基于改进 YOLOv5s 的群养猪行为识别算法研究及部署

袁伟俊[1]

（1.暨南大学信息科学技术学院，广州 510632）

1 项目概述

中国是全球最大生猪养殖和猪肉消费国家，生猪养殖业的发展对我国经济的发展具有重要的战略意义。生猪的不同行为与其健康状况息息相关，例如当生猪饮水不足时，会导致严重的猪肠道病及猪只生长不良等问题。目前，大多养殖场主要通过人工巡视猪场的方式观察生猪的健康状况，主观性强，效率低，无法满足智慧养殖业规模化和集约化的快速发展。因此，采用视频监控等手段辅助监测生猪的健康状况是规模化生猪养殖的关键技术。

本项目首先以群养生猪图像为研究对象，利用 **YOLOv5s** 及改进方法，实现群养猪**饮水、站立和躺卧**三种行为识别，图 1 为识别效果图；然后将训练得到的群养猪行为识别模型，转换为机器学习通用中间格式 ONNX，使用 ONNX Runtime 在腾讯云服务器上进行推理，本项目最终以**网站平台、微信小程序**和 **PyQt 应用软件**三种不同的形式进行部署，方便用户的使用。

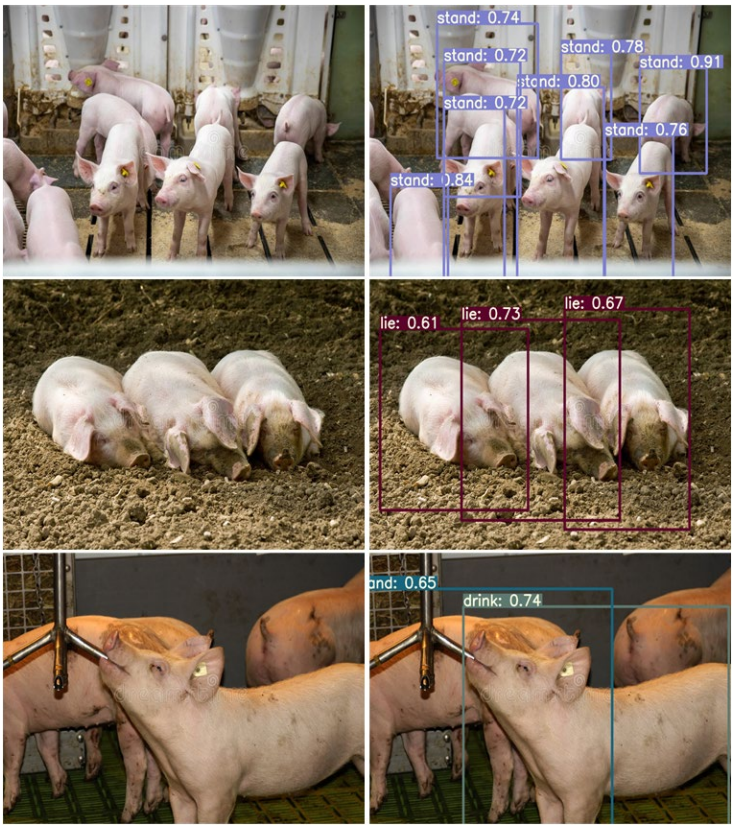

图 1 群养猪行为识别效果图

2 项目流程

图 2 是基于改进 YOLOv5s 的群养猪行为识别模型建立流程图，可概括为以下几个阶段：**数据集的准备工作，YOLOv5 模型的搭建，模型结构的调优**和**模型的部署**。

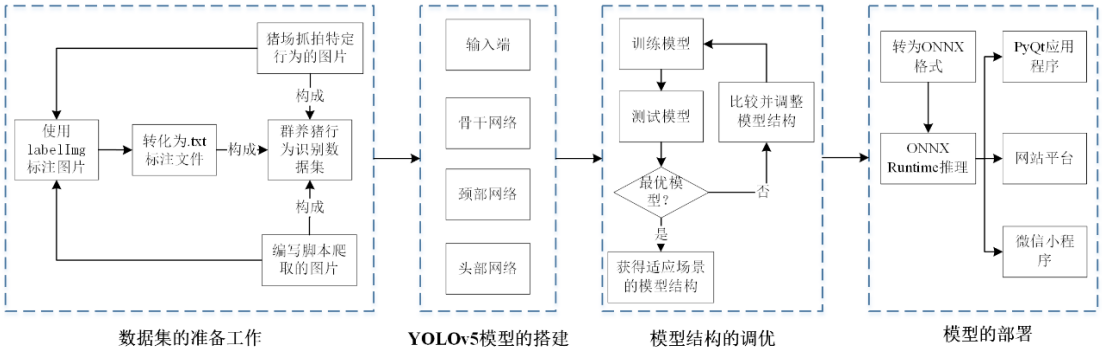

图 2 基于改进 YOLOv5s 的群养猪行为识别模型建立流程

2.1 数据集的准备

行为识别的实验数据有两个来源，分别是：（a）拍摄自广东省韶关市某生猪养殖场，拍摄时间为 2021 年 1 月 27 日-2 月 21 日 14:00-18:00，采用正拍和俯拍两种角度抓拍群养猪进行饮水、站立等行为，一共采集 156 张图片；（b）从百度、必应等不同搜索引擎中查找群养猪进行饮水、站立、躺卧行为的图片共 1200 张，编写程序代码对图片进行爬取，经过人工筛选，最后采集 224 张符合实验要求的图片。

