西北大学
Northwest University

轻量化目标检测系统的设计与实现

姓名: 薄劲阳　　学号: 2020115025

专业: 计算机科学与技术

导师: 卜起荣

答辩日期: 2024 年 5 月 21 日

目录

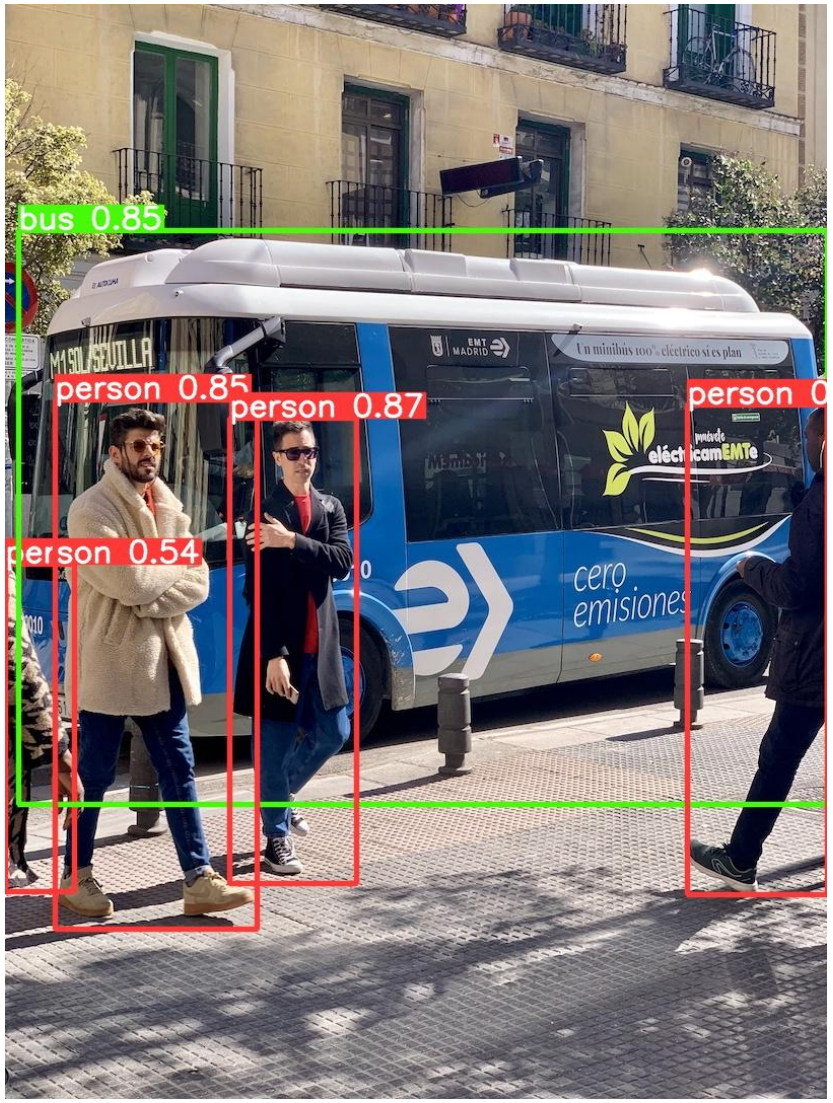
CONTENTS

PART ONE

研究背景及意义

题目：轻量化目标检测平台的设计与实现

传统的目标检测算法通常要求较大的模型大小和高计算复杂度，这会对计算和存储资源造成较大的负担。轻量化目标检测算法通过优化网络结构、减少参数数量以及采用高效的计算方式，可以大幅降低算法对资源的需求。



