



廣東藥科學大學
GUANGDONG PHARMACEUTICAL UNIVERSITY

医学图像处理 课程设计报告

题目 超声图像中孕囊区域的自动检测

班级: 电子信息工程 221、222

姓名: 张海涛 张卓权 李博宇

学号: 2230506158 2230506162 2230506222

2025 年 6 月 5 日

摘要

随着医疗影像技术的发展，精准检测超声图像中的孕囊区域对早期妊娠评估至关重要。本研究旨在提升孕囊检测的准确性和效率，为此提出一种基于深度学习的检测方案，利用卷积神经网络对超声图像进行特征提取与分析，经大量数据训练优化模型参数。实验表明，该方法在多种超声图像数据集上均能有效检测出孕囊区域，检测准确率较传统方法有显著提高，平均准确率大约为 99.5%，召回率大约为 100%，为临床诊断提供了有力支持。未来，将继续优化算法，以适应更多复杂场景下的孕囊检测需求，进一步推动妇产科超声诊断技术的发展。

关键词： 超声图像；孕囊检测；深度学习；医疗影像分析

目 录

摘 要.....	1
第 1 章 绪 论.....	3
1.1 课题研究的背景和来源.....	3
1.2 课题研究成果应用的实际意义.....	3
1.3 相关方法综述.....	4
第 2 章 设计方案.....	6
2.1 YOLOv8 算法简介.....	6
2.2 Ultralytics 核心.....	7
2.3 软件式的模型训练方法.....	8
第 3 章 实验过程及结果分析.....	10
第 4 章 结论与展望.....	14
参考文献	23
小组分工	23

第一章 绪论

1.1 课题研究的背景和来源

1.1.1 背景

随着医学影像技术的飞速发展,超声图像在临床诊断中扮演着越来越重要的角色。超声图像以其无创、实时、低成本等优势,广泛应用于妇产科、心血管科、肿瘤科等多个领域。在妇产科领域,超声图像对于早期妊娠的检测和评估至关重要,尤其是对孕囊的检测,能够帮助医生准确判断妊娠的位置、大小、形态等信息,对于早期妊娠的监测和异常妊娠的诊断具有重要的临床意义。然而,传统的人工解读超声图像存在主观性强、耗时长、对医生经验依赖度高的问题,容易导致漏诊和误诊。因此,开发一种能够自动、准确检测超声图像中孕囊区域的方法,对于提高诊断效率和准确性具有重要的现实意义。

1.1.2 来源

本课题的研究源自主临床需求与当前技术发展趋势的结合。在实际的妇产科临床工作中,医生需要快速准确地从大量超声图像中识别出孕囊区域,以便进行后续的诊断和治疗决策。然而,面对日益增长的医疗数据,传统的人工诊断方式已经难以满足高效、精准的临床需求。同时,随着深度学习技术在图像识别领域的不断突破,越来越多的研究开始探索其在医学影像分析中的应用潜力。在此背景下,本研究旨在利用深度学习技术,开发一种自动化的超声图像孕囊区域检测系统,以期为临床诊断提供更有力的支持。

1.2 课题研究成果应用的实际意义

1.2.1 临床诊断方面

该研究成果能够为妇产科医生提供一种高效、准确的孕囊检测工具,帮助其快速定位孕囊区域,减少人工诊断的主观误差和时间成本。对于早期妊娠的监测,尤其是对于异位妊娠等异常情况的诊断,该系统能够及时发现孕囊的异常位置和形态,为临床干预争取宝贵时间。此外,通过对孕囊大小和形态的精确测量,还可以辅助医生对胎儿的发育情况进行初步评估,为后续的产前检查和孕期管理提供重要依据。

1.2.2 科研价值方面

本研究为医学影像分析领域提供了一种基于深度学习的孕囊检测方法，丰富了该领域的研究内容和方法体系。通过对超声图像孕囊区域检测的深入研究，可以进一步探索深度学习技术在医学影像中的应用潜力和局限性，为其他相关研究提供参考和借鉴。同时，该研究成果还可以作为基础，进一步拓展到对胎儿其他组织结构的检测和分析，推动胎儿医学影像诊断技术的发展。

1.2.3 社会效益方面

提高孕囊检测的准确性和效率，有助于降低因漏诊和误诊导致的不良妊娠结局，保障母婴健康，具有显著的社会效益。此外，该系统可以应用于基层医疗机构，提高基层医疗的诊断水平，促进医疗资源的合理分配和医疗服务的均等化，为广大孕妇提供更优质、高效的医疗服务。

1.3 相关方法综述

1.3.1 传统图像处理方法

传统图像处理方法在超声图像孕囊区域检测中也有一定应用。例如，基于阈值分割的方法通过设定合适的灰度阈值，将图像分割为前景和背景，从而提取孕囊区域。但这种方法对图像质量要求较高，对于噪声敏感，且难以适应不同灰度分布的图像。边缘检测方法如 Sobel、Canny 算子等，能够检测出图像中的边缘轮廓，进而勾勒出孕囊边界。然而，超声图像中的孕囊边界往往不清晰、不连续，容易受到周围组织和噪声的干扰，导致边缘检测结果不准确。此外，还有基于形态学处理的方法，通过对图像进行腐蚀、膨胀等操作，突出孕囊的形态特征，但该方法同样存在对参数敏感、适应性差等问题。

1.3.2 基于深度学习的目标检测方法

随着深度学习技术的发展，基于卷积神经网络（CNN）的目标检测方法在医学影像领域取得了显著成果。R-CNN 系列方法通过提出候选区域并进行特征提取和分类回归，能够精确检测出图像中的目标物体，在超声图像孕囊检测中也表现出较好的性能。然而，R-CNN 方法存在训练过程复杂、速度较慢等缺点。YOLO 系列方法以其速度快、实时性好而受到关注，其将目标检测任务转化为一个端到端的回归问题，能够同时预测多个目标的位置和类别，在超声图像实时检测方面具有一定优势。此外，FPN（Feature Pyramid Network）结构的引入有效解决了目标尺度变化问题，

提高了检测的准确率和召回率。这些基于深度学习的方法通过学习大量的标注数据，能够自动提取图像中的特征，对孕囊区域进行准确检测，相较于传统方法具有明显优势，但仍存在一些挑战，如数据标注的难度、模型的泛化能力等。