采集完数据后，对群养猪的**饮水**、**站立**和**躺卧**三种行为进行手工标注，标签分别为 drink、stand 和 lie，将标注完成的数据随机划分为训练集和测试集，比例为 7：3，即 265 张训练集和 115 张测试集。数据集的具体分布情况如表 1 所示，训练集中共 504 头群养猪，其中标签为 drink、stand 和 lie 的数目分别为 139、203 和 162，测试集一共 213 头群养猪，其中标签为 drink、stand 和 lie 的数目分别为 65、79 和 69。

表 1 行为识别数据集的具体分布情况

数据集	张数	drink/头	stand/头	lie/头
训练集	265	139	203	162
测试集	115	65	79	69
总数	380	204	282	231

2.2 YOLOv5 模型的搭建

图 3 为基于 YOLOv5s 的行为识别算法网络结构图，主要包含 4 个部分，分别为输入端、骨干网络、颈部网络和头部网络。

YOLOv5s 网络结构的第一部分是输入端，该部分通常包含一个图像预处理阶段，即将输入的图像缩放为网络的输入大小（640×640），并进行归一化等操作，在网络的训练阶段，采用 Mosaic 数据增强等操作提升模型的训练速度；第二部分是骨干网络，该阶段经过 Focus 处理为后续的特征提取保留更完整的图片下采样信息，然后利用 C3 降低计算复杂度，采用空间金字塔池化 SPP 融合多尺度信息实现特征增强；第三部分是颈部网络，颈部网络对骨干网络中不同层的特征进行融合，提升模型的检测能力；第四部分是头部网络，该阶段负责最终结果的检测，在前面阶段生成的大小不一的特征图中预测不同的对象，预测过程采用 NMS 后处理算法移除冗余的候选框。

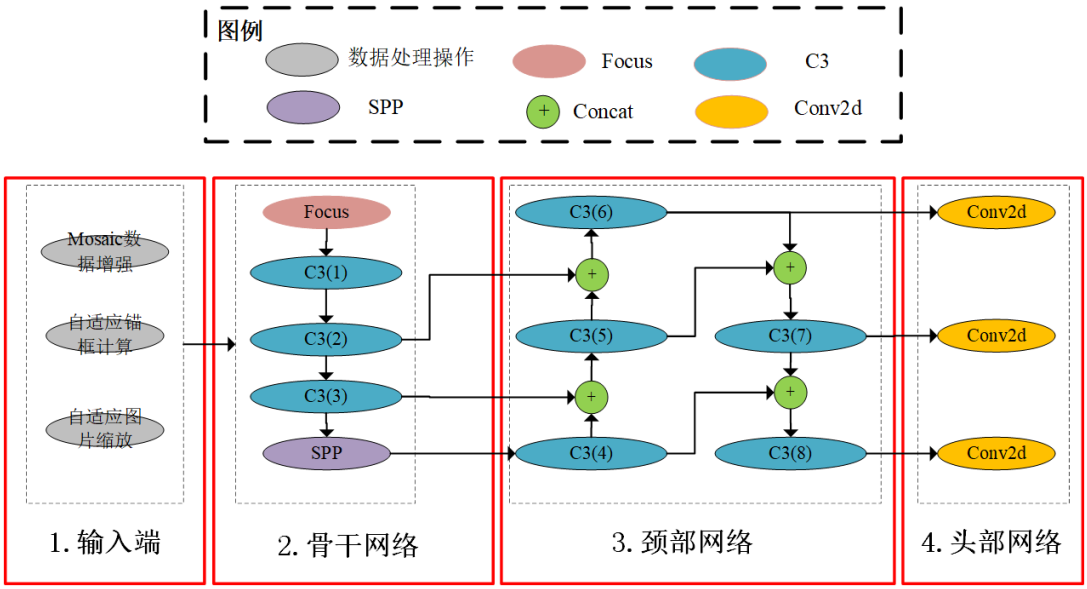

图 3 基于 YOLOv5s 的行为识别算法网络结构图

在 YOLOv5s 模型的搭建阶段，将输入端、骨干网络、颈部网络和头部网络按照一定组织结构连接起来，同时可在 yolov5s.yaml 文件中按需修改骨干网络和头部网络的结构，修改后可通过运行 yolo.py 文件查看网络结构。

