学校代码：10284
分 类 号：TP181
密　　级：公开
U D C：004.02
学　　号：MF20330024

南京大学

硕 士 学 位 论 文

论 文 题 目　基于上下文信息的视频目标检测后处理研究

作 者 姓 名　管俣祺

专业学位类别(领域)　计算机技术

研 究 方 向　人工智能与模式识别

导 师 姓 名　申富饶 教授

2023 年 5 月 22 日

答辩委员会主席 戴新宇 教授

评　　阅　　人 张剡 高工

徐明华 教授

论文答辩日期　2023 年 5 月 22 日

研究生签名:

导师签名:

Research on Post-processing of Video Object Detection Based on Context Information

by

Yuqi Guan

Supervised by

Professor Furao Shen

A dissertation submitted to
the graduate school of Nanjing University
in partial fulfilment of the requirements for the degree of
MASTER
in
Computer Science and Technology

Department of Computer Science and Technology
Nanjing University

May 22, 2023

南京大学研究生毕业论文中文摘要首页用纸

毕业论文题目：基于上下文信息的视频目标检测后处理研究

计算机技术 专业 2020 级硕士生姓名：管佩祺

指导教师（姓名、职称）：申富饶 教授

摘　要

目标检测任务根据输入数据类型进行区分，可以分为图像目标检测和视频目标检测。2014 年以前目标检测一词代指的即为静态图像目标检测，这类检测方案主要使用传统方式来处理图像特征。随着 2014 年卷积神经网络被引入图像目标检测领域，目前已经有相当数量的图像检测模型能够在模型复杂度、检测精度、检测速度上均取得不俗的表现。在图像目标检测的基础上，后续又提出了视频目标检测任务。视频数据中可能存在画面模糊、物体遮挡、镜头高速移动、物体罕见姿势等情况，故而在视频数据上直接应用图像目标检测器结果并不理想。现有的视频目标检测方案，往往需要引入额外的任务模块，从而更好的利用视频特征信息和视频数据的时间上下文信息。这类方案虽然在精度上取得了很好的成绩，但引入新模块的做法实质上使得整个视频目标检测任务变得更复杂，并且实时性也有所限制。

鉴于上述分析，本文着眼于后处理方案，力求在不引入新任务模块的前提下，通过后处理框架将视频上下文信息补充到图像目标检测模型的检测结果中。用尽可能短的耗时和尽可能简洁的模型架构，在视频数据上发挥现有的优秀图像目标检测模型的能力。本文主要研究内容如下：首先提出了一种基于视频上下文信息的非实时后处理框架。该框架通过建立跨帧检测框连接，将视频数据特有的时间一致性融合进前序图像目标检测模型的检测结果中。在序列内进行类分数更新来优化误检，并在序列间进行双向扩散来优化漏检；随后又提出了一种实时视频目标检测后处理及结果平滑框架。处理视频流时实时进行后处理优化，利用过去的一组检测结果优化当前帧结果。并使用卡尔曼滤波缓解检测框抖动的问题。

本文通过大量实验数据证明了所提方法的有效性，除此之外，本文将提出的实时后处理框架应用于煤矿作业安全预警系统，通过实际工程检验理论方法

的应用价值。该系统首先对人员和分割物体进行实时检测，并利用后处理算法来优化由于煤矿的作业环境恶劣带来的诸多误检和漏检问题。最后根据检测结果判断当前工作人员状态是否危险。

关键词： 视频目标检测；后处理；跨帧检测框连接

南京大学研究生毕业论文英文摘要首页用纸

THESIS: Research on Post-processing of Video Object Detection Based on Context Information

SPECIALIZATION: Computer Science and Technology

POSTGRADUATE: Yuqi Guan

MENTOR: Professor Furao Shen

Abstract

Object detection tasks are divided according to the type of input data, which can be divided into image object detection and video object detection. Before 2014, the term object detection referred to static image object detection, which mainly used traditional methods to process image features. With the introduction of convolutional neural networks into the field of image object detection in 2014, there are already a considerable number of image detection models that can achieve good performance in model complexity, detection accuracy, and detection speed. Based on image object detection, video object detection tasks were subsequently proposed. Video data may have situations such as blurred images, occlusion of objects, high-speed movement of lenses, and rare postures of objects. Therefore, the results of directly applying image object detectors on video data are not ideal. Existing video object detection schemes often need to introduce additional task modules to better utilize video feature information and temporal context information of video data. Although these schemes have achieved good results in accuracy, introducing new modules essentially makes the entire video object detection task more complex and also limits its real-time performance.

In view of the above analysis, this paper focuses on the post-processing scheme, and strives to supplement the video context information into the detection results of the image object detection model through the post-processing framework without introducing new task modules. Use as short as possible time consumption and as simple as possible model architecture to exert the capabilities of existing excellent image object detection models on video data. The main research content of this paper is as follows: First, a non-real-time post-processing framework based on video context information is

proposed. This framework establishes cross-frame detection box connections and integrates the temporal consistency unique to video data into the detection results of the previous image object detection model. Perform class score updates within the sequence to optimize false positives, and perform bidirectional diffusion between sequences to optimize false negatives; then a real-time video object detection post-processing and result smoothing framework is proposed. Perform post-processing optimization in real time when processing video streams, and use a group of past detection results to optimize the current frame results. And use Kalman filter to alleviate the problem of jittering of detection boxes.

This paper proves the effectiveness of the proposed method through a large amount of experimental data. In addition, this paper applies the proposed real-time post-processing framework to the coal mine operation safety early warning system and verifies the application value of theoretical methods through practical engineering. The system first performs real-time detection of personnel and segmented objects, and uses post-processing algorithms to optimize many false positives and false negatives caused by the harsh operating environment of coal mines. Finally, judge whether the current working staff status is dangerous based on the detection results.

keywords: Video Object Detection, Post Processing, Cross-Fame Detection Box Connection

目　　录

插图清单

附表清单

第一章　绪论

1.1　研究背景

近年来，基于信息时代的海量数据以及日益提高的数据处理能力，机器学习发展尤为迅猛，特别是深度学习领域。越来越多的基于深度学习的应用出现在日常生活中，这些应用降低了诸多场景中的人工成本，较为典型的一些应用是计算机视觉 [1–3]、自然语言处理 [4–8]、语音信号处理 [9–12]、推荐系统 [13–16]、时间序列分析 [17–20]、对抗攻击安全研究等 [21–23]。其中，计算机视觉是较早被应用于日常生活的一类技术。具体来说，其任务是训练计算机和系统，使之能够从视觉输入信息中提取出用户所需的有意义的目标信息，并在此基础上进行后续分析，最终反馈给用户。计算机视觉任务建立在大量图像和视频数据的基础上，结合以卷积神经网络（Convolutional Neural Networks，CNN）为例的网络结构 [24]，进行像素级内容处理，使得机器和系统模拟人类视觉系统的工作机制来处理图像画面。图像数据有多种存在形式，比如静态图像数据、动态视频序列、多个视角的摄像机画面组合，以及通过扫描得到的多维数据等。而计算机视觉技术关注的正是基于各种各样的图像数据，将学科理论和模型应用于计算机视觉系统的构建。

上述内容简要介绍了计算机视觉任务目标及基础原理。事实上，计算机视觉领域包含多个子任务，常见的有：图像内容检测 [25]、目标检测 [26]、图像分类 [27]、内容跟踪 [28]、3D 场景重建 [29, 30] 等。其中，目标检测是计算机视觉领域最重要也最具有挑战性的分支之一，其主要任务是检测图像和视频中一类或者多类语义对象（例如汽车、动物、行人等）。在日常生活中，目标检测任务的应用也随处可见，如自动驾驶、视频监控、场景分析、机器人视觉等。近两年由于目标检测日益广泛的应用和快速的技术突破，该领域也受到了学术界和工业界越来越多的关注。目标检测的实现方式通常分为非神经网络方法及基于神经网络的方法。对于非神经网络方法，通常需要预先定义特征，再利用以支持向量机（Support Vector Machine，SVM）为例的技术进行分类 [31]。而基于神经网络的方法则能够做到在不定义具体特征的情况下进行端

到端的目标检测，这类方法通常借助卷积神经网络来实现，R-CNN（Region-based Convolution Neural Networks，or Regions with CNN features，R-CNN）[32]是首个成功将深度学习应用于目标检测领域的算法。在该算法的基础上，不断有新的目标检测模型被提出，一步步拓宽了基于神经网络实现目标检测的研究思路。本文将要描述的后处理框架，也正是在此基础上提出的一类优化机制。

在前文提到的基于神经网络实现目标检测的方法中，后处理过程即为利用非极大值抑制（Non-Maximum Suppression，NMS）这类方法对模型检测结果进行修正的过程 [33]。与图像数据相比，视频数据存在丰富的上下文信息，若能够更好的利用这些上下文信息，就可以完成图像目标检测模型从图像数据到视频数据的移植。在此基础上出现了对后处理模块的优化思路，形成了一种新的视频目标检测解决方案。该类后处理方案在非极大值抑制的基础上，添加了更多的策略设计，赋予后处理模块为图像目标检测模型补充视频上下文信息的功能。综上所述，后处理方案能够在轻微增加计算代价的同时明显提升视频目标检测精度，是一种高性价比且行之有效的解决方案，具有重要的应用价值。

1.2 研究现状

本文主要围绕视频目标检测后处理方案展开。视频目标检测有两种主流的解决方案，一种是将应用于静态图像的目标检测模型逐帧应用于视频流，再结合后处理方案对视频数据的上下文信息加以利用；另一种则是基于视频级数据处理和特征工作训练专用于视频数据的目标检测模型。本节内容主要介绍目标检测（包括图像目标检测及视频目标检测）及视频目标检测后处理的研究现状。

1.2.1 目标检测

目标检测作为一项重要的计算机视觉任务，其主要目的是检测图像中特定类别的视觉对象，这一技术的应用十分广泛，包括辅助驾驶、智能避障、视频监控、图像检索等场景。在过去的几年中，深度学习技术的迅猛发展也显著加快了目标检测的发展进程。借助深度学习网络和 GPU 计算，目标检测性能得到了巨大的提升，从而奠定了诸多下游任务的基础，例如图像字幕、对象跟踪等。目标检测任务首次被提出时，机器学习尚且没有迎来第二次浪潮，当时目标检测研究对象主要是图像数据，后期产生了针对于视频数据的目标检测。因

此，早期面向图像数据的广义目标检测器后续被划分到静态图像目标检测器或通用检测器的范围内，而面向视频数据的检测器则称为视频目标检测器。

纵观静态图像目标检测这一领域的发展进程，以引入深度学习技术作为分水岭，可以分为前后两个阶段。2014 年以前主要利用传统技术手段实现目标检测，2001 年提出的 Viola-Jones 检测器开创了目标检测的先河 [34, 35]，2006 年提出的 HOG 检测器在计算机视觉和图像处理任务中首次使用了特征描述符 [36]，而 2008 年提出的 DPM 首次引入了边界框回归 [37]。2014 年，R-CNN 的提出正式宣告目标检测进入了深度学习检测的时代，典型处理范式是基于卷积神经网络构建目标检测模型。此后提出的检测器被分为两阶段检测器和一阶段检测器两个类型。通常来说，基于深度学习的目标检测器从输入图像或视频帧中提取特征，再解决后续两个任务：查找存在于画面中的任意数量的目标；对每个目标进行分类并使用边界框预测每个目标的大小和位置信息。两阶段检测器分两步处理这两个任务，清晰的任务划分为准确性提供了保障。相应的，一阶段检测器选择将这两项任务合并为一个步骤来执行，以牺牲一定的准确性为代价来换取更短的计算耗时。如前文所述，卷积神经网络的引入极大的促进了目标检测领域的发展，大量基于静态图像的检测模型，如 R-CNN 及其变体 [32, 38, 39]、YOLO 及其变体 [40–43]，为静态图像目标检测建立了良好且有效的检测框架。后续又出现了基于锚框（Anchor Box）的方法、无锚框的方法等更细粒度的检测方法，整个目标检测领域的技术积累也越发成熟和完善。

得益于卷积神经网络的引入，静态图像目标检测近几年在精度、速度等方面均获得了显著提升，但将静态图像目标检测模型直接用于视频目标检测任务的效果却不尽如人意。这一现象主要是静态图像目标检测模型无法利用视频流中时间信息及上下文信息导致的。在视频数据中，前一帧中比较容易被检测出且置信度高的目标，在下一帧中可能难以被检测到或者置信度明显降低，视频数据存在的画面模糊、物体遮挡、镜头高速移动、物体罕见姿势等现象都有可能导致上述问题，这也就使得直接将静态图像目标检测模型应用于视频数据变得更为困难。为了解决视频目标检测存在的难题，研究者们主要探索出了两种策略：基于视频级数据处理和特征工作来训练视频目标检测模型的策略；静态图像目标检测器与后处理算法结合的策略。前者需要对视频数据进行分析，在此基础上设计特征处理机制，从而进行当前帧和附近帧亦或是全局帧的特征聚合，实现对视频数据中局部上下文信息及全局上下文信息的利用。这类方法需要大量的计算，方可达到与静态目标检测相同或者更高的检测精度，相应的其

计算速度普遍较慢，无法灵活运行于低性能设备或者需要实时检测的使用场景。另一类视频目标检测策略是将静态图像目标检测器与后处理算法结合，将视频流中的每一帧看作独立图像实施图像目标检测，并在后处理阶段补足视频数据中的上下文信息。后处理模块的输入即为图像目标检测模型中已检测出但未被过滤的检测结果，根据时间一致性建立各目标在帧之间的关联，并使用得到的关联信息来优化检测结果。这一类方案由于计算代价低且设计逻辑更为清晰，更适用于工业生产及实时检测的场景。

1.2.2 视频目标检测后处理

视频目标检测与静态图像目标检测最大的区别在于，视频目标检测面向的是视频数据，而视频数据的关键因素是其内部丰富的时间信息和上下文信息。视频中物体的位置和外观遵循时间一致性，换言之，同一个视频片段上的检测结果在很短的时间间隔内不应该发生剧烈的检测框位置和检测框置信度变化。也正因如此，性能良好的静态图像检测器直接运用于视频数据上，由于无法利用视频流中的时间信息和上下文信息，均会出现性能下滑的情况。另一方面，视频数据中存在的画面模糊、物体遮挡、镜头高速移动、物体罕见姿势等现象，也增加了视频目标检测的难度，此时视频流中的时间信息和上下文信息就显得尤为重要。基于计算性能和研发成本的考虑，通过后处理机制弥补静态图像检测模型无法利用视频数据时间信息和上下文信息的缺陷，是一种在成本和收益方面都较为可观的解决方案。

后处理框架的提出是为了将静态目标检测器更好的运用于视频数据上，与此同时保证网络模型不会变得更为复杂。这类后处理方法应用于图像检测器每一帧的检测结果，结合视频数据的时间信息和上下文信息，给出最终的预测结果。从检测精度上看，后处理方案很好的完成了辅助图像目标检测器从图像数据移植到视频数据的工作，并且在计算代价和开发成本上也远小于基于视频数据设计视频目标检测器所需要的代价和成本。

常见的后处理方法可以被分成两类：一类是建立帧之间的关联，结合上下文信息来进行后处理；另一类则是从特征工程出发，通过视频数据的特征工作来优化检测结果。一般来说，基于特征工作的后处理方法所需的人工设计部分较少。但不得不考虑的问题是，基于特征工作的后处理方法需要视频数据集进行训练。与传统图像数据集相比，视频数据集收集和标注所需的工作量更大，因此可用的公开数据集也少之又少。而基于帧间关联的后处理使用公开的图像

数据集即可完成训练，是一种成效理想且实现代价较低的后处理方案，故而在实际工程项目中使用频率也更高。现有的基于帧间关联的后处理方法大多依赖光流计算、目标跟踪或跨时间排序等技术手段。从计算复杂性的角度来分析，光流的计算复杂性更高。而且光流分析与目标检测这一主线任务没有太直接的联系，仅仅只是作为后处理的一种辅助手段而存在于整个处理流程中。因此引入光流计算或目标跟踪这类技术手段，会使得计算开销明显增加，在获得准确率提升的同时却降低了检测效率。与上述问题相似的是，现存的很多基于帧间关联的后处理方案都存在计算代价高，不能应用于在线视频目标检测的问题。回溯到前文提及的目标检测常见应用场景，自动驾驶、视频监控、机器人视觉等诸多任务都是需要实时视频流检测及分析的，少有利用本地视频文件进行分析的应用场景，因此实时性也是视频目标检测一个很重要的考量标准，而现存的基于帧间关联实现后处理的方案在实时性上仍有很大的优化空间。

1.3 本文研究内容

本文主要研究应用于视频目标检测的后处理方案，其目的是在静态目标检测模型的基础上，通过后处理方案的设计和实现，将静态目标检测模型从图像数据移植到视频数据上。此类方案主要利用视频数据中丰富的时间信息和上下文信息，优化图像检测模型在视频数据上性能下滑的问题。本文所研究的后处理方案所需的计算成本较低，是一种高性价比的视频目标检测优化策略，并且对于不同的图像目标检测模型具有普适性。

更进一步的，本文对后处理方案的研究从非实时后处理和实时后处理两个角度展开。前者致力于最大程度的提高图像目标检测器在视频数据上的检测精度，建立全局的帧间关联来进行后处理。后者回归到视频目标检测的在线使用场景，将工作人员对检测结果的观测需求纳入考虑，首要任务是保证实时性和稳定性，在此基础上尽可能提高检测精度。最后，本文将实时后处理方案应用于实际系统工程中，通过实际效果检验其有效性和实用性。文本的主要研究内容总结如下：

- 首先是对非实时后处理的研究，为了利用视频流中的时间信息和上下文信息，本文设计了一种能应用于静态目标检测模型的后处理框架（Context Information Based Post-Processing，CIBPP）。CIBPP 使用位置信息、语义信息、外观信息来描述一个目标检测框，并通过距离函数计算相邻视频帧之

间任意两个检测框的距离，由此建立距离矩阵，实现跨帧检测框连接。这种连接通过检测框序列的形式记录，每个检测框序列内通过类分数平均来优化该序列内的误检。除此之外，在不同的两段序列之间，通过双向扩散来优化漏检。CIBPP 利用整个视频片段内的上下文信息，对每个独立视频帧的检测结果进行优化。该框架在 ImageNet VID 数据集上的实验结果证明了其有效性，并且具有普适性，并不限制具体的图像目标检测模型选择，因此是一种高性价比的视频目标检测优化策略。

- 考虑到视频目标检测的使用场景中存在诸多实时检测的情况，本文提出了一种适用于在线实时检测场景的后处理策略（Online Post-Processing and Smooth，OPP-Smooth），并使用卡尔曼滤波对检测结果进行平滑。OPP-Smooth 在处理每一帧时，仅利用当前帧及前序帧的信息，建立信息队列，在队列内对前向若干帧的检测结果进行连接，并实时优化检测框的位置、类分数等信息。通过卡尔曼滤波来平衡由于像素信息引起的相邻帧之间检测结果的抖动，以达到稳定检测框的效果，使得用户可以更好的对实时检测结果进行观测。和其他在线视频目标检测后处理方案相比，本文所提方法取得了更优异的效果，并且在实时检测场景下能够得到更稳定的检测框输出。
- 最后本文将提出的后处理框架融合进实际系统工程中，该系统场景为井下煤矿作业场景。系统通过实时目标检测识别出井下的工作人员，同时标识出电缆槽的位置作为安全工作区域和危险工作区域的分界，利用人员位置和电缆槽位置来判断当前是否有工作人员处于危险区域，若有则及时给出预警。该系统支持连接摄像头设备，并在用户页面实时显示当前摄像头画面经过目标检测、后处理、状态分析之后的结果，在提供井下煤矿作业安全保障的同时，也很好的体现了本文提出的后处理框架在实际应用中的价值。

1.4 本文结构安排

本文的研究围绕视频目标检测后处理策略展开，分为非实时后处理和实时后处理两个方向。非实时后处理聚焦于视频目标检测准确度的提升，使用后处理机制弥补图像目标检测模型无法利用视频数据上下文信息的缺陷。除此之外，本文同样关注在线视频目标检测的实时性要求，又提出了另一种实时后处

理方案，在保证实时性和稳定性的前提下尽可能提升检测精度。最后将所提后处理框架应用于实际的工业系统应用中。全文共有六个章节，整体结构关系如图 1-1所示。

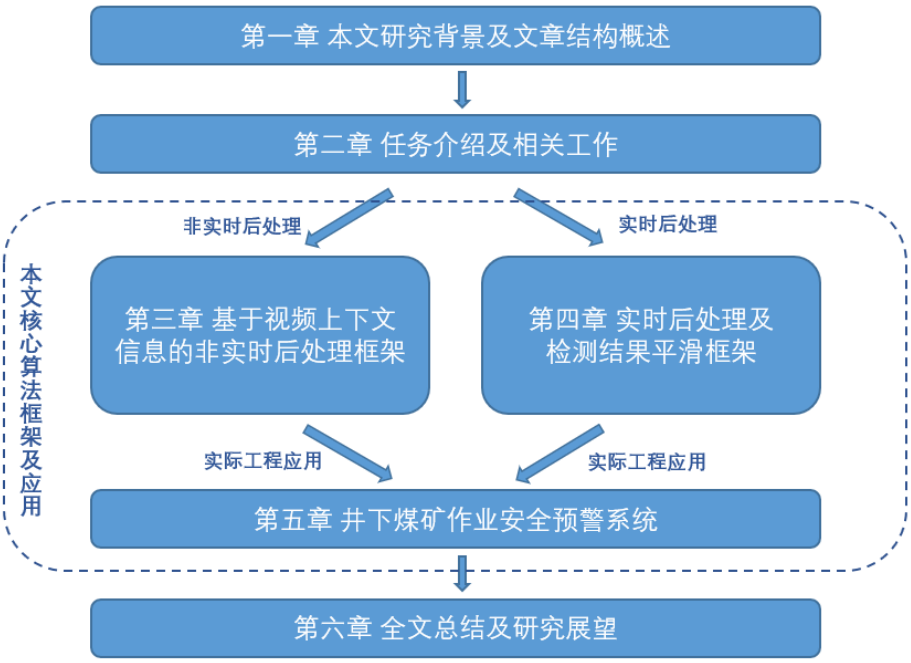


图 1-1: 本文章节结构

第一章绪论部分，主要阐述本文的研究背景，从机器学习到目标检测，再到视频目标检测及后处理，一步步细化分析。同时还介绍了目标检测及后处理这两个领域的研究现状。最后点明本文研究内容；第二章为相关工作部分，首先给出目标检测领域的任务描述及术语介绍，进而从图像目标检测、视频目标检测、后处理策略三个角度介绍一些代表性工作；第三章主要介绍本文提出的基于视频上下文信息的非实时后处理框架，从设计到实现，再到实验验证，对该框架的细节进行阐述和论证；第四章主要介绍本文提出的在线视频目标检测后处理框架，与第三章内容相比，第四章所介绍的框架需要在实时检测这一条件的限制下，完成视频目标检测结果优化。其设计及实现细节均在第四章中进行阐述；第五章介绍在线视频目标检测后处理框架在实际工程中的应用，对整个井下煤矿作业安全预警系统的研发背景、系统需求、系统设计、运行效果进行详细分析，充分说了文本所提方法的实际应用价值；第六章为本文的总结部分，以及对后续工作和研究方向的展望。

第二章　相关工作

本章主要介绍本文研究内容所涉及的相关背景知识，以及国内外具有代表性的工作内容。首先介绍目标检测的任务描述及领域内相关术语，接着分析了图像目标检测的几项代表性工作。进而又介绍了由图像目标检测引申出的视频目标检测任务，及其主流解决方案。视频目标检测的解决方案之一是静态目标检测器结合后处理机制，因此本章也对目前常见的几种后处理机制给出介绍，为后续章节中提出的后处理框架提供一定的理论基础和背景铺垫。

2.1　目标检测

2.1.1　任务描述

目标检测作为计算机视觉任务的重要分支，其主要目的是对图像中的物体进行识别和定位，通常包括以下步骤：首先对图像或视频进行预处理，如去噪、归一化等；接着通过特征提取算法，提取图像中的关键特征，如SIFT[44]、HOG 等；然后通过目标检测模型，对图像中的物体进行识别和定位，比如使用 YOLO、Faster R-CNN 等模型；最后对结果进行后处理，比如移除重叠检测框并保留可信度最高的检测框。目标检测在诸多领域都得到了广泛应用，如自动驾驶、安防监控、人脸识别等。它涉及到深度学习、计算机视觉和图形学处理等多种技术，是具有挑战性且实用价值高的机器学习任务。

2.1.2　术语介绍

随着深度学习技术的发展，目标检测从早期的图像目标检测又拓展出了视频目标检测这一任务，涉及到的技术也愈发多元化。因此，为了在本文中更清晰的介绍目标检测及后处理技术，现列举一些常用术语及其含义：

- **锚框（Anchor Box）**：锚框是一类预先定义的、固定的边界框，在目标检测中通常使用它来映射图像中的目标。锚框这一概念在 Faster R-CNN 中首次被提出，并在后来提出的诸多检测模型中得到了更广泛的应用。

- **特征图（Feature Map）**：特征图是在卷积神经网络中提取出的特征的数字表示。主要作用是将输入的图像信息转换为更加简洁的特征表示，从而为目标检测任务提供有效的特征描述。
- **检测框交并比（Intersection over Union，IoU）**：检测框交并比是衡量两个检测框重叠情况的指标。检测框交并比的值落在 [0,1] 之间，值越接近 1，说明两个边界框的重叠部分越多。关于 IoU 的具体计算方式将在本文2.1.3中详细介绍。
- **非极大值抑制（Non-Maximum Suppression，NMS）**：非极大值抑制是一种剔除多余边界框的方法，它通过预先选择得分最高的边界框，并删除与该边界框交并比超过预先定义的阈值的其他边界框，来确保最终预测结果中只包含一个目标。
- **两阶段检测器（Two-Stage Detector）**：两阶段检测器是一种常见的目标检测模型，该类模型通过两个阶段的处理来识别图像中的目标。第一阶段生成候选目标区域，第二阶段对候选区域进行评估，以确定最终的目标位置。Faster R-CNN 算法就是一种典型的两阶段检测器。
- **一阶段检测器（One-Stage Detector）**：一阶段检测器是与两阶段检测器相对的另一类目标检测模型，该类模型在单个阶段内识别图像中的目标。相比于两阶段检测器，一阶段检测器的处理速度更快，但预测精度通常略低。YOLO 和 SSD[45] 算法均属于一阶段检测器。

2.1.3 性能度量

前一节介绍了目标检测领域的相关术语，本节主要介绍的是度量目标检测模型效果优劣的方式，即评价指标。目前有多种性能指标可以从不同角度评估检测器的性能优劣，比如准确率（Precision）、召回率（Recall）、FPS、AP（Average Precision）、mAP（mean Average Precision）等，其中准确率可以由 IoU 推导得到，mAP 则是目前最为常用的单一指标。

首先给出 IoU 的计算方式如式2-1，表示两个检测框重叠情况，具体定义参考本文2.1.2。其中 A、B 为两个检测框，$Area\ of\ Overlap$ 表示 A、B 重叠部分的面积，$Area\ of\ Union$ 表示 A、B 合并之后的面积。

$$IoU = \frac{|A \cap B|}{|A \cup B|} = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{2-1}$$

对于机器学习中最基础的分类问题，可将对象真实类别与模型预测类别的组合划分为四种：TP（True Positive，真正例）、FP（False Positive，假正例）、TN（True Negative，真反例）、FN（False Negative，假反例）。混淆矩阵（Confusion Matrix）如表2-1所示。 上述划分只考虑分类问题种预测类别正