目标检测算法现状：

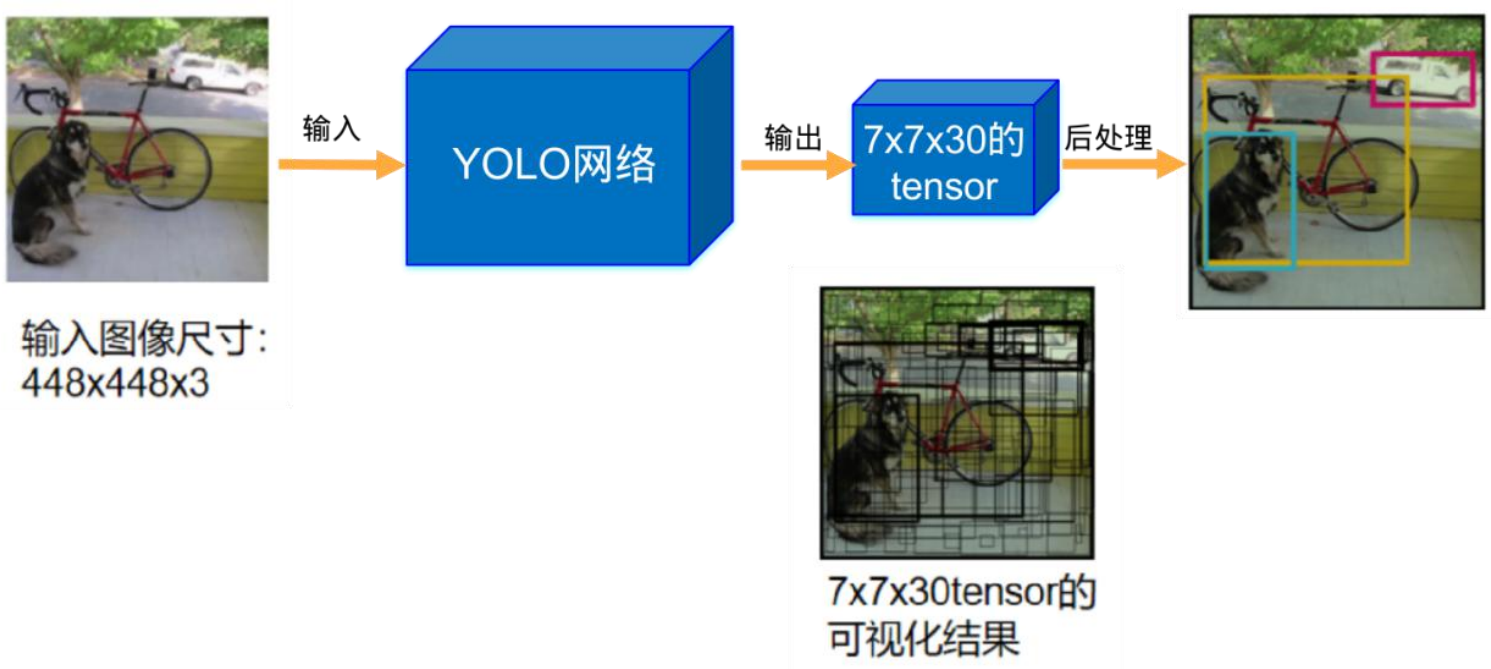
模型参数多

高计算复杂度

难以部署到嵌入式边缘设备

PART TWO

YOLO基本原理及网络架构

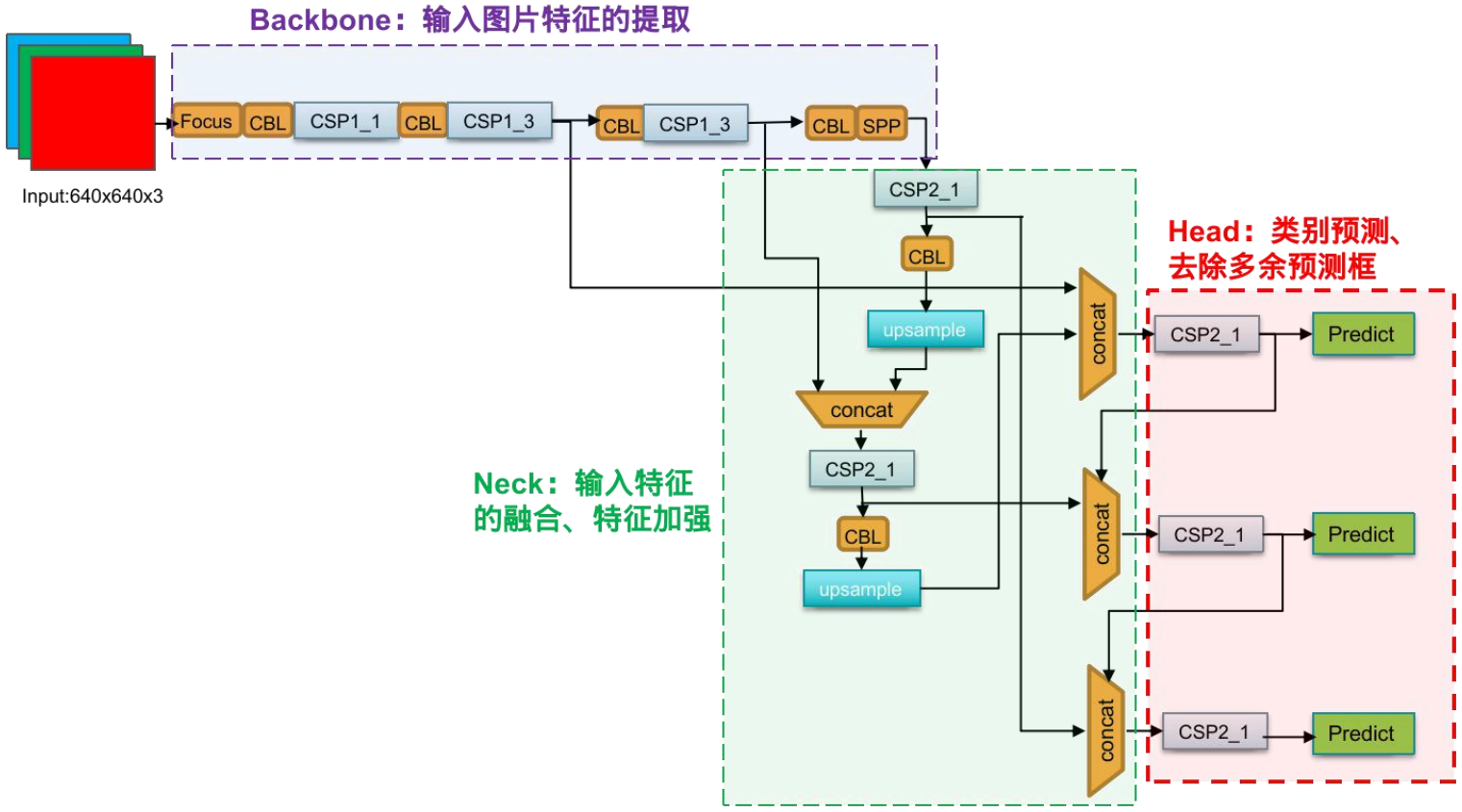


YOLO基本原理

- 将图片输入至网络模型
- 模型产生一个tensor张量，该张量包含了物体的空间位置和各个类别的概率。
- 执行后处理步骤，应用非极大值抑制（NMS）算法，去除多余的候选框，最终确定图像中物体的具体位置和类型。