基于 YOLOv5s 的群养猪行为识别结果统计如表 2 所示。其中，模型对饮水（drink）行为的识别精确率为 93.7%，召回率为 92.2%，F1 得分为 0.929，mAP@0.5 为 97.3%；对站立（stand）行为的识别精确率为 90.5%，召回率为 88.6%，F1 得分为 0.895，mAP@0.5 为 97.0%；对躺卧（lie）行为的识别精确率为 93.1%，召回率为 78.3%，F1 得分为 0.851，mAP@0.5 为 84.5%。对以上三类行为的识别结果进行汇总后，模型的平均精确率为 92.4%，平均召回率为 86.4%，平均 F1 得分为 0.893，平均 mAP@0.5 为 92.9%。在推理阶段，模型平均每张图片的推理时间为 2.475ms，即每秒能够识别约 400 张图片，在取得较高的精确率和召回率的情况下，识别速度极快，因此，该模型符合实际养殖场景下的群养猪实时行为识别需求。

从表 2 中可以发现，模型对群养猪躺卧行为识别的召回率偏低，为 78.3%，远低于对另外两种行为识别的召回率（比对饮水行为识别的召回率低 13.9%，比对站立行为识别的召回率低 10.3%），出现这种现象的原因可能是，群养猪在躺卧时喜欢聚集在一起，并且蜷缩成一团，存在严重的重叠和遮挡现象，因此漏检情况较为严重，导致对躺卧行为识别的召回率偏低。

表 2 基于 YOLOv5s 模型的群养猪行为识别结果统计

类别	群养猪个数	精确率(%)	召回率(%)	F1	mAP@0.5(%)
drink	65	93.7	92.2	0.929	97.3
stand	79	90.5	88.6	0.895	97.0
lie	69	93.1	78.3	0.851	84.5
合计	213	92.4	86.4	0.893	92.9

YOLOv5s 模型能够快速地识别群养猪的行为，但是重叠拥挤的场景对 YOLOv5s 模型识别躺卧行为是一个巨大的挑战，常规的网络结构难免会发生误检或漏检的情况。针对这些困难，**本项目将在后续工作进行模型结构调优**。

2.3 YOLOv5s 模型结构的调优

在模型结构的调优阶段，在深度学习平台中开始对模型进行训练，直到模型收敛，然后对模型进行测试，比较并调整模型结构，周而复始，直到得到适用于群养环境下密集拥挤的生猪检测场景的模型。

本项目在 YOLOv5s 基础上，提出两种改进群养猪行为识别方法。第一点改进为：将**坐标注意力模块**（Coordinate Attention，CA）融合在 YOLOv5s 的骨干网络后，试图对提取的特征图进行注意力重构，获取重要的行为特征信息，这样能够有效地提升群养猪某些行为的识别效果。第二点改进为：在前者的基础上，用 **DIoU-NMS** 后处理算法替换 NMS。该改进方法使得模型在筛选候选框的同时，既考虑候选框之间的 IoU，又考虑候选框之间中心点的距离，这有助于解决群养猪之间相互遮挡情况下的漏识别问题，进而提高模型的群养猪检测及行为识别性能。该改进模型记作**改进 YOLOv5s_CA**，其网络结构如图 4 所示。

