编　　号：2018212848

审定成绩：

重庆邮电大学

毕业设计（论文）

中文题目	基于 CNN 的蓝莓僵果病识别系统设计
英文题目	**Design of Blueberry Stiff Fruit Disease Recognition System Based on CNN**
学院名称	自动化学院/工业互联网学院
学生姓名	冉浩宏
专　　业	物联网工程
班　　级	08051804
学　　号	2018212848
指导教师	屈洪春　教授
答辩组负责人	王浩　教授

2022 年　6 月

重庆邮电大学教务处制

自动化学院/工业互联网学院本科毕业设计

(论文)诚信承诺书

本人郑重承诺：

我向学院呈交的论文《基于 CNN 的蓝莓僵果病识别系统设计》，是本人在指导教师的指导下，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，本论文不含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明并致谢。本人完全意识到本声明的法律结果由本人承担。

年级

专业

班级

承诺人签名

年　　月　　日

学位论文版权使用授权书

本人完全了解重庆邮电大学有权保留、使用学位论文纸质版和电子版的规定，即学校有权向国家有关部门或机构送交论文，允许论文被查阅和借阅等。本人授权重庆邮电大学可以公布本学位论文的全部或部分内容，可编入有关数据库或信息系统进行检索、分析或评价，可以采用影印、缩印、扫描或拷贝等复制手段保存、汇编本学位论文。

（注：保密的学位论文在解密后适用本授权书。）

学生签名：　　指导老师签名：

日期：　　年　　月　　日　　日期：　　年　　月　　日

摘要

中国是一个大型农业国家，并且农业早就已是中国的国家经济进步和繁荣的砥柱，是中国最重要的产业之一。但是农产品的病虫害问题在很长一段时间都持续地影响着农业的不断进步，每一年都会因为农产品病虫害问题造成很大的损失。深度学习技术可以用于识别植物疾病，这不仅可以提高识别的准确度和速度，而且有助于农业的数字化发展。

本论文主要针对蓝莓僵果病，即叶片、花朵和果实的正常和病态，结合已有的理论，利用卷积神经网络的学习，实现了蓝莓僵果病的识别，使其在农业生产中的应用更为广泛。本文的主要工作包括：

1）通过高斯模糊、色彩加强、直方图正规化、几何变化、伽马变换这 5 种图像处理方式，对已有的原始数据集进行图像预处理，从而进行原始数据集的扩充。

2）通过 Make Sense 图像标注工具对扩充数据集进行特征标注，完成数据集预处理。

3）根据 yolov5 算法，学习并设计了卷积神经网络模型来对蓝莓僵果病进行识别，模型使用 4200 张蓝莓图像进行训练，共分为叶、花、果的正常与感染 6 类，并使用 1800 蓝莓图片作为验证集。训练后模型的最低准确率为花正常，达到 92.6%，最高准确率为花感染，达到 99.5%。

关键词：蓝莓僵果病，卷积神经网络，深度学习，图像分析，目标分类

Abstract

China is a large agricultural country, and agriculture has long been the mainstay of China's national economic progress and prosperity, and one of the most important industries in China. However, the problem of pests and diseases of agricultural products has continued to affect the continuous progress of agriculture for a long time, and every year will cause great losses due to the problems of pests and diseases of agricultural products. Deep learning technology can be used to identify plant diseases, which can not only improve the accuracy and speed of identification, but also contribute to the digital development of agriculture.

This paper will focus on the blueberry stiff fruit disease, that is, the normal and diseased conditions of the leaves, flowers and fruits of the blueberry plant as the research object. The method of disease identification and the more diverse applications of convolutional neural networks in agriculture. The main research contents are as follows:

1）Through Gaussian blur, color enhancement, histogram normalization, geometric change, and gamma transformation, image preprocessing is performed on the existing original data set to expand the original data set.

2）Use the Make Sense image annotation tool to perform feature annotation on the expanded dataset to complete the dataset preprocessing.

3）According to the yolov5 algorithm, a convolutional neural network model was learned and designed to identify blueberry stiff fruit disease. The model was trained with 4,200 blueberry images, which were divided into 6 categories of normal and infected leaves, flowers and fruits. 1800 blueberry images are used as validation set. The lowest accuracy of the trained model is 92.6% for flower normal, and the highest accuracy is 99.5% for flower infection.

Keywords: Blueberry stiff fruit disease， convolutional neural networks， deep learning，image analysis，object classification

目录

第 1 章 引言

1.1 研究背景和意义

中国是一个传统的农业国，尽管中国自改革开放以后，各方面产业空前发展，农业也空前发达，但中国作为一个农业大国，与世界各国的差距还是很大的。尽管中国已经积累了大量的农业生产经验，也取得了明显的成绩，但是中国农业智能化水平和信息化水平还存在着很大的差距。尤其科学技术是中国农业发展的一个薄弱环节。

纵观历史，不难看出，一个国家、一个民族的兴衰，与农业的发展密不可分，农业、工业和服务业是中国经济发展的重要支撑，农业为中国 14 亿人口提供了保证。但中国在实现农业可持续、平衡发展的同时，也遇到了这种困境。