PART TWO

YOLO基本原理及网络架构

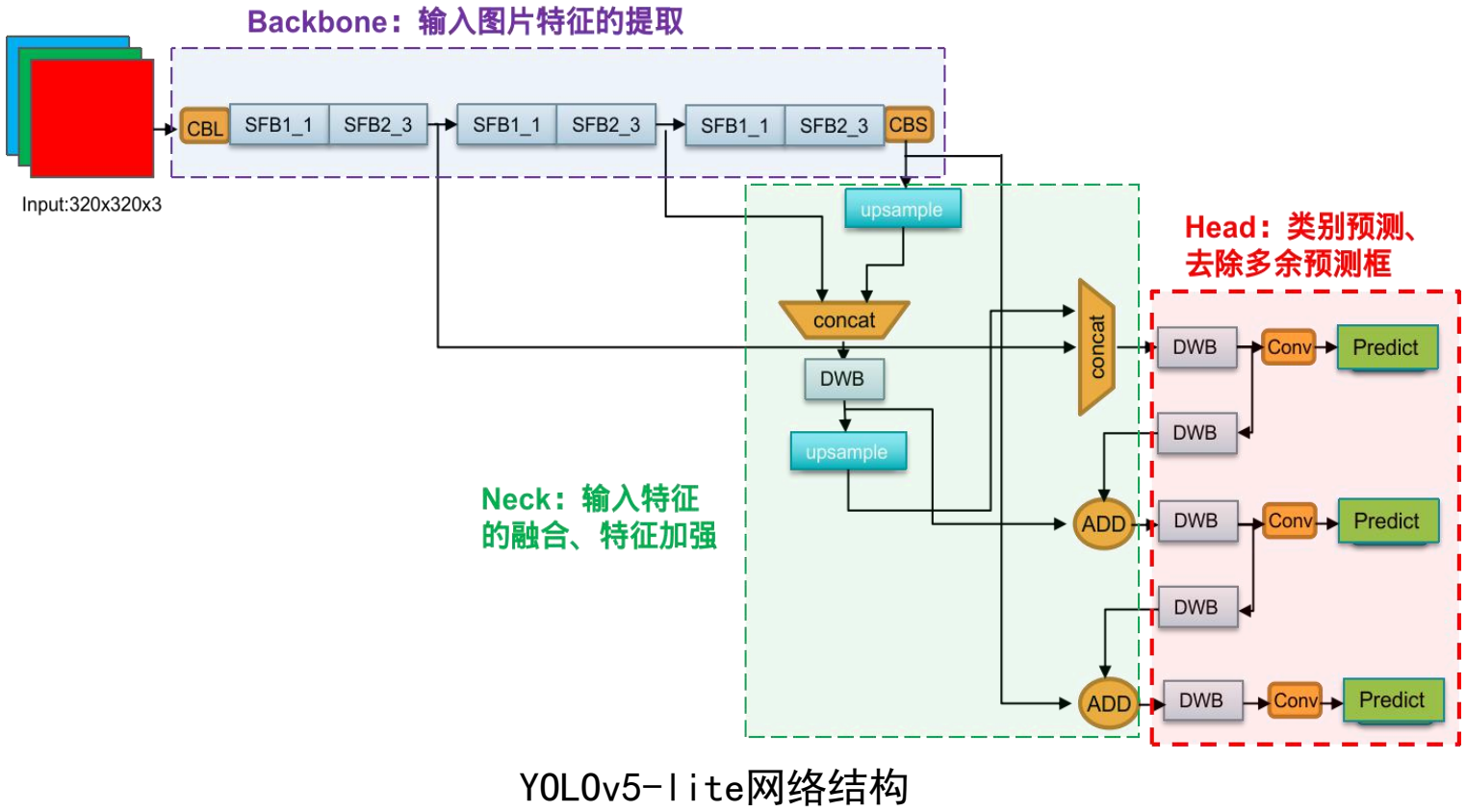


YOLOv5网络结构

YOLOv5网络结构

- Backbone结构：核心部分，负责从输入图像中提取关键特征。由Focus、CBL以及CSP组成。Focus模块负责对图像进行Slice切片操作，将高维图像转换为低维图像（会导致增加通道数）。CBL模块是由Conv卷积模块、BatchNorm归一化处理模块、Leaky ReLU激活函数模块组成。CSP由CBL、Res Unit、Conv构成。