1.3.3 机器学习方法

机器学习方法为孕囊自动检测提供了一种介于传统图像处理和深度学习之间的平衡方案，既具备一定的自动特征学习能力，又不完全依赖大规模标注数据。将复杂的孕囊检测任务分解为特征提取和分类两个阶段：先通过图像处理技术提取有区分度的特征，再用机器学习算法建立特征与孕囊区域的映射关系。机器学习方法在中等规模数据集上通常能达到约 85% 的准确率和 80% 的 Dice 系数，适合作为深度学习方案的前期验证或资源受限环境下的替代方案。其最大价值在于为后续深度学习方法提供可解释的特征重要性分析和可靠的基准性能。

1.3.4 此课题选择的方法对比

我们小组在传统图像处理方法以及基于 YOLOv8 的深度学习方法两种方法上进行了尝试，使用 MATLAB 进行传统图像处理，能对部分图像进行良好的检测但也存在一部分图像检测准确度不高的情况，而用基于 YOLOv8 的深度学习方法能有更高的精度，能自动检测到几乎全部所给测试图片的超声图像的孕囊区域。所以我们小组选择以基于 YOLOv8 的深度学习方法为主能更好地完成此课题。

1. 传统图像处理方法

优点：

无需训练数据：可直接应用于新设备采集的图像

计算效率高：在 CPU 上即可实时处理(约 50ms/帧)

可解释性强：每个处理步骤有明确物理意义

参数可控：医生可根据经验调整阈值

缺点：

鲁棒性差：对图像质量敏感(如探头压力导致的形变)

泛化能力弱：不同设备需重新调参

特征利用有限：仅使用低层图像特征

过度分割问题：易受子宫内其他囊性结构干扰

2. 基于 YOLOv8 的深度学习方法

优点：

检测精度高：mAP 可达 90% 以上

端到端学习：自动优化特征提取与检测

多尺度检测：内置特征金字塔处理不同大小孕囊

强泛化能力：适应不同设备/医院数据

缺点：

数据需求大：需数千标注样本

黑盒特性：决策过程难以解释

硬件要求高：需要 GPU 加速实现实时

第二章 设计方案

本课题采用 YOLOv8 中的 Ultralytics 核心，并使用软件式的模型训练方法进行超声图像孕囊区域检测。YOLOv8 是一种先进的目标检测算法，具有高效、准确的特点，适用于实时检测任务。Ultralytics 是 YOLO 系列算法的官方实现库，提供了丰富的功能和工具，方便用户进行模型训练、评估和部署。

2.1 YOLOv8 算法简介

YOLOv8 是 YOLO (You Only Look Once) 系列的最新版本，继承了前代算法的优点，并在性能和效率上进行了优化和提升。YOLOv8 的核心思想是将目标检测任务转化为一个端到端的回归问题，通过一个神经网络同时预测多个目标的边界框和类别概率，从而实现快速而准确的检测。

2.1.1 模型架构

YOLOv8 的模型架构主要包括以下几个部分：

1. backbone（主干网络）：用于提取图像的特征。通常采用卷积神经网络（CNN）结构，能够自动学习图像中的层次化特征表示。
2. neck（颈部网络）：对 backbone 提取的特征进行进一步处理和融合，增强特征的表达能力。YOLOv8 的 neck 部分采用了特征金字塔网络（FPN）结构，能够有效处理不同尺度的目标。
3. head（头部网络）：根据 neck 提取的特征进行目标的分类和定位。head 部分输出每个预设边界框的类别概率和坐标偏移量。

2.1.2 损失函数

YOLOv8 的损失函数综合考虑了分类损失和边界框回归损失。分类损失通常采用交叉熵损失函数，用于衡量预测类别概率与真实类别概率之间的差异。边界框回归损失则采用 Smooth L1 损失函数，用于计算预测边界框与真实边界框之间的坐标偏移量差异。通过最小化损失函数，使模型能够学习到更加准确的目标检测能力。

2.1.3 数据增强

为了提高模型的泛化能力和鲁棒性，YOLOv8 在训练过程中采用了多种数据增强技术。包括随机裁剪、随机翻转、随机调整亮度和对比度等操作，能够在不增加额外标注成本的情况下，丰富训练数据的多样性，使模型能够更好地适应不同的图像条件和变化。

2.2 Ultralytics 核心

Ultralytics 是 YOLO 算法的官方实现库，提供了简洁易用的接口和丰富的功能，方便用户快速搭建和训练目标检测模型。其主要特点包括：

2.2.1 模块化设计

Ultralytics 采用了模块化的设计理念，将 YOLO 算法的不同部分（如 backbone、neck、head 等）封装成独立的模块，用户可以根据需求灵活组合和定制模型结构。这种模块化设计不仅提高了代码的可维护性和可扩展性，还方便用户进行模型的优化和改进。

2.2.2 完备的工具链

Ultralytics 提供了一整套完备的工具链，包括数据预处理、模型训练、评估、推理和部署等功能。用户可以使用简单的命令行指令完成模型的训练和评估，也可以通过 Python API 将模型集成到自己的应用程序中。此外，Ultralytics 还支持多种模型格式的导出，方便在不同平台和设备上进行部署。

2.2.3 实时推理能力

Ultralytics 优化了 YOLO 模型的推理过程，使其在保证检测精度的同时，具备高效的实时推理能力。这得益于其对模型结构的优化和对硬件加速技术的支持，如 GPU 加速、TensorRT 等，能够满足实际应用场景中对实时性的要求。

2.3 软件式的模型训练方法

2.3.1 数据准备

在进行模型训练之前，需要准备大量的超声图像数据，并对数据进行标注。标注内容包括孕囊区域的边界框坐标和类别标签。数据的多样性和标注的准确性对模型的性能至关重要。我们从多个来源收集超声图像数据，涵盖不同孕期、不同设备采集的图像，以确保数据的代表性。

数据预处理步骤包括：

图像裁剪和缩放：将图像裁剪或缩放到统一的尺寸，以便输入到模型中。

数据增强：应用随机裁剪、翻转、调整亮度和对比度等操作，增加数据的多样性。

标注格式转换：将标注数据转换为 YOLOv8 能够识别的格式，通常是文本文件，包含边界框坐标和类别标签。

2.3.2 模型配置

使用 Ultralytics 提供的配置文件对模型进行配置。配置文件中可以指定模型的架构、训练参数、优化器选择、学习率调度等。根据超声图像孕囊检测任务的特点，对模型的参数进行调整和优化，例如设置合适的输入尺寸、锚框（anchor）尺寸等。

2.3.3 模型训练

利用 Ultralytics 提供的训练脚本启动模型训练。训练过程中，可以实时监控训练指标，如损失函数值、学习率等。Ultralytics 支持分布式训练和混合精度训练，能够加速训练过程并提高模型的收敛速度。