中国耕地面积占全球的 7%，位居世界第三[1]，但是，因为中国有 14 亿人口，每个中国人均耕地的面积比世界平均水平要低得多，那么如何才能把全世界 7%的土地用于供养 20%的世界人口呢？中国现在正面对着这个问题，而且还会持续很长一段时间。但是，由于我国的耕地面积较大，品种较多，因此，单靠人工进行病虫害识别，其工作效率已远远达不到农业生产的要求[2]。

放眼望去，到处都是人工智能，很多工作都是人类无法取代的，在这个飞速发展的世界后面，深度学习是一个重要的技术，它为我们带来了强劲的发展。世界上很多学者都对深度学习赞不绝口，把它称为一项革命性的技术，有些人甚至把它看作是最近十年的一次重大突破。实际上，“深度学习”一词频繁地出现在报刊和杂志中，引起了广泛的注意，连一般民众都听说过[3]。

而卷积神经网络作为一种典型的深度学习算法，在高科技行业中得到了广泛的应用。卷积神机网络在很多方面都有很大的应用，比如单帧图像识别，比如视频的处理，比如对自然语言的理解，比如卷积神经网络。自从 90 年代中期以后，CNN 就一直在进行理论和实践的融合，但是一直没有掀起一场学术风暴，在 21 世纪初，CNN 已经安静了十多年，而在 2012 年，AlexNet 的一鸣惊人，使 CNN 再次走上了风口浪尖[4]。

1.2 国内外研究现状

1.2.1 国外研究现状

自八十年代我国电脑技术起步之初，国外已开展了以电脑技术为基础的农作物病害诊断与鉴定工作。例如，在 1985 年，通过计算机图像技术，对小麦种子进行了相关的识别。

1987 年，Mayer[5]等利用图像处理技术对作物的形态特性进行了研究，并以此为基础，建立了基于数据的作物生长数学模型。也是在这一年，Whittaker[6]等利用数字图像技术对许多农产品进行了识别，因为它们的形态特征差异很大。

两年后的 1989 年，Slaughter[7] 利用电脑视觉技术和自动机器技术，利用电脑对橘子的位置和成熟度进行识别，利用自动采集设备采集橘子，既能提高采收的效率，又能保证成熟的果品的正确率达 75%。

1997 年，Zayas[8]的研究团队使用了一种机器视觉技术来发现有没有有害生物，而且准确率超过 85%。

1993 年，Malthus[9]利用光谱仪对大豆的侵染进行了研究，利用光谱仪检测了被侵染的大豆叶片的辐射光谱，从而实现了对大豆的侵染情况的监控。

2002 年，Delwiche[10]等人利用高广镨图像和计算机视觉对小麦的病害进行了研究。

Kulkarni 等[11]在 2008 年应用了基于神经网络的图像处理技术，通过训练 ANN 的数据采集、分割、提取和模型训练，该方法的识别准确率达到 91%。

Habib[12]将神经元模糊理论应用于棉花田间害虫的识别与分类中。

近年来，植物病害的鉴定越来越受到重视。Montalvo 等[13]利用化学成分的方法，把玉米和野草分开。

瑞士洛桑联邦技术学院的 Mohanty[14]利用 GoogleNet 的卷积神经网络结构，对昆虫进行了分类，从而获得了良好的识别效果。

1.2.2 国内研究现状

我国农业病虫害的诊断和鉴定技术相对滞后，在 21 世纪初，我国农业病虫害的诊断和鉴定工作还处于起步阶段。中国人具有很强的学习和创造精神，因此，在国外的研究基础上，国内有关领域的研究也有很好的结果[15]。

沈佐锐、于新文等[16] 在 2001 年，科学家们首先应用了图像处理技术来研究昆虫的影像。沈佐锐等采用噪声消除、边缘检测和梯度运算等方法，对大棚内的白粉虱种群进行了动态监测，从而使其识别精度得到了明显的改善。

陈佳娟[17]等，以计算机视觉为主，在此基础上，采用了基于边界追踪与扩展的方法，对叶片的空洞和边缘残缺进行了检测，并根据二者之间的差异大小来判定其损伤程度[18]。用普通的叶片作为匹配模板进行边缘缺陷的识别，但是由于叶片本身的形态差异，无法有效地进行边缘缺陷的检测[19]。

梁子安等人[20] 在收集了 5 个总科 23 种有害作物的病虫影像后，采用人工统计和神经网络模式识别相比较[21]，实验表明，该算法的识别效果和传统的分类效果是一样的，无论是在识别速度还是准确度上都要好于人工统计[22]。

近几年，随着农业信息化的快速发展，在深度学习、计算机视觉、图像处理等方面都有了很大的发展。尹显东等人[23]使用深度学习技术，例如，在 2019 年，ResNet50 和 ResNet50 将 FPN 技术与传统的深度学习方法相结合，构建了一种基于深度学习的作物病虫害智能识别技术，通过对所抽取的特征进行回归分类，对农作物病虫害进行智能识别。

而胡明越[24] 经过 10 个类别的 1593 个不同类型的树的不同颜色的图像进行了 10 次增强，然后利用 AlexNet、VggNet-16 等 4 个典型的卷积神经网络进行了训练，并将训练的结果进行加权平均，从而实现了一个基于树的分类识别的系统。