YOLO基本原理及网络架构

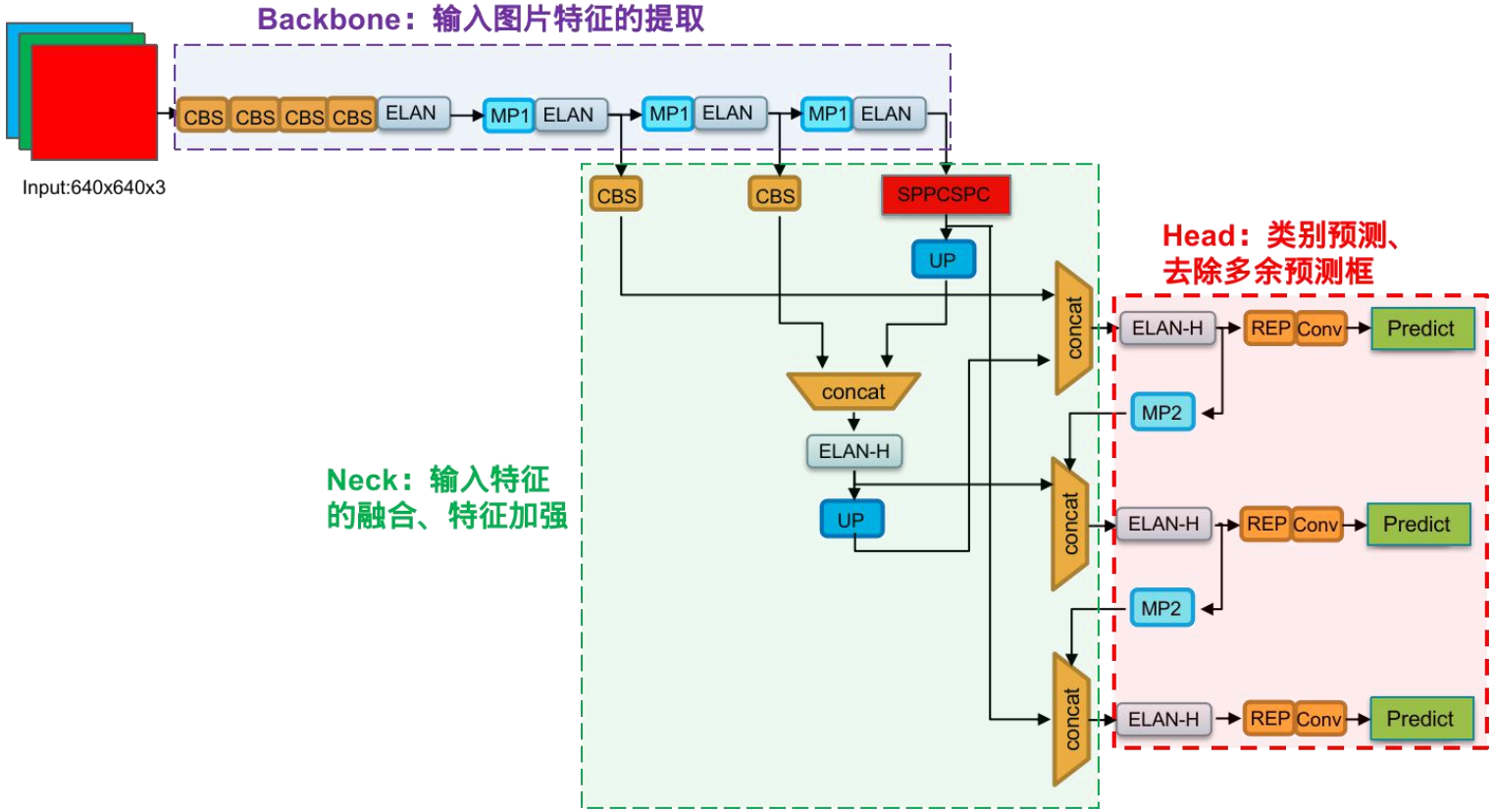


YOLOv5-lite网络结构

YOLOv5-lite网络结构

- Backbone结构：由CBL、ShuffleNet结构构成。相比于YOLOv5，去除了Focus层。另外在提取特征模块，使用ShuffleNet替代CSP模块。相比于CSP模块，ShuffleNet减少了缓存的使用。

YOLO基本原理及网络架构



YOLOv7网络结构

YOLOv7网络结构

- Backbone结构：YOLOv7的Backbone结构由CBS、ELAN、MP1模块组成。CBS模块由Conv卷积层、BatchNorm归一化层、SiLU激活函数。ELAN则是由5个连续的CBS模块组成。MP1是负责最大池化的模块。模块中存在大量并行的网络结构。

PART THREE

模型对比实验及结论

实验环境

训练好的模型分别在PC端和树莓派上运行

PC端平台配置

配置环境	版本型号
操作系统	Ubuntu 22.04.4 LTS
语言	Python 3.8
CPU	AMD® Ryzen 9 7945hx
GPU	Nvidia GeForce RTX 4060
内存	16.0GB