训练过程中的关键步骤包括：

模型初始化：加载预训练权重或随机初始化模型参数。

数据加载和迭代：将预处理后的数据分批送入模型进行训练，逐批次更新模型参数。

损失计算和反向传播：根据预测结果和真实标注计算损失函数值，通过反向传播算法更新模型参数。

模型保存和评估：定期保存训练好的模型权重，并在验证集上进行评估，监控模型的性能指标，如平均精度（mAP）、召回率等。

2.3.4 模型优化

在模型训练完成后，对其进行进一步的优化，以提高模型的性能和效率。例如，采用模型剪枝技术去除冗余的神经元和通道，减少模型的计算量和存储空间；使用量化技术将模型的权重和激活值转换为低精度数据类型，加速模型的推理速度；还可以通过调整模型的超参数，如锚框尺寸、置信度阈值等，来提高模型在特定数据集上的检测效果。

2.3.5 模型评估和测试

使用验证集对训练好的模型进行评估，计算模型的各项性能指标，如精确率（Precision）、召回率（Recall）、平均精度（mAP）等。根据评估结果，对模型进行进一步的调整和优化。在验证集评估通过后，将模型应用于测试集进行最终的测试，以验证模型在未见过的数据上的泛化能力和实际应用效果。

通过采用 YOLOv8 中的 Ultralytics 核心，并结合软件式的模型训练方法，我们致力于开发一个高效、准确的超声图像孕囊区域检测系统，为临床诊断提供有力的支持。

第三章 实验过程及结果分析

主要介绍根据本小组所设计的技术方案，对给定数据进行实验的过程，得到的结果，以及对结果优劣的评估分析。

所使用的各类软件设备：



参数设置：

```
1 task: detect
2 mode: train
3 model: yolov8s.yaml
4 data: C:/Users/20467/Desktop/V0Cdevkit/dataset.yaml
5 epochs: 200
6 time: null
7 patience: 100
8 batch: 16
9 imgsz: 640
10 save: true
11 save_period: -1
12 cache: false
13 device: '0'
14 workers: 12
15 project: C:/Users/20467/Desktop/V0Cdevkit/run
16 name: train2
17 exist_ok: false
18 pretrained: h:/yolo免环境训练工具v1.2/weight/yolov8s.pt
19 optimizer: SGD
20 verbose: true
21 seed: 0
22 deterministic: true
23 single_cls: false
24 rect: false
25 cos_lr: false
26 close_mosaic: 10
27 resume: false
28 amp: true
29 fraction: 1.0
30 profile: false
31 freeze: null
32 multi_scale: false
33 overlap_mask: true
34 mask_ratio: 4
35 dropout: 0.0
36 val: true
37 split: val
38 save_json: false
```

模型训练基于 YOLOv8 模型结构。根据 runs/train/gestational_sac_exp/args.yaml 文件，训练过程的核心参数如下：

Parameter	值 (Value)	描述
model	yolov8s.yaml	使用的预训练模型。
epochs	200	训练总轮次。
batch	16	批处理大小。
imgsz	640	输入图像统一调整的尺寸。
optimizer	SGD	优化器选择（YOLOv8 自动选择）。
lr0	0.01	初始学习率。
lrf	0.01	最终学习率（学习率调度相关）。
momentum	0.937	优化器动量。
weight_decay	0.0005	权重衰减系数，用于防止过拟合。
patience	100	Early stopping 的耐心值（100 轮无提升则停止）。
close_mosaic	10	最后 10 个 epoch 关闭 mosaic 数据增强。

训练过程：

```
train: Scanning C:\Users\20467\Desktop\VOCdevkit\labels\train.cache... 16 images, 0 backgrounds, 0 corrupt: 100%|██████████| 16/16 [00:00<?, ?it/s]
val: Scanning C:\Users\20467\Desktop\VOCdevkit\labels\val.cache... 7 images, 0 backgrounds, 0 corrupt: 100%|██████████| 7/7 [00:00<?, ?it/s]
Plotting labels to C:\Users\20467\Desktop\VOCdevkit\run\train2\labels.jpg...
optimizer: SGD(lr=0.01, momentum=0.937) with parameter groups 57 weight(decay=0.0), 64 weight(decay=0.0005), 63 bias(decay=0.0)
TensorBoard: model graph visualization added
Image sizes 640 train, 640 val
Using 12 dataloader workers
Logging results to C:\Users\20467\Desktop\VOCdevkit\run\train2
Starting training for 200 epochs...
```

Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size	640: 100%	1/1 [00:02<00:00, 2.13s/it]
1/200	4.24G	2.996	11.9	2.298	26	640: 100%	100%	1/1 [00:00<00:00, 3.23it/s]
	Class	Images	Instances	Box(P	R	mAP50	mAP50-95): 100%	
	all	7	7	0.000952	0.286	0.000737	0.000208	
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size	640: 100%	1/1 [00:00<00:00, 3.83it/s]
2/200	4.12G	2.929	9.985	2.301	30	640: 100%	100%	1/1 [00:00<00:00, 11.19it/s]
	Class	Images	Instances	Box(P	R	mAP50	mAP50-95): 100%	
	all	7	7	0.000952	0.286	0.000753	0.000212	
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size	640: 100%	1/1 [00:00<00:00, 4.11it/s]
3/200	4.12G	2.943	9.656	2.301	35	640: 100%	100%	1/1 [00:00<00:00, 12.04it/s]
	Class	Images	Instances	Box(P	R	mAP50	mAP50-95): 100%	
	all	7	7	0.000952	0.286	0.000758	0.000183	
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size	640: 100%	1/1 [00:00<00:00, 4.10it/s]
4/200	4.12G	2.687	10.65	2.268	28	640: 100%	100%	1/1 [00:00<00:00, 14.93it/s]
	Class	Images	Instances	Box(P	R	mAP50	mAP50-95): 100%	
	all	7	7	0.00143	0.429	0.00107	0.000246	
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size	640: 100%	1/1 [00:00<00:00, 4.02it/s]
5/200	4.12G	2.733	12.15	2.192	25	640: 100%	100%	1/1 [00:00<00:00, 14.81it/s]
	Class	Images	Instances	Box(P	R	mAP50	mAP50-95): 100%	
	all	7	7	0.00143	0.429	0.00125	0.000241	
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size	640: 100%	1/1 [00:00<00:00, 4.00it/s]
6/200	4.12G	2.463	9.139	2.044	32	640: 100%	100%	1/1 [00:00<00:00, 12.58it/s]
	Class	Images	Instances	Box(P	R	mAP50	mAP50-95): 100%	
	all	7	7	0.00143	0.429	0.00145	0.000363	
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size	640: 100%	1/1 [00:00<00:00, 3.95it/s]
7/200	4.13G	2.798	11.19	2.385	25	640: 100%	100%	1/1 [00:00<00:00, 15.16it/s]
	Class	Images	Instances	Box(P	R	mAP50	mAP50-95): 100%	
	all	7	7	0.00143	0.429	0.00162	0.000352	