表 2-1: 分类问题的结果混淆矩阵

	预测结果：正例	预测结果：反例
真实结果：正例	TP（真正例）	FN（假反例）
真实结果：反例	FP（假正例）	TN（真反例）

确与否，而在目标检测任务中，还需要设定一个 IoU 阈值来判定检测结果是否正确。具体来说，当预测检测框和真实值（Ground Truth）分类相同且二者 IoU 超过该阈值时，该检测结果被分类为 TP。相应的当预测检测框和真实值分类相同但 IoU 未超过该阈值时，该检测结果被分类为 FP。当预测检测框和真实值分类不相同时，该检测结果被分类为 FN。若模型给出的预测检测框在真实值中不存在则被划分为 FP。而模型未给出真实值对应的检测结果则被划分为 FN。基于上述内容，可得目标检测任务的准确率和召回率的定义如下：

$$\begin{aligned} Precision &= \frac{TP}{TP+FP} \\ Recall &= \frac{TP}{TP+FN} \end{aligned} \tag{2-2}$$

理想情况下我们希望检测模型的准确率和召回率都尽可能高，但二者在某些情况下是矛盾的。P-R 曲线（Precision-Recall Curve）是常用的辅助分析手段。对于每一组 P-R 值，以召回率为横轴，准确率为纵轴，即可绘制出 P-R 曲线。而 AP 就是平均精度，顾名思义就是对 P-R 曲线上的准确度求均值，理论上使用积分来进行计算，如式2-3。其中 r 表示召回率，p 表示准确率。

$$AP = \int_0^1 p(r)dr \tag{2-3}$$

而在实际使用中，往往是对是对 P-R 曲线进行平滑处理再累加。对于每个召回值，均可以计算出对应的最大准确率，然后对所有的准确率取平均即得到最后的 AP 值。

AP 能够衡量目标检测模型在每个类别上的检测性能，而不同的检测器对

基本检测类别的设定均有不同。因此需要单一指标来横向对比不同检测器的性能，从而就引出了 mAP。mAP 衡量的是模型在所有类别上的好坏，具体计算方式即为对所有类别的 AP 取均值，如式 2-4，其中 *Classes Count* 是类别数量，*AP*(*i*) 表示第 *i* 个基本类别的平均精度。

$$mAP = \frac{\sum_{i=1}^{Classes\ Count} AP(i)}{Classes\ Count} \tag{2-4}$$

除了上述提到的衡量检测器精度的几个相关指标之外，目标检测算法另一个不可忽视的评估指标是计算速度。尤其是在实时检测的场景下，对计算速度的要求更为苛刻。FPS（Frame Per Second）用来评估目标检测的速度，即每秒内模型处理的图片数量。FPS 越高，则处理单张图片的耗时越短。另外需要注意的是，硬件环境对计算速度会产生较大的影响，因此对比不同检测器的速度时，需要保证硬件环境相同。

2.2 图像目标检测

2014 年以前的图像目标检测工作属于“传统目标检测”，随着 2014 年 R-CNN 的提出，目标检测正式进入了“基于深度学习的目标检测”时期，本节主要介绍基于深度学习的图像目标检测工作。当前主流的图像目标检测方法可以被分为两类：两阶段检测器、一阶段检测器。两阶段检测器使用独立模块用于生成候选区域（Region Proposals），该类检测器在第一阶段产生一定数量的候选区域，接着在第二阶段对候选区域进行分类和定位。但由于整个处理逻辑被分成两个阶段，整体架构会更为复杂，检测流程总耗时也会稍长。而一阶段检测器往往通过密集采样，直接对语义目标进行分类和定位，整个流程在一个阶段内完成，在处理逻辑上更为简单，计算速度也更快，相应的其精度相比于两阶段检测器会略低。本节将对一些具有代表性的两阶段检测器和一阶段检测器进行介绍。

2.2.1 R-CNN

得益于深度学习的高速发展，机器学习多个领域均取得了重大技术突破，目标检测领域的研究者们自然而然提出了这样一个问题：卷积神经网络能够学习图像的高级特征表示，是否可以将其引入目标检测领域？Ross B. Girshick

等人于 2014 年发表在 CVPR 上的一篇工作，率先打破了这一僵局，该工作在 CNN 的基础上设计了 R-CNN 模型，在目标检测领域的发展历史中具有里程碑式的意义 [32]。R-CNN 摒弃了传统的滑动窗口和人工选取特征的方法如 HOG 和 SIFT，将候选区域算法和卷积神经网络相结合，使得检测速度和精度明显提升 [46]。

R-CNN 的整体流程如图 2-1所示 [32]，系统首先获取输入图像，从输入图像中提取大约 2000 个自下而上的候选区域。每个候选区域被重新缩放为固定大小，随后输入到使用 ImageNet 预训练过的卷积神经网络模型中，以提取每个候选区域的特征。最后使用线性 SVM 分类器来预测每个区域中对象的存在，并识别出对象类别。R-CNN 显著提升了目标检测的性能，并在 PASCAL VOC 2010 数据集上实现了 53.7% 的平均精度（mAP）。

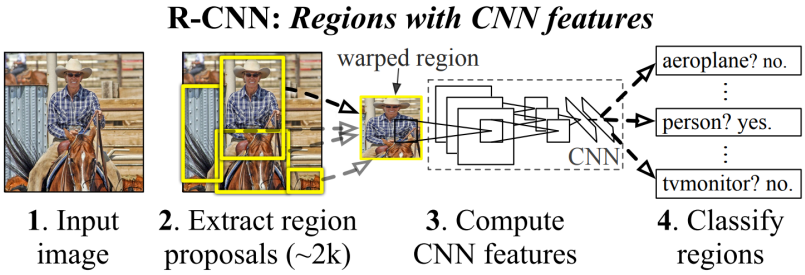


图 2-1: R-CNN 流程图

整个处理流程中最重要的两步即为提取候选区域和计算特征。R-CNN 使用选择性搜索算法（Selective Search）来从输入图像中生成候选区域 [46]。这一算法的优点是对于图像中不同尺寸的物体有很好的适应性并且计算速度快。提取出候选区域后需要完成的任务即为计算特征 [47]。特征计算是指将原始特征转换成一组有明显物理意义的特征，使得构建出的模型能够取得更好的性能。

R-CNN 的提出正式将目标检测引入了基于深度学习的时代，该模型大幅提升了图像目标检测性能，但其缺点也是显而易见的。首先是大量重叠的候选区域会导致重复计算问题，进而影响检测速度；其次是部分候选区域被缩放到固定尺寸之后，产生的畸变可能会引起输入 CNN 的信息部分丢失，从而造成性能瓶颈。也正因为 R-CNN 在这些方面存在有待优化的地方，后续才会有更多的检测框架被提出。

2.2.2 Faster R-CNN

R-CNN 系列目标检测算法包括如下三个：R-CNN、Fast R-CNN、Faster R-CNN[32, 38, 39]。2015 年，R-CNN 的作者 Ross Girshick 等人提出了新的工作 Fast R-CNN[38]，该方法支持在相同的网络结构设置下同时训练检测器和边界框分类回归器。这一方法将 R-CNN 所取得的 58.5% 的 mAP 进一步提高至 70.0%，并且其计算速度比 R-CNN 快两百多倍。尽管如此，Fast R-CNN 的检测速度仍然没有跳出候选区域生成这一步骤的限制。因此，有研究者提出了这样一个问题：能否使用 CNN 模型直接生成区域提议？而随后提出的 Faster R-CNN 就很好地回答了这个问题。

Faster R-CNN 是 R-CNN 系列中最为成熟和完善的算法，也是目标检测领域第一个接近实时的深度学习检测器（使用 ZF-Net[48] 时可以达到 17fps）。作为两阶段检测器中最为经典的目标检测算法。Faster R-CNN 同样是在第一阶段生成候选检测框，随后在第二阶段对这些检测框进行分类。但与 R-CNN 系列前序两个算法不同的是，Faster R-CNN 使用卷积神经网络生成候选检测框，从而实现了端到端的模型训练及预测，突破了 R-CNN 系列算法的速度瓶颈。

Faster R-CNN 最主要的贡献是引入了区域提议网络（Region Proposal Network，RPN），整个流程中用于产生候选区域的卷积神经网络和目标检测的卷积神经网络是共享的。Faster R-CNN 将生成候选区域、特征提取、检测框回归集成到同一个端到端学习框架内。整体流程可以分为四个步骤：首先使用基础的骨干网络实现图像的特征提取，得到的特征图在后续流程中被共享；随后通过 RPN 层生成候选区域；紧接着使用前序两个步骤得到的特征图和候选区域，综合二者信息提取出候选区域对应的特征图，送入后续的全连接层进行分类判定；最后对候选框进行分类，并微调其坐标值。

R-CNN 系列检测器均属于两阶段检测算法，整体上遵循由粗到细的处理范式，粗检主要是为了提高召回率，而精检则进一步细化，注重的是准确率。因此两阶段检测器的计算速度普遍较慢。而后续提出了新的处理范式，牺牲一定的检测精度来换取更短的计算耗时，这类检测模型被称为一阶段检测器。

2.2.3 YOLOv1-v3

YOLO 名字的由来是 You Only Look Once，指的是对整张图片扫描一次即可从中获取物体的类别与位置信息。YOLO 没有显式求取候选区域的过程，是

深度学习目标检测时期首个一阶段检测器。Joseph 是 YOLOv1-YOLOv3 的作者，该系列后续经历了诸多完善和优化，已经更新至 YOLOv8 版本 [40–42]。其他后处理方法多数以 YOLOv3 版本的模型作为基检测器，因此本节主要对 YOLOv1-YOLOv3 进行分析，这三个模型在技术实现上层层递进、逐步优化。

YOLOv1 首次通过一阶段的方式实现目标检测任务。该项工作在全图上利用卷积神经网络，从而完成关于目标类别和位置的预测。这一思路的优点是计算速度快，前文提到 Faster R-CNN 在最好情况下能达到 17fps，而标准 YOLOv1 的检测速度可以达到 45fps。YOLOv1 将一幅图像分成 s×s 个网格，如果某个目标的中心落在这个网格中，那么该网格就需要负责预测这一目标。YOLOv1 在速度上远远领先 Faster R-CNN，但是在精度方面却没有取得很大提升，其原因分析如下：虽然每个网格可以预测一定数量的检测框，但是最终只会保留一个作为输出。因此，YOLOv1 对相互靠近的目标以及画面占比很小的目标检测效果不佳。

YOLOv2 在保持 YOLOv1 良好计算速度的基础上，从预测更精准（Better）、速度更快（Faster）、识别对象更多（Stronger）这三个方面进行了改进。作者在原著论文中指出，YOLOv2 的首要目标是提高召回率，其次在提升定位准确率的同时保持分类的准确率。具体来说，YOLOv2 比较重要的几点优化如下：

1. 使用批归一化（Batch Normalization）：主要用于解决反向传播过程中梯度消失和梯度爆炸的问题，降低对一些超参数的依赖，让网络提高收敛性。
2. 使用高分辨率分类器（High Resolution Classifier）：主要为了避免模型只能适应低分辨率的训练数据。
3. 使用锚框：前文提到的两阶段检测器经典算法 Faster R-CNN 提出了 RPN 用来预测锚框的偏移量和置信度。因此，Joseph Redmon 借鉴了 Faster R-CNN 的思想，将锚框引入了 YOLOv2。
4. 使用 K-means 聚类算法 [51–53]：YOLOv2 尝试统计出更符合样本对象尺寸的先验框，因而引入了 K-means 聚类分析。由于检测任务的目标是提高 IoU 分数，那么使用传统的欧式距离作为度量会产生较大偏差。因此定义距离度量如下：

$$d(box, centroid) = 1 - IoU(box, centroid) \tag{2-5}$$

其中 *centroid* 是聚类时被选作中心的边框，*box* 为其它边框，由上式可知，IoU 越大，得到的距离值越小，距离越近。

5. 使用多尺度训练（Multi-Scale Training）：主要目的是提高模型的健壮性，使之能够检测不同尺寸的图片。

在经过上述几项优化之后，YOLOv2 的性能和速度均得到了提升，前一个版本中存在的小面积目标难以被检测的缺陷在 YOLOv2 中也得到了很好的解决。在小物体较多的 COCO 数据集上，YOLOv2 性能与 Faster R-CNN 不相上下但是检测速度更快。尽管如此，YOLOv2 仍存在美中不足的地方：没有结合多尺度特征进行预测，其骨干网络结构也仍有优化的空间。经历了前两个 YOLO 版本的迭代之后，YOLOv3 的改进更多的是基于工程的角度而非算法思想的角度。YOLOv3 相对于 YOLOv2 的主要改进有以下几点：

1. 调整骨干网络结构：引入 ResNet[54, 55] 残差结构，加深了骨干网络的层数但又能够避免梯度消失问题，在低运算量下仍能保持较好的性能。
2. 利用多尺度特征进行目标检测：YOLOv3 采用了三个不同尺度的特征图来检测目标，这一点借鉴了 FPN（Feature Pyramid Networks）[56]。
3. 分类用 Logistic 函数取代 Softmax 函数 [49, 50]：在预测目标类别时使用 Logistic 函数的输出，使得模型能够支持多标签对象的预测。

通过对 YOLOv2 的几点改进，YOLOv3 的计算速度和精度均不逊色于前两个版本。在 COCO 数据集上 mAP 为 50 时，同等精度下 YOLOv3 的计算速度比前两个版本的 YOLO 模型更快。但如果要求更精准的预测边框，YOLOv3 的性能则会有所下滑。具体模型性能分析可以参考原著论文 [42]。综合来看，YOLO 系列算法的提出使得目标检测任务能够满足实时性要求，但相较于两阶段检测器，精度方面会有一定限制。

2.3 视频目标检测

如前文所述，随着卷积神经网络的引入，图像目标检测领域取得了巨大的技术进步。但如果将图像目标检测模型直接移植到视频目标检测的场景中，其性能往往会大幅下降。视频数据可能会存在画面模糊、物体遮挡、镜头高速移动、物体罕见姿势等情况，这些都会降低静态图像目标检测的性能。2015 年的 ImageNet 大规模视觉识别挑战赛（ImageNet Large Scale Visual Recognition Challenge，ILSVRC）将视频目标检测列为了一项新的任务，并提供了一个后续被广为使用的视频目标检测公开数据集 ImageNet VID，促进了这一领域的研究成果在接下来的几年中不断涌现。整体上来看，基于深度学习的视频目标

检测方法可以分为基于光流的方法 [57–59]、基于长短期记忆（Long Short Term Memory，LSTM）的方法 [60–62]、基于后处理策略的方法 [63–65]、基于跟踪的方法 [66–68]、其他方法 [69–73]。本节主要基于上述分类，对一些具有代表性的方法进行阐述，其中基于后处理的方法将在下一节详细阐述。

2.3.1 基于光流的方法

光流分析是视频分析中的一项基本任务，光流这一概念由 Malcolm 和 Gibson 于 1950 年首次提出 [74, 75]。光流的测算实际上就是通过检测图像像素点的强度随时间的变化，进而推断出物体移动速度及方向的方法。

鉴于光流测算方法在视频分析领域的有效性，有研究人员开始考虑将光流与深度学习结合从而形成新的视频目标检测框架。2015 年 Dosovitskiy 等人提出的 FlowNet[76] 首次将光流分析法扩展到深度神经网络。而在视频目标检测中应用光流估计的两个主要难点即为关键帧（key frame）选择和特征聚合（feature aggregation）。

2017 年提出的 DFF（Deep Feature Flow）[59] 利用光流网络来模拟原始像素中的对应关系。DFF 所提出的视频检测网络分为两个子网，N_{feat} 是特征子网，N_{task} 是任务子网，具体结构如图2–2所示 [59]。DFF 只提取关键帧上的信息，随后通过光流场将关键帧的信息传播到其他帧（即图中的 propagation 操作）。DFF 引入了一项关键操作：光流变换（Wrap），用于传播关键帧的特征到非关键帧。该算法在很大程度上加速了非关键帧的检测，DFF 在 ImageNet VID 数据集上能够取得 20fps+73.1%mAP 的结果。但显而易见的是，DFF 在关键帧的选择方面存在一定的局限性，人为设计的选择策略存在很大的优化空间。

和 DFF 相比，2017 年提出的 FGFA（Flow-Guided Feature Aggregation）使用了截然不同的关键帧选择方式 [58]。该算法主要致力于提高检测准确性而非加快计算速度。FGFA 使用 ResNet-101 作为骨干网络在视频的每一帧中都进行特征提取，再根据光流信息，将临近帧的特征传播到当前帧，用来强化当前帧的特征信息。FGFA 所提出的特征融合方法在 ImageNet VID 数据集上取得了 1.36fps+%76.3mAP 的结果，该方法很好的改善了由于运动模糊、罕见姿态、视频失焦等现象引起的检测精度下降的问题，但明显增加了计算时间。

在上述方法中，光流计算被广泛用于跨帧传播特征，提供了单个视频帧和其他帧之间的信息交换。但是引入额外的光流模型显著增加了整个目标检测器

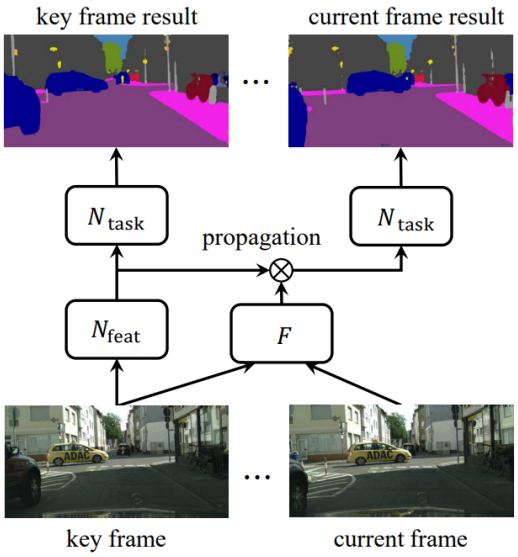


图 2-2: DFF 网络结构图

的模型大小。并且当前光流估计方法并不适用于高级特征，比如特征描述中某方向的轻微偏移可能对应图像内容的大幅偏移，因此使用光流计算来辅助视频目标检测会将检测器复杂化，且在性能上也有局限。

2.3.2 基于 LSTM 的方法

LSTM 这一概念于 1997 年由 Hochreiter 等人首次提出 [60]，而 Xingjian 等人在 2015 年提出的卷积 LSTM（Convolutional LSTM）[77] 可以被视为一种特殊类型的递归神经网络，其模型结构比光流模型要简单很多，非常适合插入深度学习网络。

Lu 等人在 2017 年提出了联合 LSTM 方法（Association LSTM）[78]，用来提高视频目标检测的精度。与传统 LSTM 不同，联合 LSTM 在生成相关特征的同时，直接返回目标的位置和类别信息。整个算法的主要框架由目标检测模型 SSD[45] 和卷积 LSTM 组成。SSD 模型在视频的每一帧中进行目标检测，再根据其检测结果提取物体特征，最后将这些特征组合并送入卷积 LSTM。联合 LSTM 算法充分利用了卷积 LSTM 的优点，增强了视频目标检测的鲁棒性，设计思路清晰且行之有效。但该方法也存在一定的局限性，即相邻帧中提取的目标运动变化信息是有限的，只能反映目标在短时间内的运动情况，对于长期运动变化则无法获知相关信息。

Xiao 等人于 2018 年发表的工作中提出了时空记忆网络 STMN（Spatial-

Temporal Memory Network）[79]。STMN 实质上是一个循环神经网络，整个架构包括时空记忆模块 STMM（Spatial-Temporal Memory Module）和 RNN（Recurrent Neural Network）两部分，能够学习包含在多个视频帧中的运动信息。STMM 是 STMN 的核心模块，其本质是一个卷积递归计算单元，每一帧都对应一个 STMM，各个 STMM 模块之间能够相互通信。整个流程通过组合跨帧信息来传播和聚合特征，以此来获得目标在长时间的变化信息。

使用 LSTM 模型或者光流模型进行视频目标检测的初衷是相似的，这两种方法本质上都是利用视频数据中的上下文信息来优化网络结构。二者主要区别是基于光流分析的方法侧重于目标运动，而基于 LSTM 的方法侧重于空间信息。相比之下基于 LSTM 的方法比基于光流的方法所需的计算成本更低，但是在传输的信息中会包含过多的冗余信息。

2.3.3 基于跟踪的方法

视频目标检测和目标跟踪是两个密切相关的任务，在某种程度上，目标跟踪可以看成是特殊的目标检测。将二者集成在一个算法中来进行视频目标检测时，目标跟踪往往作为一种优化视频目标检测结果的手段而存在。视频对象跟踪模型中包含了视频目标检测所需要的与目标相关的时空信息，并且跟踪模型和检测模型中基本的特征提取网络是可以共享的，使得视频目标检测和目标跟踪这两个任务的融合更加合理。

Mao 等人提出的级联跟踪检测模型 CaTDet[80] 主要使用跟踪算法来减少视频检测算法的计算量。CaTDet 秉持的思想是验证和校准的代价要小于重新检测的代价。整个框架包含跟踪器和检测器两部分，如图2-3所示 [80]，处理步骤如下：

1. 将视频帧输入轻量级检测器（即图中的 Proposal Network），在每帧中检测可能存在目标的区域，称为候选区域。
2. 使用跟踪器跟踪当前帧中的高置信度检测框（即图中的 Tracker），并预测该检测框在下一帧中的位置。
3. 对于每一帧，将候选区域和跟踪器当前跟踪结果相结合，得到一个目标区域，输入细化网络对检测结果进行标定（即图中的 Refinement Network）。

上述策略无法避免的问题是，物体或者摄像机的位置偏移会增加预测难度，因此整个策略的鲁棒性较低。

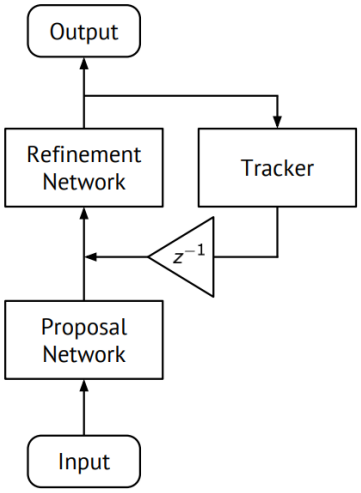


图 2-3: CaTDet 中跟踪器和检测器协同示意图

Feichtenhofer 等人提出的检测和跟踪（Detect to Track and Track to Detect，D&T）[81] 方法将检测和跟踪集成到一个框架中，并使用感兴趣区域（Region of Interest，RoI）来辅助完成视频目标检测任务。D&T 采用了一种基于区域的全卷积网络 R-FCN（Region-based Fully Convolutional Networks）[82]，并将该策略扩展到多帧检测和跟踪。检测器和跟踪器共享一些特征，以此缓解不正确的跟踪结果带来的精度损失，并且有效降低了计算量。该方法在 ImageNet VID 数据集上将 mAP 提升至了 82.0%。

尽管利用跟踪算法辅助进行视频目标检测获得了较好的精度表现，但本质上仍然引入了目标跟踪这一额外任务模块。整个流程需要完成两项任务，一定程度上使视频目标检测这一问题变得更为复杂。而更理想的方案是在不增加模型复杂度的情况下，利用视频数据的上下文信息，而非引入另一个新任务用于优化视频目标检测。

2.3.4 其他方法

除了前文中提到的几类视频目标检测方法之外，还有一些使用其他策略来优化检测性能的方法。Chin 等人于 2019 年提出的一项工作研究了输入图像大小对视频目标检测性能的影响 [83]。基于上述思路，提出了 AdaScale 框架，用于自适应的选择输入图像的大小，具体来说即为根据当前帧的信息来预测下一帧的最佳尺寸。通过这种方式，增加检测结果中真正例的数量，从而实现精度上的提升。

减少视频目标检测任务所需的计算资源，有研究人员考虑引入注意力机制进行特征图对齐。这种思路首先是在机器翻译领域得到运用，而后又被应用于视频目标检测领域 [84–87]。

Deng 等人发表于 2019 年的一项工作中提出了关系提取网络 RDN[88]，RDN 利用视频中的对象关系传播和聚合特征图。这项工作的创新点是用多阶段推理来提取关系，并且用多阶段推理所获得的候选检测框进行特征聚合，再使用聚合后的特征生成最终分类结果和检测框信息。该方法在 ImageNet VID 上获得了 83.2% 的 mAP。

RDN 仅仅利用了局部时间信息，Wu 等人提出的工作中，引入了 SELSA[89] 模块，与 Deng 等人在视频目标检测框架中使用的 OGEMN[90] 模块类似，这两个算法都利用了全局时间信息。SELSA 利用整个序列上的候选检测框之间的关系，并融合相关的特征图用于分类和回归。而 OGEMN 使用面向对象的外部存储器来存储像素和实例级别的特征，用于更进一步的全局信息融合。

Chen 等人在 2020 年的一项工作中提出了 MEGA[91]。MEGA 利用全局信息来识别模糊目标，利用局部信息定位目标在视频帧中的具体位置。Guo 等人提出的渐进式稀疏局部注意力方法（Progressive Sparse Local Attention，PSLA）[92] 主要利用长时信息，使用注意力机制增强每个特征单元的信息，通过跨帧的空间信息以及逐渐稀疏的步幅在局部区域中传播特征。这两种方式都属于同时使用了全局时间信息和局部时间信息的方案。

综上所述，除了基于光流的方法、基于长短期记忆的方法、基于跟踪的方法以及将在下一节中讨论的基于后处理策略的方法，仍有许多角度新颖的方法用于视频目标检测结果优化。但这些方法的主流思想往往是将其他领域的优秀成果引入视频目标检测领域，检测性能得到提升的同时，整个流程和任务的复杂度也会随之增加。

2.4 后处理策略

图像目标检测在过去的几年中获得了大幅性能提升，基于后处理的视频目标检测正是利用性能优异的图像目标检测器来解决视频目标检测问题。对于后处理策略而言，其任务就是在图像目标检测器完成每个视频帧的目标检测后，通过一定的机制对视频数据时间信息和上下文信息加以利用，最终提升视频目标检测的精度。而后处理方法之间的主要区别之一即为跨帧建立检测结果连接

时所使用的策略。除此之外，近两年也提出了一些更为新颖的启发式后处理方法，本节将对几种后处理策略进行分析说明。

2.4.1 Seq-NMS

本文2.2中提及的目标检测模型均是为静态图像目标检测而设计。当任务场景变成视频目标检测时，忽略时间信息和上下文信息则是对视频语义的明显浪费。

Han 等人提出的 Seq-NMS[63] 后处理算法主要使用上下文信息来对检测框进行重排列。Seq-NMS 算法执行的前提是：视频片段内的相邻帧往往存在相似的目标，并且检测框也在位置和大小上高度相似，即具有时间一致性。Seq-NMS 正是为了利用这种时间一致性而提出的启发式方法，一般目标检测模型中进行极大值抑制操作时只考虑当前视频帧中的检测框，而 Seq-NMS 考虑的是整个视频流的检测框。Seq-NMS 整体可以分为三步：序列选择（Sequence Selection）；序列重赋分（Sequence Re-scoring）；检测框抑制（Suppression）。工作机制如图2-4所示 [63]。