史冰莹[25]等采用卷积神经网络的迁移学习方法，对 11 种类型 28 种病害 58 种进行了分类，并利用典型网络建立了一种识别模型，具有 86%的准确率。通过对系统的损伤函数进行优化，使其正确率可达 86.5%。

同年，刘志勇[26] 在此基础上，作者提出了 DropOut 策略和 PRELU 激活功能等方法，对传统 leNet-5 网络进行了改进，该网络在 142800 个采样点上进行了学习，得到了 95.3%的准确率和准确率。

随着物联网技术的不断发展，智能农业的应用也日益广泛。在 2021 年，高德民等人[27] 在农业的疾病监控中，采用了物联网和无人机技术，通过构建光照模式，让农田内的太阳能面板一直处于太阳光的正下方，既可以增加太阳能的利用率，又可以有效地降低农田的电力消耗。而秦彩杰等人[28] 同时还将图像处理、目标检测等技术与无人机技术相结合[29]，对农产品的识别精度和效率都达到了智能化的要求。

1.3 主要研究内容

本论文的主要工作是学习卷积神经网络的基本原理，并设计和开发了一个用于检测和鉴别蓝莓僵果病的卷积神经网络识别系统。通过阅读大量的文献资料和相关的视频解说，进一步加深了对卷积神经网络的理论结构的认识。经过反复的实验、测试和修改，最后得到了一个符合实验要求的卷积神经网络识别系统。本文的主要工作包括：

1）数据集预处理。针对数据名称、数据大小不统一、数据集稀少、数据数量不平衡等问题，本文提出了一种基于频域、颜色、几何空间等概念的预处理方法。基于这一点，选取 6000 张不同类型的影像资料，以 7:3 的比率，训练并验证了卷积神经网络。在此基础上，利用 Make Sense 的图象标记技术，实现了数据集的预处理。

2）卷积神经网络训练。采用 YOLOv5s 网络模型，并针对识别对象蓝莓植株的特征及类别，对经典 YOLOv5s 网络模型进行部分修剪，使其具有更好的适应性，构建的模型更加轻量化，从而实现蓝莓僵果病识别系统。

3）模型分析与指标评价。在对 4200 份样本进行 100 轮的训练和学习后，所建立的卷积神经网络对 1800 张图片的预测精度达 96.6%，并且对 300 张不同类型的图片的预测精度都在 92.6%以上。除了正确率以外，还通过多种方法对模型的预测进行了分析，从而从多个方面对该模型的可信度进行了检验。

第 2 章 卷积神经网络的基本介绍

卷积神经网络是一种卷积运算，其优点在于卷积运算。相对于其它神经网络和其它深度学习方法，卷积神经网络有如下优点：一是可以进行卷积运算，利用卷积运算来抽取输入数据的特征，从而使模型在更深的层面上得到更高效的输入；其次，采用了层次分明的层次结构，便于学习、修改和维护；在此基础上，引入了一种前向神经网络，简化了算法的流向，方便了算法的实时更新。这些优势和对数据的提取能力，这使得卷积神经网络在目前的深度学习领域中得到了许多应用[30]。

卷积神经网络因其局部连通、权重分享等优势而被广泛地用于计算机视觉中，而且还能减少参数的计算，而且卷积神经网络具有更小的体积和更快速的训练速度，卷积神经网络在大规模的数据处理中有着无可比拟的优越性。

2.1 历史发展

从最初的理论知识被引入到现在的研究领域，其发展历程也仅为六十年。按照时间顺序，卷积神经网络的发展历程可以分为初步探索、全面兴起以及跃进式发展三个阶段[31]。

1）初期勘探期

令人难以置信的是，现在已经成为研究热点的卷积神经网络，其研究的源头竟然只是一个巧合。在 1962 年，Hubel 和 Wiesel[32] 将电极插入猫的脑内，让猫咪观察图像，然后测量电流、电压等电气参数的改变，从而了解猫的大脑对图像的刺激。然而，在一次偶然的观察中，电子参数发生了明显的变化，最后，休伯特和维塞尔改变了研究方法，从静止到动态，最后得出了一个结论，那就是感知场的概念。

他们在视觉资讯处理上的卓越成就，使他们在一九八一年荣获诺贝尔生理或医学奖。到了 1980 年，Fukushimel[33]在卷积神经的概念中，我们提出了卷积神经元，这个时候，卷积神经层的概念就是它的权值分享，而后面的卷积神经网络则是基于卷积神经。在 1989 年，LeCun[34]利用逆向传递和加权分享的方法，设计和开发了一种新的卷积神经网络，并在美国邮政公司的对写字符识别中应用。

2）全面兴起阶段

LeCun[35]等人建立了一个经典的 LeNet-5 卷积神经网络网络模型，并在此基础上引入了梯度逆向传播，使其在实际操作中的正确性得到了进一步的改善。而 LeNet-5 模型所提出的“权重分享”，则是在 2010 年早期的卷积神经网络中得到了广泛的应用。