训练集 (Training Set):

孕囊图像: 16 个病例的切片

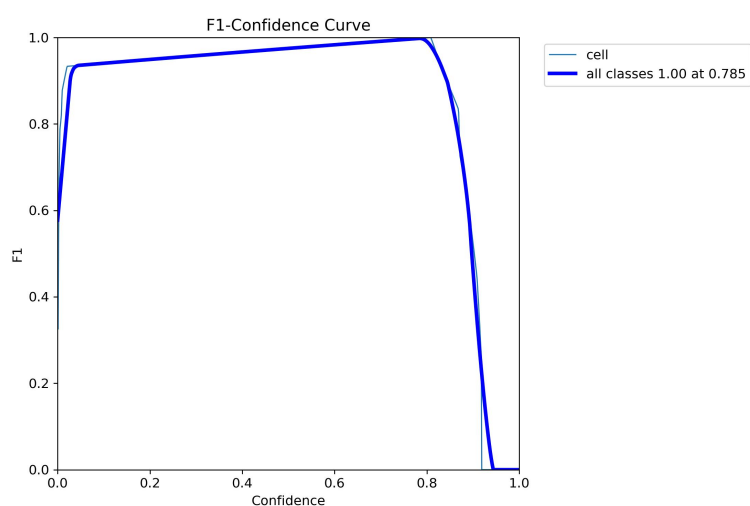
总计: 16 个病例的切片被用于训练。

验证集 (Validation Set):

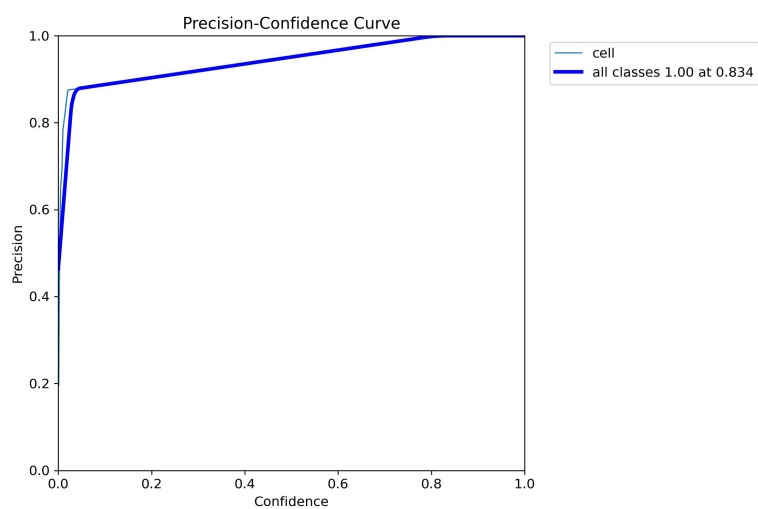
孕囊图像: 7 个病例的切片

总计: 7 个病例的切片被用于验证。

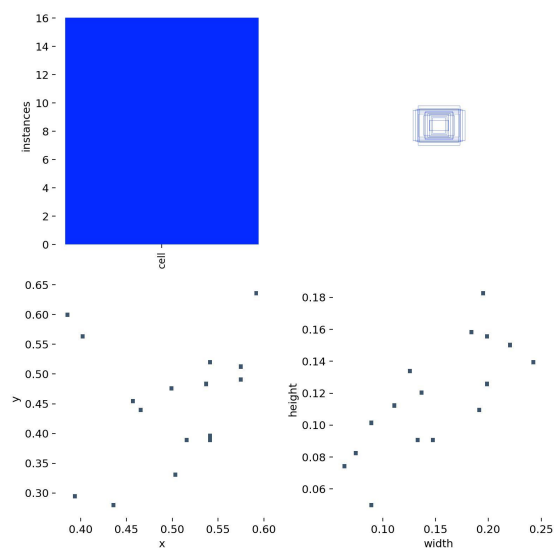
置信度曲线:



精确度—置信度曲线:



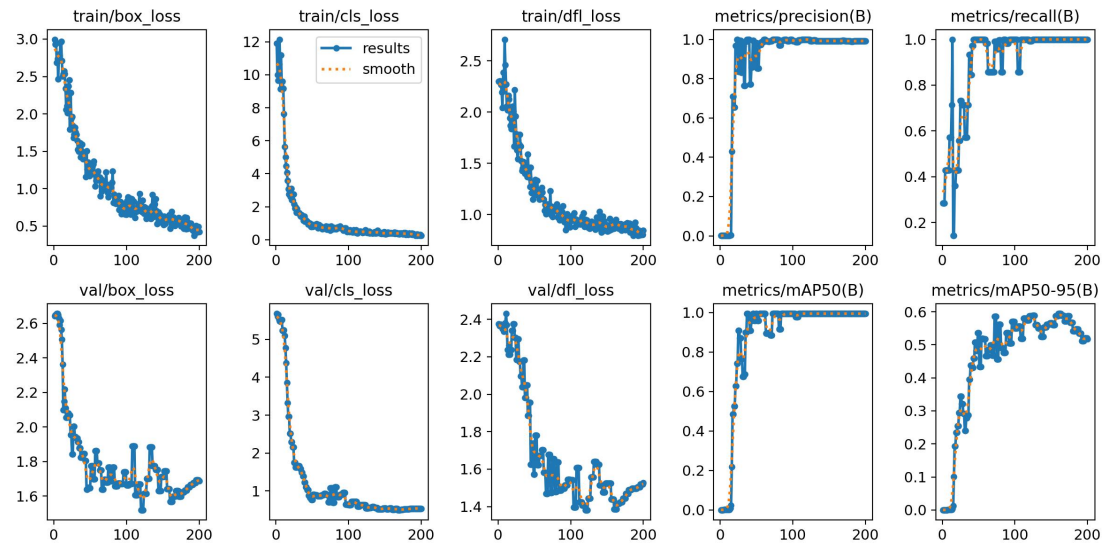
标签分布图和边界框分别图：



训练过程：

涉及私人内容不予展示

训练结果图表：



医学图像处理课程设计报告

epoch	time	train/box	train/cls	train/df1	metrics/p1	metrics/r1	metrics/m1	metrics/m2	val/box	val/cls	val/df1	lr/pg0	lr/pg1	lr/pg2
1	5.20471	2.99617	11.9029	2.29795	0.00095	0.28571	0.00074	0.00021	2.64262	5.67929	2.37539	0.1	0	0
2	5.92783	2.92862	9.9846	2.30102	0.00095	0.28571	0.00075	0.00021	2.64736	5.6082	2.36602	0.0990995	9.9505e-05	9.9505e-05
3	6.62758	2.94345	9.6562	2.3014	0.00095	0.28571	0.00076	0.00018	2.6476	5.65534	2.36668	0.098198	0.000198	0.000198
4	7.23849	2.68739	10.6506	2.26841	0.00143	0.42857	0.00107	0.00025	2.65569	5.52815	2.37236	0.0972955	0.0002955	0.0002955
5	7.87592	2.73342	12.1457	2.19204	0.00143	0.42857	0.00125	0.00024	2.65574	5.47563	2.36687	0.0963921	0.0003921	0.0003921
6	8.53286	2.46263	9.13933	2.04366	0.00143	0.42857	0.00145	0.00036	2.64712	5.47482	2.35077	0.0954876	0.0004876	0.0004876
7	9.1667	2.79814	11.1855	2.38479	0.00143	0.42857	0.00162	0.00035	2.62756	5.4974	2.34461	0.0945822	0.0005822	0.0005822
8	9.81544	2.77881	9.21306	2.2274	0.00143	0.42857	0.00174	0.00054	2.58986	5.51751	2.33517	0.0936757	0.0006757	0.0006757
9	10.4891	2.95564	9.59501	2.709	0.00143	0.42857	0.00192	0.00058	2.614	5.23352	2.37293	0.0927683	0.0007683	0.0007683
10	11.2787	2.97123	9.15639	2.45962	0.0019	0.57143	0.00225	0.00066	2.56354	5.22597	2.38037	0.0918599	0.0008599	0.0008599
11	12.0587	2.70958	7.62619	2.27272	0.0019	0.57143	0.0023	0.0008	2.50641	5.24678	2.4308	0.0909505	0.0009505	0.0009505
12	12.8347	2.50529	5.64463	2.05675	0.0019	0.57143	0.00296	0.00087	2.36222	5.09949	2.36957	0.0900401	0.0010401	0.0010401
13	13.5899	2.53262	5.01224	2.01834	0.00238	0.71429	0.00294	0.00103	2.09887	4.77147	2.23726	0.0891287	0.0011287	0.0011287
14	14.3929	2.57067	4.4551	2.15996	0.00333	1	0.00932	0.00356	2.1476	4.38299	2.21343	0.0882163	0.0012164	0.0012164
15	15.1496	2.53703	4.07555	2.12489	0.00477	1.4286	0.0244	0.0127	2.21684	3.85462	2.26052	0.087303	0.001303	0.001303
16	15.9446	2.341	3.57763	1.94356	0.42966	0.14286	0.21805	0.10079	2.1081	3.32715	2.21293	0.0863886	0.0013886	0.0013886
17	16.6972	2.05646	3.05909	1.86996	0.71004	0.36007	0.48578	0.19256	2.05446	2.95768	2.23818	0.0854733	0.0014733	0.0014733
18	17.4892	2.10964	2.76965	1.83664	0.71004	0.36007	0.48578	0.19256	2.05446	2.95768	2.23818	0.0845569	0.0015569	0.0015569
19	18.2666	2.01298	2.94105	1.8895	0.65514	0.42857	0.52591	0.23439	2.07935	2.51941	2.34141	0.0836396	0.0016396	0.0016396
20	19.0217	2.36955	2.62879	2.05825	0.65514	0.42857	0.52591	0.23439	2.07935	2.51941	2.34141	0.0827213	0.0017213	0.0017213
21	19.7748	2.45539	3.10129	2.0104	0.96818	0.42857	0.62847	0.25376	2.0686	2.28805	2.37573	0.081802	0.001802	0.001802
22	20.5053	1.79253	2.42706	1.66813	0.96818	0.42857	0.62847	0.25376	2.0686	2.28805	2.37573	0.0808817	0.0018817	0.0018817
23	21.2402	2.28417	2.37373	2.21532	1	0.55726	0.74315	0.29439	1.95312	2.15812	2.30202	0.0799604	0.0019604	0.0019604
24	21.9861	1.93584	2.74727	1.9599	1	0.55726	0.74315	0.29439	1.95312	2.15812	2.30202	0.0790381	0.0020381	0.0020381
25	22.7197	1.96382	4.07555	1.78401	0.83631	0.73215	0.91038	0.34421	1.84161	1.75257	2.18489	0.0781149	0.0021149	0.0021149
26	23.4479	1.84544	1.90645	1.63111	0.83631	0.73215	0.91038	0.34421	1.84161	1.75257	2.18489	0.0771906	0.0021906	0.0021906
27	24.162	1.67809	1.98826	1.54244	0.99409	0.71429	0.87158	0.32067	2.0034	1.61916	2.29648	0.0762654	0.0022654	0.0022654
28	24.8404	1.7239	1.63488	1.6731	0.99409	0.71429	0.87158	0.32067	2.0034	1.61916	2.29648	0.0753391	0.0023391	0.0023391