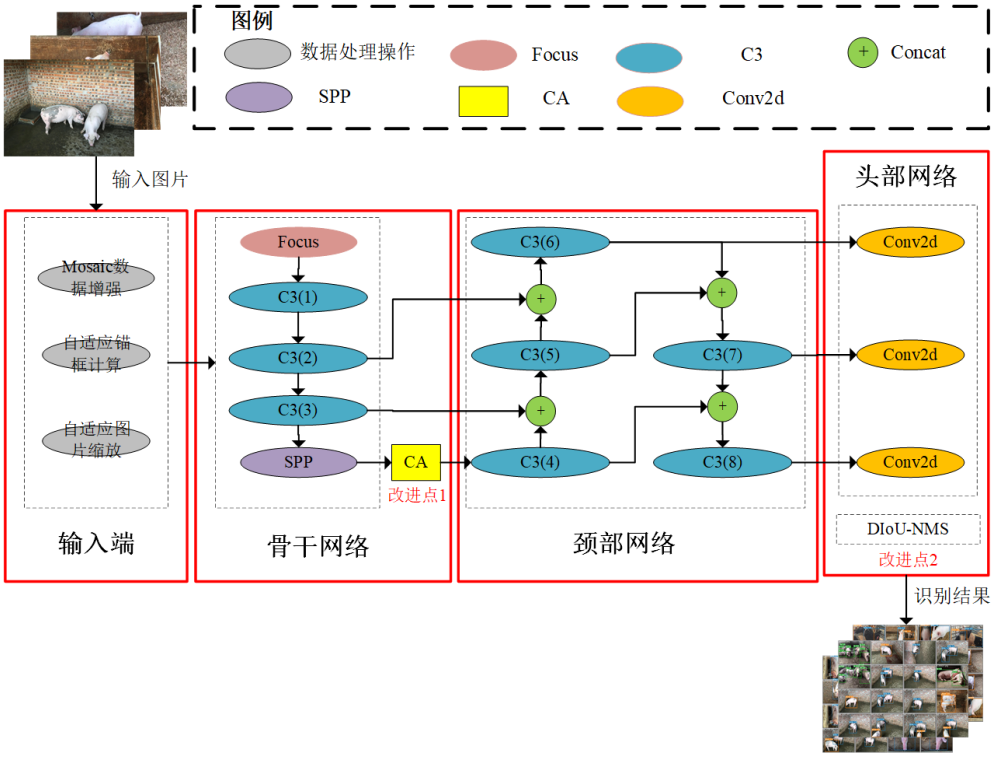

图 4 基于改进 YOLOv5s_CA 的群养猪行为识别模型结构图

为了全面评估所提出的对 YOLOv5s 的两种改进策略的有效性，在群养猪行为识别数据集上开展了**消融实验**。消融实验在训练过程使用相同的训练参数及超参数，根据所提出的改进策略，依次在 YOLOv5s 网络的基础上添加 CA、使用 DIoU-NMS 算法。表 3 为消融实验的结果，数值加粗则代表该指标是所有实验中最好的，原始的 YOLOv5s 模型取得了最高的精确率，但是召回率是最低的，分别添加 CA 和使用 DIoU-NMS 算法后，模型的召回率、F1 得分和 mAP@0.5 均有提升。但是由于 DIoU-NMS 算法需要额外计算候选框之间中心点的距离，因此在推理时间上有所增加，也就是花费部分时间的代价来提升模型的

性能，尽管如此，模型仍旧能够满足实时行为识别的要求。同时添加 CA 和使用 DIoU-NMS 算法后，模型的精确率和召回率达到了更高的均衡，获得了 F1 得分为 0.912，mAP@0.5 为 94.1%的最好结果，相比原始的 YOLOv5s 网络，F1 得分和 mAP@0.5 分别提升了 0.019 和 1.2%。

表 3 基于 YOLOv5s 网络的消融实验结果

CA	DIoU-NMS	精确率(%)	召回率(%)	F1	mAP@0.5(%)	推理时间
-	-	**92.4**	86.4	0.893	92.9	2.475ms/张
√	-	88.8	90.6	0.897	93.0	2.725ms/张
-	√	88.4	**93.1**	0.907	93.6	6.600ms/张
√	√	90.9	91.5	**0.912**	**94.1**	6.763ms/张

此外，为了比较改进 YOLOv5s_CA 模型与经典二阶段模型 **Faster R-CNN** 的性能，本文在相同的群养猪行为识别数据集上展开实验。训练 Faster R-CNN 模型前，将 batch_size 设置为 8，初始学习率设置为 0.01，训练次数为 8000 个 iter，训练结束时已经收敛。改进 YOLOv5s_CA 模型与 Faster R-CNN 模型的比较如表 4 所示，数值加粗则代表该指标是在对比实验中最好的，可以看出改进 YOLOv5s_CA 模型在精确率、F1 得分和 mAP@0.5 上分别比 Faster R-CNN 模型高 6.1%、0.007 和 2.4%，因此改进后的模型更适合于对群养猪行为进行识别，并且在速度上比二阶段模型有明显优势。

表 4 改进 YOLOv5s_CA 模型与 Faster R-CNN 模型比较

模型	精确率(%)	召回率(%)	F1	mAP@0.5(%)
Faster R-CNN	84.8	**93.1**	0.905	91.7
改进 YOLOv5s_CA	**90.9**	91.5	**0.912**	**94.1**

2.4 模型的部署

首先将训练得到模型转换为机器学习通用中间格式 **ONNX**，为后续 **ONNX Runtime** 部署奠定基础。导出 ONNX 需要声明输入图像的大小，本项目指定为 640×640。

在导出改进 YOLOv5s_CA 模型为 ONNX 格式时，CA 坐标注意力中使用了 **nn.AdaptiveAvgPool2d**，这个操作符在 ONNX 是动态的，**ONNX 暂时不支持导出**，报错如图 5 所示。因此，导出的是基于 YOLOv5s 的群养猪行为识别模型，并在后续的部署中使用该模型。

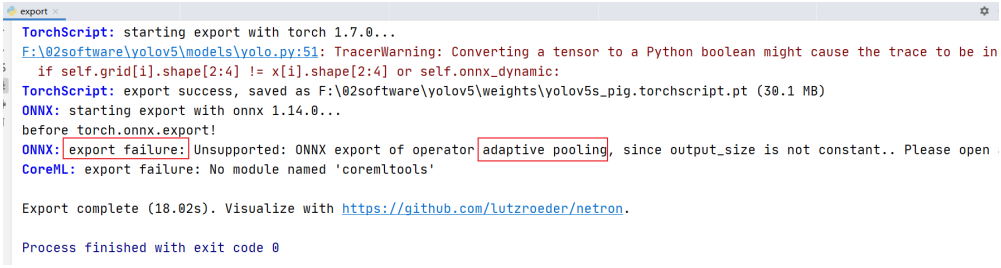

图 5 ONNX 不支持 nn.AdaptiveAvgPool2d 操作的导出

将模型导出成 ONNX 中间格式后，就可以使用 ONNX Runtime 进行推理。**本项目的部署平台是腾讯云，操作系统为 Ubuntu-20.04，服务器具有 2 核的 CPU 和 2GB 的内存**。该硬件配置属于**消费级别**，因此项目有较强的复现性和落地性。为方便用户的使用，本项目最终以网站平台、微信小程序和 PyQt 应用软件三种不同的形式进行部署。