树莓派平台配置

配置环境	版本型号
操作系统	Raspberry Pi OS
语言	Python 3.8
CPU	BCM2711 SoC
内存	4GB

实验数据集

训练数据集采用Fruits detection数据集，

测试数据选择了一段时长58s的水果视频。

Fruits detection 数据集

水果视频

模型对比实验及结论

评测指标

- **FPS**：表示模型网络在一秒钟内能够完成处理的图像张数。目标检测处理一张图片的时间，从图片处理处开始计时，到推理结束停止计时。

$$FPS = \frac{1s}{processing\ time\ per\ frame}$$

- **参数量Params**：表示网络结构的总参数量。参数量越多，模型越复杂。
- **模型占用内存**：表明模型所占用的存储空间大小。

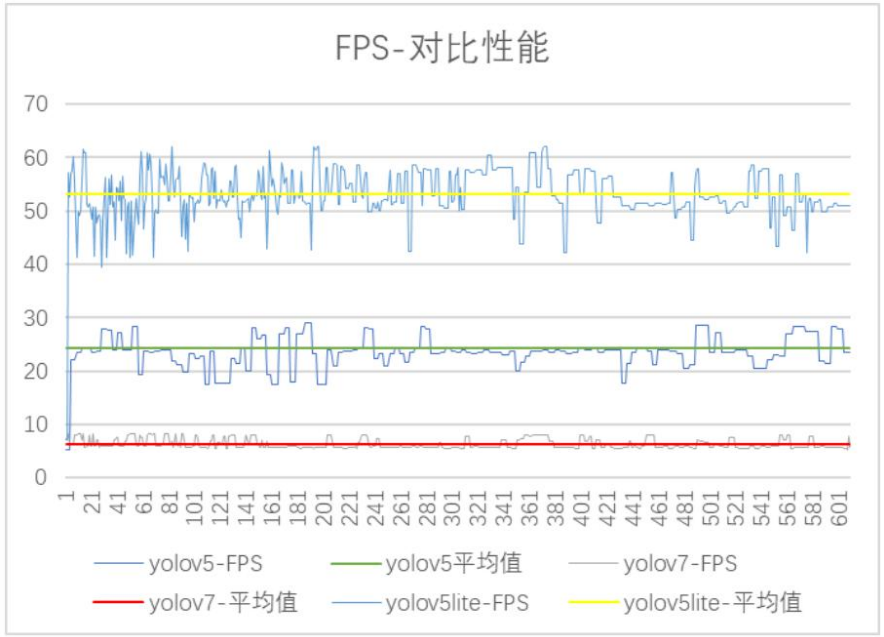
PART THREE

模型对比实验及结论

实验结果

YOLOv5、YOLOv5-lite、YOLOv7在PC端的FPS对比表

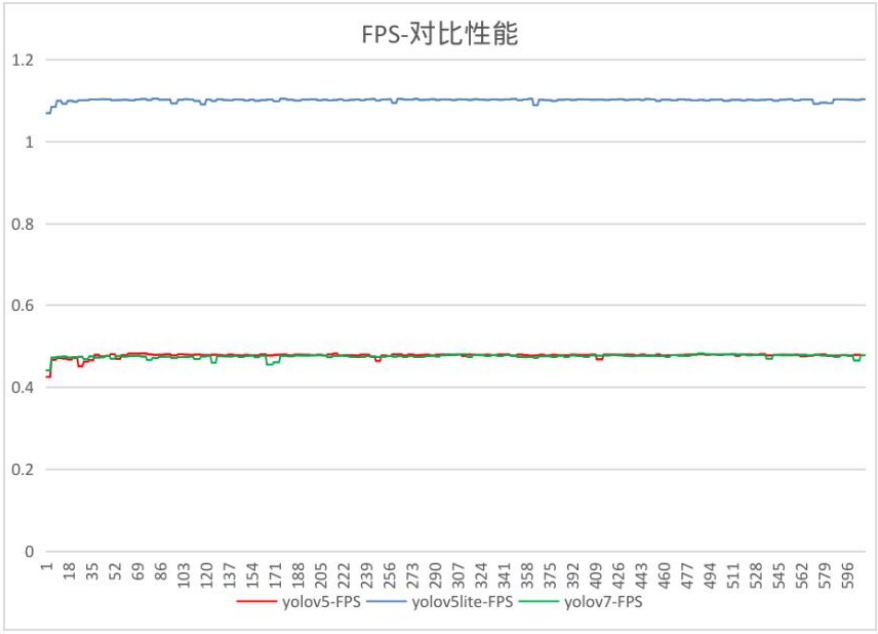
目标检测模型	FPS平均值
YOLOv5	24.325
YOLOv5-lite	53.106
YOLOv7	6.330



YOLOv5、YOLOv5-lite、YOLOv7 FPS 检测结果图(PC 端)

YOLOv5、YOLOv5-lite、YOLOv7在树莓派的FPS对比表

目标检测模型	FPS平均值
YOLOv5	0.478
YOLOv5-lite	1.102
YOLOv7	0.477



YOLOv5、YOLOv5-lite、YOLOv7 FPS 检测结果图(树莓派)