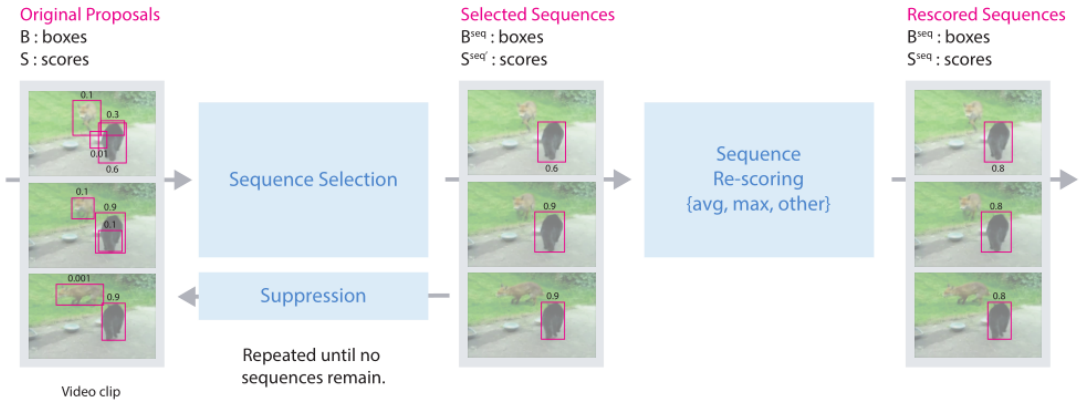


图 2-4: Seq-NMS 中的序列选择、重赋分、检测框抑制机制

其中，在序列选择部分，对于视频片段 V 中的每对相邻帧，前一帧中的检测框与后一帧中的检测框若 IoU 高于某个阈值，则进行连接操作。通过这种方式在整个视频片段中建立检测框之间的联系。更进一步的，在这种约束下，尝

试最大化检测框连接的得分。形式化的计算方式如下：

$$\begin{aligned} i' &= \underset{i_{t_s},\ldots,i_{t_e}}{\operatorname{argmax}} \sum_{t=t_s}^{t_e} s_t\left[i_t\right] \\ & s.t. 0 \le t_s \le t_e < T \\ & s.t. \operatorname{IoU}\left(b_t\left[i_t\right], b_{t+1}\left[i_{t+1}\right]\right) > 0.5, \forall t \in [t_s, t_e) \end{aligned} \tag{2-6}$$

视频序列由 T 个视频帧组成，对于每个视频帧 t，都对应该帧中的一组检测框 b_t 和一组检测框得分 s_t，i_t 表示视频帧 t 上的检测框索引。整个视频对应的检测框集合记为 $B = \{b_0, \ldots, b_T\}$，整个视频中检测框对应的得分集合记为 $S = \{s_0, \ldots, s_T\}$。建立检测框连接可以通过动态规划来实现，算法返回一组索引 i' 组成的集合，根据该索引集合提取出一个检测框序列 $B^{\text{seq}} = \{b_{t_s}\left[i_{t_s}\right], \ldots, b_{t_e}\left[i_{t_e}\right]\}$ 和对应的得分序列 $S^{seq'} = \{s_{t_s}\left[i_{t_s}\right], \ldots, s_{t_e}\left[i_{t_c}\right]\}$。$s$ 和 e 表示一个已连接的检测框序列的起始位置和结束位置。

得到检测框序列和得分序列后，在得分序列上应用函数修改每个检测框的得分。在 Seq-NMS 的实验中，分别使用了平均值函数和最大值函数。。在 ImageNet VID 上的结果表明，相比于直接应用图像目标检测器，增加了 Seq-NMS 之后视频目标检测的 mAP 可以提高 4% 左右。但该方法较为明显的缺陷是，可能会在序列匹配阶段从一个目标漂移到另一个目标，引起后续的错误。

2.4.2 T-CNN

前一小节介绍的 Seq-NMS 是较为经典的后处理策略，该方案处理机制简单，相应的鲁棒性比较差。后续 Kang 等人提出的 T-CNN（Tubelets with Convolutional Neural Networks）[64] 在 Seq-NMS 的基础上，建立了更为完善的机制。T-CNN 中的检测框序列通过 Oneata[93] 等人的工作来获取，获取的序列称为“tubeletes”。对于这些 tubeltes，T-CNN 将其视为独立的单元，在单元内应用长时约束。

T-CNN 的整体结构如图2-5所示 [64]，分为四个步骤：静止图像检测器（对应图中 Still-image Detection 模块）对所有视频帧独立地生成候选检测框并分配初始得分；多上下文抑制 MCS（Multi-Context Suppression）模块结合上下文信息以降低误检的数量，此外还设计了基于运动的传播模块 MGP（Motion-Guided Propagation），利用运动信息将一帧中的检测结果传播到相邻几帧，从

而减少漏检的情况；接着在一个检测框序列内刷新各个检测框得分（对应图中 Tubelet Re-scoring 模块），这部分思路和 Seq-NMS 一致；最后对于不同基检测器的结果进行组合（对应图中 Model Combination 模块），得到最终输出。

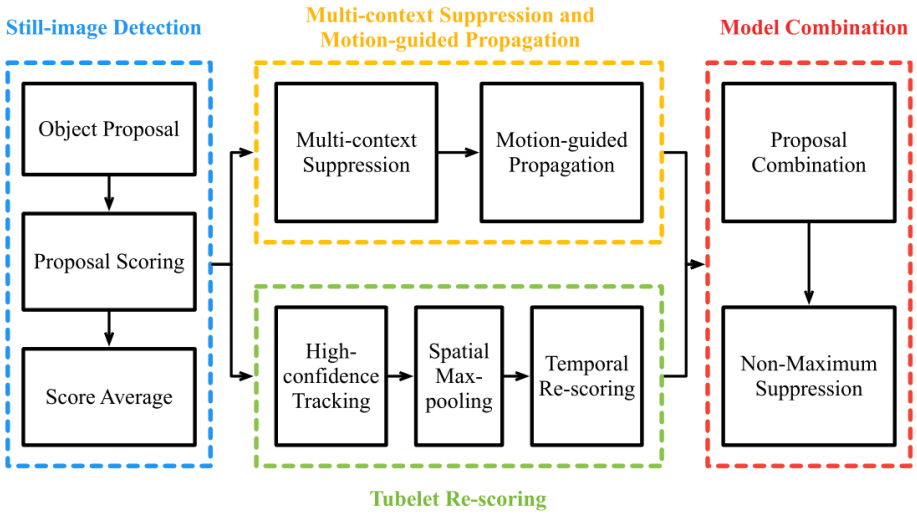


图 2-5: T-CNN 框架图

在 ImageNet VID 数据集上的结果表明，添加 T-CNN 之后视频目标检测的 mAP 可以提高约 6.7%，很好的利用了视频数据的时间一致性。但是与本文2.3中所述的几种视频目标检测方法类似，T-CNN 引入了光流计算模块，增加了整个框架的复杂度。并且 T-CNN 使用了不止一个静态目标检测器（DeepID[94] 和 CRAFT[95]），导致整体计算速度较慢。关于如何达到检测性能和检测速度上的平衡，还需要进一步探索。

2.4.3 Seq-Bbox

视频目标检测后处理算法主要通过建立检测框之间的联系来利用时间信息和上下文信息，而如何建立检测框连接则是决定后处理算法好坏的关键。如果连接机制过于简单，会导致整个方法鲁棒性差，而如果使用光流计算、目标跟踪这类方法，则会大幅增加计算开销。为了维持精度和速度的平衡，Belhassen 等人提出了 Seq-Bbox[65] 算法。

Seq-Bbox 包括两个核心步骤：帧级检测框重赋分；序列级连接。而这两个核心步骤的前序操作是通过计算得到检测框序列，然后方可实现序列内的帧级检测框重赋分和序列间的连接。

Seq-Bbox 中建立连接的方式是计算检测框之间的距离，距离定义如下：

$$distance = \frac{1}{similarity} = \frac{1}{IoU(Vctr_i \cdot Vctr_j)} \tag{2-7}$$

其中两个检测框之间的距离记为 *distance*。*similarity* 表示相似度，通过两个检测框的类分数 $Vctr_i$、$Vctr_j$ 点乘得到，相似度越高，检测框之间的距离也就越小。IoU 描述了两个检测框在位置上的相似度，而检测框对应的类分数向量点乘的结果描述了二者的语义信息相似度。

通过计算每对相邻帧之间所有的检测框距离，可以得到距离排序并以此建立检测框序列，进而利用获得的序列进行两个核心步骤的处理。第一个核心步骤为帧级检测框重赋分，指的是在一个检测框序列内，通过类分数平均来提高一些低置信度检测框的类分数，从而减少检测框由于置信度较低而被过滤的情况。第二个核心步骤是序列级连接，即在不同的两个序列之间尝试连接首尾检测框，从而减少漏检的情况。但这两个序列的间隔必须小于某个阈值，才允许连接。

除上述内容之外，对于单个序列内的重赋分操作，Seq-Bbox 也提供了一个可以用于实时检测的版本，具体修改是将类分数计算从原先的序列内平均改成实时加权平均。但漏检问题的优化无法在实时 Seq-Bbox 方法中实现。整体来看，Seq-Bbox 在不引入新模块的基础上，很大程度提高了检测框连接算法的鲁棒性，检测框距离的提出也提供了一种建立检测框连接的新思路。但其性能仍有一定的提升空间，并且 Seq-Bbox 在解决漏检问题时，仍依赖于人工设定的间隔阈值。

2.4.4 PBDE

除了前文中提到的通过跨帧建立检测框连接从而实现后处理操作的方法，也存在其他后处理思路。Li 等人提出的基于检测框密度的后处理方法 PBDE（Post-processing based on Box DEnsity）[96] 就提供了一个新方向。将一段视频作为样本进行连续测试，可以观察到假正例检测框的存在是非常不稳定的，而真正例检测框的存在是趋于稳定的。基于这一现象的启发，Li 等人提出了 PBDE 算法。

PBDE 定义了检测框密度的概念，对于一个真正例检测框而言，理想情况是有大量检测框与之重叠。考虑到所有的检测框都是矩形，所以用每个检测框

的中心点来简化检测框表示。更进一步的，PBDE 使用两点之间的几何距离来描述检测框密度，若一个检测框所在区域检测框密度较高，说明存在大量的其他检测框与其距离相近。

在实际算法实现过程中，Li 等人将 PBDE 植入目标检测的非极大值抑制过程，结合 Soft-NMS[97] 方法，通过检测框密度来抑制假正例检测框，从而提高整个视频目标检测精度。PBDE 算法为我们引入了后处理方案的一个全新研究角度，也在一定程度上提高了视频目标检测的性能。但其缺点主要在于，单独使用 PBDE 作为后处理方案时对检测结果的提升微乎其微，更好的方式是作为其他后处理算法的辅助和补充。

2.5 本章小结

本章主要对视频目标检测后处理方案的背景知识与相关工作进行了介绍。首先介绍了目标检测的任务内容、相关术语及性能度量。其次，本文提出的后处理框架需要以静态图像目标检测器作为基检测器，因而在本章第二节中对于图像目标检测的几项经典工作给出了详细介绍。再次，对于视频目标检测这一场景，除了基于后处理机制这类方法外，还有基于光流的方法、基于长短期记忆的方法、基于跟踪的方法、其他方法这几类。因此需要对这几类方法进行介绍，从而通过对比阐明后处理方案的研究意义。其他主流视频目标检测方法的切入点各有不同，但普遍处理范式是借鉴其他领域的研究成果，并将相应模块引入视频目标检测框架内，比如引入光流计算模块、LSTM 模块、跟踪模块。这类方法虽然能获得精度上的提升，但也增加了整个检测流程的复杂度，并且引入的模块与目标检测任务并没有直接关系，仅仅起到优化视频目标检测结果的作用。最后，本章在第四节中介绍了几种后处理机制，这类方法的目的是在不增加检测流程复杂度的前提下，以最小的计算代价利用视频数据中的上下文信息。设计后处理方案的关键即为如何建立跨帧的检测框连接，这一点在本文提出的方法中也会有所体现。

第三章　基于视频上下文信息的非实时后处理框架

后处理框架是为了将静态图像目标检测器更好的应用于视频数据上而提出的一种算法。本章介绍的基于视频上下文信息的非实时后处理框架 CIBPP，不考虑实时检测，尽可能利用视频数据上下文信息来提高视频目标检测精度。本章首先分析了基检测器融合及数据准备的过程，随后介绍本文提出的后处理框架构成及工作机制，最后通过实验验证该方法的有效性。

3.1　后处理框架与前序检测模型的融合

3.1.1　原始检测结果的形式化表示

如前文2.4所述，基检测器完成对视频数据每帧图像的目标检测之后，才会进入后处理流程。简单来说首先将视频数据输入至基检测器，基检测器给出每个视频帧中所有的检测框位置及其类别、类分数，后处理框架则负责处理基检测器在每一个视频帧上的检测结果。关于是否能在后处理阶段充分利用视频上下文信息，一个很重要的影响因素即为如何建立检测框的跨帧连接。建立跨帧连接时，考虑的信息越多，建立的连接也就越可靠，相应的后处理算法的鲁棒性也就越高。因此，除了基检测器给出的检测框坐标、类分数等信息，还可以考虑引入新的信息作为建立检测框连接的考量因素。参考 Sabater 等人提出的 REPP[98]，CIBPP 同样考虑从基检测器的特征图入手，提取每个检测框对应的语义信息，并使用这类语义信息来辅助计算相邻帧中任意一对检测框的距离。

对于任意一个视频帧中的任意一个检测框，我们使用如下一组信息来描述该检测框：

- 几何信息：基检测器给出的每个检测框的位置和几何特征，$bbox = \{x, y, w, h\}$，其中 $bbox$ 表示检测框（bounding box），(x, y) 为检测框中心点坐标，w 和 h 表示该检测框的宽度和高度。

- 语义信息：基检测器给出的每个检测框对应的类分数向量，$cc \in \mathbb{R}^C$，其中 cc 表示类分数（class confidence），C 是数据集中基本类别的个数。
- 外观信息：一个 L2 正则化的嵌入向量 $app \in \mathbb{R}^{256}$，app 表示外观信息（appearance），其具体计算方式在后文3.1.2中详述。

形如 $\{bbox, cc, app\}$ 的一组描述符包含了检测框不同维度的信息。后续设计距离度量函数时，需要充分考量这些影响因素，进而建立可信的跨帧检测框连接，提高检测结果的时间一致性。

3.1.2 外观信息的获取

前文3.1.1中定义的检测框描述符包括几何信息、语义信息、外观信息三部分，其中几何信息和语义信息可以从图像目标检测模型的检测结果中直接获取，因此需要构建一个单独的模块用于获取外观信息。具体来说，从单帧图像的分析过程来看，基于深度学习的静态目标检测模型使用卷积层来提取图像特征信息，这一步骤会生成若干个特征图。该图像最终会对应一组检测框结果，而每个检测框可以被视为一个感兴趣区域 RoI。基于这一思路，可以增加一个 RoI 池化层，根据每个检测框的位置信息，从整张输入图像对应的特征图中提取出检测框对应的子特征图。随后将各个子特征图缩放成统一大小以解决每个检测框尺寸不一致带来的子特征图大小差异。最后通过一个全连接层将子特征图转化为更低维度的嵌入向量，从而在很大程度上降低数据复杂度，节约整个后处理阶段的计算时间。上述思路在 Faster R-CNN、FaceNet[99]、REPP 中均有涉及。具体结构及流程如图3-1所示 [98]。

鉴于上述结构中存在嵌入层，因此涉及到网络模型的训练过程。本文将训练数据组织成三元组的形式进行训练，具体数据集处理方式及实验设置在3.3中详细说明，训练时使用的损失函数参考 FaceNet，如式3-1所示。

$$L = \sum_{i}^{N} \left[\left\| f(x_i^a) - f\left(x_i^p\right) \right\|_2^2 - \left\| f(x_i^a) - f(x_i^n) \right\|_2^2 + \alpha \right]_+ \tag{3-1}$$

其中 x_i^a 为每次选择的基样本，x_i^p 是相对于 x_i^a 的正样本，而 x_i^n 是相对于 x_i^a 的负样本，$f(*)$ 为样本对应的特征嵌入，α 是正负样本的边界。该损失函数可以最小化同类别同实例检测结果的欧几里得距离，最大化不同类检测结果的欧几里得距离。关于三元组数据及损失函数的详细介绍将在本文3.2.2(3) 中给出。

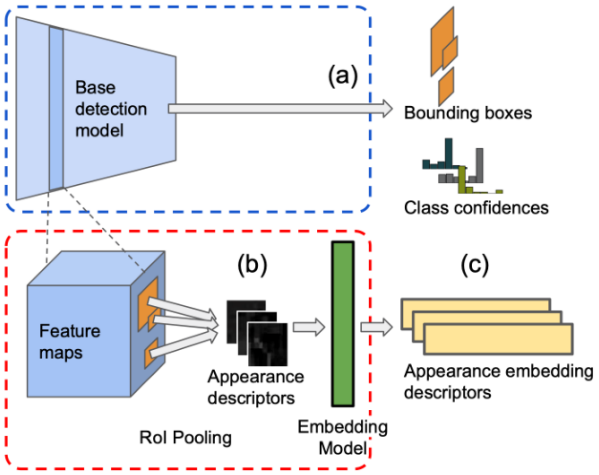


图 3-1: 从基检测器生成的特征图中提取外观信息

3.1.3 检测框距离度量

在明确每个检测框信息的形式化表示后，首先从基检测器中获得每个检测框的几何信息 *bbox* 及语义信息 *cc*，随后通过训练好的 RoI 池化层及嵌入层架构，可以提取到每个检测框的外观信息 *app*，至此即可得到一组完整的检测框信息表示 $\{bbox, cc, app\}$。在此基础上下一个需要解决的问题即为：如何定义两个相邻帧之间检测框的距离度量。这种距离度量是后续进行检测框连接的根据。理想情况是在计算距离时，将每个检测框的几何信息、语义信息、外观信息均纳入考量，距离越近的两个检测框，在各个维度上的信息理论上也是越相似的，建立的检测框连接可信度也就越高。

因此，基于 $\{bbox, cc, app\}$ 这样一组检测框信息描述，需要计算相邻两帧之间任意两个检测框的距离，也可以理解成计算这两个检测框的相似程度，距离和相似度存在如下关系：

$$distance = \frac{1}{similarity} \tag{3-2}$$

这一思想与 Seq-Bbox 中对检测框距离的定义是一致的，显而易见的，两个检测框越相似，其距离应该越小。我们希望最小化 *distance*，即最大化 *similarity*，Seq-Bbox 中仅仅考虑了 IoU 和类分数对距离的影响。而 CIBPP 参考 REPP 的设置，使用一个基础的逻辑回归模型，用于辅助计算 *distance*。通过这样一个逻辑回归模块，可以将检测框的几何信息、语义信息、外观信息都视为对 *distance* 产生影响的因素，并将这种影响体现于逻辑回归模块的输出数据中。

如式2-7，Seq-Bbox 中对 *similarity* 的定义是 $similarity = IoU\left(Vctr_i \cdot Vctr_j\right)$，其中 $Vctr_i$ 和 $Vctr_j$ 分别表示两个检测框的类分数向量。但该定义式无法作为逻辑回归模块的损失函数进行训练，原因是当两个检测框不相交时，根据 IoU 的定义，IoU 取值为 0，无法反映两者相距远近。此时损失函数值为 0，没有梯度回传，无法进行学习训练。除此之外，IoU 存在的另一个缺陷是其无法精确反应两个检测框的重合“质量”，当 IoU 相等时，也存在重合情况的优劣之分，图3-2中间的检测框重叠情况显然更优，在逻辑回归时应当对应更高的分数。

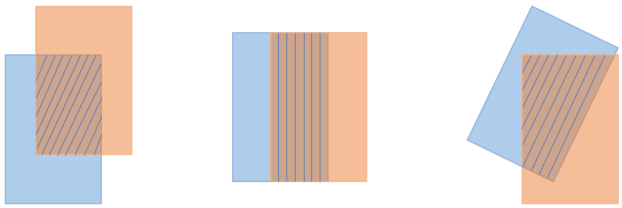


图 3-2: 在 IoU 相等时不同的检测框重叠情况

因此，本文提出的 CIBPP 使用 GIOU（Generalized Intersection over Union）作为检测框重叠程度的度量 [100]。对于任意两个检测框，需要找到一个能覆盖这两个检测框的最小矩形 *rec*，随后计算这个矩形的面积 $Area_{rec}$ 以及两个检测框的 *IoU*，使用 *IoU* 的值减去矩形中不属于两个测检测框的面积和整个矩形面积的比值，即为 *GIoU*。具体定义如式3-3。

$$GIoU = IoU - \frac{|Area_{rec} - Union|}{|Area_{rec}|} = \frac{Overlap}{Union} - \frac{|Area_{rec} - Union|}{|Area_{rec}|} \tag{3-3}$$

其中 *Union* 为检测框合并后的面积，*Overlap* 为检测框重叠部分面积。除了 *GIoU* 之外，本文3.1.1中定义的检测框几何信息 $\{x, y, w, h\}$ 还可以提供两个检测框中心点的欧几里得距离 $distance_{points}$、宽度比值 $ratio_{width}$、高度比值 $ratio_{height}$。这四项数据中，*GIoU* 及 $distance_{points}$ 反映的是两个检测框在位置上的特征关系，而 $ratio_{width}$ 及 $ratio_{height}$ 反映的是检测框在形状上的特征关系。综上所述，在计算任意两个检测框的距离前，它们各自的几何信息 $bbox_i$ 和 $bbox_j$ 通过上述操作，被转化成一组数据：$\{GIoU, distance_{points}, ratio_{width}, ratio_{height}\}$，其中的每一项都是同时与两个检测框相关的。

类似的，在本文3.1.1和3.1.2中介绍的检测框外观信息 *app*，是对单个检测框外观维度的描述。CIBPP 引入 $distance_{app}$ 来描述一对检测框的外观距离，即为两个检测框外观向量对应的欧式距离。综上所述，将目前为止得到的

描述一对检测框几何及外观关系的数据进行整合，随后将其送入逻辑回归分类模型，即可得到一个几何外观相似度分数输出，如式3-4。

$$score_{ga} = X\left(GIoU, distance_{points}, ratio_{width}, ratio_{height}, distance_{app}\right) \tag{3-4}$$

其中 X 即为逻辑回归模型，$score_{ga}$ 得分越高，表示这一对检测框在几何和外观上越接近。而本文3.1.1中提及的检测框信息描述，到目前为止仅仅使用了其中的几何信息和外观信息。对于语义信息，具体使用方式是对两个检测框的类分数向量进行点乘，从而产生一个标量。当两个检测框不属于同一类时，其类分数点乘得到的结果趋近于 0，点乘结果越大，则越有可能是同一个类的实例。具体计算方式如式3-5。

$$score_{sem} = cc_i \cdot cc_j \tag{3-5}$$

其中，cc_i、cc_j 为两个检测框的类分数向量。至此，CIBPP 已经利用了3.1.1中提及的检测框描述符 $\{bbox, cc, app\}$ 中的全部信息，也得到了各个维度上的相似度得分。而两个检测框的综合相似度即为这些得分的乘积，因此可得 CIBPP 的检测框距离详细计算如式3-6。

$$\begin{aligned} distance &= \frac{1}{similarity} \\ &= \frac{1}{score_{ga} \times score_{sem}} \\ &= \frac{1}{X\left(GIoU, distance_{points}, ratio_{width}, ratio_{height}, distance_{app}\right)(cc_i \cdot cc_j)} \end{aligned} \tag{3-6}$$

其中 $score_{ga}$ 为逻辑回归的输出结果，其值落在 [0,1] 内，$score_{sem}$ 的值是两个检测框类分数向量的点积结果，因此其值也是大于 0 小于 1 的。故 *similarity* 的取值落在 [0,1] 内，取值越大表示两个检测框在各个维度的综合相似度越高，相应的距离也就越近。而当语义相似度极低时，类分数点乘的结果趋近于 0，上述分式的分母无限趋近于 0，则对应的 *distance* 为无穷大，这一结果也是合乎逻辑的。后文中建立检测框间的距离矩阵及跨帧连接检测框，均基于这一距离度量实现。

3.2 基于视频上下文信息的非实时后处理框架细节

在本文3.1中，我们详细介绍了进行后处理操作前所需的数据处理和模型融合工作。首先给出了每个检测框信息的形式化定义，即通过一组信息描述符来刻画每个检测框的各维度信息，接着解释了如何对基检测器的结构进行微调使之能够提供检测框外观特征，最后将每一对检测框的描述符转化成一组与两个检测框均相关的数据，这组数据被用于检测框距离计算。其中检测框距离定义在本文3.1.3中给出了计算式及具体解释。而在本节中，我们将基于上述内容介绍本文提出的后处理框架 CIBPP（Context Information Based Post-Processing，CIBPP）。

3.2.1 整体流程

图3-3及图3-4所示展示了整个基于视频上下文信息的非实时后处理框架工作流程。后处理框架从基检测器中获取数据，对于单个视频帧中的每一个检测框，后处理模块从基检测器中都会获取到一组描述符：几何信息、语义信息、外观信息，记为 $\{bbox, cc, app\}$。前两项由基检测器直接提供，而外观信息的获取方式已在本文3.1.2中给出说明。

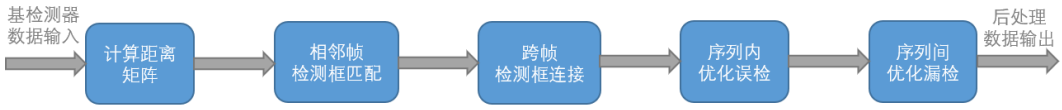


图 3-3: CIBPP 后处理简要流程

图3-4给出了后处理框架的详细处理流程，图中使用红色虚线框来展示每一模块的数据变化，使用彩色方块表示一个目标在不同帧之间的检测框，使用白色字体表示检测框的类分数变化。现结合该图，给出本章中的相关符号说明，见表3-1。

得到后处理模块输入数据后，对一组相邻帧中所有的检测框对，都使用本文3.1.3中介绍的距离函数计算其距离。因此每一组相邻视频帧，都会构建出一个距离矩阵。通过该矩阵，进行这一组相邻帧中的检测框匹配，在整个视频帧序列上循环这一操作，即可得到整个视频片段上所有相连的检测框序列。