2.4.1 网站平台

在网站平台的实现上，实验采用 **Vue.js** 作为前端框架，使用轻量级的 **Flask 框架**编写调用 YOLOv5s 模型的接口，采用前后端分离的方式进行网站平台的开发。其中，使用 Flask 框架编写各个接口的功能如表 5 所示，部分 Flask 后端接口的浏览器展示如图 6 所示。

表 5 Flask 框架编写的接口功能展示

接口	功能
http://1.12.231.219:8083/demo	上传图片并进行推理识别的 demo
http://1.12.231.219:8083/upload_image	上传图片
http://1.12.231.219:8083/results/<imageId>	查看图片
http://1.12.231.219:8083/yolo	调用 ONNX 模型进行推理

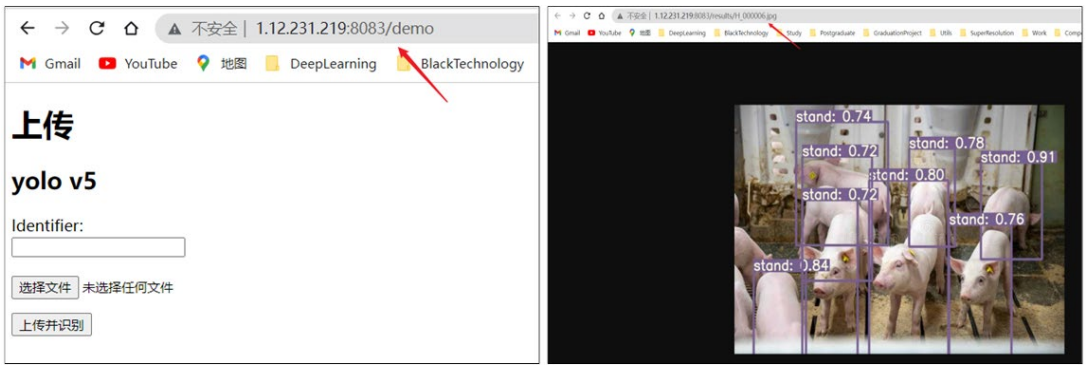

图 6 部分 Flask 后端接口的浏览器展示

使用 Vue 前端框架编写的网站平台的部署网址为：http://1.12.231.219/，首页如图 7 所示。如图 8 所示，将网页下拉到底部，用户可以点击“选择文件”的按钮上传群养猪图片，并点击“行为识别”。Vue 项目的打包部署已上传至谷歌云端硬盘 https://drive.google.com/fil