PART THREE

模型对比实验及结论

实验结果

YOLOv5、YOLOv5-lite、YOLOv7的参数量对比

目标检测模型	参数量Params
YOLOv5	725885
YOLOv5-lite	1649853
YOLOv7	37622682

YOLOv5、YOLOv5-lite、YOLOv7的模型占用内存大小对比

目标检测模型	参数量Params
YOLOv5	28.6MB
YOLOv5-lite	6.2MB
YOLOv7	146.7MB

PART THREE

模型对比实验及结论

实验结果分析

- FPS
  - PC端：YOLOv5-lite > YOLOv5 > YOLOv7
  - 树莓派：YOLOv5-lite > YOLOv5 ≈ YOLOv7
- 参数量Params：YOLOv5-lite < YOLOv5 < YOLOv7
- 模型占用内存：YOLOv5-lite < YOLOv5 < YOLOv7

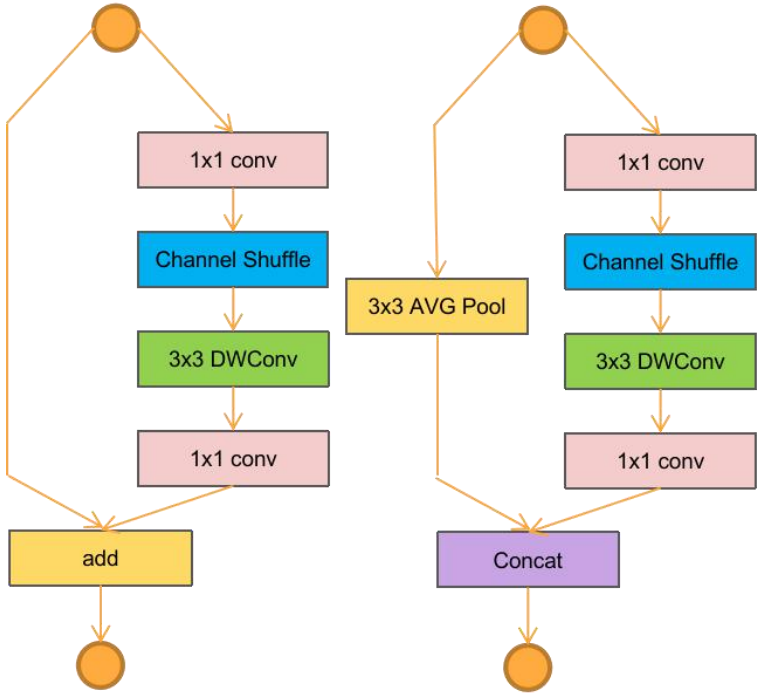
轻量化的四个原则：Shufflenet v2论文从内存访问代价（Memory Access Cost，MAC）和GPU并行性的方向分析了网络应该怎么设计才能进一步减少运行时间，减少存储器开销。所以轻量化的思路就是要减少存储器的使用，增加网络的并行度。

1. **同等通道大小下能够最小化内存访问量**，即当输入通道数和输出通道数相同时，MAC最小；
2. **过度使用组卷积会提高MAC**；
3. **网络过于碎片化（特别是多路）会降低并行度**；因为更多的分支需要更多的卷积核加载和同步操作。
4. **不能忽略元素级操作（比如Add操作）**；