获取某视频片段上所有的检测框序列集合之后，分两个阶段进行处理。首先，在每个检测框序列内，对检测框的类分数进行平均，用以解决部分帧中检

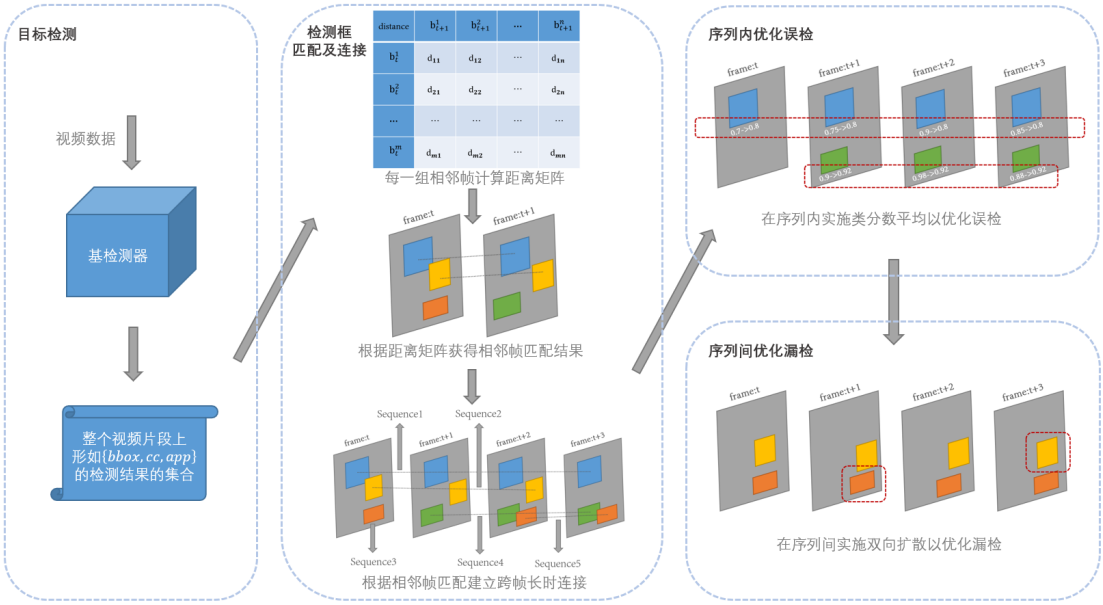


图 3-4: CIBPP 后处理详细流程

测框类分数低而被过滤掉的情况。其次，针对于不同的检测框序列，给定一个扩散比例阈值，根据当前序列长度，求得最大允许扩散长度，并根据这一扩散长度将当前检测序列的检测结果扩散到到附近几帧。通过这种机制来优化由于物体遮挡、特殊角度、视频失焦带来的同一个对象在视频片段内少数几帧检测结果丢失的情况。通过这两项操作，优化基检测器的初始检测结果中可能存在的误检和漏检问题，利用整个视频片段中的上下文信息，实现视频目标检测精度的提升。

3.2.2 关键模块

本小节对本文提出的基于视频上下文信息的非实时后处理框架 CIBPP 中的关键模块进行详细介绍。

（1）计算距离矩阵

对于任意一组相邻帧而言，每帧中都可能存在多个检测框，因而需要计算所有检测框对的距离，使用 3.1.3中定义的检测框距离函数构建距离矩阵。

给出一组时间上相邻的视频帧 t、$t+1$，二者分别对应一组初始检测框结果 $B_t = \{b_t^1, b_t^2, \ldots, b_t^m\}$、$B_{t+1} = \{b_{t+1}^1, b_{t+1}^2, \ldots, b_{t+1}^n\}$，其中视频帧 t 含有 m 个检测框，而视频帧 $t+1$ 含有 n 个检测框。对于视频帧 t 上任意一个检测框 b_t^i，需要计算

表 3-1: CIBPP 后处理框架涉及到的相关数学符号说明

数学符号	含义
T	视频片段对应帧数
t	视频片段上某一视频帧
B_t	视频帧 t 上的检测框集合
B_{ori}	整个视频片段上的基检测器原始检测结果集合
B_{final}	整个视频片段上经后处理操作后的最优检测结果集合
b_t^*	视频帧 t 上的某一检测框
$b.cc$	检测框 b 的语义信息，即类分数
D	视频帧 t 和视频帧 $t+1$ 的距离矩阵
$P_{t\,t+1}$	视频帧 t 和视频帧 $t+1$ 的检测框匹配结果
$Pairs$	视频片段上所有相邻帧匹配结果集合
Seq	视频片段上某一检测框序列
S_0	视频片段经基检测器检测后所有检测框序列集合
S_1	经过序列内优化误检处理后的检测框序列集合
S_2	经过序列间优化漏检处理后的检测框序列集合
$threshold_{iou}$	检测框重叠抑制阈值
$threshold_{diffuse}$	检测框序列扩散比例
$diffuse$	应用 $threshold_{diffuse}$ 得到的最大允许扩散长度

其与 B_{t+1} 中每一个检测框的距离。因而可以得到如图3-5所示的距离矩阵。

根据本文3.1.3中定义的 $distance = \frac{1}{similarity} = \frac{1}{score_{ga} \cdot score_{sem}}$，每一对检测框的距离 *similarity* 落在 [0,1] 内。*similarity* 越大则两个检测框越接近，其 *distance* 就越小。对于视频帧 t 上的某个检测框 b_t^i 而言, 该检测框对应的距离矩阵的一行中存在最小值 d_{ij}，则 b_{t+1}^j 是视频帧 $t+1$ 中与 b_t^i 距离最近、最相似的检测框。但这一结论反向并不成立，即对于视频帧 $t+1$ 上的检测框 b_{t+1}^j 而言，视频帧 t 中与之距离最近、最相似的检测框未必是 b_t^i。同样的，b_{t+1}^j 对应的距离矩阵的一列中也存在最小值 d_{kj}，因此视频帧 t 中与 b_{t+1}^j 距离最近、最相似的检测框是 b_t^k，k 与 i 有概率相等但并非恒等（设想视频帧 t 中存在两个高度重叠的对应同一实例的检测框，而视频帧 $t+1$ 中该实例仅有一个检测框这种情况）。

在处理一段视频数据时，需要对每一组相邻帧都构建一个距离矩阵，即视

distance	b_{t+1}^{1}	b_{t+1}^{2}	…	b_{t+1}^{n}
b_{t}^{1}	d_{11}	d_{12}	…	d_{1n}
b_{t}^{2}	d_{21}	d_{22}	…	d_{2n}
…	…	…	…	…
b_{t}^{m}	d_{m1}	d_{m2}	…	d_{mn}

图 3-5: 一组相邻帧的距离矩阵

频序列由 T 个视频帧组成，则需要计算 $T-1$ 个距离矩阵，后续进行检测框的连接和抑制操作均基于上述距离矩阵。

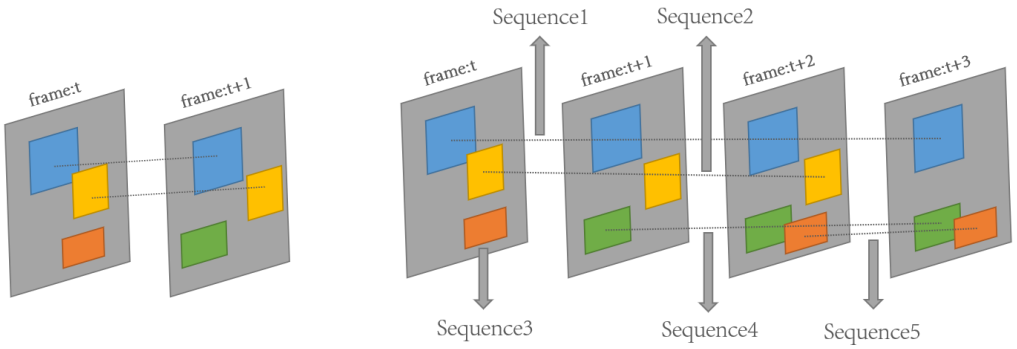
（2）相邻帧检测框匹配及跨帧检测框连接

对于每一个视频片段而言，我们需要获得这个视频片段上所有的检测框序列集合，才可以进行后续优化误检和漏检的操作。3.2.2（1）已经详细给出了建立一组相邻帧对应的距离矩阵的方式。而根据这一距离矩阵，得到整个视频片段上的检测框序列集合，则需要分两个步骤处理：首先在两帧之间通过距离矩阵建立检测框的对应关系，并在找到一对匹配的检测框之后，抑制这两帧中其他重叠的检测框；其次是跨帧维护长时的检测框序列，每在一对相邻帧中找到一对匹配的检测框，都需要确定新匹配到的检测框与现存序列的关系，从而将成功匹配到的检测框加入到正确的现存序列中。

我们使用“匹配”来描述一组相邻帧之间检测框的处理，使用“连接”来描述多帧长时的检测框处理，如图3-6所示。图中使用不同颜色的方块代表不同类别的检测框或者同类但不同实例的检测框，这意味着不同颜色的方块所代表的检测框计算出的 *distance* 取值趋近于无穷大，则成功建立检测框连接的概率很低。

t 和 $t+1$ 为两个相邻的视频帧，在这一组视频帧上的匹配的结果如子图3-6(a)所示。其中视频帧 t 中橙色检测框和视频帧 $t+1$ 中绿色检测框不会与任何检测框匹配，因为另一帧中的检测框均为非同类或者非同一实例的检测框，前者会导致语义相似度极低，后者会导致外观相似度低，都会得到一个接

近于无穷大的 *distance* 取值，因此这两个检测框只能单独存在。



(a) 相邻帧检测框匹配　　(b) 跨帧检测框连接

图 3-6: 相邻帧检测框匹配及跨帧检测框连接

参照 3.2.2（1），相邻的视频帧 t、$t+1$ 对应两组初始检测框结果记为 $B_t = \{b_t^1, b_t^2, \ldots, b_t^m\}$、$B_{t+1} = \{b_{t+1}^1, b_{t+1}^2, \ldots, b_{t+1}^n\}$。为了得到子图 3-6(a)中的匹配结果，CIBPP 中具体的实现方式是，根据 3.2.2（1）中求得的距离矩阵，选取整个矩阵中的最小值 d_{ij}，这样就可以保证两个检测框互相都是和对方距离最近也即最相似的。显而易见的，在每次进行选择时，都需要保证不会选择无穷大的 *distance* 值，距离为无穷大时没有连接意义。此时视频帧 t 中的检测框 b_t^i 和视频帧 $t+1$ 中的检测框 b_{t+1}^j 会优先被匹配。那么对于 b_t^i，首先删去视频帧 t 中与 b_t^i 的 IoU 超过阈值 $threshold_{iou}$ 的检测框信息及其在距离矩阵中对应的行，随后删去距离矩阵中 b_t^i 本身对应的行。第一步是在匹配到一对检测框之后，去掉同一帧内重合度过高的冗余检测框，第二步是为了保证已经匹配的检测框不再参与其他检测框对的匹配。同样的，对于 b_{t+1}^j，也需要删去视频帧 $t+1$ 中与 b_{t+1}^j 的 IoU 超过 $threshold_{iou}$ 的检测框信息及其在距离矩阵中对应的列，最后删去 b_{t+1}^j 在距离矩阵中对应的列。

前文描述了如何通过距离矩阵得到一组相邻帧间的检测框匹配关系，而下一阶段即为根据匹配关系实现子图 3-6(b)中的跨帧检测框连接。初始时整个视频片段对应的检测框序列集合为空，视频序列由 T 个视频帧组成，则需要计算 $T-1$ 个距离矩阵。因此从第二帧开始，每次都对当前帧及前一帧计算距离矩阵并基于这一距离矩阵进行检测框的匹配及抑制操作。对于每一对匹配成功的检测框，若能找到已存在的检测框序列与当前检测框信息吻合，则认为当前匹配的是该序列的一个延续，将当前帧的检测框合并到该序列中。若未找到与当前

检测框信息吻合的现存序列，则认为是新出现的一个序列，将其视为长度为 1 的序列，加入序列集合中，并且下一帧中有可能存在与之匹配的检测框。具体的实现细节在3.2.3中详述。

经过相邻帧检测框匹配及跨帧检测框连接这两个步骤后，可以得到整个视频片段上的检测框序列集合，设某个视频片段上共存在 c 个检测框序列，由于后续该集合还需要经过优化误检和优化漏检处理，因此记为 S_0，$S_0 = \{Seq_0^1, Seq_0^2, \ldots, Seq_0^c\}$，也就是检测框连接及抑制模块的输出。

（3）序列内优化误检

序列内优化误检模块的输入是检测框连接及抑制模块中生成的整个视频片段上初始检测框序列集合 $S_0 = \{Seq_0^1, Seq_0^2, \ldots, Seq_0^c\}$。此模块处理目标是对每个序列内存在的一些正确但是类分数低的检测结果，通过和该序列内其他检测结果的类分数进行平均，来实现对低分数的适当补偿。从而在每个序列内保证检测结果相对稳定，进而增强整个视频片段上检测结果的时间一致性。连续帧的检测结果中出现少数误检的情况如图3-7所示，而优化之后的结果如图3-8。

图 3-7: 连续帧中少数误检情况

图 3-8: 通过序列内分数平均优化误检的结果

这一操作的理论基础是一个视频片段内，时间上相近的帧在内容上也是更相似的，前文3.2.2（2）得到的序列集合，本质上是将每个序列都看成是同一个物体在不同帧上的检测结果集合。因此，同一物体在不同帧上的检测结果不

应该发生分类突变，那么任意一个原始检测框序列 Seq 中，即使存在几个由于低置信度导致的错误检测结果，在经过整个序列内类分数平均之后，也可以被纠正。CIBPP 正是通过这种机制来优化基检测器中由于缺乏视频上下文信息而存在的部分误检情况。

对于 S_0 中的每一个 Seq_0^i，都在序列内对所有检测框的类分数进行平均，得到 Seq_1^i，S_0 中全部检测框经过平均处理后得到 S_1，$S_1 = \{Seq_1^1, Seq_1^2, \ldots, Seq_1^c\}$，即为序列间优化误检模块的输出。

（4）序列间优化漏检

本文3.2.2（3）中介绍了 CIBPP 通过检测框序列内分数平均来优化误检的过程，但除了误检之外，基检测器的初始检测结果中还会存在漏检的问题。在某些特殊角度、画面模糊等情况下，对同一个对象的检测结果可能会在某些帧中丢失，而经过数帧之后又可以正确检测，如图3-9所示。当检测结果出现上述漏检情况时，3.2.2（2）中的检测框连接及抑制模块会将一个对象的检测结果对应到两个或多个检测框序列内，这是由于在丢失检测结果的几帧上进行检测框匹配时无法建立连接，因此会被分割成多个序列。

对于整个视频片段上的检测框序列集合 S_1（此处的检测框序列集合指的是经3.2.2（3）处理后的集合），其中的两个或多个序列，可能是对应同一个对象的检测结果，理想情况下需要对这些集合进行合并操作，并针对丢失检测结果的几个间隔帧进行检测结果补充。因此，CIBPP 提出一种双向扩散机制。设定一个阈值 $threshold_{diffuse}$，该阈值为一个比例值，表示当前序列的最大允许扩散比例。假设经3.2.2（3）中介绍的模块处理后的某个检测框序列为 Seq_1^i，此序列长度为 len，则该序列对应的最大允许扩散长度为 $threshold_{diffuse} \times len$，那么对应到双向扩散，向前和向后两个方向分别可以扩散 $0.5 \times threshold_{diffuse} \times len$ 帧。

检测框结果扩散本质上就是对于当前序列内的所有的检测框，取所有检测框中心点坐标均值、类分数均值，得到一个新的检测框，将这个检测框扩散到当前序列的前后几帧。而对于图3-9所示的第三帧目标未被检测到的情况，前两帧中的序列会通过向后扩散赋予第三帧一个检测框，同样的，后两帧向前扩散时也会赋予第三帧一个检测框，这两个检测框的几何信息、语义信息、外观信息分别取自于各自对应的序列，因此不完全相同。此时第三帧上就会存在两个重叠度很高的并且对应于同一个实例的检测框，所以在经过序列间扩散之

后，对于所有接收到新检测框的帧，都需要进行检测框抑制操作，这一步骤与本文3.2.2（2）使用的抑制操作类似，目的都是移除同一个视频帧内冗余的检测框。经过序列间扩散及抑制后，得到的检测结果如图3-10所示。

图 3-9: 连续帧中少数漏检情况

图 3-10: 通过序列间双向扩散优化漏检的结果

本文3.2.2（3）中得到的检测框序列集合 S_1，其中所有的检测框序列都需要经过上述序列间双向扩散优化漏检的操作，而后形成一个新的集合 S_2。该集合内对应的即为整个 CIBPP 后处理框架优化后的检测框序列集合。基于集合 S_2，提取出每一个视频帧上的检测结果，也即完成了全部的基检测器初始检测结果后处理工作。

3.2.3 算法流程

前文介绍了基于视频上下文信息的非实时后处理框架 CIBPP 的整体结构，以及其中的几个关键模块。本节主要根据实际数据流来分析整个框架的处理机制，并对部分具体模块的实现给出算法说明。

对于一个视频片段而言，假设该视频片段对应 T 个视频帧，经过基检测器处理之后，每个视频帧 t 对应一组检测框集合 $B_t = \{b_t^1, b_t^2, \ldots, b_t^m\}$，其中 $b_t^i = \{bbox, cc, app\}$，即使用一组形式化的信息来描述一个检测框。进而整个视频片段上的检测结果集合记为 $B_{ori} = \{B_1, B_2, \ldots, B_T\}$。

视频片段经过基检测器处理后输入到后处理框架 CIBPP 中的数据即为整个视频片段上的检测结果集合 B_{ori}，从第二个视频帧开始，直到最后一个视频帧，共存在 $T-1$ 组相邻帧，每一组相邻帧都根据本文3.2.2（1）中描述的方

式进行距离矩阵的计算。随后在距离矩阵的基础上逐帧进行检测框匹配，遍历完所有相邻视频帧的检测框匹配结果后，即可得到该视频片段上所有的检测框序列集合 S_0。对该集合进行串行的两步处理，首先对 S_0 中的所有序列都进行序列内分数平均，得到新的检测框序列结合 S_1；再对 S_1 中的每个序列进行双向扩散及抑制，得到最终的检测框序列集合 S_2。最后根据 S_2，还原其检测框信息得到整个视频片段上的最优检测结果集合 B_{final}，至此整个后处理阶段结束。下文中将对这一流程中每个模块的实现进行详细分析。

（1）计算距离矩阵

每有一组相邻帧出现时，即需执行一次距离矩阵的计算，因此本模块输入数据是视频帧 t 中的所有检测框集合 $B_t = \{b_t^1, b_t^2, \ldots, b_t^m\}$，及视频帧 $t+1$ 中所有的检测框集合 $B_{t+1} = \{b_{t+1}^1, b_{t+1}^2, \ldots, b_{t+1}^n\}$，算法输出是视频帧 t 及 $t+1$ 对应的距离矩阵 D。具体流程如算法 3.1所示。

算法 3.1 计算距离矩阵

Require: 视频帧 t 中的所有检测框集合 $B_t = \{b_t^1, b_t^2, \ldots, b_t^m\}$，视频帧 $t+1$ 中所有的检测框集合 $B_{t+1} = \{b_{t+1}^1, b_{t+1}^2, \ldots, b_{t+1}^n\}$

Ensure: 视频帧 t 及 $t+1$ 对应的距离矩阵 D

1: 初始化一个 m 行 n 列的矩阵，元素初始均为无穷大
2: **for** $i = 0; i < m; i++$ **do**
3: **for** $j = 0; j < n; j++$ **do**
4: 使用式 3-6计算 b_t^i 和 b_{t+1}^j 的检测框距离并填入 D_{ij}
5: **end for**
6: **end for**

通过上述算法，可以得到任意一组相邻帧中每一对检测框之间的距离，为后序检测框连接提供合理有效的度量信息。

（2）相邻帧检测框匹配

在 3.2.2（2）中提到，CIBPP 将检测框序列的建立划分成了两个步骤：第一步是在每一组相邻帧间建立检测框匹配关系，该步骤在每次计算得到距离矩阵之后就会执行。而第二步是根据前一步骤的匹配结果，处理当前帧中成功匹配的检测框与现存检测框序列的关系，从而建立长时的跨帧检测框连接。第一步建立匹配关系时，需要的输入数据为相邻帧的检测框信息，以及这一组相邻帧对应的距离矩阵。而其输出则是这一组相邻帧的匹配结果集合，记为 $P_{t\,t+1}$，$P_{t\,t+1} = \{\{b_t^i, b_{t+1}^j\}, \{b_t^p, b_{t+1}^q\}, \ldots, \{b_t^g\}, \{b_{t+1}^h\}\}$，其中 $\{b_t^i, b_{t+1}^j\}$ 表示检测框 b_t^i 和检测框

b_{t+1}^{j} 建立了匹配关系，而结果中可能含有未成功匹配的独立存在的检测框，比如 b_t^g、b_{t+1}^h。具体过程如算法 3.2所示。

算法 3.2 相邻帧检测框匹配

Require: 视频帧 t 中的所有检测框集合 $B_t = \{b_t^1, b_t^2, \ldots, b_t^m\}$，视频帧 $t+1$ 中所有的检测框集合 $B_{t+1} = \{b_{t+1}^1, b_{t+1}^2, \ldots, b_{t+1}^n\}$，视频帧 t 及 $t+1$ 对应的距离矩阵 D，检测框抑制阈值 $threshold_{iou}$

Ensure: 视频帧 t 及 $t+1$ 的检测框匹配结果 $P_{t\,t+1}$

```
初始化 P_{t t+1}=B_t ∪ B_{t+1}
while D 中元素最小值不为无穷大 do
    找到 D 中元素最小值 D_{ij}
    从 P_{t t+1} 中删去 b_t^i、b_{t+1}^j
    将 {b_t^i, b_{t+1}^j} 加入 P_{t t+1}
    将 D 的第 i 行所有元素置为无穷大
    将 D 的第 j 列所有元素置为无穷大
    for p = 0; p < m&&p! = i; p + + do
        if b_t^i 与 b_t^p 的 IoU 超过 threshold_iou then
            将 D 的第 p 行所有元素置为无穷大
            从 P_{t t+1} 中删去 b_t^p
        end if
    end for
    for q = 0; q < n&&q! = j; q + + do
        if b_{t+1}^j 与 b_{t+1}^q 的 IoU 超过 threshold_iou then
            将 D 的第 q 列所有元素置为无穷大
            从 P_{t t+1} 中删去 b_{t+1}^q
        end if
    end for
end while
```

将 $P_{t\,t+1}$ 初始化为 B_t 和 B_{t+1} 的并集，在每次成功匹配一对检测框时，删去 $P_{t\,t+1}$ 中这两个检测框单独存在的信息，转而将成对的信息添加进去。从而确保 $P_{t\,t+1}$ 最后包含所有匹配成功的检测框，以及独立存在未被匹配的检测框。这些独立的检测框与其他检测框的距离始终为无穷大，因此其独立存在的信息在 $P_{t\,t+1}$ 内始终不会被删除。

（3）跨帧检测框连接

在完成每一组相邻帧之间的检测框匹配后，需要根据匹配结果来建立长时的跨帧检测框连接。这一部分的输入数据即为视频片段上所有相邻帧的匹配结果，在此基础上建立的连接方为“跨帧的”、“长时的”，而算法得到的则是原始视频片段上所有的检测框序列集合 S_0，初始时 S_0 为空集。对于每一组相

邻帧，都需要判断当前匹配结果是否属于前序已经存在的某个序列，若存在，则认为当前匹配的检测框和该现存序列属于同一个对象在较短时间间隔内的检测结果，进而将新匹配到的检测框加入该现存序列中。若当前帧上某个检测框未能找到信息吻合的现存序列，则将其视为一个新序列加入 S_0 中。遍历所有的相邻视频帧匹配结果，即可得到该视频片段对应的原始检测框序列集合 S_0。

算法 3.3 跨帧检测框连接

Require: 整个视频片段上所有相邻帧的检测框匹配结果集合 $Pairs = \{P_{12}, P_{23}, \ldots, P_{T-1T}\}$
Ensure: 整个视频片段上所有检测框序列集合 S_0

```
初始化 S0 为空集
len1 = Pairs 中元素个数
for i = 1; i <= len1; i++ do
    len2 = P_{i i+1} 中元素个数
    for j = 1; i <= len2; j++ do
        if P^j_{i i+1} 是形如 {b*_t} 的单个检测框信息 && j==1 then
            S0 = S0 ∪ {P^j_{i i+1}}
        end if
        if P^j_{i i+1} 是形如 {b*_{t+1}} 的单个检测框信息 then
            S0 = S0 ∪ {P^j_{i i+1}}
        end if
        if P^j_{i i+1} 是形如 {b*_t, b*_{t+1}} 的成对检测框信息 then
            len3 = S0 中元素个数
            for k = 1; k <= len3; k++ do
                if 序列 S^k_0 的最末尾元素与 b*_t 相同 then
                    将 b*_{t+1} 加入序列 S^k_0 最末尾
                end if
            end for
        end if
    end for
end for
```

跨帧检测框连接模块的输入是所有相邻帧匹配结果的集合 $Pairs = \{P_{1\,2}, P_{2\,3}, \ldots, P_{T-1\,T}\}$，其中 $P_{t\,t+1} = \{\{b_t^i, b_{t+1}^j\}, \{b_t^p, b_{t+1}^q\}, \ldots, \{b_t^s\}, \{b_{t+1}^h\}\}$，表示视频帧 t 及视频帧 $t+1$ 的检测框匹配结果，视频片段共 T 帧，则该集合共有 $T-1$ 个元素。遍历元素中的每一个元素，即处理每一个 $P_{t\,t+1}$ 时都更新一次 S_0。具体算法流程如算法 3.3所示。

为了更好的解释上述算法的工作流程，对于一组相邻帧，我们将时间上更早的帧称为左帧，另一帧称为右帧。因此每一组相邻帧的匹配结果集合，其中的元素存在三种情况：来自左帧的单独检测框、来自右帧的单独检测框、左帧

和右帧的一对匹配检测框。对于左帧中单独存在的检测框，如果该检测框属于整个视频片段第一帧的检测结果，则将该单独存在的检测框加入 S_0。其后的每一组匹配结果，都只将右帧中单独存在的检测框加入 S_0。而对于成对存在的检测框，根据匹配规则，该成对的结果一定是某个已存序列的延续，因此将新匹配到的右帧中的检测框加入该序列。按照上述步骤遍历整个 *Pairs* 集合，即可得到视频片段上所有初始检测结果序列集合 S_0。

（4）序列内优化误检

得到某个视频片段上所有检测框序列后，下一步操作需要对每个序列内的检测框进行类分数平均处理。此模块的输入数据即为包含了全部检测框信息的序列集合 S_0，本模块对每个序列中各个检测框的语义信息进行修改，并不影响序列的元素组成。算法返回结果则是经过序列内优化误检处理的新的检测框序列集合，记为 S_1。具体算法流程如算法3.3所示。

算法 3.4 序列内类分数平均优化误检

Require: 整个视频片段上初始检测框序列集合 S_0
Ensure: 整个视频片段上经类分数平均优化后的检测框序列集合 S_1
1: $len1 = S_0$ 中元素个数
2: **for** $i = 1; i <= len1; i++$ **do**
3: $sum \in \mathbb{R}^C$，初始时 *sum* 各维度均为 0
4: **for** 遍历 S_0^i 中所有的检测框 b **do**
5: $sum = sum + b.cc$
6: **end for**
7: $avg = \frac{sum}{len1}$,$avg \in \mathbb{R}^C$
8: **for** 遍历 S_0^i 中所有的检测框 b **do**
9: $b.cc = avg$
10: **end for**
11: **end for**

其中，*b.cc* 指的是本文3.1.1中定义的每个检测框的语义信息，即类分数向量。*sum* 和 *avg* 为局部变量，用于对所有检测框类分数进行求和及取平均操作。对于每个检测框序列，将其中所有检测框的类分数向量累计求和，再根据序列长度，求得一个平均类分数向量，进而用这个平均类分数向量替换原检测框的语义信息。实际代码执行过程中是对 S_0 中的数据做原地修改，但此处为了区分各模块的操作，将序列内类分数平均之后的检测框序列集合记为 S_1。

（5）序列间优化漏检

对于本文3.2.3（3）描述的建立跨帧检测框连接的过程，每一个检测框序列都对应一个序列起始帧号以及序列长度。每产生一个新序列时，赋予该序列起始帧号，有新检测框加入该序列时，相应的更新该序列长度。在序列间进行双向扩散时，会涉及到序列起始帧及长度的变化，因此需要维护序列信息来记录检测框和帧的对应关系。