3）飞跃发展时期

Kichevsoy 等[36]2012 年推出的 AlexNet 网络大行其道，AlexNet 在 ImageNet 的大规模视觉识别大赛中获得了压倒性的胜利，AlexNet 在图片识别方面的精确度高出了 11 个百分点。在 2014 年，Szegedy 等人[37] 推出 Google Net 以扩大网络的深度和广度；同年，Simonyan 等人[38] 本文着重探讨了网络深度对网络的重要性，并给出了 VGGNet 模型。并且，这两款模型在 2014LSVRC 的影像分类大赛中，都获得了一等奖和二等奖。卷积神经网络在近几年的发展中得到了飞速的发展，它不断地和传统的算法结合在一起，并且由于它的引入和计算机的运算量得到了极大的提高。

2.2 网络结构

卷积神经网络虽然在网络中有着很高的层次，但是其网络结构非常简单，通常是卷积层、池化层、归一化层以及全连接层。

2.2.1 卷积层

卷积层是由人工神经网络模拟大脑对图像进行识别的一种方法，它具有一定的局部性。在对图像进行识别时，大脑会先将图像的各个区域进行感知，然后将这些区域的特征结合起来，这样才能全面地感受到整个画面。卷积层的主要作用是对图像进行局部识别，通过逆向传播可以优化每个卷积核的参数，从而提高了图像特征的抽取精度。为了防止卷积核数过多，使模型参数急剧增加，且由于卷积运算所造成的计算量较大，因此，每个卷积层都采用了局部链接和权重分享的方法，这种算法不但可以解决以上问题，还可以大大提高卷积的效率。

在卷积层中，通过设定卷积核的大小、形状和运动模式，可以对原始图像进行提取，并对其进行密度控制，最后获得形状图的形状大小，并增加卷积核的数目，

以获得更多的特征信息。在图 2.1 中，利用卷积方法对输入的数据进行卷积，可以得到更多的特征信息，在一定程度上减少了数据的数量，并为下一阶段的网络层提供了更加有效的数据输入。

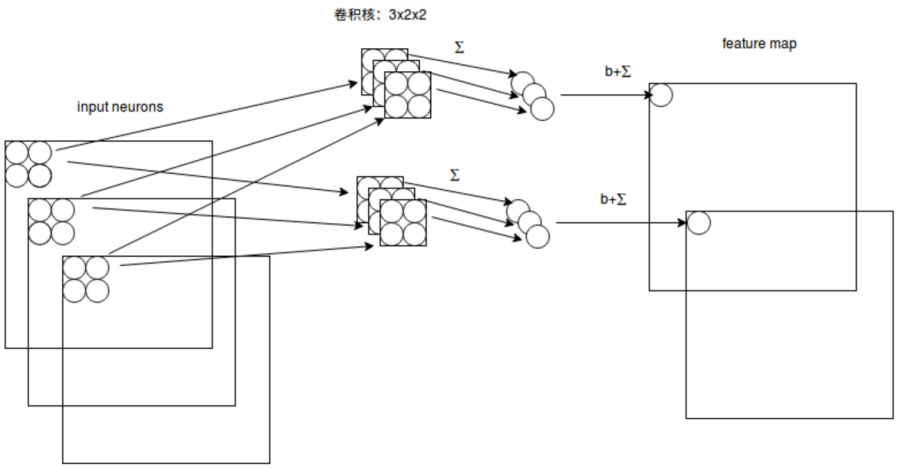


图 2.1 卷积行为示意图

2.2.2 池化层

池化层又叫下取样层，它的工作原理是把特征图分成若干小块，在每个小块中选取或求出一组数值，以取代某一区域的资料，通常采用最大和平均两种方式，即最大和平均。该方法能有效地压缩输入的特征信息，减少参数和参数，并能从输入的特征图中选择特征进行后续处理，有效地解决了过拟合问题。在进行池化时，还可以设定池的尺寸和步长，从而实现对输出特性图的尺寸的控制。池化过程见图 2.2。

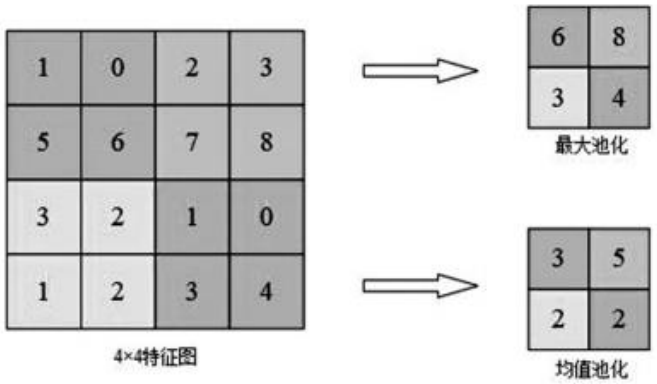


图 2.2 池化行为示意图

2.2.3 归一化层

Batch Normalization，简称 BN，也就是“批归一化”，是神经网络中的一种特别的层次，目前已经成为许多主流网络的标准。

将每个神经元的输入值都拉回一个平均为 0 的正态分布，这就是将一个不断偏离的分布拉回一个相对标准的分布。

在 CNN 中，BN 应该在非线性映射之前起作用。在神经网络学习过程中，当学习过程中出现了一些不能训练的情况，如梯度爆发等情况时，可以采用 BN 方法进行求解。此外，还可以在常规应用中增加 BN，从而提高了模型的精度。BN 更适合于以下情况：每一个微型电话更大，并且数据分布更紧密。在练习之前，必须要有足够的准备，不然的话，训练的效果就会大打折扣。此外，BN 在运行时必须对每一个微型节点进行一阶统计和二阶统计，因而不适合于 RNN 网络和动态网络结构。

