cambridge University Press, 2009: 26-28

[42] ang G, Hoiem D, Forsyth D. Learning image similarity from flick groups using fast kernel machines. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(11): 2177-2188

[43] Jussi Hanhirova, Teemu Kämäräinen, Sipi Seppälä, et al. Latency and throughput characterization of convolutional neural networks for mobile computer vision. the 9th ACM Multimedia Systems Conference, 2018

[44] M. Fatica. CUDA toolkit and libraries. 2008 IEEE Hot Chips 20 Symposium(HCS), 2008: 1-22

[45] Everingham M., Van Gool, L. Williams C. K. I., et al. The PASCAL Visual Object Classes(VOC) Challenge. International Journal of Computer Vision, 2010, 88(2): 303-338

[46] Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. International Conference on Learning Representations(ICLR), 2015

[47] Matthew D. Zeiler, Rob Fergus. Visualizing and Understanding Convolutional Networks. European Conference on Computer Vision(ECCV), 2014

[48] Abhinav Shrivastava, Abhinav Gupta, Ross Girshick. Training Region-based Object Detectors with Online Hard Example Mining. IEEE Conference on Computer Vision and Pattern Recognition(CVPR), 2016