对于每一个序列，都需要根据序列长度进行向前和向后的检测结果扩散。扩散的目的是为了解决同一个对象的检测结果在少数几帧中丢失，最终被分割成两个或多个序列的情况。通过扩散的形式对中间帧进行相应填充，以此解决漏检问题。但检测结果扩散除了会填充丢失结果的帧，也可能会对一些包含检测结果的帧再次赋予新的检测框，导致结果冗余。因此对每一个被赋予新检测框的视频帧，需要检查是否存在由于检测结果扩散导致的检测框冗余，如存在，则将冗余检测框删除。具体算法流程如算法3.5所示。

算法 3.5 序列间双向扩散优化漏检

Require: 整个视频片段上经类分数平均优化后的检测框序列集合 S_1，检测结果扩散比例 $threshold_{diffuse}$

Ensure: 整个视频片段上经序列间双向扩散后的检测框序列集合 S_2

```
len1 = S1 中元素个数
for i = 1; i <= len1; i++ do
    len2 = S1^i 中元素个数
    diffuse = len2 × threshold_diffuse
    for j = 1; j <= ⌊diffuse/2⌋; j++ do
        在 S1^i 头部插入 S1^i[0]
        S1^i 起始帧号- -
        S1^i 序列长度 ++
    end for
    for j = 1; j <= ⌊diffuse/2⌋; j++ do
        在 S1^i 尾部插入 S1^i[len2-1]
        S1^i 序列长度 ++
    end for
end for
```

上述算法根据每个序列长度，求得当前序列的扩散距离 $diffuse$，则向前和向后的扩散距离分别是 $\lfloor\frac{diffuse}{2}\rfloor$。扩散时要注意边界约束，即不能扩散到第一帧及最后一帧以外的地方。前序的类分数平均操作保证了同一序列内所有检测框的类分数都是一致的，因此在扩散时，仅需选取序列内第一个及最后一个检

测框作为扩散样本。整个扩散过程实际上是对 S_1 的内部修改，但这里为了区分模块输入输出，将处理后的检测框序列集合记为 S_2。从 S_2 中可以还原得到整个视频片段上的检测结果集合 B_{temp}，最后对 B_{temp} 进行去除冗余检测框的操作，这一操作过程与算法3.2所使用的抑制过程类似，因此不再赘述。最后即可得到整个视频片段上的最终检测结果集合 B_{final}，后处理部分结束。

3.3 实验与分析

前文3.1及3.2已给出了后处理框架 CIBPP 的数据准备、整体设计及各模块的详细算法流程。本节主要介绍 CIBPP 的实验设置及实验结果。首先明确所有实验的软硬件环境，随后给出所用数据集、评价指标、损失函数等信息，并对训练数据的制备也进行了阐述。其次对于基检测器的选取、结构设置及部分参数设置也给出了相应分析。最后通过在不同模型上进行对比实验以验证 CIBPP 的有效性，并通过和其他后处理方案对比验证 CIBPP 的性能。

3.3.1 实验设置

（1）实验环境

本文提出的 CIBPP 后处理框架所涉及的实验均在以下环境中进行。

- 系统：Ubuntu 18.04.5 LTS
- GPU：GeForce RTX 2080 Ti
- CPU：Intel(R) Xeon(R) CPU E5-2678 v3 @ 2.50GHz
- 编程语言：Python3.6

（2）数据集及评价指标

2015 年 ImageNet 大规模视觉识别挑战赛将视频目标检测列为了一项新的任务，而该挑战赛提供的 ImageNet VID 数据集则成为了后续视频目标检测领域最常用的数据集，也是迄今为止用于视频目标检测任务的规模最大的数据集。为了更好的进行结果对比，参考其他视频检测模型及后处理算法，本文同样选择使用 ImageNet VID 数据集。与 ImageNet DET 数据集相比，VID 数据集不是独立图像组成的数据集合，而是将同一视频中的帧归为一组，该视频每一帧中所有目标都有对应的类别和矩形边框标注。VID 数据集对应 30 个基本类别，含

有 3862 个视频用于训练、555 个视频用于验证、937 个视频用于测试。数据分布如表 3-2所示。

表 3-2: ImageNet VID 数据集数据组成

	训练集	验证集	测试集
视频数量	3862	555	937
图像数量	1122397	176126	315176

本文后处理框架的实验使用本文 2.1.3中介绍的 mAP 指标作为评价标准，这也是常用于 ImageNet VID 数据集上的指标之一。

（3）三元组数据及损失函数

本文 3.1.2介绍的获取外观信息的嵌入层及本文 3.1.3 介绍的用于度量检测框距离的逻辑回归结构均使用三元组数据进行训练。而三元组（triplet）及其对应的损失函数（triplet loss）最早在 FaceNet 中被提出，这一思路受到了 Weinberger 等人关于近邻分类算法工作的启发 [101]。FaceNet 采用三元组损失函数来衡量训练过程中的样本距离。在随机梯度下降过程中，最大化同一对象的样本距离，最小化不同对象的样本距离，从而训练得到的网络能够获取更高质量的人脸特征信息。FaceNet 中定义的三元组损失函数及转化如式 3-7所示。

$$\begin{aligned} &\left\|x_i^a - x_i^p\right\|_2^2 + \alpha < \left\|x_i^a - x_i^n\right\|_2^2, \forall \left(x_i^a, x_i^p, x_i^n\right) \in \mathcal{T} \\ &L = \sum_i^N \left[\left\|f\left(x_i^a\right) - f\left(x_i^p\right)\right\|_2^2 - \left\|f\left(x_i^a\right) - f\left(x_i^n\right)\right\|_2^2 + \alpha\right]_+ \end{aligned} \tag{3-7}$$

式 3-7中，x_i^a 为每次选择的基样本（anchor），x_i^p 是相对于 x_i^a 的正样本（positive），而 x_i^n 是相对于 x_i^a 的负样本（negative），$f(*)$ 为样本对应的特征嵌入，α 是正负样本的边界，$\mathcal{T}$ 是训练集中所有三元组的集合。先选定基样本和正样本组成的两元组，接着从非同一实例的样本中找到一个与基样本距离比“基样本—正样本距离”大 α 的样本。进而损失函数转化为最小化 L，最终得到的损失函数也即式 3-1。该函数目标是使同一实例的样本距离最小化，而不同类或者同类不同实例的样本距离最大化，可视化形式如图 3-11所示 [99]。

针对于视频目标检测后处理任务，需要在 ImageNet VID 数据集上构建符合上述要求的三元组训练集和验证集。回到 InameNet VID 数据集所包含的信息，

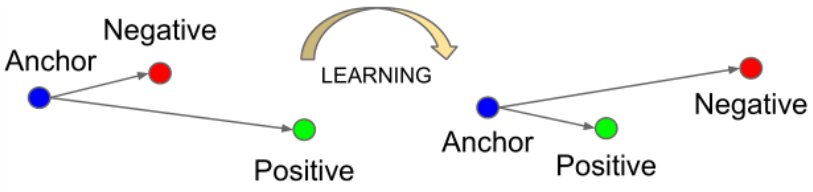


图 3-11: 三元组损失函数可视化

每个视频帧的检测结果注释文件中都包含 trackid 这项数据，用于区分画面内不同物体。因此，对于每个视频片段，首先随机采样一个 trackid，即确定一个目标实例，随后在检测到该实例的帧中随机取一个作为基样本，而正样本从基样本前后 25 帧的范围内随机采样获得。负样本则从另一个目标实例（即另一个不相等的 trackid）对应的视频帧中获得，亦或从另一个视频片段中获得。负样本可以与基样本不同类，也可以是同类但不同实例。通过上述方式，即可得到用于训练和验证的三元组数据。

（4）基检测器及参数设置

原则上本文提出的后处理框架 CIBPP 可以应用于任何基于卷积神经网络的目标检测模型，基检测模型提供每一个检测框的坐标和类分数，而卷积神经网络保证了图像目标检测模型可以得到特征图用于进行检测框特征提取。为了更好的和其他后处理方案进行性能上的对比，我们选择 YOLOv3 作为主要的基检测模型。YOLOv3 属于一阶段目标检测模型，在性能和速度上有较好的均衡，该系列模型的演变及对比已在本文2.2.3中给出详细分析。

在视频目标检测过程中，对于视频流中的每一帧图像，都统一将其缩放为 512×512 像素大小。通过本文3.1.2中提到的 RoI 池化层提取每个检测框对应的特征后，将提取出的子特征图缩放至 5×5×256 的大小。用于检测框特征嵌入的全连接层有 256 个神经元，输出为 256 维的 L2 归一化向量表示。为了适当增加输入进后处理阶段的检测结果数量，我们将 YOLOv3 中的检测结果置信度阈值下调至 0.3，这样可以保证尽可能少的遗漏合理的检测结果。其他模块的细节设置如下所述：

- **计算距离矩阵**：尝试匹配一对检测框时，若这两个检测框各自的最大类分数仍小于 0.2，则这两个检测框类分数向量点积值会较小，即语义相似度极低。此时认为这两个检测框均为误检，将二者距间离置为无穷大。
- **相邻帧检测框匹配**：产生一对匹配的检测框时，需要抑制同一帧内其他与

之高度重叠的检测框，则需要设置检测框重叠阈值 $threshold_{iou}$，本文将该阈值设置为 0.6。

- **跨帧检测框连接：**此模块的输入数据是整个视频片段上的相邻帧匹配结果，因而存在很多前期出现但后期消失在视频内的对象。本文规定超过 60 帧仍未匹配到下一个检测框的序列，将其导出，不再尝试该序列和新检测框的匹配，这样做是为了节约计算资源。
- **序列内优化误检：**对于任意一个检测结果序列，通过类分数平均之后，选择整个序列上所有检测框累计得分最高的类标签作为整个序列的类标签。
- **序列间优化漏检：**每个序列应用扩散比例阈值 $threshold_{diffuse}$ 即可得到每个序列的扩散长度，实验中 $threshold_{diffuse}$=0.1，同时增加了最大扩散长度的限制。假设一个检测框序列的长度为 len，则其计算出的扩散距离为 $threshold_{diffuse} \times len$，最后应用的扩散距离为 $min(threshold_{diffuse} \times len, 60)$，鉴于 ImageNet VID 数据集中大部分视频的 fps 都在 30 上下，因此上述限制将允许丢失检测结果的时间控制在 2s 内。在经过序列间双向扩散之后，序列长度仍小于 10 的则认为是误检序列，将其从集合中剔除。

3.3.2 有效性实验及结果分析

为了证明本文提出的基于视频上下文信息的非实时后处理框架 CIBPP 的有效性，我们使用 ImageNet VID 数据集进行了一些实验验证。除了本文3.3.1（4）中提到的使用 YOLOv3 作为主要的基检测模型，为了保证结果准确，我们同样使用其他视频目标检测模型进行了对比实验。这里使用到的视频目标检测模型主要为 FGFA、SELSA 及 MEGA，关于这几种视频目标检测模型的介绍已在本文2.3.1及2.3.4中给出。从视频目标检测模型中进行检测框特征提取和从静态图像目标检测模型中进行特征提取方式不同，而本文提出的 CIBPP 主要运用于静态图像目标检测器上，因此在与 SELSA、FGFA 及 MEGA 进行对比时，没有使用外观特征信息。各个模型的实验环境与本文3.3.1（1）给出的实验环境保持一致，得到的实验结果如表3-3所示。

其中，ProcessingTime 为每张图片的总处理时长，以毫秒（ms）为单位。上述结果均基于 ImageNet VID 的验证集。将各组模型增加后处理前后的 mAP 对比转化成更为直观的图表形式，如图3-12所示。从表3-3及图3-12中可以看出，虽然各个基检测器对每帧图像的处理时长相差较大，但后处理框架 CIBPP 对处理时长的增加幅度是基本稳定的。并且，上述几个模型在增加了 CIBPP 作

表 3-3: CIBPP 在不同基检测器上的实验结果

Method	Base Detector	Backbone	mAP	ProcessingTime
YOLOv3	YOLOv3	Darknet-53	70.21%	18.57
YOLOv3+CIBPP	YOLOv3	Darknet-53	79.36%	24.33
FGFA	R-FCN	ResNet-101	75.93%	91.20
FGFA+CIBPP	R-FCN	ResNet-101	82.17%	98.34
SELSA	Faster R-CNN	ResNet-101	82.01%	133.16
SELSA+CIBPP	Faster R-CNN	ResNet-101	85.69%	143.47
MEGA	Faster R-CNN	ResNeXt101	83.94%	164.58
MEGA+CIBPP	Faster R-CNN	ResNeXt101	86.21%	175.32

为后处理模块后，其检测精度都得到了一定提升，也说明了 CIBPP 的有效性。相应的，基检测模型的 mAP 越低，增加后处理之后 mAP 的提升幅度越大。但基检测器本身精度较高时，CIBPP 对其提升较小。需要说明的是，后处理算法本身属于一种不额外引入新的模块和任务，但能够补充上下文信息的视频目标检测优化策略，因此这一定位决定了其无法取得巨大的检测精度提升。而从时间成本和策略复杂程度的角度考量，后处理方案仍是一种性价比很高的解决方案。

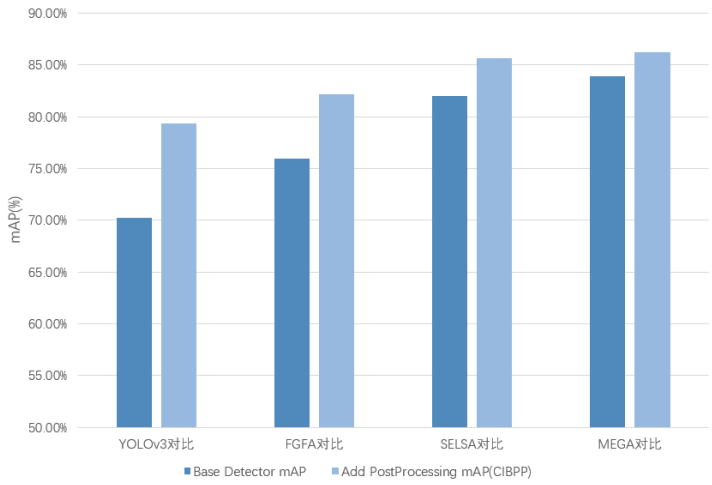


图 3-12: CIBPP 在各模型上的 mAP 结果

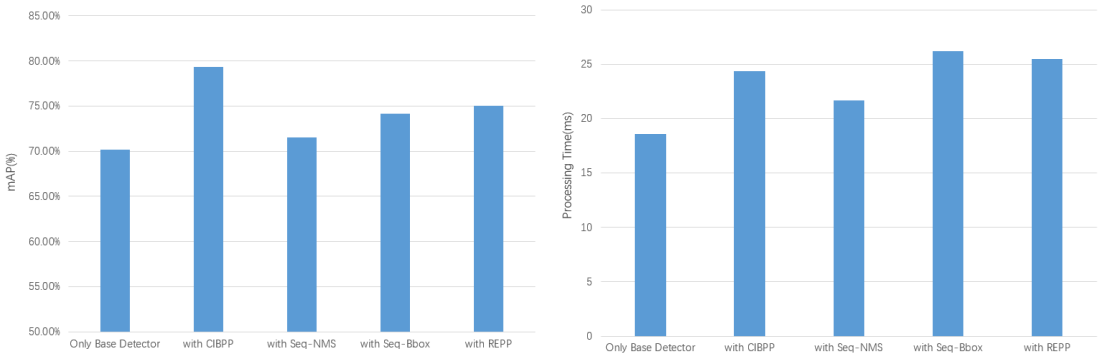
3.3.3 对比实验及结果分析

在本文3.3.2中，我们将 CIBPP 应用于不同的基检测器上来验证其有效性，除此之外，对于本文2.4中介绍的几种后处理策略，也需要逐个与 CIBPP 进行对比，以观测这些后处理方案的优劣。具体来说本小节主要将 CIBPP 与下述几种后处理方案进行了对比：Seq-NMS、Seq-Bbox、REPP。T-CNN 由于其基检测器的设置与其他后处理方案不同，因此为了尽可能保证实验条件一致，没有与 T-CNN 的结果进行对比。实验设置仍与本文3.3.1（1）给出的实验环境保持一致，得到的实验结果如表3-4所示。

表 3-4: CIBPP 与其他后处理方案的结果对比

Base Detector	Post Processing Method	mAP	ProcessingTime
YOLOv3	-	70.21%	18.57
YOLOv3	CIBPP	79.36%	24.33(18.57+5.76)
YOLOv3	Seq-NMS	71.51%	21.68(18.57+3.11)
YOLOv3	Seq-Bbox	74.19%	26.19(18.57+7.62)
YOLOv3	REPP	75.06%	25.46(18.57+6.89)

将表3-4所示结果转化成直方图的形式如图3-13所示。其中子图3-13(a)表示不同后处理方案在检测精度上的的差异，子图3-13(b)表示不同后处理方案在单帧图像处理耗时上的差异。



(a) mAP 对比 (b) 单帧图片处理时长对比

图 3-13: CIBPP 与其他后处理方案的对比

表3-4及图3-13所示的结果中，Only Base Detector 指的是 YOLOv3 模型在训练之后直接用于视频目标检测的 mAP 结果，没有添加任何后处理过程。其

余几种分别为 YOLOv3 模型上添加了相应的后处理方案得到的结果。从上述结果中不难发现，本文提出的 CIBPP 方法相较于其他几种后处理方案，对检测精度的提升幅度更大。其中 Seq-NMS 由于设计思路简洁，计算速度比 CIBPP 稍快，相应的 Seq-NMS 在精度上的提升有些不尽如人意。相比于 Seq-Bbox，CIBPP 运用了更多的检测框信息，因此建立的跨帧检测框连接更为可靠，在检测速度与 Seq-Bbox 不相上下的同时精度更高。而对于 REPP，虽然 REPP 同样利用了检测框的外观信息，但 CIBPP 增加了序列间双向扩散以优化误检的模块，因此得到的最终检测精度比 REPP 更高。综上所述，和 Seq-NMS、Seq-Bbox、REPP 几种后处理方案相比，本文提出的 CIBPP 在检测精度和计算速度上的整体表现略胜一筹。

3.4 本章小结

本章主要介绍了一种基于视频上下文信息的非实时后处理框架 CIBPP。CIBPP 重点在于提升检测准确性而非实时性。本章首先介绍了后处理框架需要的一些数据准备工作，包括检测框信息的形式化表示、外观信息的获取，以及检测框间的距离度量，这部分内容可沿用于本文第四章介绍的实时后处理框架。随后对于 CIBPP 后处理框架进行了一些介绍，从整体流程到具体模块都给出了相关说明。系统地描述了 CIBPP 在整个视频片段上建立长时的跨帧检测框连接的过程。CIBPP 使用全局上下文信息来构造序列，进而在此基础上实现序列间优化误检及序列间优化漏检的操作，这也使得 CIBPP 无法应用于实时视频目标检测。除此之外，本章对于后处理框架 CIBPP 中每一个模块的算法实现，也通过伪代码和数学符号等形式进行了进一步阐释。最后将该后处理框架应用于几种不同的基检测器上来验证其有效性，并与其他后处理方案进行对比来评估其优劣。

第四章　实时视频目标检测后处理及结果平滑框架

本文第三章介绍了一种基于视频上下文信息的非实时后处理框架，该框架主要应用于本地视频分析，致力于提升图像目标检测器在视频数据上的检测精度，而无法用于在线目标检测后处理。因此本章介绍了一种用于实时视频目标检测的后处理及结果平滑框架，首先分析问题背景，然后给出框架的具体设计和模块介绍，最后通过实验来验证其有效性。

4.1　问题分析

4.1.1　背景分析

前文所述的基于视频上下文信息的非实时后处理框架 CIBPP 的输入实际上是整个视频片段的检测结果。换言之，在处理每一帧中的检测框时，既可以使用该帧之前的信息，也可以使用该帧之后的信息。即利用整个视频片段上的全局上下文来实现优化误检和漏检的目的。这一点在优化漏检的操作上体现的尤为明显，CIBPP 对漏检的优化是通过序列间双向扩散实现的。因此必须在扫描完整个视频片段的检测结果并构建出检测框序列之后，才能实现双向扩散。

而在视频目标检测的实际应用场景中，很大一部分是实时检测场景，比如安防监控，这类场景均使用摄像头数据作为实时输入，并且需要在用户端实时返回检测结果，以便工作人员观测。实时视频目标检测要求在处理每一帧的信息时，不能使用未出现的帧信息，因此第三章中介绍的后处理框架无法工作于实时检测场景下。

除此之外，实时目标检测的用户端往往需要用图形化界面实时显示检测结果。比如前文提到的安防监控，其用户端往往由工作人员通过显示器查看监控画面及检测结果。因此，实时检测需要考虑的另一个问题即为检测结果的稳定性。对于视频目标检测，往往会出现视频画面及待检目标位置均保持不变，但

该目标的检测框在较短时间间隔内经常抖动的情况。其原因是基于 CNN 的目标检测算法输入数据是像素值，在人眼观测下没有变化的图像场景，对于 CNN 而言可能在像素信息上已经发生了改变。细微的光线变化或者视角偏移，都会导致检测结果不稳定，最终显示在用户界面上的即为抖动的检测框，非常影响观测。

鉴于 CIBPP 无法用于在线视频目标检测后处理，以及在线视频目标检测需要在用户端提供尽可能稳定的检测结果，我们在本章中介绍了一个实时视频目标检测后处理及结果平滑框架 OPP-Smooth（Online Post-Processing and Smooth，OPP-Smooth）。其中检测结果平滑模块主要使用卡尔曼滤波来实现，关于这一算法的具体细节在本文4.1.2中给出相应介绍。

4.1.2 卡尔曼滤波

1960 年，R.E.Kalman 发表的一篇工作描述了离散数据线性滤波问题的递归解 [102]，这也是卡尔曼滤波（Kalman Filter）算法名称的由来。再之后由于数字计算的不断进步，卡尔曼滤波器一直是广泛研究和应用的主题，特别是在计算机视觉领域 [103]。

卡尔曼滤波器实际是一组数学方程，它提供了一种有效的计算（递归）方法，以最小化均方误差（Mean Square Error）的方式来估计过程中的状态。该滤波器在几个方面都非常有效：它支持对过去、现在甚至未来状态的估计。即使在建模系统的精确性质未知的情况下也能做到这一点 [103]。

为了更好的阐述卡尔曼滤波相关内容，现给出本小节内使用的数学符号及其含义说明，如表4-1所示。

卡尔曼滤波适用于线性高斯系统的状态分析，首先给出这类被观测系统的方程式描述，分为状态方程和观测方程，状态方程如式4-1所示。

$$\begin{aligned} x_k &= Ax_{k-1} + Bu_k + q_k \\ s.t. q_k &\sim N(0, Q) \end{aligned} \tag{4-1}$$

其中 x_k 表示系统当前时刻，即 k 时刻的状态，x_{k-1} 表示系统前一时刻，即 $k-1$ 时刻的状态。u_k 为控制量，q_k 为符合高斯分布的过程噪声，其协方差为 Q。A、B 为系统参数，A 为状态转移矩阵，B 为控制矩阵，二者维度由系统状态维

表 4-1: 卡尔曼滤波涉及到的相关数学符号说明

数学符号	含义
x_k	系统当前时刻，即 k 时刻的真实状态
x_{k-1}	系统前一时刻，即 $k-1$ 时刻的真实状态
$\hat{x}_k^-$	系统在 k 时刻的先验状态估计值，或称状态预测值
$\hat{x}_k$	系统在 k 时刻的后验状态估计值，或称状态最优估计值
z_k	系统在 k 时刻的状态观测值
u_k	系统控制量，若没有则设为 0
q_k	符合高斯分布的过程噪声
r_k	符合高斯分布的测量噪声
A	状态转移矩阵
B	可选的控制矩阵
C	状态观测矩阵
Q	过程噪声 q_k 的协方差矩阵
R	测量噪声 r_k 的协方差矩阵
$\hat{P}_k^-$	k 时刻的先验估计协方差
$\hat{P}_k$	k 时刻的最优估计协方差
K_k	卡尔曼增益

度，即控制量维度决定。观测方程如式4-2所示。

$$\begin{aligned} y_k &= Cx_k + r_k \\ s.t. r_k &\sim N(0, R) \end{aligned} \tag{4-2}$$

其中 y_k 表示系统当前时刻，即 k 时刻的观测值，r_k 为符合高斯分布的测量噪声，其协方差为 R。C 为系统参数，表示状态观测矩阵，其维度也由系统状态的维度决定。在后续卡尔曼滤波算法中实际上并不会直接使用到过程噪声 q_k 和测量噪声 r_k，而是使用它们的协方差矩阵 Q 和 R。

对于状态估计算法而言，每个时刻的状态对应四个相关值：$\hat{x}_k^-$（先验状态估计值，状态预测值）、$\hat{x}_k$（后验状态估计值，状态最优估计值）、z_k（状态观测值）、x_k（状态真实值）。而卡尔曼滤波的原理就是使用前一时刻的最优估计值 $\hat{x}_{k-1}$ 预测当前状态 $\hat{x}_k^-$，再根据当前状态观测值 z_k 进行修正，得到当前状

态最优估计值 $\hat{x}_k$，使其尽量逼近状态真实值 x_k。基于上述思路，卡尔曼滤波可以被分为两个步骤：预测和更新。预测阶段的方程如式4-3所示。

$$\begin{aligned} \hat{x}_k^- &= A\hat{x}_{k-1} + Bu_{k-1} \\ \hat{P}_k^- &= A\hat{P}_{k-1}A^T + Q \end{aligned} \tag{4-3}$$

预测阶段使用前一时刻的最优估计 $\hat{x}_{k-1}$，结合系统控制量 u_{k-1}，得到当前时刻的先验估计 $\hat{x}_k^-$。并在此基础上计算出先验估计对应的协方差矩阵 $\hat{P}_k^-$，至此卡尔曼滤波的预测阶段结束。整个滤波算法本质上是一个迭代的过程，对于任意一个时刻，都需要更新该时刻的系统状态和误差协方差，在预测阶段也会使用到前一时刻的最优估计和最优估计协方差矩阵。相应的，在更新阶段，需要计算出当前时刻的最优估计和最优估计协方差矩阵，提供给下一时刻做计算。更新阶段的方程式如式4-4所示。

$$\begin{aligned} K_k &= \frac{\hat{P}_k^- C^T}{C\hat{P}_k^- C^T + R} \\ \hat{x}_k &= \hat{x}_k^- + K_k\left(z_k - C\hat{x}_k^-\right) \\ \hat{P}_k &= \left(I - K_k C\right)\hat{P}_k^- \end{aligned} \tag{4-4}$$

更新阶段首先使用先验估计对应的协方差矩阵 $\hat{P}_k^-$、状态观测矩阵 C 及测量噪声协方差矩阵 R 计算出卡尔曼增益 K_k。而这一增益作为系数用于先验估计的修正，结合系统观测值，将先验估计 $\hat{x}_k^-$ 修正为当前时刻的最优估计 $\hat{x}_k$。因此卡尔曼增益体现了当前计算是更信任观测值 z_k 还是更信任先验估计值 $\hat{x}_k^-$。在得到最优估计后，最后一步需要计算最优估计对应的协方差矩阵，用于下一时刻的迭代计算。鉴于文本篇幅的限制，上述过程中涉及到的具体推导过程不作详细说明，后续对卡尔曼滤波算法的具体使用将在实验与分析部分详述。