BN 的功能是将输入分布从一个渐近的非线性函数映射到一个数值区间的极值饱和区，强行将其拉回一个平均方差为 1 的正态分布，从而使该非线性转换函数的输入值进入一个对输入非常敏感的区域，从而避免了梯度消失问题。因为梯度总是保持在一个较大的范围内，所以很显然，神经网络的参数调节效率更高，但变化更大，这意味着，最优的损失函数的步长，也就是最小的收敛速度。

2.2.4 全连接层

全连通层除包含完整的连结神经元外，还包含活化功能。激活函数一般为非线性，其中最常见的是 Sigmod 函数、Tanh 函数和 ReLU 函数。

从池化层获得的样本数据一般都是线性的，因为采用了卷积层，但是实际的样本表与样本表的相关性却是非线性的，采用线性变换方法所获得的结果往往是不可靠的，而激发函数作为非线性函数，恰好能够将样本的特征信息转化成样本标记，并利用多个非线性函数拟合出任何函数，从而使得触发函数在整个连接层中起到了非常关键的作用。

Sigmod 函数的图象是一条简单、易于求导、易于求导的单调上升曲线，易于在模型逆向传播时进行梯度求解，但若导数太低，则会导致梯度的消失。

Tanh 函数的图象与 Sigmod 函数相似，由函数公式可知，Tanh 函数为 2 xsigmod (2 x)-1，所以有一个关于梯度消失的问题。

与 Sigmod、Tanh 等算法进行了比较，结果表明，ReLU 函数并不具有渐近消失的特性，并且该函数可以将数据置零，并保持不变的单向抑制特性，从而大大降低了运算的工作量。

2.3 经典网络模型

随着卷积神经网络技术的发展，卷积神经网络的研究和开发也越来越多，一些典型的卷积神经网络模型也在不断地发展，例如 LeNet、PreresNet、Google Net、DenseNet 等。

2.3.1 LeNet

LeNet5 是 1994 年问世的第一批卷积神经网络，它在深度学习领域中起到了很大的促进作用。从 1988 年起，经过几年的研究和多次成功的反复试验，这个先驱者的结果被称为 LeNet5。

1989 年，Yann LeCun 在贝尔实验室的一项研究中，第一次将逆向传输算法用于实践，提出了一种基于任务域的限制，可以极大地提高学习网络的普遍性。在此基础上，给出了一种新的基于任务域的约束方法，从而使网络的通用性得到了很大的改善。他把“手写”数字和反向转移算法结合起来，并在美国邮政系统中应用了。即便这个问题是线性可分解的，单层次网络仍然具有很低的推广性能。LeNet 网络结构如图 2.3。

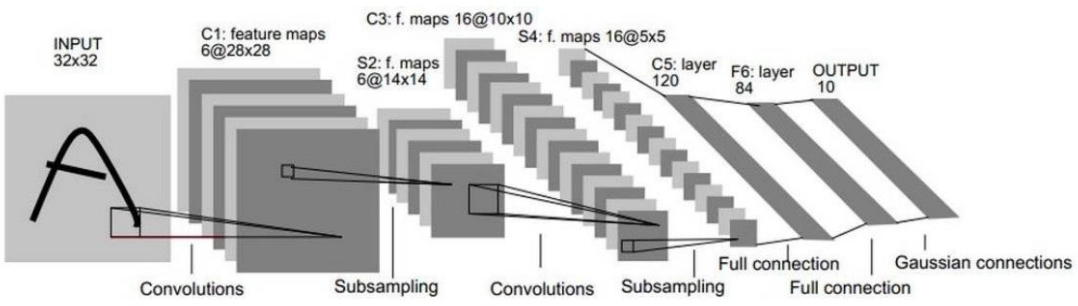


图 2.3 LeNet 模型结构图

LeNet-5 包含 7 个层次，其中没有输入，每个层次都包含了一个可训练的参数（加权），这个时候所用的是 32*32 个象素的图片。LeNet-5 的构造将在下文中逐层描述，然后用 Cx 来表示卷积层，用 Sx 来表示次抽样层，用 Fx 来表示完整的连接层，这里 x 是层的索引。C1 为卷积层，其尺寸为 28*28，其 6 个 5*5 的卷积核，从而避免了从卷积核边缘掉落的信息。

C1 包括 156 个可培训的参数，122304 个链接.层 S2 是一个子取样层，其输出 6 个尺寸 14*14 的特征图。在 C1 中，每一个特征映射都与 2*2 相邻的区域相关联。

在 S2 中，将四个输入单元相加，并将其与可训练因子（加权）相乘，并将其添加至可训练偏差（bias）。结果是用 S 型函数进行的。因为 2*2 个感知区域不交迭，所以 S2 的特征图仅包含 C1 特征图的一半行号和列号。S2 层包含了 12 个可供学习的参数，以及 5880 个连接。