173	123.355	0.59038	0.38471	0.89244	0.99265	1	0.995	0.58781	1.59795	0.50624	1.44324	0.001486	0.001486	0.001486
174	124.024	0.57455	0.37463	0.88249	0.99265	1	0.995	0.58781	1.59795	0.50624	1.44324	0.0014365	0.0014365	0.0014365
175	124.67	0.49982	0.33664	0.85181	0.99265	1	0.995	0.58781	1.59795	0.50624	1.44324	0.001387	0.001387	0.001387
176	125.358	0.58605	0.40984	0.91008	0.9912	1	0.995	0.56774	1.62366	0.51545	1.46981	0.0013375	0.0013375	0.0013375
177	125.989	0.52865	0.36585	0.87444	0.9912	1	0.995	0.56774	1.62366	0.51545	1.46981	0.001288	0.001288	0.001288
178	126.624	0.57842	0.41506	0.9207	0.9912	1	0.995	0.56774	1.62366	0.51545	1.46981	0.0012385	0.0012385	0.0012385
179	127.257	0.4686	0.29483	0.85629	0.9912	1	0.995	0.56774	1.62366	0.51545	1.46981	0.001189	0.001189	0.001189
180	127.919	0.61818	0.38372	0.88807	0.99044	1	0.995	0.54252	1.63303	0.51813	1.48905	0.0011395	0.0011395	0.0011395
181	128.572	0.48462	0.34161	0.91394	0.99044	1	0.995	0.54252	1.63303	0.51813	1.48905	0.00109	0.00109	0.00109
182	129.198	0.4606	0.31965	0.82992	0.99044	1	0.995	0.54252	1.63303	0.51813	1.48905	0.0010405	0.0010405	0.0010405
183	129.821	0.60937	0.37881	0.84674	0.99044	1	0.995	0.54252	1.63303	0.51813	1.48905	0.000991	0.000991	0.000991
184	130.514	0.55265	0.40187	0.8707	0.99143	1	0.995	0.54748	1.65174	0.53538	1.49582	0.0009415	0.0009415	0.0009415
185	131.192	0.46639	0.32169	0.82289	0.99143	1	0.995	0.54748	1.65174	0.53538	1.49582	0.000892	0.000892	0.000892
186	131.862	0.53378	0.33259	0.79483	0.99143	1	0.995	0.54748	1.65174	0.53538	1.49582	0.0008425	0.0008425	0.0008425
187	132.554	0.50308	0.37921	0.90283	0.99143	1	0.995	0.54748	1.65174	0.53538	1.49582	0.000793	0.000793	0.000793
188	133.228	0.52318	0.30984	0.85838	0.99113	1	0.995	0.53459	1.66692	0.53426	1.49586	0.0007435	0.0007435	0.0007435
189	133.896	0.53721	0.35178	0.92209	0.99113	1	0.995	0.53459	1.66692	0.53426	1.49586	0.000694	0.000694	0.000694
190	134.57	0.46136	0.33693	0.86911	0.99113	1	0.995	0.53459	1.66692	0.53426	1.49586	0.0006445	0.0006445	0.0006445
191	142.295	0.43999	0.29284	0.81132	0.99113	1	0.995	0.53459	1.66692	0.53426	1.49586	0.000595	0.000595	0.000595
192	142.993	0.44929	0.25956	0.8224	0.9918	1	0.995	0.51206	1.67793	0.53891	1.50319	0.0005455	0.0005455	0.0005455
193	143.649	0.3719	0.26408	0.79594	0.9918	1	0.995	0.51206	1.67793	0.53891	1.50319	0.000496	0.000496	0.000496
194	144.282	0.42404	0.32431	0.84761	0.9918	1	0.995	0.51206	1.67793	0.53891	1.50319	0.0004465	0.0004465	0.0004465
195	144.915	0.46602	0.24474	0.80038	0.9918	1	0.995	0.51206	1.67793	0.53891	1.50319	0.000397	0.000397	0.000397
196	145.565	0.50785	0.30826	0.83052	0.99205	1	0.995	0.52019	1.69302	0.53787	1.5199	0.0003475	0.0003475	0.0003475
197	146.196	0.41454	0.2786	0.83013	0.99205	1	0.995	0.52019	1.69302	0.53787	1.5199	0.000298	0.000298	0.000298
198	146.87	0.43716	0.2639	0.80617	0.99205	1	0.995	0.52019	1.69302	0.53787	1.5199	0.0002485	0.0002485	0.0002485
199	147.546	0.49312	0.31856	0.80611	0.99205	1	0.995	0.52019	1.69302	0.53787	1.5199	0.000199	0.000199	0.000199
200	148.246	0.42322	0.26087	0.8465	0.9921	1	0.995	0.51774	1.68692	0.53274	1.52793	0.0001495	0.0001495	0.0001495

train/box_loss（训练集的边界框损失）：随着训练轮次的增加，边界框损失逐渐降低，定位目标更加准确

train/cls_loss（训练集的分类损失）：分类损失在初期迅速下降，然后趋于平稳，逐渐提高了分类准确性。

train/df1_loss（训练集的分布式焦点损失）：呈现下降趋势，即优化了预测框与真实框之间的匹配。

metrics/precision(B)（精确度）：精确度随着训练轮次的增加而提高。

metrics/recall(B)（召回率）：召回率逐渐上升。

val/box_loss（验证集的边界框损失）：验证集的边界框损失下降，但存在一些波动。迹象。

val/cls_loss（验证集的分类损失）：验证集的分类损失呈下降趋势，但在某些点上出现波动。

val/df_l_loss (验证集的分布式焦点损失) : 验证集的分布式焦点损失也在下降, 存在一些波动, 可能需要进一步观察以确定是否是过拟合的迹象。

metrics/mAP50(B) (在 IoU 阈值为 0.5 时的平均精度) : mAP50 随着训练轮次的增加而提高。

metrics/mAP50-95(B) (在 IoU 阈值从 0.5 到 0.95 的平均精度) : mAP50-95 的提高即模型在不同 IoU 阈值下的性能都在提升。

检测代码:

```
import nibabel as nib

import numpy as np

import cv2

from ultralytics import YOLO

import os

import matplotlib.pyplot as plt

from mpl_toolkits.mplot3d import Axes3D


# 1. 加载 NIfTI 文件

def load_nii(path):

    nii_img = nib.load(path)

    data = nii_img.get_fdata()

    return data # 返回原始数据


# 2. 直接转换切片为图像格式

def slice_to_image(slice_2d):

    """将原始切片数据转换为 8 位灰度图像"""

    # 归一化到 0-255 范围

    slice_normalized = (slice_2d - np.min(slice_2d)) / (np.max(slice_2d) - np.min(slice_2d) +
```


1e-8)

```
image = (slice_normalized * 255).astype(np.uint8)

# 转换为 3 通道(RGB)

return cv2.cvtColor(image, cv2.COLOR_GRAY2RGB)
```

3. 处理单张切片

```
def process_slice(slice_2d, model, slice_idx):

    # 直接转换格式

    rgb_slice = slice_to_image(slice_2d)

    # YOLOv8 推理

    results = model(rgb_slice, conf=0.3)

    # 绘制检测结果

    annotated = results[0].plot()

    cv2.putText(annotated, f'Slice: {slice_idx}', (10, 30),

                cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)

    return annotated, results[0].boxes.data.cpu().numpy()
```