4.2 实时视频目标检测后处理及结果平滑框架细节

前一节中我们给出了实时后处理和非实时后处理在使用场景上的区别，并根据实时视频检测往往伴随着用户端界面实时观测这一特点，点明了实时视频目标检测不容忽视的一点即为检测结果的稳定性。而本节将介绍为了解决上述问题提出的实时视频目标检测后处理及结果平滑框架 OPP-Smooth。该框架分两个主要模块：实时类分数修正，实时检测结果平滑。本节首先给出整个框架

的工作流程，再分别对这两个模块及其他辅助模块的设计和实现进行阐述。

4.2.1 整体流程

OPP-Smooth 属于实时后处理框架，每获取到一帧图像的检测结果时，后处理算法就会被执行一次。因此其数据输入与本文第三章介绍的 CIBPP 算法不同，CIBPP 是将整个视频片段上所有的检测结果统一送入后处理模块进行分析，而 OPP-Smooth 是实时检测、逐帧分析。

对于实时后处理，若要结合视频数据的时间信息及上下文信息，则必须利用当前帧的前序帧来实现时间一致性。因此，在 OPP-Smooth 中，需要对前序一定数量的帧进行结果暂存。由于实时检测的视频流可能是无限长的，因此暂存的结果长度也需要一定限制，超出这一限制时，使用队列的先进先出机制进行更新。每次得到一帧图像的检测结果时，整体后处理流程如图 4-1及图 4-2所示。

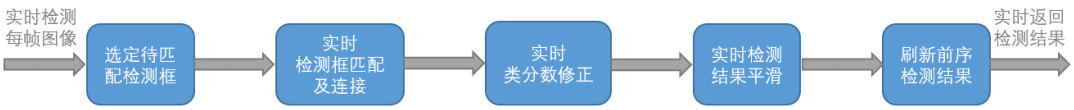


图 4-1: OPP-Smooth 后处理简要流程

在得到每一帧的实时检测结果后，向前寻找一组待匹配的检测框，使用找到的这组检测框和当前帧所有的检测框，按照本文3.2.2（1）及3.2.3（1）中所述的方式构建距离矩阵。在此距离矩阵的基础上得到一组匹配结果，分别使用这一组匹配结果进行实时类分数修正和检测结果平滑，前者对当前帧检测结果的语义信息进行修正，后者对当前帧检测结果的几何信息进行修正，使连续帧的检测结果更为稳定。而在修正完成后，即可将当前帧的最终结果绘制在用户界面上，同时刷新暂存序列信息，用于下一帧的修正。

图4-2给出了 OPP-Smooth 的详细处理流程，图中使用红色虚线框来展示每个模块的数据变化，使用彩色方块表示一个目标在不同帧之间的检测框，使用白色字体表示检测框的类分数。现结合该图，给出本章中的相关符号说明，见表4-2，其中卡尔曼滤波的相关符号已在本文4.1.2中给出，因此不再赘述。

整个实时后处理框架有两个主要模块：实时类分数修正、实时检测结果平滑。实时类分数修正实际上是 CIBPP 中序列内优化误检模块修改成在线处理之后的版本，而实时检测结果平滑是通过卡尔曼滤波实现的。这两个模块都需要

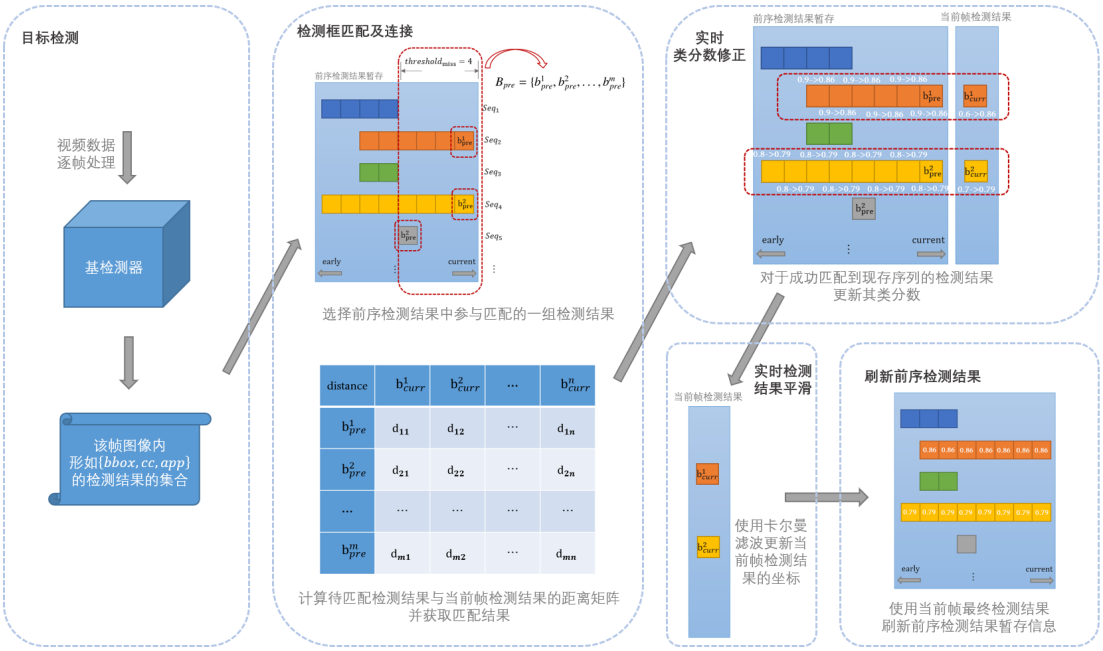


图 4-2: OPP-Smooth 后处理详细流程

使用到当前帧的前序帧信息，用以实现检测结果的时间一致性及滤波算法。因此就涉及到对已检测帧的信息进行暂存，并在每一帧的结果被修正后刷新暂存信息。上述两个主要模块及辅助的信息暂存机制将在本文4.2.2介绍。

4.2.2 关键模块

本小节主要对实时视频目标检测后处理及结果平滑框架 OPP-Smooth 中的关键模块进行详细介绍。

（1）前序检测结果暂存

当前时刻以前的检测结果以序列的形式保存，每个序列最少含有一个检测框。这里使用的序列概念和本文3.2.2（2）中描述的检测框序列本质上是一致的。而每个已出现的序列，都记录其序列长度、未出现长度、平均类分数。信息暂存形式如图4-3所示。

图4-3中，使用不同颜色区分不同对象，$threshold_{max}$ 表示所有序列的最大记录长度。实时视频目标检测的数据输入可能是摄像头数据，故而无法保存无限长的序列，并且序列长度过长对检测结果的优化也没有帮助。因此，长度超过 $threshold_{max}$ 的序列，使用队列的数据处理方式，将队头检测框删除。图4-3为了说明信息暂存的方式将 $threshold_{max}$ 设置为 8，不代表实际测试和实

表 4-2: OPP-Smooth 后处理框架涉及到的相关数学符号说明

数学符号	含义
$INFO$	前序检测结果记录
B_{pre}	前序检测结果中参与当前帧匹配的检测框集合
B_{curr}	当前帧原始检测框集合
B_{refine}	当前帧经实时后处理优化后的检测框集合
b	某一检测框
$b.cc$	检测框 b 的语义信息，即类分数
D	B_{pre} 和 B_{curr} 的距离矩阵
$P_{pre\ curr}$	B_{pre} 和 B_{curr} 的匹配结果
$index_{temp}$	参与匹配的前序序列索引
$index_{pre}$	匹配成功的前序序列索引
Seq	某一检测框序列
$Seq.len$	序列长度
$Seq.miss$	序列未出现长度
$Seq.cc$	序列平均类分数
$threshold_{max}$	暂存区最大序列长度
$threshold_{miss}$	最大允许目标未出现长度

验中的数值。图中靠近左侧的表示时间上更早的检测框，而靠近右侧的表示最近视频帧的检测框。每个序列都维护其序列长度和未出现长度，对应的序列类分数即为当前序列内所有检测框的类分数平均。除此之外，对于保存的最大序列数量（并非最大序列长度）也有相应约束，但一般情况下一个视频内不会同时存在大量的待检对象，因此该参数的设置手工完成即可，具体数值于本文4.3中给出。初始时整个暂存区为空，得到每帧的优化结果后不断对暂存区进行刷新，直到视频全部分析完成或停止实时摄像头输入后清空该区域。

（2）实时类分数修正

实时类分数修正模块完成的工作实质上与本文3.2.2（3）序列内优化误检完成的工作是一致的，同样借助检测框序列来对低置信度检测框进行优化，实时修正当前帧检测结果的语义信息。第三章介绍的 CIBPP 在匹配检测框时，使用的是相邻视频帧 t、$t+1$ 对应两组初始检测框结果 $B_t = \{b_t^1, b_t^2, \ldots, b_t^m\}$、

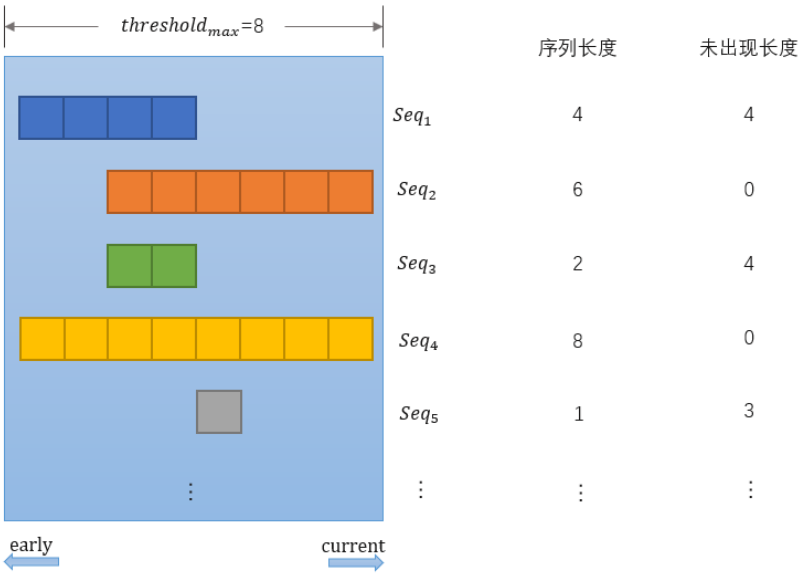


图 4-3: 前序检测结果暂存机制

$B_{t+1} = \{b^1_{t+1}, b^2_{t+1}, \ldots, b^n_{t+1}\}$。而在实时后处理框架中，进行检测框匹配的两组数据是当前帧上的检测结果 $B_{curr} = \{b^1_{curr}, b^2_{curr}, \ldots, b^n_{curr}\}$，以及前序的一组检测结果 $B_{pre} = \{b^1_{pre}, b^2_{pre}, \ldots, b^m_{pre}\}$。当前帧检测结果可以从基检测器中直接获得，而选择 B_{pre} 的过程如图4-4所示。

这里引入了另一个阈值 $threshold_{miss}$，即最大允许的未出现长度。初始时 B_{pre} 为空，对于未出现长度小于 $threshold_{miss}$ 的序列，取序列尾部检测框加入 B_{pre}，同时还需要记录对应的序列下标，便于后续使用该下标定位到该序列的类分数、序列长度等数据。最后使用 B_{pre} 和 B_{curr} 构建距离矩阵，构建距离矩阵的方式与本文3.2.2（1）所述一致。

OPP-Smooth 中检测框匹配的方式和 CIBPP 保持一致（见本文3.2.2（3）），算法 3.2返回结果为视频帧 t 及 $t+1$ 的检测框匹配结果 $P_{t\,t+1}$，而 OPP-Smooth 每帧图像的匹配得到的是前序检测框和当前帧检测框的匹配结果 $P_{pre\,curr}$。除此之外，还会返回一个索引序列 $index_{pre}$，用于标记哪些前序序列和当前帧的检测结果成功匹配。本文4.2.2（1）提到每个已出现的序列都需要维护其序列长度、未出现长度、平均类分数，索引序列 $index_{pre}$ 主要用于定位到上述数据并更新。得到匹配结果后，对于当前帧中所有匹配成功的检测框，将

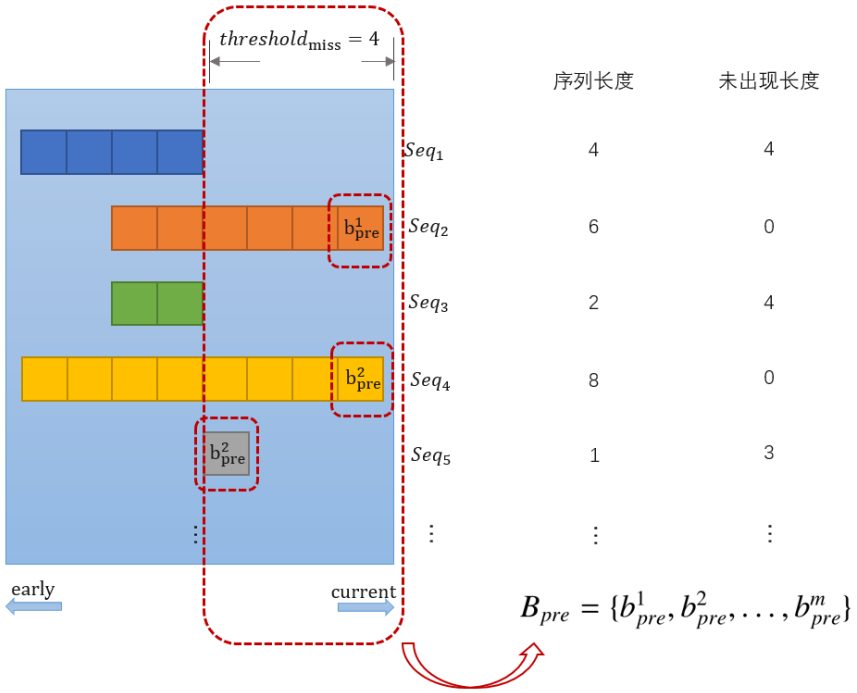


图 4-4: 选择待匹配的前序检测结果

其类分数作如下修改：

$$b.cc = \frac{1}{Seq.len + 1} b.cc + \frac{Seq.len}{Seq.len + 1} Seq.cc \tag{4-5}$$

其中，b 为当前帧的某个检测框，$b.cc$ 为基检测器给出的该检测框原始类分数，Seq 为该检测框匹配到的前序序列，$Seq.cc$ 为该序列的平均类分数，$Seq.len$ 为该序列长度。式4-5实质上是根据前序序列长度，将序列平均类分数和检测框原始类分数进行加权平均。通过上述方式，对于每一个视频帧，向前进行检测框序列连接，并实时更新当前检测框类分数，以实现对误检的优化。

（3）实时检测结果平滑

前文4.2.2（2）描述的实时类分数修正实现了在线的检测框语义信息修正，而实时检测结果平滑模块主要是通过检测框几何信息的修正，来达到稳定检测结果的目的。对于同一对象在相邻帧之间的检测结果，由于视角变化或者对象姿态变化而引起的检测框长宽变化是属于合理范围内的，而当对象保持不变且拍摄视角也稳定不变时，此时由于光线或图片像素点数据的轻微变化，导致的检测框偏移，则被认为是噪声。人眼捕捉到的未发生变化的图像内容，经

过图像采集设备及解码操作之后，可能在像素信息上已经发生了改变，对于目标检测模型而言变成了两张截然不同的图像。OPP-Smooth 希望利用卡尔曼滤波算法尽可能的过滤掉这类噪声，使得用户端实时观测的检测框能够更加稳定。

在本文4.1.2中，我们详细介绍了卡尔曼滤波算法的适用系统以及工作机制。而对于视频目标检测这一任务来说，由于像素变化而非检测对象移动带来的检测框偏移，被看成是高斯白噪声。前一帧的最终检测结果（即前一帧经过实时后处理优化后的结果）被用于预测当前帧的结果，再结合当前帧实际检测到的结果（即基检测器得到的结果）对预测进行优化。整个过程中使用检测框的中心点坐标参与卡尔曼滤波的预测及更新。鉴于上述过程是发生在一段视频片段内的，为了更好的在本文中进行分析，我们将滤波过程映射到一张二维图像上，如图 4-5。

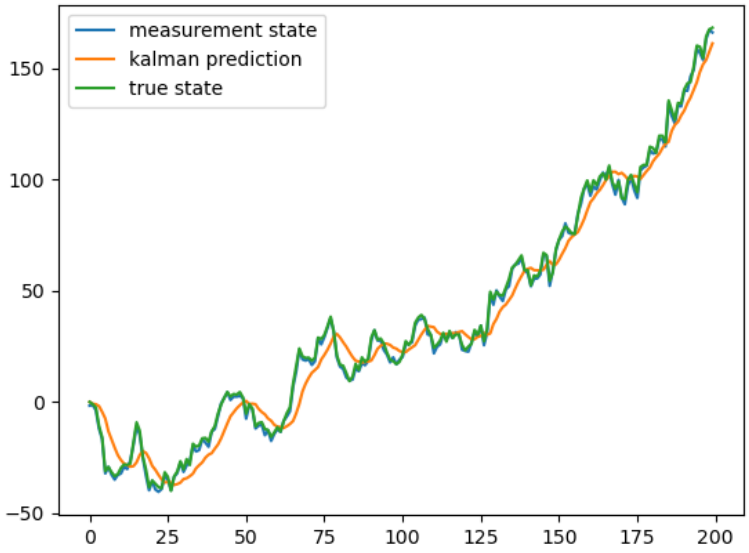


图 4-5: 实时检测结果平滑的工作原理

图 4-5显示了一组坐标点的位置变化，黄色为卡尔曼滤波后的结果，可以发现其相对于蓝色测量值而言明显更为平滑，抖动的频率更小。在 OPP-Smooth 中，将检测框抽象成坐标点，使用前序帧优化后的结果和当前帧的测量结果，得到当前帧优化后的检测框中心点坐标。再将原始检测框的长宽应用于优化后的中心点坐标上，即为更新后的检测框几何信息。而关于卡尔曼滤波算法在应用时的具体参数设置将在本文4.3中阐述。

4.2.3 算法流程

前文介绍了实时视频目标检测后处理及结果平滑框架 OPP-Smooth 的整体结构，以及其中的关键模块和辅助模块。本节主要根据实际的数据流来分析整个框架的流程，并对部分模块的具体实现给出算法说明。

对于实时后处理框架而言，整个框架的输入数据是当前帧的信息，用 *curr* 表示正在处理的视频帧，则其上所有检测框的集合为 $B_{curr} = \{b^1_{curr}, b^2_{curr}, \ldots, b^n_{curr}\}$。另外，本文4.2.2（1）中介绍的前序检测结果记录记为 $INFO$，$INFO$ 实际上是检测框序列的集合，即 $INFO = \{Seq_1, Seq_2, Seq_3, \ldots\}$。其中每一个 Seq 都表示一个对象的跨帧检测结果连接。$threshold_{max}$ 用于约束每一个对象对应的最大序列长度，另一个阈值 $threshold_{miss}$ 负责筛选前序哪些检测框用于和当前帧检测结果 B_{curr} 进行匹配，具体选择及匹配方式在本文4.2.2（2）中已给出说明。匹配之后得到一组匹配结果 $P_{pre\,curr}$ 以及一个索引序列 $index_{pre}$。根据该匹配结果，找到当前帧检测结果和现存序列的关系，利用这种关系实现实时类分数修正和检测结果平滑操作。则原始检测结果 B_{curr} 经过优化之后变成 B_{refine}，将 B_{refine} 作为最终的实时视频目标检测结果绘制在视频帧上或返回至用户界面，并利用这一优化后的结果刷新 $INFO$ 内相关序列的信息。至此后处理算法结束一轮迭代，在下一帧数据输入时，再次执行上述操作。

其中遍历前序检测结果找到一组检测框，用于与当前帧检测框进行匹配的过程如算法4.1所示。

算法 4.1 选择待匹配的前序检测结果

Require: 前序检测结果记录 $INFO = \{Seq_1, Seq_2, Seq_3, \ldots\}$，最大允许对象未出现长度阈值 $threshold_{miss}$

Ensure: 参与匹配的前序检测框 $B_{pre}\{b^1_{pre}, b^2_{pre}, \ldots, b^m_{pre}\}$ 以及这些检测框所在序列的索引 $index_{temp}$

初始化 B_{pre} 为空，$index_{temp}$ 为空
$len1 = INFO$ 中元素个数
for $i = 1; i <= len1; i++$ **do**
  $Seq = INFO_i$
  if $Seq.miss <= threshold_{miss}$ **then**
    $len2 = Seq.len$
    将 Seq_{len2} 加入 B_{pre}
    将 $Seq.index$ 加入 $index_{temp}$
  end if
end for

算法4.1将前序每一个序列的未出现长度 $Seq.miss$ 与阈值 $threshold_{miss}$ 进行比较，若在范围内，则将序列尾部的检测框 Seq_{len2} 加入待匹配集合 B_{pre} 中，并记录下这些序列对应的索引 $Seq.index$。未必所有的现存序列都符合匹配要求，因此要利用索引加以区分。同理，目前选择的待匹配序列及序列尾部检测框，也未必都能和当前帧实时检测结果匹配成功，因此待匹配的序列索引只能是临时值 $index_{temp}$，在匹配成功之后被修正成 $index_{pre}$。

得到待匹配检测框 B_{pre} 之后，需要使用 B_{pre} 与当前帧的检测结果进行匹配。匹配方式与算法3.1和算法3.2基本相同，均需要建立距离矩阵，随即寻找成对的检测框得到匹配结果，OPP-Smooth 将匹配结果记为 $P_{pre\,curr}$。稍有不同的是，在 OPP-Smooth 框架的匹配过程中，需要对 $index_{temp}$ 加以修正，$index_{temp}$ 表示参与匹配的检测框所在序列索引，而成功匹配到一对检测框时，需要记录匹配成功的检测框所在序列索引，据此，将 $index_{temp}$ 修正成为 $index_{pre}$。

完成前文所述操作后，已获知当前检测框和前序序列的匹配关系，随后即可进行实时类分数修正。本质上即为，根据匹配关系，使用前序序列的平均类分数，结合序列长度，与当前检测框的类分数做加权平均。具体过程如算法4.2所示。

算法 4.2 实时类分数修正

Require: 前序检测结果记录 $INFO = \{Seq_1, Seq_2, Seq_3, \ldots\}$，匹配结果 $P_{pre\,curr}$，匹配成功的检测框所在序列的索引 $index_{pre}$，当前帧检测结果 $B_{curr} = \{b^1_{curr}, b^2_{curr}, \ldots, b^n_{curr}\}$

Ensure: 当前帧经过类分数修正之后的检测结果 B_{temp}

1: $len1 = B_{curr}$ 中元素个数
2: **for** $i = 1; i <= len1; i++$ **do**
3: **if** b^i_{curr} 与前序检测框匹配成功 **then**
4: 根据 $P_{pre\,curr}$ 及 $index_{pre}$ 找到 b^i_{curr} 匹配到的序列 Seq
5: $b^i_{curr}.cc = \frac{1}{Seq.len+1} b^i_{curr}.cc + \frac{Seq.len}{Seq.len+1} Seq.cc$
6: **end if**
7: **end for**

算法4.2实际上是对当前帧检测结果进行原地修改，但为了体现出该模块的工作机制，将输出记为 B_{temp}。这一结果还需要经过卡尔曼滤波，对每个检测框的几何信息进行修正后方可得到最终的优化结果 B_{refine}。算法4.2中 cc 表示检测框及序列的类分数，$Seq.len$ 表示序列长度。

获得当前帧最终检测结果的最后一步是实时检测结果平滑，即卡尔曼滤波，该模块输入数据是匹配结果中每一对检测框的中心点，将检测框看成是质

点来进行滤波。输出得到每个检测框新的中心点，再结合检测框原有的长宽，即可得到修正后的几何信息，相应的也就得到了 B_{refine}。卡尔曼滤波的应用主要是本文4.1.2所述公式的实现和参数设置，不涉及到过多的算法逻辑，因此没有给出伪代码说明。

行文至此，每帧图片的检测结果实时优化实质上已全部完成，但还需要利用已经优化的检测结果，将其作为前序信息，更新前序检测结果记录 $INFO$，以便在下一帧图像输入时能够利用最新的前序信息。具体更新过程如算法4.3所示。

算法 4.3 刷新前序检测结果

Require: 前序检测结果记录 $INFO = \{Seq_1, Seq_2, Seq_3, \ldots\}$，匹配结果 $P_{pre\ curr}$，匹配成功的检测框所在序列的索引 $index_{pre}$，当前帧最终检测结果 B_{refine}，允许记录的最大序列长度 $threshold_{max}$

Ensure: 更新后的前序检测结果记录 $INFO = \{Seq_1, Seq_2, Seq_3, \ldots\}$

$len1 = B_{refine}$ 中元素个数
for $i = 1; i <= len1; i++$ **do**
  if b^i_{curr} 与前序检测框匹配成功 **then**
    根据 $P_{pre\ curr}$ 及 $index_{pre}$ 找到 b^i_{curr} 匹配到的序列 Seq
    将 b^i_{curr} 加入 Seq
    $Seq.cc = b^i_{curr}.cc$
    $Seq.miss = 0$
    $Seq.len++$
    if $Seq.len > threshold_{max}$ **then**
      移除 Seq 中第一个检测框
      $Seq.len = threshold_{max}$
    end if
  end if
  if b^i_{curr} 是当前帧中未匹配成功独立存在的检测框 **then**
    $Seq' = \{b^i_{curr}\}$
    $Seq'.cc = b^i_{curr}.cc$
    $Seq'.miss = 0$
    $Seq'.len = 1$
    将 Seq' 加入 $INFO$
  end if
end for
for $INFO$ 中每一个未更新信息的序列 Seq'' **do**
  $Seq''.miss++$
  if $Seq''.miss > threshold_{max}$ **then**
    从 $INFO$ 中移除 Seq''
  end if
end for

算法4.3中，Seq_cc 表示序列平均类分数，Seq_miss 表示序列未出现长度，Seq_len 表示序列长度。该算法对于每一对匹配成功的序列，都定位到一个现存序列，并更新该现存序列的类分数、未出现长度、序列长度等信息，若序列超过最大允许记录长度，则按照队列的数据处理方式移除队头。而对于当前帧中出现了但未匹配成功的检测框，将其视为一个长度为 1 的新序列加入 $INFO$ 中。上述两个阶段均未访问到的 $INFO$ 中的剩余序列，将其未出现长度加一，若超过 $threshold_{max}$ 帧仍未出现新匹配的检测框，将整个序列移除，至此，前序检测结果记录的更新已完成。

4.3 实验与分析

前文4.2已给出了实时视频目标检测后处理及结果平滑框架 OPP-Smooth 的整体设计及各模块的详细算法流程。本节主要介绍 OPP-Smooth 的实验设置及实验结果。

4.3.1 实验设置

OPP-Smooth 的相关实验中，使用的实验环境、数据集及评价指标与本文3.3.1（1）及3.3.1（2）中介绍的保持一致，因此不再赘述。同样的，基检测器的选择、特征提取模块、逻辑回归模块的设置也均与本文3.3.1（4）所述的设置同步。因此，本小节仅对 OPP-Smooth 部分模块的细节设置及参数设置进行阐述，具体如下：

- **前序检测结果暂存**：设置序列最大记录长度 $threshold_{max}$=90，鉴于 ImageNet VID 数据集中大部分视频的 fps 都在 30 上下，该阈值可以理解为最多记录同一对象三秒内的检测结果，超出该范围的数据对当前帧的优化结果影响很小，利用队列的机制剔除队头位置的检测框。另外，对于超过 $threshold_{max}$ 帧仍未出现下一个检测框的序列，则将其导出，不再尝试匹配，从而节约一定的计算资源。与此同时，设置最大序列数量为 100，由于前序结果暂存模块实质上只会保留几秒内的检测结果，而视频流中几秒内不太可能同时存在几百个不同的检测对象，因此这一参数只起到约束作用，对算法本身没有影响。
- **实时检测框匹配及连接**：使用 $threshold_{miss}$ 来代表最大允许的未出现长度，即进行检测框匹配时，除了前一帧的检测框可以尝试匹配，前两帧、