C3 为卷积层，其卷积核心为 16 个 5~5。第一个 C3 特征图的输入为 S2 的三个特征图的每一个相继的子集，后六个特征图的输入分别是四个相继的子集，后三个特征图的输入分别是四个不连续的子集。最后，从 S2 的全部特性曲线中提取出最终的特征曲线。C3 层的可培训参数 1516 条，连接 156,000 条。

层 S4 与 S2 相似，具有 2×2 的尺寸，并且具有 16 个 5×5 的特性曲线。S4 层包含了 32 个可培训的参数，以及 2000 个连接。

C5 是一个卷积层，它有 120 个 5×5 的卷积核心。每一个单位都与 5×5 相邻的 S416 个特征图相连接。在此，由于 S4 的特性图尺寸也是 5*5，因此 C5 的输出尺寸是 1*1。这样，S4 与 C5 就能充分地连接在一起。C5 被标注为卷积层，而非完整的联接层，原因在于，如果 LeNet-5 的输入增加，它的结构没有改变，那么它的输出将会比 1*1 大，也就是说，它并没有完整的连接。在 C5 层，可以进行培训的链接有 48120 个。F6 层与 C5 充分相连，输出 84 个特性曲线。可培训的参数为 10164。

2.3.2 PreresNet

随着深度学习技术的发展，随着深度的增长，深度学习技术对图像的深度产生了决定性的影响。但是随着网络深度的增加，如果只是在以后不断叠加，那么将会产生许多问题：首先是梯度爆发/消失，backprop 不能将梯度有效地更新到前一层，

从而使上一层的参数不能被更新。第二个问题是退化问题，如果网络的层数太多，那么优化就会变得更加困难，训练误差和预测误差也会变得更大。

PreresNet 的目的是为了克服网络深度增加后的培训困难。它提供了包括两个 3x3 的卷积和一个 shortcut 连接的 residual 模块。shortcut 连接可以有效地减轻在逆向传输中因深度太深而造成的梯度消失，从而保证了在网络深化后的性能不会恶化。

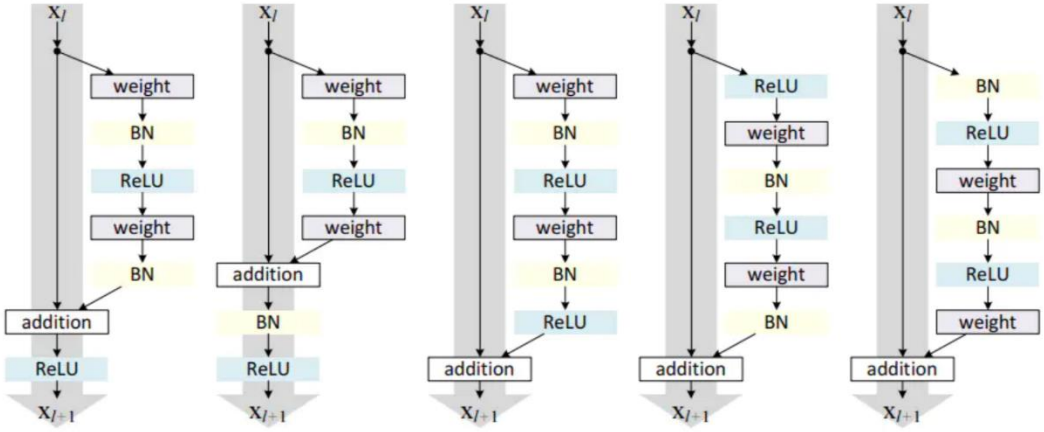


图 2.4 PreresNet 模型结构图

2.3.3 GoogLeNet

GoogleNet 是一种全新的卷积神经网络模型，该模型于 2014 年由 Christian Szegedy 提出，GoogLeNet 不再像 VGGNet 那样在 AlexNet 的基础上进行优化，从而增强了网络的等级，从而增强了识别能力。

在 GoogLeNet 以前，大多数的卷积神经网络都是侧重于网络的深度，利用更多的网络层次来提高模型的学习效率，但是随着网络的深入，会产生许多负面影响，比如，随着网络的深入，模型的输入和最后的输出之间的计算链路变得更长，所需要的参数也会成倍的增长，并且随着时间的推移，这种变化会越来越明显。基于这一点，GoogLeNet 在模型的设计上一改以往的设计模式，采用了“Inception”模式，由原来的单一卷积模式转变为多分支并行卷积，同时指出了不同尺度卷积校验特征抽取的重要性。Inception 模块采用多分支并行计算的方法进行卷积运算，可以更好地利用计算机的计算量，在较小的网络层级内完成与深层网络同样的卷积，从而

减少了输入和输出间的计算链，避免了参数多、计算量大、梯度消失等问题。相对于 AlexNet 和 VGGNet，GoogLeNet 可以从同样数量的数据中获取更多的特性，从而获得更好的培训结果。与 AlexNet 8 级的网络架构相比，GoogLeNet 的深度是 22 个层次，但是随着深度的增大，它的参量并没有增长，反而是它的总数的 1/12。Inception 结构如图 2.5 所示。