e/d/17CWt7FCrEEd5Sz6-ndRPeLlh911-gtUr/view?usp=sharing。

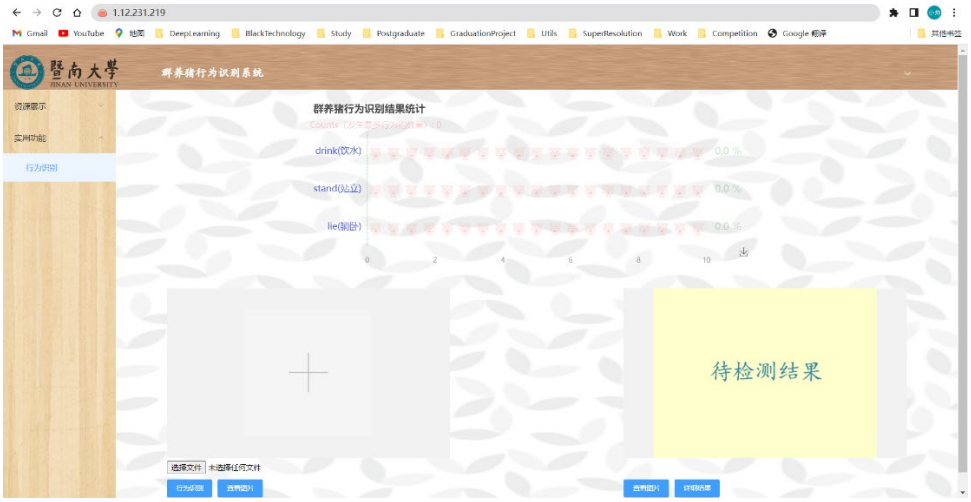

图 7 网站平台首页

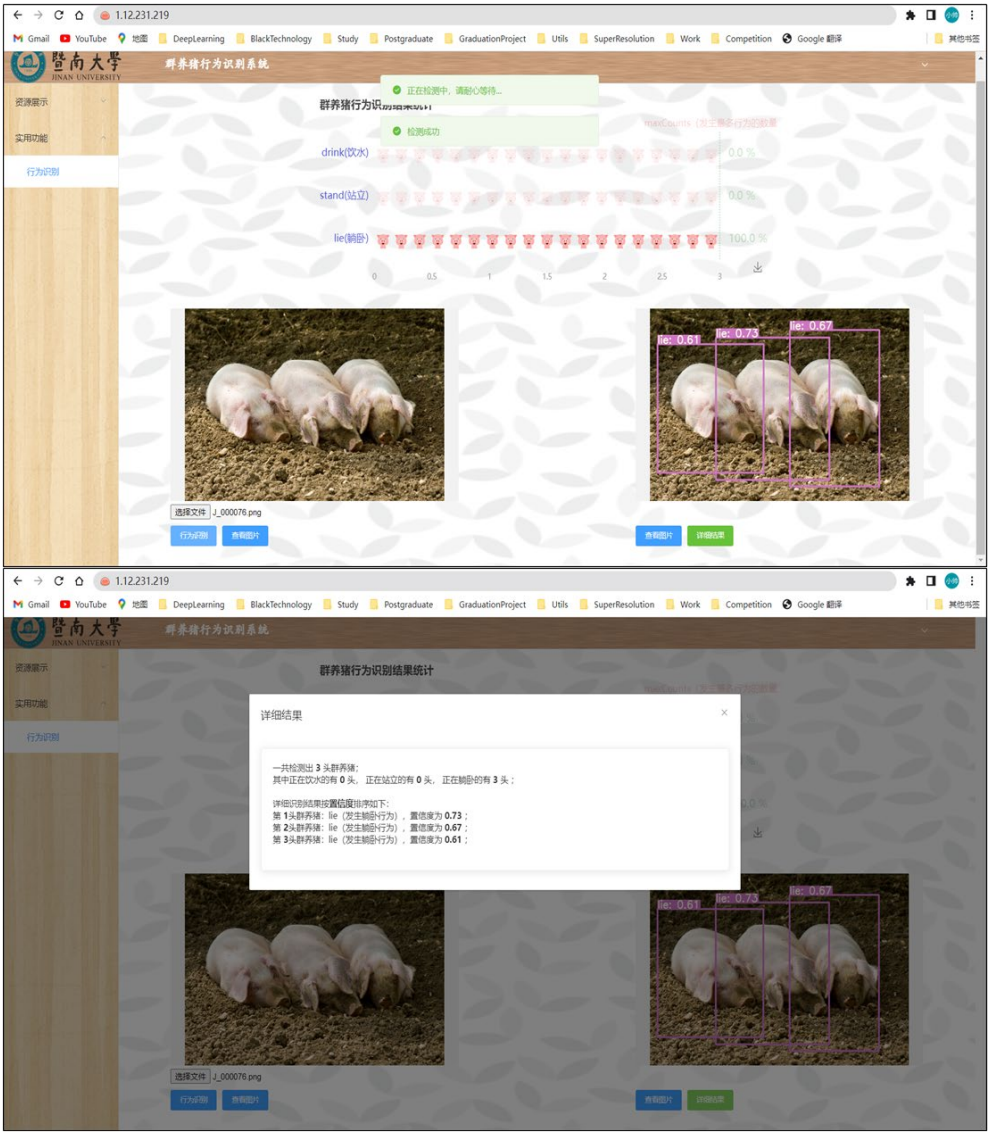

图 8 网站平台操作展示

在部署网站平台时，使用的是 **Nginx** 中间件做的反向代理。我之前在 CentOS 上部署过前端项目，但是因为 Ubuntu 对于 Python 比较友好，因此使用是在 Ubuntu 上部署的。两者的部署会有些细微的差别，比如 Nginx 配置文件的位置会不一样，但是本质上还是一样的，都需要通过修改配置文件，重启 Nginx 来部署 Vue 网站平台。

2.4.2 微信小程序

在微信小程序的实现上，实验使用微**信开发者工具**进行开发，后端采用 Flask 框架进行编写，以前后端分离的方式完成该项目设计与开发。由于需要域名才能上线小程序，因此微信端的部署仅限于本地开发，之后有充足的经济支持会考虑上架小程序。微信小程序的界面如图 9 所示，界面划分为“登录”、 “行为识别”和“个人中心”三个部分。图 9（1）为授权登录界面，用户只有授权登录后，才能够使用微信小程序的全部功能。图 9（2）为群养猪行为识别界面，该界面的操作会在后面详细说明。图 9（3）为个人中心界面，点击“联系作者”能够复制开发者的博客链接，点击“退出登录”将会退出微信的授权登录。。