前三帧、前 $threshold_{miss}$ 帧中出现过的检测框都可以尝试匹配，实验中将 $threshold_{miss}$ 设置为 10。

- **实时检测结果平滑（即卡尔曼滤波）**：OPP-Smooth 对每个检测框的中心点坐标进行滤波，并且对于目标的运动我们无法预估其运动状态，因此将卡尔曼滤波中的控制量设为 0，相应的也不再需要考虑控制矩阵 B。而状态转移矩阵 A 的设置如式 4-6。

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4-6}$$

状态观测矩阵 C 的设置如式 4-7，经过状态观测矩阵处理之后可以使得先验状态估计和状态观测值的维度一致。

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{4-7}$$

第一帧预测时没有前序帧，相应的也就没有前序最优结果用于卡尔曼滤波的预测阶段，可以使用假设值作为前序最优结果，并不影响后续处理。无论前序最优结果设置为何值（不全取为 0），经过几次迭代之后，卡尔曼滤波都能收敛到合理的预测值，OPP-Smooth 将第一帧的观测值设置成前序帧的最优结果。除此之外，卡尔曼滤波预测阶段还需要使用前一帧的最优估计协方差矩阵，这一矩阵也可以人为设定，随后经几次迭代收敛到正确的数值。但该矩阵取值不能全为 0，否则滤波器会认为已经不存在误差，进而无法继续迭代。关于过程噪声协方差矩阵和测量噪声协方差矩阵的设置，实验中我们对比了不同的取值，对滤波结果均无明显影响，因此这里使用两个全 0 矩阵，随机取部分元素置为 1，保证至少有 1/4 的元素为 1，分别作为过程噪声协方差矩阵和测量噪声协方差矩阵。

4.3.2 有效性实验及结果分析

为了证明本文提出的实时视频目标检测后处理及结果平滑框架 OPP-Smooth 的有效性，我们使用 ImageNet VID 数据集进行了一些实验验证。基检测器使用 YOLOv3，同时也使用了其他几种视频目标检测模型进行对比分析，以保证

结果可信。与本文3.3.2类似，有效性实验中使用到的视频目标检测模型主要是 FGFA、SELSA 及 MEGA。而关于检测结果平滑的效果评估，目前没有相关数据集或统一指标用于衡量，因此我们只对实时类分数修正之后的结果进行评估。除此之外，由于从视频目标检测模型中进行检测框特征提取和从静态图像目标检测模型中进行检测框特征提取的方式不同，而本文提出的 OPP-Smooth 主要运用于静态图像目标检测器上，因而在与 SELSA、FGFA 及 MEGA 进行对比时，没有使用外观特征信息，这一点也与 CIBPP 的实验设置保持一致。实验结果如表4-3所示。

表 4-3: OPP-Smooth 在不同基检测器上的实验结果

Method	Base Detector	Backbone	mAP	ProcessingTime
YOLOv3	YOLOv3	Darknet-53	70.21%	18.57
YOLOv3+ OPPSmooth	YOLOv3	Darknet-53	75.41%	20.76
FGFA	R-FCN	ResNet-101	75.93%	91.20
FGFA+ OPPSmooth	R-FCN	ResNet-101	78.8%	94.72
SELSA	Faster R-CNN	ResNet-101	82.01%	133.16
SELSA+ OPPSmooth	Faster R-CNN	ResNet-101	84.14%	138.26
MEGA	Faster R-CNN	ResNeXt101	83.94%	164.58
MEGA+ OPPSmooth	Faster R-CNN	ResNeXt101	85.32%	169.51

表4-3中，Processing Time 为每张图片检测及实时类分数修正的总时长，不包括卡尔曼滤波的处理时间，单位为毫秒（ms）。从表中可以看出，对于各个基检测器，加入实时后处理过程之后其精度都有一定的提升，并且每张图片的处理时长仅有小幅度增长。尤其对于一些原始检测精度就比较好但耗时较长的模型，可以做到在耗时几乎持平的情况下将精度提升 1.8% 5%，充分说明了 OPP-Smooth 作为实时后处理框架的有效性及良好的计算速度。再将实时视频目标检测后处理及结果平滑框架 OPP-Smooth 与基于视频上下文信息的非实时后处理框架 CIBPP 进行对比，此处省略基检测器信息及骨干网络信息，具体如表4-4所示。

从表4-4中可以更直观的得到非实时后处理框架 CIBPP 和实时后处理框架 OPP-Smooth 的对比。不难看出，无论应用在何种基检测器上，后处理模块的加入都能使得前序视频目标检测的精度有一定提升。相应的，使用非实时后处

表 4-4: CIBPP 和 OPP-Smooth 在不同基检测器上的对比

Method	mAP	ProcessingTime
YOLOv3	70.21%	18.57
YOLOv3+CIBPP	79.36%	24.33
YOLOv3+ OPPSmooth	75.41%	20.76
FGFA	75.93%	91.20
FGFA+CIBPP	82.17%	98.34
FGFA+ OPPSmooth	78.8%	94.72
SELSA	82.01%	133.16
SELSA+CIBPP	85.69%	143.47
SELSA+ OPPSmooth	84.14%	138.26
MEGA	83.94%	164.58
MEGA+CIBPP	86.21%	175.32
MEGA+ OPPSmooth	85.32%	169.51

理模块 CIBPP 时精度提升幅度稍大一些，而使用实时后处理模块 OPP-Smooth 时对耗时的增加幅度是最小的。这一点也是符合预期的，非实时后处理可以使用全局时间信息，在处理每一帧结果时，既可以使用前序帧也可以使用后序帧，而实时后处理只能使用前序一定数量的帧用于辅助当前帧的结果优化。这也就导致了非实时后处理在精度提升上有着更好表现，相应的处理时间也会更长，表4-4中的检测结果也是符合上述逻辑的。

4.3.3 对比实验及结果分析

在本文4.3.2中我们通过将 OPP-Smooth 应用于不同的基检测器上来验证其有效性，而本小节主要使用其他实时后处理算法与 OPP-Smooth 进行比较，以验证其性能优劣。目前公开发表的实时后处理算法并不多见，以及部分方法没有给出具体的代码或实现参数，也使得无法将 OPP-Smooth 与之进行比较。因此，参考 Seq-Bbox 的工作，该工作分实时和非实时两个版本，本文主要将以下几种情况进行对比：基检测器不加后处理、基检测器使用本文提出的非实时后处理 CIBPP、基检测器使用本文提出的实时后处理 OPP-Smooth、基检测器使用 Seq-Bbox 后处理非实时版本、基检测器使用 Seq-Bbox 后处理实时版本。

具体实验结果如表4-5所示。

表 4-5: OPP-Smooth 与其他后处理方案的结果对比

Base Detector	Post Processing Method	mAP	ProcessingTime
YOLOv3	-	70.21%	18.57
YOLOv3	CIBPP	79.36%	24.33(18.57+5.76)
YOLOv3	OPP-Smooth	75.41%	20.76(18.57+2.19)
YOLOv3	Seq-Bbox	74.19%	26.19(18.57+7.62)
YOLOv3	Seq-Bbox(online)	73.11%	23.84(18.57+5.27)

表4-5中，Seq-Bbox 是非实时版本的后处理框架，而 Seq-Bbox(online) 表示实时版本的后处理框架，二者的区别在 Seq-Bbox 原文中有详细说明 [65]。从表中可以看出，对检测精度提升最大的仍然是 CIBPP，但如果限制后处理方案必须为实时后处理，则 OPP-Smooth 的效果最好。并且在上述几种方法中，OPP-Smooth 的后处理阶段是耗时最短的，证明了 OPP-Smooth 在精度和速度上均不逊色于其他实时后处理方案。

4.4 本章小结

本章主要介绍了一种实时视频目标检测后处理及结果平滑框架 OPP-Smooth。第三章已给出了后处理框架所需的数据处理、模型融合等工作，因而本章首先介绍实时后处理和非实时后处理在使用场景和工作机制上的异同点，以及卡尔曼滤波的理论基础。随后对于实时后处理框架 OPP-Smooth 进行了一些介绍，先给出整体流程以及辅助机制的说明，进而对其中的关键模块逐个分析，并结合伪代码和数学符号阐明每一个模块的数据处理过程。详细阐述了 OPP-Smooth 逐帧处理视频流数据，使用前序帧的检测结果优化当前帧中检测框几何信息及语义信息的过程。同时通过维护检测结果暂存区的信息来为后续帧的优化提供数据支持。整个过程不会涉及到任何未来视频帧的信息，因而可以保证实时性。本章最后将 OPP-Smooth 应用于不同的基检测器上来验证其有效性，并与其他实时后处理方案进行对比来评估其优劣。

第五章　实时视频目标检测后处理及结果平滑框架在系统中的应用

本文第四章在叙述实时视频目标检测后处理和非实时视频目标检测后处理的区别时提到，之所以会需要实时后处理，本质上是为实际使用场景服务。为了更好的将本文提出的后处理框架应用于实际系统工程中，我们使用实时后处理框架 OPP-Smooth，搭建了一个井下煤矿作业安全预警系统。本章将对该系统的研发背景、系统需求、系统设计及实际效果逐一进行分析。

5.1　相关背景

在工业生产中，井下煤矿作业安全事故的发生屡见不鲜。井下煤矿作业由于光线条件差，工作人员的活动空间和视野都严重受限，因此整体危险系数也比较高。如果工作人员进入危险区域进行作业，发生安全事故的概率也急剧升高，因此需要在工作人员所处位置存在安全隐患时，及时判断并给出提醒。需要完成的具体任务则是通过井下作业环境中安装的摄像头设备获取视频数据，根据画面内容来检测“工作面人员闯入禁区”这一行为并给出预警。具体来说，在采煤机运行期间，人员应禁止出现在采煤机周围的一定范围内。通过实时监测，进行视频分析，当人员进入划定的危险区域时，立即发出预警。图 5-1展示了一张采煤机正在运行的工作面场景照片。图中工作人员处在画面右侧区域，而采煤机在画面左侧区域运行，二者间通过一个电缆槽（图中黄色物体）分割。系统需要将该电缆槽作为安全区和危险区的边界（如图 5-1中红色辅助线所示），电缆槽左侧（即采煤机运行区）为危险区，另一侧则为安全区。

基于上述背景分析，我们开发了一个实时井下煤矿作业安全预警系统，其输入数据为摄像头捕捉到的实时画面，输出则是对当前时刻工作人员安全状态

图 5-1: 采煤机正在运行的工作面场景照片及安全区划分示意

的判断，及相关目标的检测结果。将该系统纳入井下煤矿的辅助安保体系，尽可能利用井下煤矿可以采集到的视觉信息，密切关注工作人员是否有安全风险。

5.2 系统需求

5.2.1 功能需求

尽管本系统的研发是面向视频的任务，但现阶段工业界在实际任务中更倾向于使用静态图像目标检测模型来逐帧处理视频。我们使用静态图像目标检测模型与后处理框架相结合的形式搭建该系统。整个系统需要根据输入的摄像头实时监控画面，进行相应的目标检测和后处理工作，最后实时给出画面内所有人员位置和画面内危险边界位置，以及一个当前情况是否危险的预警信号。我们对该系统的子任务进行抽取，则可以将整个任务实现拆分成两个步骤：一是对井下工作人员和电缆槽的识别，其中包含后处理模块；二是对当前画面是否有危险情况的判断。前者是典型的视频目标检测问题，其任务目标也非常明确，即找到画面中的工作人员以及电缆槽，对于其他对象类别则并不关心。后者需要判断是否存在危险情况，可以通过计算机视觉领域的一些经典处理方法来完成，系统结果拟合也需要在这一步骤中进行处理，比如如何从电缆槽对象

检测结果得到危险边界，得到危险边界之后如何判断当前工作人员是否处于危险状态。综上所述，可以将系统的功能需求总结如下：

- **工作人员检测**：识别出当前画面内所有的工作人员，该结果由基检测器给出，后续需要送入后处理模块进行优化，最终以矩形检测框的形式绘制于视频帧上。
- **电缆槽检测**：前文5.1提到井下作业使用电缆槽作为危险区域和安全区域的区分，而拟合危险边界首先需要检测出电缆槽对象。该对象的检测结果从基检测模型中获取，后续需要经过后处理模块优化，最后被用于边界计算和状态判断。
- **检测结果后处理**：对于前序目标检测模型给出的工作人员检测框和电缆槽检测框，通过本文第四章中提出的实时视频目标检测后处理及结果平滑框架 OPP-Smooth 进行优化，得到优化后的工作人员检测结果和电缆槽检测结果。
- **划分危险区域及安全区域**：根据每帧画面内的电缆槽位置，拟合出画面内的危险边界，并明确指出边界哪一侧为安全区域哪一侧为危险区域。
- **判断当前安全状态**：根据画面内工作人员和危险边界的相对位置判断当前工作人员状态是否安全，若为危险状态则给出预警信号。

上述内容分析了整个系统的功能需求，后续系统架构也是在这一基础上设计的。

5.2.2 性能需求

性能需求即非功能需求，在系统需求分析中担任着补充功能需求的角色。性能需求描述了系统设计需要遵循的规范和标准，以及一些设计细节和精度要求，它在很大程度上决定着系统的质量优劣。总体来说，井下煤矿作业安全预警系统应达到以下几个方面的要求：

- **准确性**：准确性包括两方面，首先是对于用户在页面上的操作能给出正确的交互响应，其次是对每个视频帧的处理要尽可能准确，不允许出现过多的预警误报或者无法检测出危险状态的情况。
- **计算速度**：该系统的输入是与摄像头绑定的，因此需要保证实时计算，具体来说，系统处理速度至少需要达到 25fps。
- **开放性**：系统的操作界面为网页，因此需要保证系统前端可以运行于目前

常用的大部分浏览器上。

- **可维护性**：如果需要对系统进行升级，新增加的功能模块应独立于原系统，后续开发不需要对整个系统代码进行重构。
- **耦合性**：系统各个功能模块应避免高耦合的情况，重复代码应写成子模块，以供其他模块调用。

以上是对整个系统性能需求的分析，后续在进行具体的系统设计时，也应当保证系统满足上述要求。

5.3 系统架构及实现

基于前文5.2对系统需求的分析，我们将整个系统分为四个模块，整体流程如图5-2及图5-3所示。其中我们略去了目标检测模型训练的部分，这是由于本章对系统的分析实质上是为了体现实时后处理框架在实际工程中是如何工作的，而模型训练并不能体现出这一内容。严格来说，训练好的模型作为基检测器用于井下煤矿作业安全预警系统中，本身也没有和用户交互的部分，并且收集数据及训练的方式与前文的实验过程类似，因此不再过多描述。需要明确的是，收集井下煤矿的图像数据进行目标标注时只对画面中的工作人员和电缆槽进行标注，训练及后续检测时也仅仅考虑这两类对象，若出现了其他类别的检测结果，直接判定为误检。 图5-2展示了整个系统的简要处理流程，输入数据

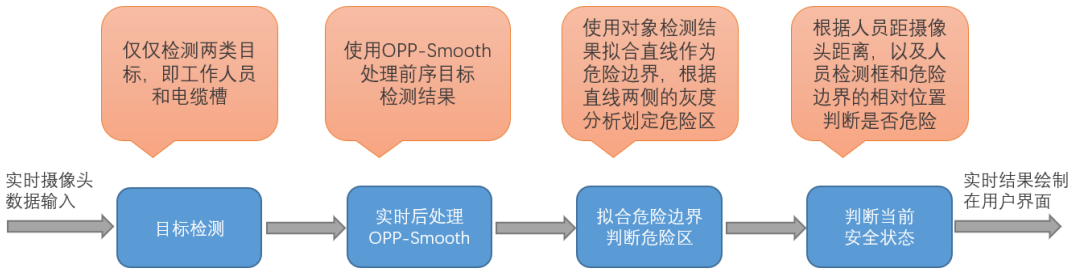


图 5-2: 井下煤矿作业安全预警系统简要工作流程

与摄像头绑定，但实质上仍然是逐帧处理图像，这也与第四章中描述的后处理框架 OPP-Smooth 的处理机制是一致的。因此图中所示的流程在整个视频流中是逐帧迭代的，每一帧图像都对应着一次流程迭代。具体流程细节如图5-3所示。图中将每帧图像的处理过程分为四个阶段，图像输入进系统，第一阶段需要经过以卷积神经网络模型为基础的目标检测模块，得到该图像上的原始检测

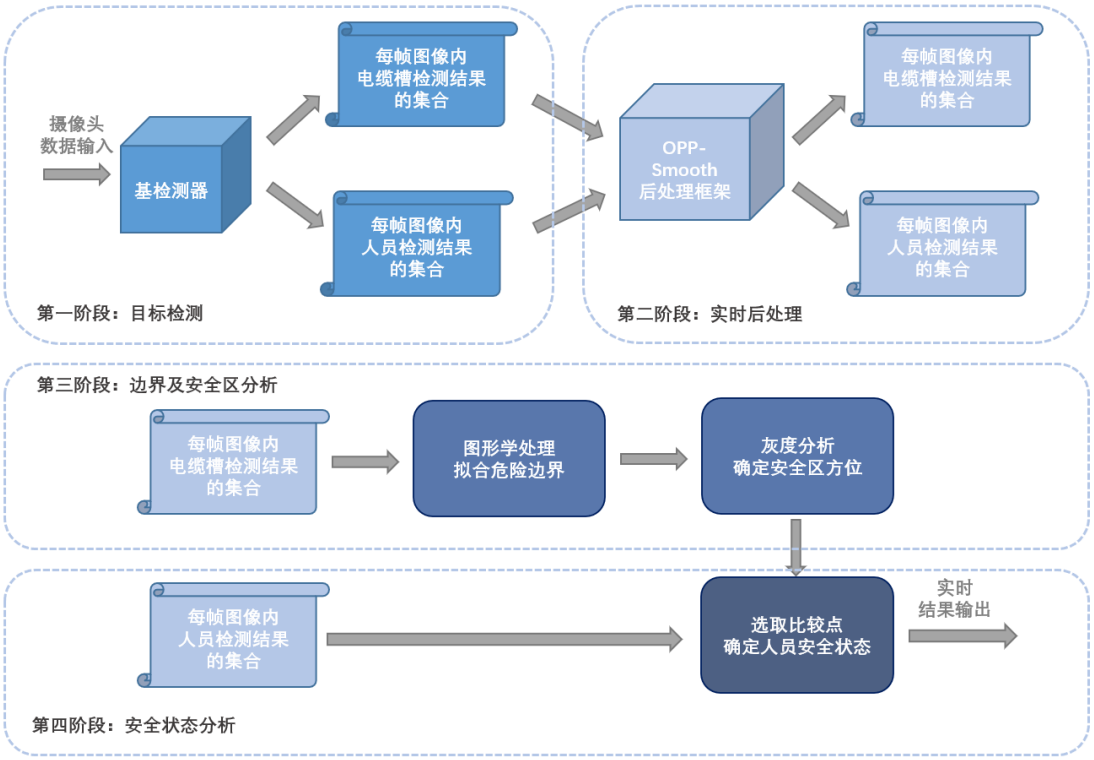


图 5-3: 井下煤矿作业安全预警系统详细工作流程

结果；第二阶段将检测结果送入后处理模块，结合前序检测结果来优化当前检测结果；至此已得到每帧图像的最优检测结果，第三阶段需要进行信息提取，根据电缆槽的检测结果信息拟合出该帧图像中的危险边界；第四阶段根据人员位置和危险边界的位置，判断当前画面中人员是否处于危险位置，从而决定是否需要给出预警信号。经过上述四个阶段的处理，一帧图像的处理流程结束，将得到的最终结果显示绘制在用户界面。

其中，目标检测阶段是通过训练好的图像目标检测模型完成的，而后处理阶段的工作机制已经在第四章中给出详细描述，因此不再赘述。剩余两个阶段的实现细节如下所述（其中划分危险区域及安全区域被分成了拟合危险边界和判断危险区方位两个子模块来描述）：

- **拟合危险边界：**系统之所以选择使用目标检测模型先找到电缆槽目标再进行后续处理，是由于图像中没有可以直接提取的边界信息，加之电缆槽本身也是具有一定宽度和高度的目标，而非一个线性（或非线性）的边界。因此系统需要对电缆槽这一目标的检测结果进行处理，拟合出危险边界。本系统选择用线性函数作为边界拟合的目标函数，理论上也可以使用二次函数进行拟合，来模拟一定的图像畸变效果。但实际测试表明，使用一次

线性函数进行拟合也足以满足检测需求，并且在算法逻辑和代码实现上更为简单直观。系统具体处理方式是提取画面中电缆槽检测结果的所有边界像素点，并使用 opencv 及其他图像学方法进行直线拟合，拟合得到的直线即为危险边界。

- **判断危险区方位：**在得到画面中的危险边界之后，对危险边界进行延长使之穿过整个画面，将画面分成两部分。鉴于井下煤矿光线条件差、摄像头画面内颜色信息极少，系统将图像转化成灰度图进行处理。通过对大量样本数据进行对比和测试，得出如下结论：安全区域的灰像素点数量/亮像素点数量 > 危险区域的灰像素点数量/亮像素点数量；安全区域的暗像素点数量/亮像素点数量 > 危险区域的暗像素点数量/亮像素点数量。因此，判断危险区方位时，我们主要使用危险边界两侧的像素灰度信息进行分析。

图 5-4: 远距离人员位于安全区域误报成危险

- **判断当前安全状态：**当前画面中没有工作人员或者当前画面中没有电缆槽时，系统默认为安全状态，当所有工作人员均位于安全区域时，也为安全状态，若任何一名工作人员出现在了危险区域内，就需要给出预警。系统从人员检测框下边缘中取一个点作为比较点，判断该点和危险边界的相对位置，若该点落在危险区域内则给出预警。但这里需要设计策略来选取这一比较点。当人员位于画面靠下方的位置时，离摄像头较近，此时检测框下边缘可能有一部分落在危险区但实际上人员并不在危险区内。而当人员位于画面靠上方的位置时，检测框的长宽都会非常小，危险边界稍有抖动可能就会导致人员状态从安全变成危险，图 5-4就是人员出现在远距离位置

的时候，系统给出错误的危险预警的示例。因此，合理策略是在人员检测结果的位置变化时，这一比较点也随之变化，这样就可以避免人员处于远处的安全区时误判成危险的情况。具体来说，系统只需要根据检测框在图像坐标系内的纵坐标来判定应当选择人员检测框下边缘的哪个像素点作为比较点即可。

上述内容描述了整个井下煤矿作业安全预警系统的工作流程，以及部分模块的设计细节，后文将对系统实际运行效果进行分析。

5.4 效果展示

前文详细介绍了井下煤矿作业安全预警系统的研发背景、需求分析、系统架构。本节将对系统实际运行时不同状态的效果进行展示。用户访问系统管理页面时触发模型加载工作，根据预设路径及参数信息导入训练好的目标检测模型，若模型文件不存在，用户可以调试后点击左侧按钮重新导入模型。模型导入成功后进度条如图5-5所示，此时右侧的视频画面展示模块为空白，等待用户接入摄像头数据。

图 5-5: 井下煤矿作业安全预警系统用户登录并加载模型后的界面

系统的数据输入需要用户线下完成摄像头设备的连接，摄像头接入后显示界面如图5-6所示，此时右侧视频画面显示的即为摄像头实时捕捉的画面。而系统主界面上“开始检测”的按钮从禁止触发状态变成待触发状态。若用户点

击该按钮，则会进入实时监测状态，对接下来的每一帧图像都进行人员检测和电缆槽检测，并将实时检测结果显示于用户界面。

图 5-6: 井下煤矿作业安全预警系统摄像头设备成功连接后的界面

用户启动监测之后，系统界面如图5-7所示。此时右侧视频画面展示模块显示内容不再是摄像头输入数据，而是摄像头输入数据经过目标检测及后处理之后的结果。画面中用绿色矩形检测框的形式显式标注了工作人员的位置，同时绘制出了电缆槽检测结果掩膜及拟合的危险边界。画面左上角使用了两个标签来表示监测信息，“left”标签表示当前画面中安全区位于危险边界左侧，这一标签可能会根据实际场景取不同值：left、right、below、above，表示相对于危险边界的哪一侧是安全区域。“safe”标签表示当前画面内工作人员所处的位置是安全的，而在工作人员越过危险边界进入危险区时，该标签会变成红色字体的“warning”。除此之外，用户界面也可以通过单选框的形式来选择展示哪些检测信息，可选项有：FPS、已检测时长、当前安全状态、安全区方向。用户可以通过主界面进度条左侧的“停止检测”按钮来终止检测。

除了图5-5、图5-6和图5-7中展示的几种用户界面工作状态，实际上系统运行时在各个阶段都会存在中间状态，涉及到灰度图、二值图、边缘提取、直线拟合等内容，但这些内容均是封装于系统内部的，不参与用户交互，因此不作为系统效果展示的一部分。

图 5-7: 井下煤矿作业安全预警系统实时监测界面

5.5 本章小结

本章主要介绍了实时视频目标检测后处理及结果平滑在工程系统中的应用，使用第四章提出的实时视频目标检测后处理及结果平滑框架 OPP-Smooth 搭建了一个井下煤矿作业安全预警系统，用于实时监测井下煤矿的作业环境。该系统的安全预警功能可以在很大程度上降低井下作业由于光线差、粉尘多导致的避障不及时这类安全风险。本章从研发背景出发，详细介绍了该系统的功能需求、性能需求，并给出相应的系统架构和模块设计，最后通过图片展示了其实际运行效果。当目标检测模型在较差的画面质量下出现检测错误或者漏检时，后处理的加入可以很好的利用时间信息来优化结果，降低错误率，并且也没有过多增加处理时长。该系统支持实时显示检测结果，具有很强的应用性，也充分说明了本文所提算法的实际应用价值。

第六章　总结与展望

本文主要着眼于视频目标检测这一任务，该任务与图像目标检测的主要区别在于，视频目标检测的输入数据是实时视频流或者视频文件，自然会涉及到视频数据特有的时间信息和上下文信息。若直接将图像目标检测器应用于视频数据，那么连续帧中往往会出现对目标的漏检，即连续帧中并非所有对象都可以被准确的检测到。除此之外，相邻两帧中检测到的同一目标也会存在类分数上的差异，进而由于这种类分数差异导致目标分类错误，即可能出现目标误检。最后，连续帧中对同一目标的检测框可能会不稳定，即摄像头静止不动时，由于像素信息差异或解码结果变化可能会引起检测框坐标发生变化。上述问题都是将图像目标检测器应用于视频数据会出现的情况，而本文则致力于使用后处理方案解决上述问题。

关于视频目标检测的主流解决方案，在本文第二章中也给出了介绍，目前基于光流的方法、基于 LSTM 的方法、基于跟踪的方法都被设计出来并运用于视频目标检测任务，但这些方法共同存在的问题即为：它们在解决视频目标检测任务的同时也引入了新的任务，比如光流计算、LSTM、跟踪模块，获得精度提升的同时，处理时长和整个任务的复杂度也相应增加。

对于视频目标检测任务与图像目标检测任务，图像目标检测随着多年的技术积累和卷积神经网络的引入，已经有相当数量的模型能够在精度、速度、模型体量上都取得很好的表现。如果能将优秀的图像目标检测模型合理运用于视频数据上，无疑是站在了巨人的肩膀上。而后处理方法正是一类通用策略，应用于图像目标检测器的结果输出，以改善视频目标检测性能。后处理方法的优化机制本质上是对图像目标检测模型处理视频数据时无法利用时间信息和上下文信息这一点进行补充，并且也没有额外引入新的计算任务，是一种非常高效的解决方案。

本文第三章提出的基于视频上下文信息的非实时后处理框架 CIBPP 属于非实时后处理方案，非实时就意味着该框架在处理视频片段中的每一帧时可以利用全局上下文信息。进而建立跨帧的长时检测框序列，并分别在序列内和序列间执行误检和漏检的优化操作。而第四章提出的实时视频目标检测后处理及结

果平滑框架 OPP-Smooth 则属于实时后处理方案，从时序的角度来看，每输入一帧图像，只能利用其前序帧的检测结果来辅助优化当前帧的检测结果，这也正是 OPP-Smooth 的核心设计思路。除此之外，实时后处理往往需要实时显示检测结果于用户界面，以供工作人员观测。因此 OPP-Smooth 也针对性的设计了滤波模块用于增强检测结果的稳定性。最后，本文第五章介绍了一个井下煤矿作业安全预警系统，通过实际工程应用证明了本文所提后处理框架的实际应用价值。

对于本文提出的非实时后处理框架 CIBPP 及实时后处理框架 OPP-Smooth，它们自身仍有可优化之处。对于非实时后处理框架 CIBPP，可以考虑设计成一个通用工具，以供其他用户使用任意一个图像目标检测模型时调用，这是偏向于工程应用这一方向的优化。而实时后处理框架 OPP-Smooth 中使用卡尔曼滤波对检测结果进行了降噪操作，使同一对象在连续帧内的结果更加稳定，关于检测结果稳定性这一点目前尚且没有通用指标可供衡量，这也是一个有待研究的问题。综上所述，本文提出的方法致力于在不增加复杂度的情况下将视频数据的时空信息融入图像目标检测结果中，但仍旧有需要研究和优化的内容。

参考文献

[1] LONG J, SHELHAMER E, DARRELL T. Fully Convolutional Networks for Semantic Segmentation[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015.