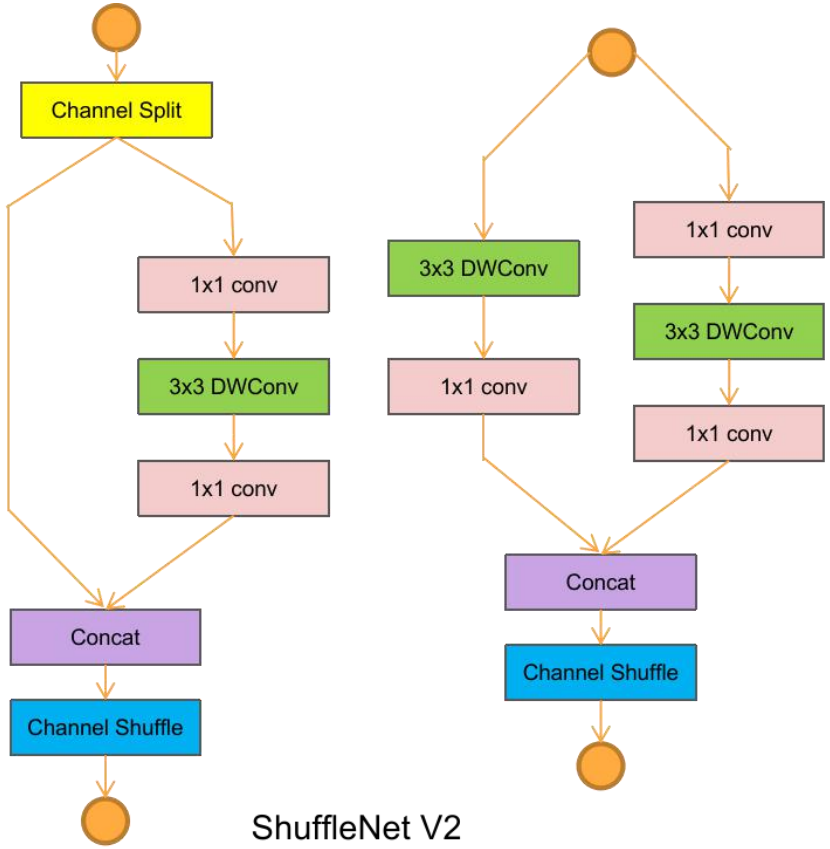
PART THREE

模型对比实验及结论

实验结果分析



ShuffleNet V1



ShuffleNet V2

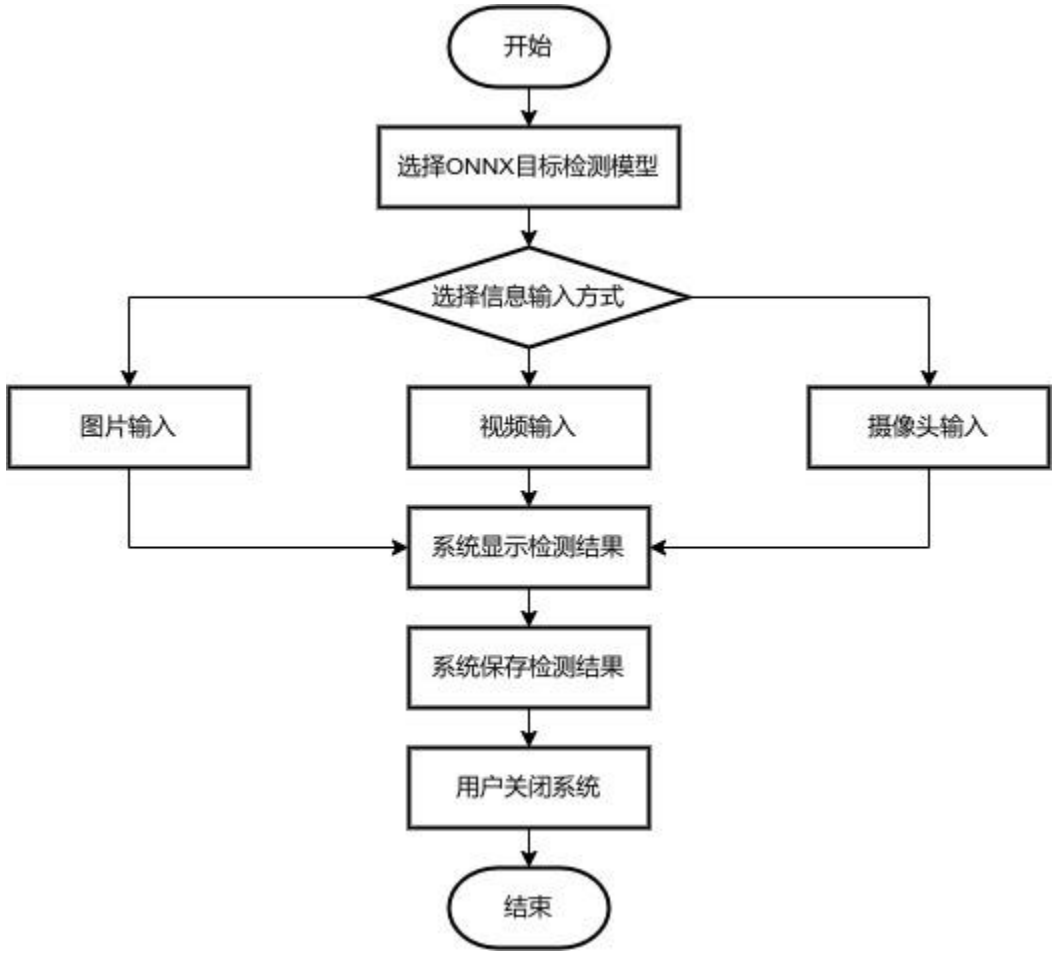
YOLOv5-lite采用ShuffleNet V2作为Backbone的结构。

- ShuffleNetV2采用了**Channel Split通道分离操作**。Channel Split只是将通道分成两组，而不像Slice操作增加了通道数。这样ShuffleNet并没有因过量使用组卷积增加MAC。
- 此外最后**使用concat操作**，而没有使用Tensor Add操作，这也提升了运行速度。