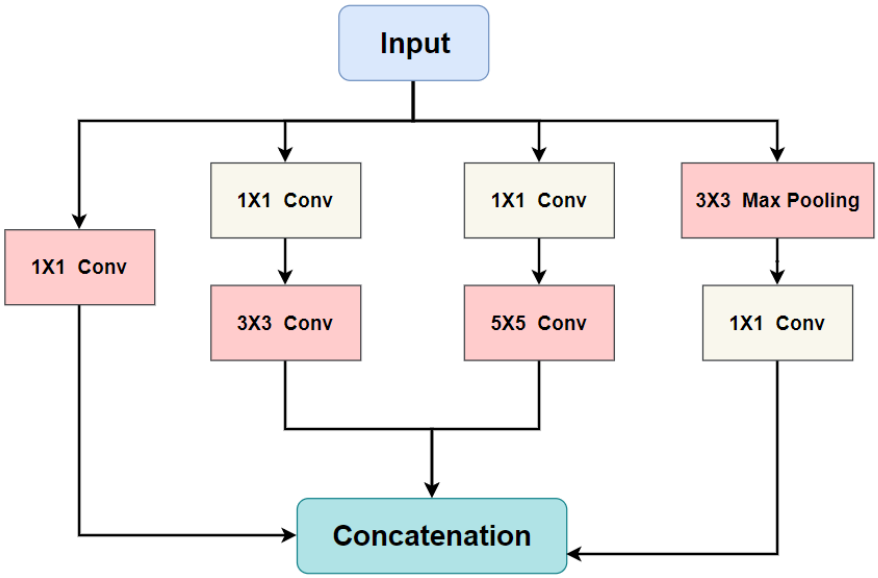


图 2.5 Inception 结构模块

2.3.4 DenseNet

随着 CNN 的深入，新的问题也随之产生：在进入网络的末尾（或一开始），输入和梯度都会在层间流动。近来许多工作都致力于此。它们有一个共同点：它们都是在不同的层次上建立简短的联系。

Dense Convolutional Network（DenseNet）将 Short 路径简化为一种简单的连接方式：为了使层之间的信息流通最大化，将各层与其他层相连（但必须确保每个层的特性图都是一样的）。为保持前向传输的特点，每个层都把前一层当作输入，然后把它自身的特征映射传给下一层。更重要的是，与 ResNet 相比，其他层没有加和的方法，而是把前面的一些特性连接起来。这样，第 L 层就有 l 个输入（之