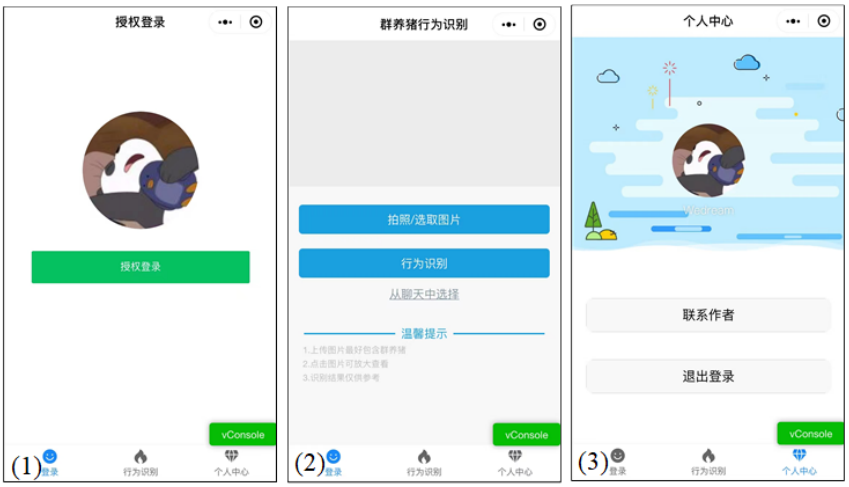

图 9 微信小程序界面展示

在群养猪行为识别界面，用户可以点击如图 9（2）所示的“拍照/选取图片”按钮上传图片，图片上传成功的界面如图 10（1）所示。点击“行为识别”按钮，界面将会显示群养猪行为识别的结果，如图 10（2）所示，点击图片还可放大查看识别的结果图片。

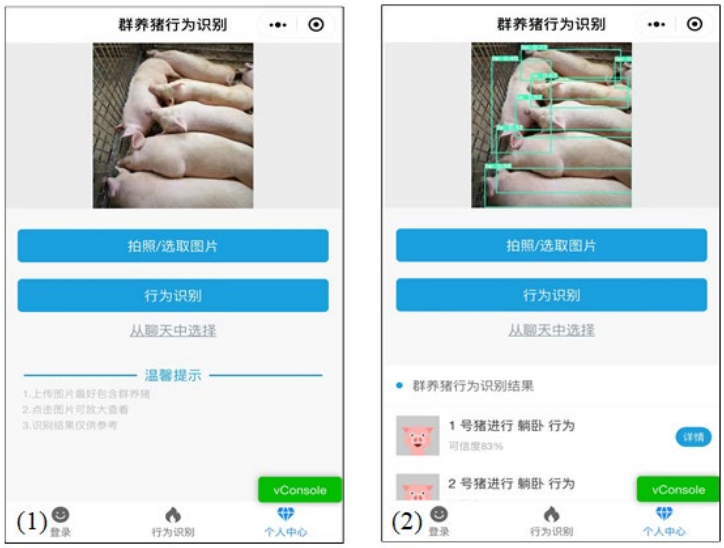

图 10 微信小程序操作展示

用户可在图 10（2）的界面中下划，查看识别结果，如图 11（1）所示，包含行为识别及置信度等信息。点击“详情”按钮，界面将会显示如图 11（2）所示的行为识别详情。

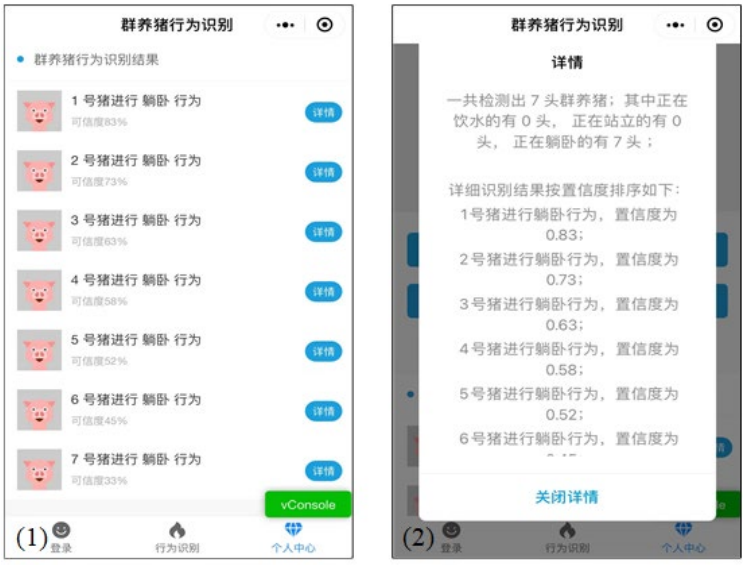

图 11 微信小程序行为识别结果展示

2.4.3 PyQt 应用软件

在应用软件的实现上，实验采用 Python 和 **PyQt5** 完成对群养猪行为识别检测系统的开发，借助界面设计辅助工具 QtDesigner 完成应用软件的界面设计工作。应用软件的主界面如图 12 所示，界面划分为五个模块，分别为“图片显示模块”、“图片调整模块”、“图片信息模块”、“识别结果模块”和“功能按钮模块”。PyQt 应用软件的部署 exe 文件已上传至谷歌云端硬盘 https://drive.google.com/file/d/1CWKLvpxws9CyfnINqJQ0MoXvYblgpaRo/view?usp=sharing。