4. 主流程

```
if __name__ == "__main__":

    # 配置参数

    nii_path = "F:/BaiduNetdiskDownload/normal/2/transectionorg.nii"

    model_path = "H:/ultralytics-main/best.pt" # 替换为超声专用模型

    output_dir = "H:/1"
```

```
# 加载模型

model = YOLO(model_path)

# 加载 NIfTI 数据

volume = load_nii(nii_path)

depth = volume.shape[2] if len(volume.shape) > 2 else 1

# 创建输出目录

os.makedirs(output_dir, exist_ok=True)

# 逐片处理

all_detections = []

for z in range(depth):

    # 提取切片

    if len(volume.shape) == 3: # 3D 体积

        slice_2d = volume[:, :, z]

    else: # 2D 图像

        slice_2d = volume

    # 跳过空切片

    if np.max(slice_2d) < 1e-5:

        continue

    # 处理切片

    annotated_img, detections = process_slice(slice_2d, model, z)

    # 保存结果

    cv2.imwrite(f'{output_dir}/slice_{z:04d}.png', annotated_img)

    for det in detections:

        all_detections.append([z] + det.tolist())
```

```
# 如果是 2D 图像，只处理一次
if len(volume.shape) == 2:
    break

# 保存检测结果
if all_detections:
    np.savetxt(f'{output_dir}/detections.csv', all_detections,
               delimiter=",", fmt="%0.4f",
               header="slice,x1,y1,x2,y2,conf,class")

# 3D 可视化（仅当有多个切片时）
if len(volume.shape) == 3 and all_detections:
    fig = plt.figure(figsize=(10, 8))
    ax = fig.add_subplot(111, projection='3d')

    # 为不同类别设置颜色
    class_colors = ['r', 'g', 'b', 'c', 'm', 'y']

    for det in all_detections:
        z, x1, y1, x2, y2, conf, cls = det
        cls = int(cls)
        color = class_colors[cls % len(class_colors)]

        # 绘制中心点
        center_x = (x1 + x2) / 2
        center_y = (y1 + y2) / 2
        ax.scatter(center_x, center_y, z, c=color, s=50)

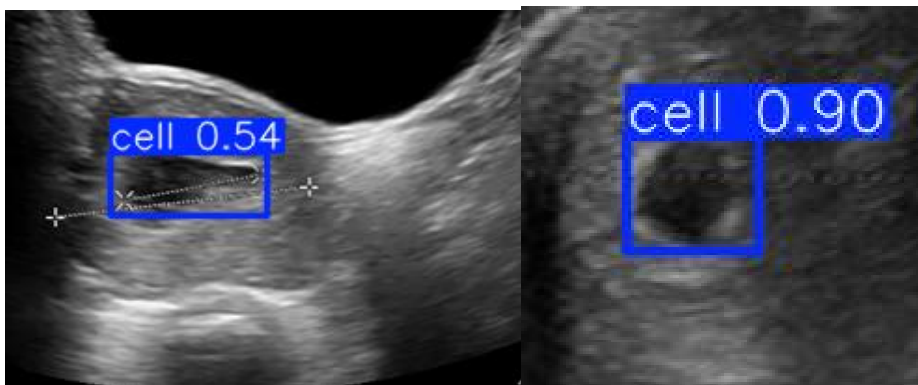
    # 绘制边界框（简化表示）
```

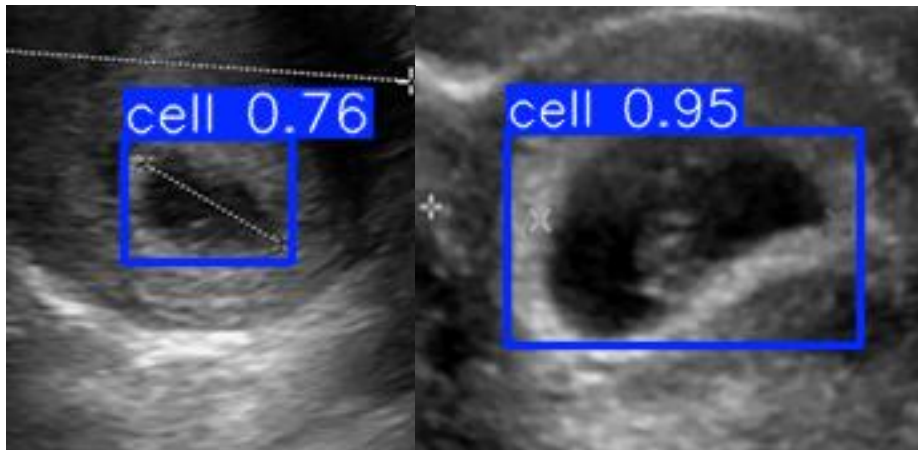
```
ax.plot([x1, x2], [y1, y1], [z, z], c=color, alpha=0.7)
ax.plot([x1, x2], [y2, y2], [z, z], c=color, alpha=0.7)
ax.plot([x1, x1], [y1, y2], [z, z], c=color, alpha=0.7)
ax.plot([x2, x2], [y1, y2], [z, z], c=color, alpha=0.7)
```

```
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Slice')
plt.title('3D Detection Results')
plt.savefig(f'{output_dir}/3d_detections.png', dpi=300)
plt.close()
```

```
print(f'处理完成! 共生成 {depth} 张切片结果, 检测到 {len(all_detections)} 个目标")
```

Nii 图像自动检测的结果:





附：前期使用 MATLAB 进行传统图像处理的尝试

1. 读取 nii 文件并进行预处理

```
niiFilePath = 'C:\数字图像处理\孕囊超声图像\normal\9\transectionorg.nii';
niiData = niftiread(niiFilePath);
niiInfo = niftiinfo(niiFilePath);

% 如果NIFTI文件是4D的(如时间序列)，选择第一个时间点
if ndims(niiData) == 4
    originalImage = niiData(:,:,1);
else
    originalImage = niiData;
end

% 对于3D数据，选择一个切片进行处理（例如中间切片）
if ndims(originalImage) == 3
    sliceNum = round(size(originalImage, 3)/2);
    originalImage = originalImage(:,:,sliceNum);
end

% 转换为灰度图像（NIFTI可能已经是灰度）
if ~ismatrix(originalImage)
    % 如果图像有多个通道，转换为灰度
    grayImage = mat2gray(originalImage); % 归一化到[0,1]范围
    grayImage = im2uint8(grayImage); % 转换为8位灰度
else
    grayImage = originalImage;
end
```