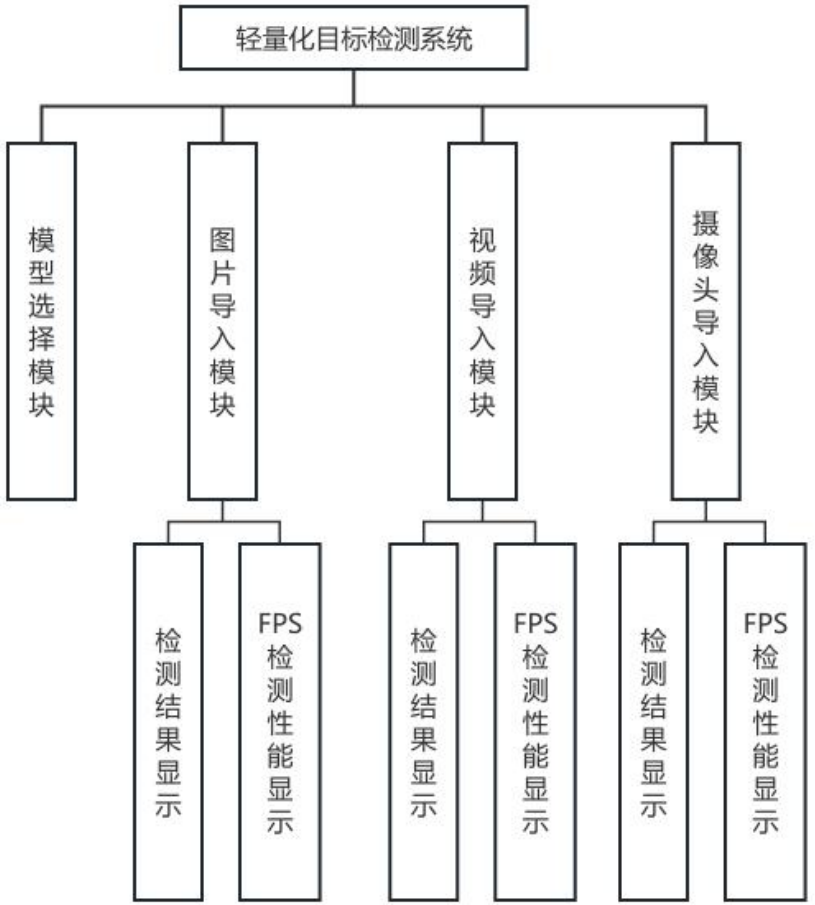
PART FOUR

成果展示

系统流程图



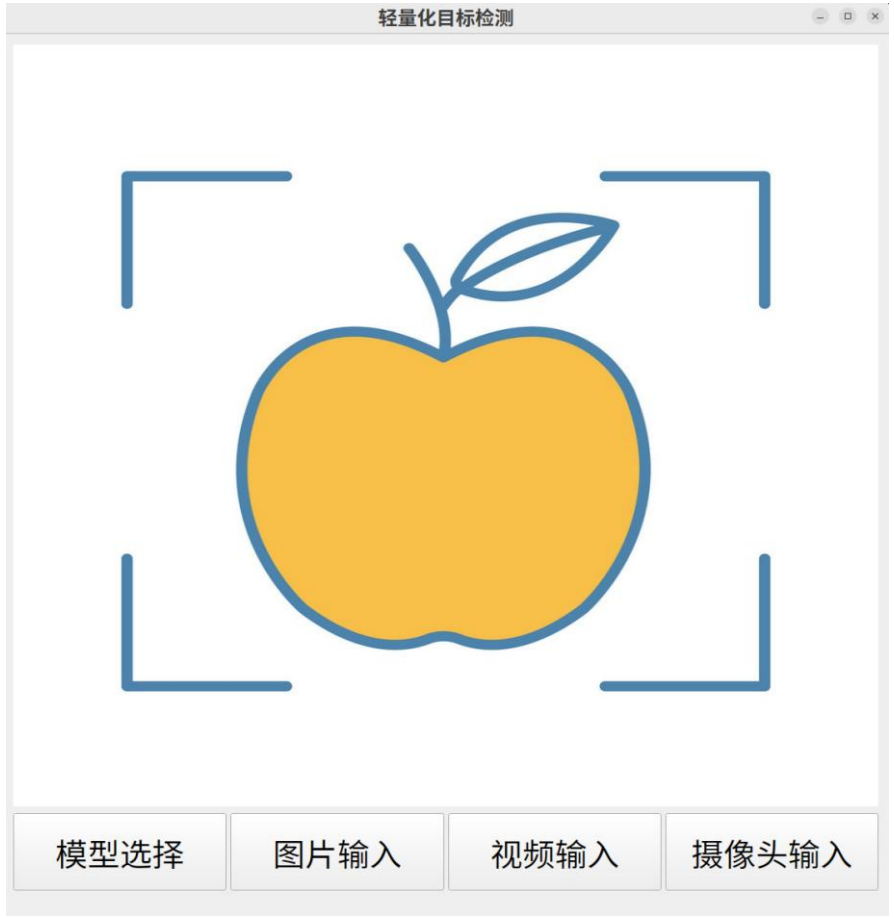
功能模块图



PART FOUR

成果展示

主界面



检测展示

PART FIVE

总结与展望

1. **本系统在树莓派上运行速度较慢**。由于PyQt界面的使用，在树莓派这种资源有限的设备上运行，界面运行速度很慢。因此，如何构建轻量化的界面，或者远程连接树莓派运行目标检测模型，将检测结果返回给本地终端，需要后续的改进。
2. **本系统支持的模型较少，只适用YOLOv5、YOLOv5-lite、YOLOv7**。对于其他目标检测模型，效果支持并不好。如何解决多种模型的选择，以及针对不同模型，编写对应的推理算法，有待后续的研究。
3. **本系统并没有做出算法创新，只是对现有模型的复现和应用**。如何在现有模型的基础上，对模型进行改进，后续需要继续深入研究。；

感谢观赏

THANKS FOR WATCHING