图 12 应用软件主界面

群养猪行为识别检测系统（应用软件）的操作步骤如图 13 所示。

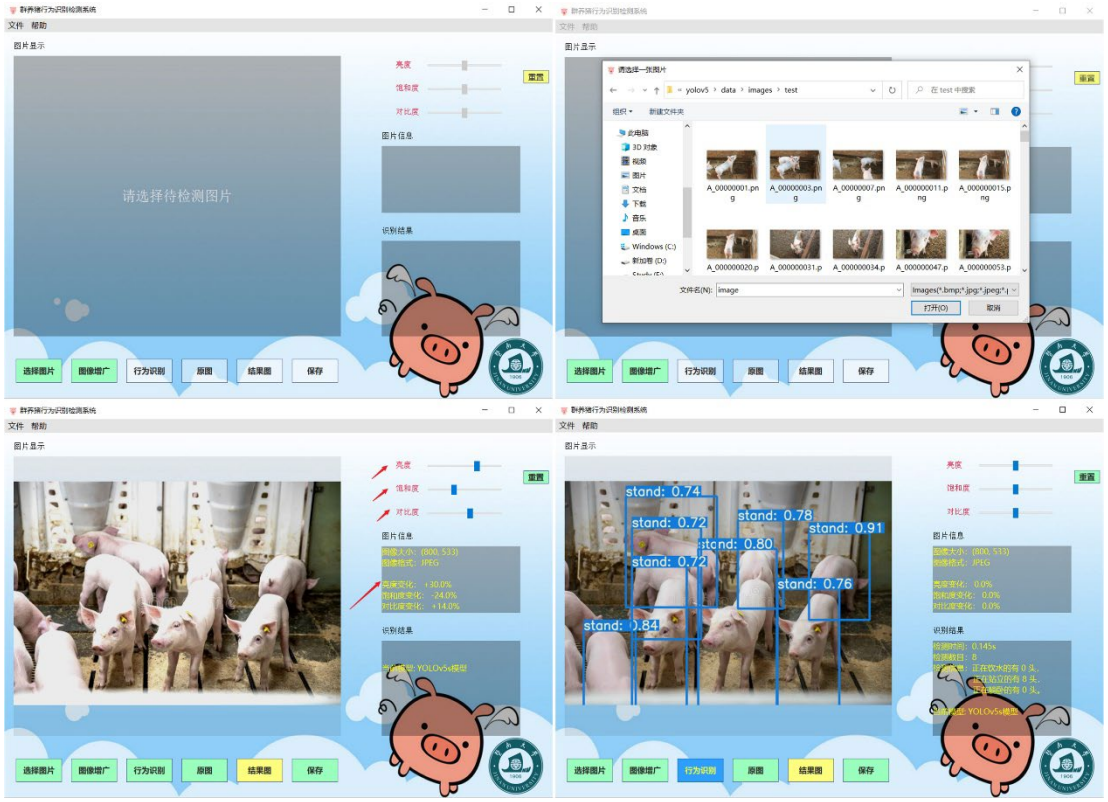

图 13 应用软件操作步骤展示

3 项目总结

本项目通过在生猪养殖场拍摄和网络爬取的方法获取群养猪行为识别数据集，将群养猪的行为分为饮水、站立和躺卧三个类别，按一定比例随机划分数据集用于模型的训练和测试。为了提升对某些行为的识别效果，提出引入 CA 到 YOLOv5s 中，用于获取重要的行为特征信息。**实验结果表明 CA 能够提取行为表征能力更强的特征，缓解部分行为漏识别的情况。**在前者的基础上，**通过将后处理算法 DIoU-NMS 替换掉原本的 NMS，有助于解决群养猪之间相互遮挡情况下的漏识别问题。**实验结果表明，改进 YOLOv5s_CA 模型对群养猪行为的识别更为全面，鲁棒性更强。

在算法部署方面，本项目以网站平台、微信小程序和应用软件三种形式进行部署，涉及的实现技术有 **Vue.js、Flask、ONNX、ONNX Runtime、微信小程序开发**和 **PyQt** 等，极大地提高了我的动手能力，也丰富了我的系统部署知识。当然，这些过程不可能是一帆风顺的，在部署过程中，我遇到了部分操作符在 ONNX 中不支持导出的问题，遇到了由于 PyTorch 版本原因导致的报错，遇到了使用 PyInstaller 将 PyQt 应用程序导出的 exe 不能使用的问题……但是，俗话说得好，兵来将挡水来土掩，这些问题都在我的一步一步调试中得到了解决。比如说导出的 exe 应用软件不能使用的问题，我先使用命令生成带终端窗口显示的 exe 文件，运行软件，观察终端输出的信息，发现是缺少一些图像、模型权重和配置文件等外部资源，而这些资源，由于某种原因，使用 PyInstaller 时并不能将这些外部资源打包进来，因此导出的 exe 不能使用。于是，我将缺失的外部资源文件一一复制到 exe 所在的目录中，最终 exe 顺利启动。

此外，通过这次项目整个流程的实现，从**收集数据**，到**训练模型**，再到**改进模型**，最后

到**部署模型**，我学到了很多东西。比如说**机器学习通用中间格式 ONNX 真的是一个伟大的发明，使得算法部署具有很强的迁移性，开发者不仅可以使用 Python 语言进行部署，当有推理高效率的需求时，也可以使用 C++这种偏底层硬件的语言进行部署，效果会更好**。总的来说，这次实验让我系统性地感受到了算法系统从零诞生到广泛使用的过程，收获颇丰，受益匪浅。