2. 图像增强、滤波去噪处理

```
% 图像增强 - 对比度调整
enhancedImage = histeq(grayImage);
subplot(2,3,2);
imshow(enhancedImage);
title('对比度增强');

% 去噪处理 - 结合高斯滤波和中值滤波
sigma = 1.5;
filteredImg = imgaussfilt(enhancedImage, sigma);
filteredImage = medfilt2(filteredImg, [3 3]);
subplot(2,3,3);
imshow(filteredImage);
title('组合去噪');
```

3.边缘检测、并且去除图片四角

```
% 边缘检测 - 优化Canny算子参数
[~, threshold] = edge(filteredImage, 'sobel');
fudgeFactor = 4;
edgeImage = edge(filteredImage, 'Canny', threshold * fudgeFactor);

% 创建中心区域掩模（去除四个角）
[rows, cols] = size(edgeImage);
centerMask = false(rows, cols);
margin = round(min(rows, cols) * 0.25); % 保留中心75%的区域

% 定义中心矩形区域
centerMask(margin:end-margin, margin:end-margin) = true;

% 应用掩模，去除四角
centerEdgeImage = edgeImage & centerMask;

subplot(2,3,4);
imshow(centerEdgeImage);
title('中心区域边缘检测');
```

4.形态学处理

```
% 形态学操作 - 根据图像特性调整参数
se1 = strel('disk', 1);
dilatedImage = imdilate(centerEdgeImage, se1);

se2 = strel('disk', 1);
closedImage = imclose(dilatedImage, se2);

filledImage = imfill(closedImage, 'holes');
% 提取最大连通区域（确保只保留孕囊主体）
%filledImage = bwareafilt(filledImage, 4); % 保留面积最大的区域

subplot(2,3,5);
imshow(filledImage);
title('优化形态学处理');
```

5.筛选孕囊条件

```

% 区域属性分析
[labeledImage, numRegions] = bwlabel(filledImage);
regionProps = regionprops(labeledImage, 'Area', 'BoundingBox', 'Eccentricity', 'Solidity', ...
    'MajorAxisLength', 'MinorAxisLength', 'Orientation', 'Centroid', 'Perimeter');

% 获取图像中心坐标
[imgHeight, imgWidth] = size(filledImage);
imageCenter = [imgWidth/2, imgHeight/2]; % [x,y]格式

% 筛选孕囊区域 (基于面积、偏心率和实心底)
minArea = 500; % 根据图像大小调整
maxArea = 2000;
maxEccentricity = 0.7;
minSolidity = 0.3;
maxDistanceToCenter = 200; % 像素单位, 允许的最大中心偏移距离 (根据图像尺寸调整)
minCircularity = 0.3;
maxEllipseError = 0.2; % 椭圆拟合的最大误差 (新增条件)
minAspectRatio = 0.5; % 长短轴最小比例 (避免过于狭长的区域)
maxAspectRatio = 2.0; % 长短轴最大比例

sacRegions = [];

for i = 1:numRegions
    props = regionProps(i);
    area = props.Area;
    eccentricity = props.Eccentricity;
    solidity = props.Solidity;
    centroid = props.Centroid; % 区域质心坐标[x,y]
    % 计算到图像中心的距离
    distanceToCenter = norm(centroid - imageCenter);
    majorAxis = props.MajorAxisLength;
    minorAxis = props.MinorAxisLength;

    circularity = (4 * pi * area) / (props.Perimeter^2 + eps); % 避免除以0
    % 计算长短轴比例
    aspectRatio = minorAxis / (majorAxis + eps);
    % 椭圆拟合误差 (需要自定义函数, 见下文)
    ellipseError = computeEllipseError(props, filledImage,i);

    % 综合筛选条件
    if area > minArea && ...
        area < maxArea &&...
        distanceToCenter < maxDistanceToCenter %&&...
        %eccentricity < maxEccentricity %&& ...
        %solidity > minSolidity %&& ...
        %circularity > minCircularity %&& ...
        %aspectRatio >= minAspectRatio && aspectRatio <= maxAspectRatio && ...
        %ellipseError < maxEllipseError
        sacRegions = [sacRegions; props.BoundingBox];
    end
end
end

```

测试结果：

涉及私人内容不予展示

第四章 结论与展望

通过本次课设，掌握了 YOLOv8 的模型训练、调参及部署流程，同时也能借此理解了目标检测在医疗影像中的挑战。在实验过程中模型的性能受训练数据质量和数量的影响较大，数据增强和标注精度对提升检测效果至关重要。通过调整模型结构（如减少参数量、优化损失函数），可以在保证精度的同时提升推理速度，适合边缘设备部署。

这种方法也存在着部分局限性，在复杂背景或低对比度图像中，模型可能出现误检或漏检，需进一步优化。本次课设的训练数据相对较少，若有更多的孕囊超声图像，能使得训练、测试的准确度得以提高。

前面用 MATLAB 对图像用传统处理方法的研究还比较简略，若是能在提高在噪声处理、边缘检测、形态学处理这几方面的精度，应该还能让孕囊筛选的条件更加严格从而提高检测精度。

总的来说，通过实验中各种方法的尝试，提升了我们解决问题的能力，对于数据我们学会了系统性分析和迭代优化，认识到医学与 AI 结合的重要性，要考虑算法性能的同时，也不能脱离临床实际需求，这也启发我们要未来研究需更注重模型的可靠性和伦理规范。

参考文献

- [1] 超详细的 YOLOv8 项目组成解析：一站式指南了解其架构与组件 2023-12-26
https://blog.csdn.net/qq_42452134/article/details/135151827
- [2] Matlab 图像处理 2025-04-12
https://blog.csdn.net/m0_53004397/article/details/122779619
- [3] 聚类分析医学图像分割毕业论文【matlab】 2023-10-14
<https://blog.csdn.net/checkpaper/article/details/133824004>
- [4] 超详细 YOLOv8 图像分类全程概述：环境、训练、验证与预测详解 2023-12-26
https://blog.csdn.net/qq_42452134/article/details/135168930?ops_request_misc=%257B%2522request%255Fid%2522%253A%252206eb8944d4a8c96a7678e93579d0ecb9%2522%252C%2522scm%2522%253A%252220140713.130102334..%2522%257D&request_id=06eb8944d4a8c96a7678e93579d0ecb9&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduend~default-1-135168930-null-null.142^v102^pc_search_result_base8&utm_term=%E5%9F%BA%E4%BA%8Eyolov8%E7%9A%84%E5%9B%BE%E5%83%8F%E5%A4%84%E7%90%86&spm=1018.2226.3001.4187

组员分工

背景调研：张海涛

方案设计：李博宇、张海涛、张卓权

代码实现：李博宇、张卓权

报告撰写：张卓权 、李博宇