[2] KARPATHY A, TODERICI G, SHETTY S, et al. Large-scale Video Classification with Convolutional Neural Networks[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2014.

[3] GREGOR K, DANIHELKA I, GRAVES A, et al. DRAW: A Recurrent Neural Network For Image Generation[C/OL] // BACH F, BLEI D. Proceedings of Machine Learning Research, Vol 37 : Proceedings of the 32nd International Conference on Machine Learning. Lille, France : PMLR, 2015 : 1462 – 1471. https://proceedings.mlr.press/v37/gregor15.html.

[4] ZHANG X, ZHAO J, LECUN Y. Character-level Convolutional Networks for Text Classification[C/OL] // CORTES C, LAWRENCE N, LEE D, et al. Advances in Neural Information Processing Systems : Vol 28. [S.l.] : Curran Associates, Inc., 2015. https://proceedings.neurips.cc/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf.

[5] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient Estimation of Word Representations in Vector Space[C/OL] // . [S.l.] : arXiv, 2013. https://arxiv.org/abs/1301.3781.

[6] GRAVES A. Generating Sequences With Recurrent Neural Networks[C/OL] // . [S.l.] : arXiv, 2013. https://arxiv.org/abs/1308.0850.

[7] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is All you Need[C/OL] // GUYON I, LUXBURG U V, BENGIO S, et al. Advances in

Neural Information Processing Systems: Vol 30. [S.l.]: Curran Associates, Inc., 2017.
https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[8] DEVLIN J, CHANG M-W, LEE K, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding[C/OL] // . [S.l.]: arXiv, 2018.
https://arxiv.org/abs/1810.04805.

[9] GRAVES A, JAITLY N. Towards End-To-End Speech Recognition with Recurrent Neural Networks[C/OL] // XING E P, JEBARA T. Proceedings of Machine Learning Research, Vol 32: Proceedings of the 31st International Conference on Machine Learning. Bejing, China: PMLR, 2014: 1764–1772.
https://proceedings.mlr.press/v32/graves14.html.

[10] COHEN I, BERDUGO B. Speech enhancement for non-stationary noise environments[C/OL] // . 2001: 2403–2418.
https://www.sciencedirect.com/science/article/pii/S0165168401001281.

[11] MALAH D, COX R, ACCARDI A. Tracking speech-presence uncertainty to improve speech enhancement in non-stationary noise environments[C/OL] // 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258): Vol 2. 1999: 789–792 vol.2.
https://dx.doi.org/10.1109/ICASSP.1999.759789.

[12] DENG F, BAO C, KLEIJN W B. Sparse Hidden Markov Models for Speech Enhancement in Non-Stationary Noise Environments[J/OL]. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2015, 23(11): 1973–1987.
https://dx.doi.org/10.1109/TASLP.2015.2458585.

[13] CHENG H-T, KOC L, HARMSEN J, et al. Wide and Deep Learning for Recommender Systems[C/OL] // DLRS 2016: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. New York, NY, USA: Association for Computing Machinery, 2016: 7–10.
https://doi.org/10.1145/2988450.2988454.

[14] COVINGTON P, ADAMS J, SARGIN E. Deep Neural Networks for YouTube Recommendations[C/OL] // RecSys '16 : Proceedings of the 10th ACM Conference on Recommender Systems. New York, NY, USA : Association for Computing Machinery, 2016 : 191 – 198.
https://doi.org/10.1145/2959100.2959190.

[15] BARKAN O, KOENIGSTEIN N. ITEM2VEC: Neural item embedding for collaborative filtering[C/OL] // 2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP). 2016 : 1 – 6.
https://dx.doi.org/10.1109/MLSP.2016.7738886.

[16] LI C, LIU Z, WU M, et al. Multi-Interest Network with Dynamic Routing for Recommendation at Tmall[C/OL] // CIKM '19 : Proceedings of the 28th ACM International Conference on Information and Knowledge Management. New York, NY, USA : Association for Computing Machinery, 2019 : 2615 – 2623.
https://doi.org/10.1145/3357384.3357814.

[17] WEN R, TORKKOLA K, NARAYANASWAMY B, et al. A Multi-Horizon Quantile Recurrent Forecaster[C/OL] // . [S.l.] : arXiv, 2017.
https://arxiv.org/abs/1711.11053.

[18] KARIM F, MAJUMDAR S, DARABI H, et al. Multivariate LSTM-FCNs for time series classification[J/OL]. Neural Networks, 2019, 116 : 237 – 245.
https://www.sciencedirect.com/science/article/pii/S0893608019301200.

[19] KARIM F, MAJUMDAR S, DARABI H, et al. LSTM Fully Convolutional Networks for Time Series Classification[J/OL]. IEEE Access, 2018, 6 : 1662 – 1669.
https://dx.doi.org/10.1109/ACCESS.2017.2779939.

[20] AMASYALI K, EL-GOHARY N M. A review of data-driven building energy consumption prediction studies[J/OL]. Renewable and Sustainable Energy Reviews, 2018, 81 : 1192 – 1205.
https://www.sciencedirect.com/science/article/pii/S1364032117306093.

[21] GOODFELLOW I J, SHLENS J, SZEGEDY C. Explaining and Harnessing Adversarial Examples[C/OL] // . [S.l.] : arXiv, 2014.
https://arxiv.org/abs/1412.6572.

[22] CARLINI N, WAGNER D. Towards Evaluating the Robustness of Neural Networks[C/OL] // 2017 IEEE Symposium on Security and Privacy (SP). 2017 : 39 – 57.
https://dx.doi.org/10.1109/SP.2017.49.

[23] MADRY A, MAKELOV A, SCHMIDT L, et al. Towards Deep Learning Models Resistant to Adversarial Attacks[C/OL] // . [S.l.] : arXiv, 2017.
https://arxiv.org/abs/1706.06083.

[24] O'SHEA K, NASH R. An Introduction to Convolutional Neural Networks[C/OL] // . [S.l.] : arXiv, 2015.
https://arxiv.org/abs/1511.08458.

[25] DUBEY S R. A Decade Survey of Content Based Image Retrieval Using Deep Learning[J/OL]. IEEE Transactions on Circuits and Systems for Video Technology, 2022, 32(5) : 2687 – 2704.
https://dx.doi.org/10.1109/TCSVT.2021.3080920.

[26] ZOU Z, CHEN K, SHI Z, et al. Object Detection in 20 Years: A Survey[C/OL] // . [S.l.] : arXiv, 2019.
https://arxiv.org/abs/1905.05055.

[27] SZEGEDY C, LIU W, JIA Y, et al. Going Deeper With Convolutions[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015.

[28] YILMAZ A, JAVED O, SHAH M. Object Tracking: A Survey[J/OL]. ACM Comput. Surv., 2006, 38(4) : 13 – es.
https://doi.org/10.1145/1177352.1177355.

[29] MA Z, LIU S. A review of 3D reconstruction techniques in civil engineering and their applications[J/OL]. Advanced Engineering Informatics, 2018, 37 : 163 –

174.
https://www.sciencedirect.com/science/article/pii/S1474034617304275.

[30] IZADI S, KIM D, HILLIGES O, et al. KinectFusion: Real-Time 3D Reconstruction and Interaction Using a Moving Depth Camera[C/OL] // UIST '11 : Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology. New York, NY, USA : Association for Computing Machinery, 2011 : 559 – 568.
https://doi.org/10.1145/2047196.2047270.

[31] HEARST M, DUMAIS S, OSUNA E, et al. Support vector machines[J/OL]. IEEE Intelligent Systems and their Applications, 1998, 13(4) : 18 – 28.
https://dx.doi.org/10.1109/5254.708428.

[32] 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 (CVPR). 2014.

[33] NEUBECK A, VAN GOOL L. Efficient Non-Maximum Suppression[C/OL] // 18th International Conference on Pattern Recognition (ICPR'06) : Vol 3. 2006 : 850 – 855.
https://dx.doi.org/10.1109/ICPR.2006.479.

[34] VIOLA P, JONES M. Rapid object detection using a boosted cascade of simple features[C/OL] // Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001 : Vol 1. 2001 : I – I.
https://dx.doi.org/10.1109/CVPR.2001.990517.

[35] VIOLA P, JONES M J. Robust real-time face detection[J]. International journal of computer vision, 2004, 57 : 137 – 154.

[36] DALAL N, TRIGGS B. Histograms of oriented gradients for human detection[C] // 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05) : Vol 1. 2005 : 886 – 893.

[37] FELZENSZWALB P F, GIRSHICK R B, MCALLESTER D, et al. Object detection with discriminatively trained part-based models[J]. IEEE transactions on pattern analysis and machine intelligence, 2009, 32(9): 1627–1645.

[38] GIRSHICK R. Fast r-cnn[C] // Proceedings of the IEEE international conference on computer vision. 2015: 1440–1448.

[39] REN S, HE K, GIRSHICK R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[J]. Advances in neural information processing systems, 2015, 28.

[40] REDMON J, DIVVALA S, GIRSHICK R, et al. You Only Look Once: Unified, Real-Time Object Detection[C/OL] // . [S.l.]: arXiv, 2015. https://arxiv.org/abs/1506.02640.

[41] REDMON J, FARHADI A. YOLO9000: better, faster, stronger[C] // Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263–7271.

[42] REDMON J, FARHADI A. YOLOv3: An Incremental Improvement[C/OL] // . [S.l.]: arXiv, 2018. https://arxiv.org/abs/1804.02767.

[43] BOCHKOVSKIY A, WANG C-Y, LIAO H-Y M. YOLOv4: Optimal Speed and Accuracy of Object Detection[C/OL] // . [S.l.]: arXiv, 2020. https://arxiv.org/abs/2004.10934.

[44] G. L D. Distinctive Image Features from Scale-Invariant Keypoints[J/OL]. International Journal of Computer Vision, 2004. https://cir.nii.ac.jp/crid/1574231875443569408.

[45] Wei Anguelov LIU D E D S C R S F C-Y B A C. SSD: Single Shot MultiBox Detector[C] // Bastian Matas LEIBE J S N W M. Computer Vision – ECCV 2016. Cham: Springer International Publishing, 2016: 21–37.

[46] J R Rvan de Sande UIJLINGS K E A T A W M. Selective Search for Object Recognition[J/OL]. International Journal of Computer Vision, 2013. https://doi.org/10.1007/s11263-013-0620-5.

[47] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet Classification with Deep Convolutional Neural Networks[J/OL]. Commun. ACM, 2017, 60(6): 84–90.
https://doi.org/10.1145/3065386.

[48] Matthew D Fergus ZEILER R. Visualizing and Understanding Convolutional Networks[C] // David Pajdla FLEET T S B T T. Computer Vision – ECCV 2014. Cham: Springer International Publishing, 2014: 818–833.

[49] BROWN P F, DELLA PIETRA V J, DESOUZA P V, et al. Class-based n-gram models of natural language[J]. Computational linguistics, 1992, 18(4): 467–480.

[50] MORIN F, BENGIO Y. Hierarchical probabilistic neural network language model[C] // International workshop on artificial intelligence and statistics. 2005: 246–252.

[51] MACQUEEN J. Some methods for classification and analysis of multivariate observations[C] // Proc. 5th Berkeley Symposium on Math., Stat., and Prob. 1965: 281.

[52] HARTIGAN J A, WONG M A. Algorithm AS 136: A K-Means Clustering Algorithm[J/OL]. Journal of the Royal Statistical Society. Series C (Applied Statistics), 1979, 28(1): 100–108.
http://www.jstor.org/stable/2346830.

[53] LIKAS A, VLASSIS N, VERBEEK J J. The global k-means clustering algorithm[J]. Pattern recognition, 2003, 36(2): 451–461.

[54] HE K, ZHANG X, REN S, et al. Deep Residual Learning for Image Recognition[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016.

[55] WU Z, SHEN C, VAN DEN HENGEL A. Wider or deeper: Revisiting the resnet model for visual recognition[J]. Pattern Recognition, 2019, 90: 119–133.

[56] LIN T 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 (CVPR). 2017.

[57] LI G, XIE Y, WEI T, et al. Flow guided recurrent neural encoder for video salient object detection[C] // Proceedings of the IEEE conference on computer vision and pattern recognition. 2018 : 3243 – 3252.

[58] ZHU X, WANG Y, DAI J, et al. Flow-Guided Feature Aggregation for Video Object Detection[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017.

[59] ZHU X, XIONG Y, DAI J, et al. Deep Feature Flow for Video Recognition[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017.

[60] HOCHREITER S, SCHMIDHUBER J. Long Short-Term Memory[J/OL]. Neural Computation, 1997, 9(8) : 1735 – 1780. https://dx.doi.org/10.1162/neco.1997.9.8.1735.

[61] ZHANG K, CHAO W-L, SHA F, et al. Video summarization with long short-term memory[C] // Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14. 2016 : 766 – 782.

[62] KANG K, LI H, XIAO T, et al. Object detection in videos with tubelet proposal networks[C] // Proceedings of the IEEE conference on computer vision and pattern recognition. 2017 : 727 – 735.

[63] HAN W, KHORRAMI P, PAINE T L, et al. Seq-NMS for Video Object Detection[C/OL] // . [S.l.] : arXiv, 2016. https://arxiv.org/abs/1602.08465.

[64] KANG K, LI H, YAN J, et al. T-CNN: Tubelets With Convolutional Neural Networks for Object Detection From Videos[J/OL]. IEEE Transactions on Circuits and Systems for Video Technology, 2018, 28(10) : 2896 – 2907. https://dx.doi.org/10.1109/TCSVT.2017.2736553.

[65] BELHASSEN H, ZHANG H, FRESSE V, et al. Improving Video Object Detection by Seq-Bbox Matching.[C] // VISIGRAPP (5: VISAPP). 2019 : 226 – 233.

[66] RAGLAND K, THARCIS P. A survey on object detection, classification and tracking methods[J]. Int. J. Eng. Res. Technol, 2014, 3(11) : 622 – 628.

[67] DEDEOĞLU Y. Moving object detection, tracking and classification for smart video surveillance[D]. [S.l.] : Bilkent Universitesi (Turkey), 2004.

[68] LUO H, XIE W, WANG X, et al. Detect or track: Towards cost-effective video object detection/tracking[C] // Proceedings of the AAAI Conference on Artificial Intelligence : Vol 33. 2019 : 8803 – 8810.

[69] DASIOPOULOU S, MEZARIS V, KOMPATSIARIS I, et al. Knowledge-assisted semantic video object detection[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2005, 15(10) : 1210 – 1224.

[70] WU H, CHEN Y, WANG N, et al. Sequence level semantics aggregation for video object detection[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019 : 9217 – 9225.

[71] WANG S, ZHOU Y, YAN J, et al. Fully motion-aware network for video object detection[C] // Proceedings of the European conference on computer vision (ECCV). 2018 : 542 – 557.

[72] BERTASIUS G, TORRESANI L, SHI J. Object detection in video with spatiotemporal sampling networks[C] // Proceedings of the European Conference on Computer Vision (ECCV). 2018 : 331 – 346.

[73] DENG J, PAN Y, YAO T, et al. Relation distillation networks for video object detection[C] // Proceedings of the IEEE/CVF international conference on computer vision. 2019 : 7023 – 7032.

[74] JIAO L, ZHANG R, LIU F, et al. New Generation Deep Learning for Video Object Detection: A Survey[J/OL]. IEEE Transactions on Neural Networks and Learning Systems, 2022, 33(8) : 3195 – 3215. https://dx.doi.org/10.1109/TNNLS.2021.3053249.

[75] GIBSON J J. The perception of the visual world.[J], 1950.

[76] DOSOVITSKIY A, FISCHER P, ILG E, et al. FlowNet: Learning Optical Flow With Convolutional Networks[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2015.

[77] SHI X, CHEN Z, WANG H, et al. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting[C/OL] // CORTES C, LAWRENCE N, LEE D, et al. Advances in Neural Information Processing Systems : Vol 28. [S.l.] : Curran Associates, Inc., 2015. https://proceedings.neurips.cc/paper/2015/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf.

[78] LU Y, LU C, TANG C-K. Online Video Object Detection Using Association LSTM[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017.

[79] XIAO F, LEE Y J. Video Object Detection with an Aligned Spatial-Temporal Memory[C] // Proceedings of the European Conference on Computer Vision (ECCV). 2018.

[80] MAO H, KONG T, DALLY B. CaTDet: Cascaded Tracked Detector for Efficient Object Detection from Video[C/OL] // TALWALKAR A, SMITH V, ZAHARIA M. Proceedings of Machine Learning and Systems : Vol 1. 2019 : 201 – 211. https://proceedings.mlsys.org/paper/2019/file/698d51a19d8a121ce581499d7b701668-Paper.pdf.

[81] FEICHTENHOFER C, PINZ A, ZISSERMAN A. Detect to Track and Track to Detect[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017.

[82] DAI J, LI Y, HE K, et al. R-FCN: Object Detection via Region-based Fully Convolutional Networks[C/OL] // LEE D, SUGIYAMA M, LUXBURG U, et al. Advances in Neural Information Processing Systems : Vol 29. [S.l.] : Curran Associates, Inc., 2016.

https://proceedings.neurips.cc/paper/2016/file/577ef1154f3240ad5b9b413aa7346a1e-Paper.pdf.

[83] CHIN T-W, DING R, MARCULESCU D. AdaScale: Towards Real-time Video Object Detection using Adaptive Scaling[C/OL] // TALWALKAR A, SMITH V, ZAHARIA M. Proceedings of Machine Learning and Systems : Vol 1. 2019 : 431 – 441.
https://proceedings.mlsys.org/paper/2019/file/b1d10e7bafa4421218a51b1e1f1b0ba2-Paper.pdf.

[84] MESULAM M-M. Large-scale neurocognitive networks and distributed processing for attention, language, and memory[J]. Annals of Neurology: Official Journal of the American Neurological Association and the Child Neurology Society, 1990, 28(5) : 597 – 613.

[85] GALASSI A, LIPPI M, TORRONI P. Attention in natural language processing[J]. IEEE transactions on neural networks and learning systems, 2020, 32(10) : 4291 – 4308.

[86] KUMAR A, IRSOY O, ONDRUSKA P, et al. Ask me anything: Dynamic memory networks for natural language processing[C] // International conference on machine learning. 2016 : 1378 – 1387.

[87] BAHULEYAN H, MOU L, VECHTOMOVA O, et al. Variational attention for sequence-to-sequence models[J]. arXiv preprint arXiv:1712.08207, 2017.

[88] DENG J, PAN Y, YAO T, et al. Relation Distillation Networks for Video Object Detection[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2019.

[89] WU H, CHEN Y, WANG N, et al. Sequence Level Semantics Aggregation for Video Object Detection[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2019.

[90] WU H, CHEN Y, WANG N, et al. Sequence Level Semantics Aggregation for Video Object Detection[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2019.

[91] CHEN Y, CAO Y, HU H, et al. Memory Enhanced Global-Local Aggregation for Video Object Detection[C] // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2020.

[92] GUO C, FAN B, GU J, et al. Progressive Sparse Local Attention for Video Object Detection[C] // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 2019.

[93] Dan Revaud ONEATA J V J S C. Spatio-temporal Object Detection Proposals[C] // David Pajdla FLEET T S B T T. Computer Vision – ECCV 2014. Cham : Springer International Publishing, 2014 : 737 – 752.

[94] OUYANG W, WANG X, ZENG X, et al. DeepID-Net: Deformable Deep Convolutional Neural Networks for Object Detection[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015.

[95] BAEK Y, LEE B, HAN D, et al. Character Region Awareness for Text Detection[C] // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019.

[96] LI Z, JIA B, HE Y, et al. PBDE: an effective post-processing method based on box density for object detection[J]. Applied Intelligence, 2022 : 1 – 12.

[97] BODLA N, SINGH B, CHELLAPPA R, et al. Soft-NMS – Improving Object Detection With One Line of Code[C] // Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017.

[98] SABATER A, MONTESANO L, MURILLO A C. Robust and efficient post-processing for video object detection[C/OL] // 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2020 : 10536 – 10542. https://dx.doi.org/10.1109/IROS45743.2020.9341600.

[99] SCHROFF F, KALENICHENKO D, PHILBIN J. FaceNet: A Unified Embedding for Face Recognition and Clustering[C] // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015.

[100] REZATOFIGHI H, TSOI N, GWAK J, et al. Generalized Intersection over Union A Metric and A Loss for Bounding Box Regression[C/OL] // . [S.l.]: arXiv, 2019.
https://arxiv.org/abs/1902.09630.

[101] WEINBERGER K Q, BLITZER J, SAUL L. Distance Metric Learning for Large Margin Nearest Neighbor Classification[C/OL] // WEISS Y, SCHöLKOPF B, PLATT J. Advances in Neural Information Processing Systems: Vol 18. [S.l.]: MIT Press, 2005.
https://proceedings.neurips.cc/paper/2005/file/a7f592cef8b130a6967a90617db5681b-Paper.pdf.

[102] KALMAN R E. A New Approach to Linear Filtering and Prediction Problems[J/OL]. Journal of Basic Engineering, 1960, 82(1): 35 – 45.
https://doi.org/10.1115/1.3662552.

[103] WELCH G, BISHOP G, OTHERS. An introduction to the Kalman filter[J], 1995.

致　　谢

随着毕业论文的最后落笔，三年的研究生生活也即将画上句号。在这三年中，我经历过许多低谷的时期，幸有身边人的关心和鼓励，才能走出低谷并一点点成长。而对于那些值得纪念的珍贵时刻，我始终铭记于心。回想三年来的生活点滴，以及此次毕业设计的相关工作，我始终对给予我帮助的人心怀感激，并向他们致以最真诚的感谢。

首先，我要感谢我的导师申富饶教授。为人师表，言传身教，导师的言行举止总是值得我去学习，耐心的教导也让我受益良多。在我整个硕士研究生生涯中，申老师一直给予我悉心的指导和关心，为我提供了许多有价值的建议。导师在研究中的严谨治学精神，让我深受启发，而导师的支持和鼓励，让我在困难和挫折面前不灰心丧气，始终保持了对研究工作的热情。

其次，我要感谢 RINC 研究组的同学们，他们是我在学术道路上的战友和伙伴。我们相互鼓励、相互学习、共同进步。在我的学术研究中，他们给予了我很多支持和帮助，从研究设计、数据分析到论文写作等多个方面都给了我很多想法和思路。也十分荣幸作为 RINC 研究组的一员，陪大家一同度过三年的研究生生涯。

最后，我要感谢我的家人和朋友们。正是他们对我的支持和鼓励，让我在这条学术之路上坚定了自己的信念和方向。在我完成研究工作的过程中，他们一直在我的身边，给予我精神上的支持和关心，为我提供了无私的帮助，始终作为我最坚实的后盾，希望自己通过日后的努力成为家人的骄傲。

在此，我再次对所有支持我的人表示衷心的感谢。以后的人生旅程，定当不负恩师栽培，不负父母养育，不忘同窗情谊，不惧前方险途。

简历与科研成果

基本信息

管俣祺，女，汉族，1999 年 7 月出生，安徽宣城人。

教育背景

2020 年 9 月 — 2023 年 6 月 南京大学计算机科学与技术系 硕士
2016 年 9 月 — 2020 年 6 月 北京工业大学计算机科学与技术系 本科

攻读硕士学位期间完成的学术成果

1. 申富饶，管俣祺，李金桥。《一种港口码头辅助障碍物过滤的快速车道线检测方法》（专利申请号 202111282391.7）

攻读硕士学位期间参与的科研课题

1. 国家自然科学基金面上项目 “基于深度感知增量式联想记忆神经网络的信息融合系统研究”（项目编号 61876076，课题年限 2019 年 1 月 — 2022 年 12 月），负责目标检测相关问题的研究。

版权及论文原创性说明

任何收存和保管本论文的单位和个人，未经作者本人授权，不得将本论文转借他人并复印、抄录、拍照或以任何方式传播，否则，引起有碍作者著作权益的问题，将可能承担法律责任。

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含其他个人或集体已经发表或撰写的作品成果。本文所引用的重要文献，均已在文中以明确方式标明。本声明的法律结果由本人承担。

作者签名：_____

_____年_____月_____日

《学位论文出版授权书》

本人完全同意《中国优秀博硕士学位论文全文数据库出版章程》（以下简称“章程”），愿意将本人的学位论文提交“中国学术期刊（光盘版）电子杂志社”在《中国博士学位论文全文数据库》、《中国优秀硕士学位论文全文数据库》中全文发表。《中国博士学位论文全文数据库》、《中国优秀硕士学位论文全文数据库》可以以电子、网络及其他数字媒体形式公开出版，并同意编入《中国知识资源总库》，在《中国博硕士学位论文评价数据库》中使用和在互联网上传播，同意按“章程”规定享受相关权益。

作者签名：_____

_____年_____月_____日

论文题名	基于上下文信息的视频目标检测后处理研究				
研究生学号	MF20330024	所在院系	计算机科学与技术系	学位年度	2023
论文级别	☐ 学术学位硕士　☑ 专业学位硕士 ☐ 学术学位博士　☐ 专业学位博士 （请在方框内画钩）				
作者 Email	MF20330024@smail.nju.edu.cn				
导师姓名	申富饶 教授				

论文涉密情况：

☑ 不保密

☐ 保密，保密期 (_____年_____月_____日 至 _____年_____月_____日)

注：请将该授权书填写后装订在学位论文最后一页（南大封面）。