前的全部层的结合）。其自身的特性图被传给了后续的 L-l 层。与传统的 L 型网路不同，L 型网路有 L 型连结。

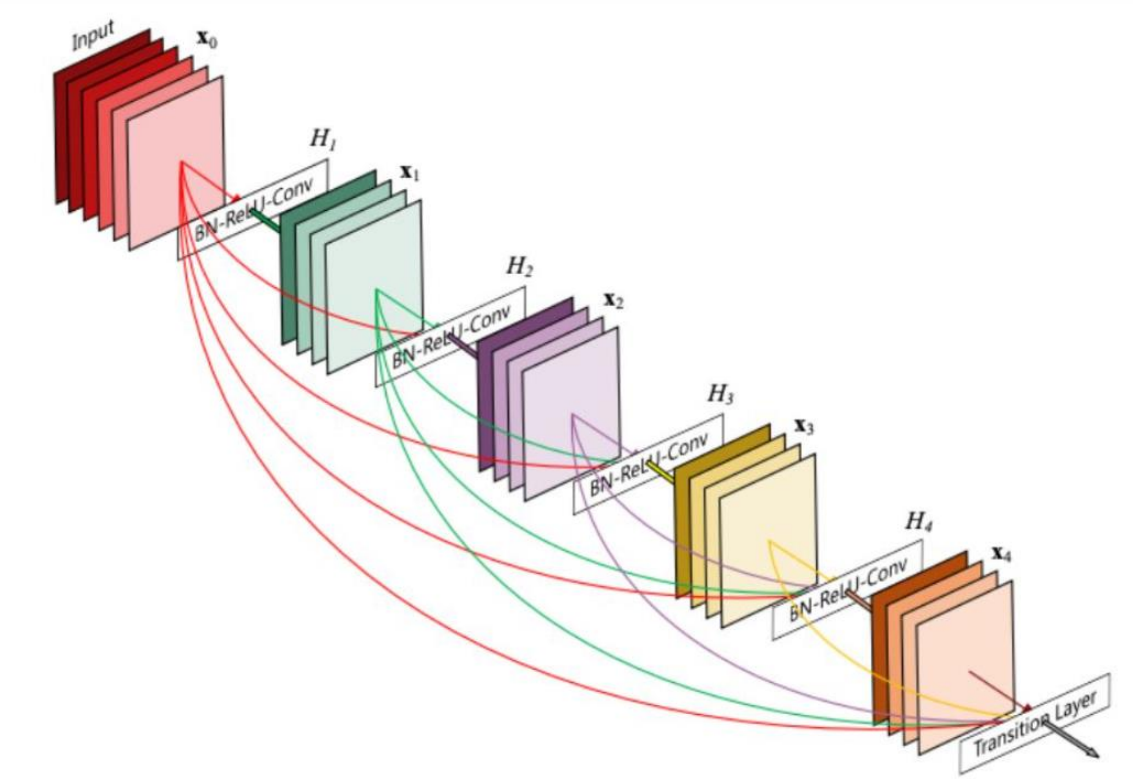


图 2.6 DenseNet 结构示意图

尽管从直观上看，DenseNet 的密集的连接方法有很多的参数，但实际上，由于不需要再学习额外的特征图信息，所以实际上要比传统的网络参数要少。传统的正向结构可以被视为一种在不同层次间进行转移的状态。每个层次都会从先前的网络层中读出并传送到下一层。当转换状态发生变化时，所传送的信息也会被保留下来。ResNet 将所传送的信息通过附加的恒定连接保存下来。

DenseNet 的另外一个优点是，它可以提高网络间的信息和梯度，从而使网络更加易于培训。每个层次都能与损失函数所传递的梯度和初始输入信息直接相关，从而构成了一种隐性的深层监控。这对提高网上培训很有帮助。再往下，可以发现，密集的连接有一个正则化的功能，可以减轻小的数据集的过度拟合。

2.4 本章小结

本文首先简要介绍了卷积神经网络的概念，并对其发展进行了回顾；同时，还对卷积层、池化层、归一化层、全连通层等几个关键部分进行了描述，并对卷积神经网络的发展过程中的四大典型网络模型 LeNet、PreresNet、GoogLeNet 和 DenseNet 进行了描述。

第 3 章 病害数据图像处理

原始资料往往不能完全符合试验要求，不能直接用于试验，正确地处理原始资料，对试验的成功起到一定的促进作用。这一章将简单地介绍试验的原始资料，并对原始资料进行统计分析，找出问题，解决问题，达到试验所需的资料。

3.1 数据集介绍

本论文以蓝莓植株为样本，从植株的叶片、花朵、果实三个器官的正常和受疾病侵袭的六个类型进行了分析，共有基础图像 424 幅。其中蓝莓植株的叶片、花朵、果实三个器官的正常和受疾病侵袭的六个类型图像数量如表 3.1 所示，原始数据集图像如图 3.1 所示。

表 3.1 原始数据集中各类别图像数量表

类别	叶片	花朵	果实	总计
正常图像数	45	83	56	184
病害图像数	123	76	41	240
总计	168	159	97	424

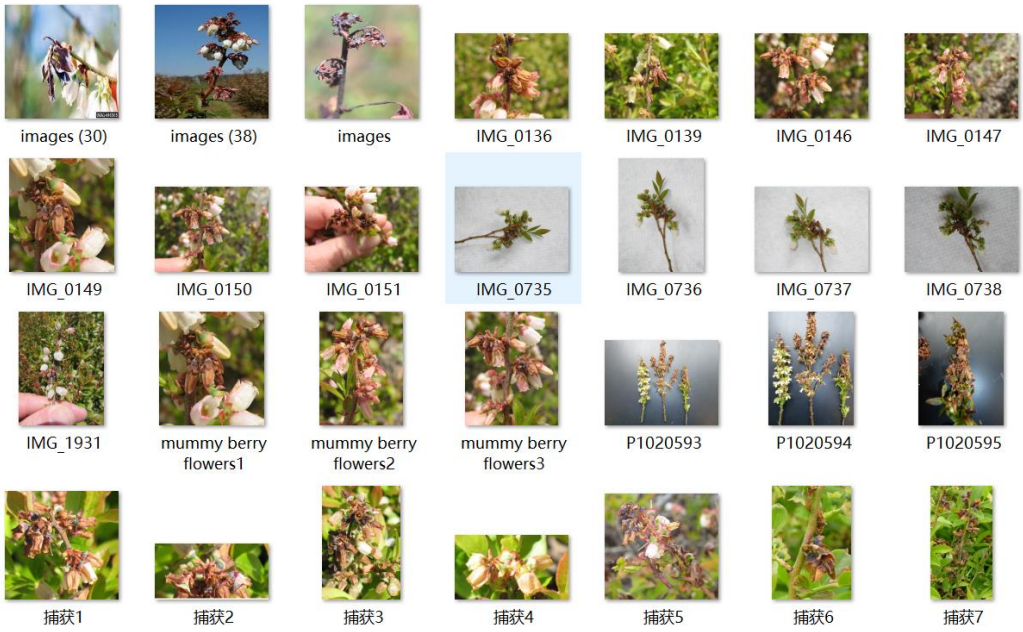


图 3.1 原始数据集

3.2 数据预处理

3.2.1 图像数据分析

对原始资料进行了统计和分析，结果表明：

1）图片资料的名称是任意的，没有一个标准的图片命名格式，无法通过图片的名字来判断图片的分类。

2）图像资料的总信息量很少，而且各个类别的图像资料差异很大。

3）图片的大小不一致。在原始数据集中，有 264 张图片尺寸为 256*256，10 张图片大小为 3072*2304，其他图片的大小不一，在原始数据集中，图片的各个尺寸所占的比例在图 3.2 中显示，从图 3.1 可以看出，主要有 256*256 和 3072*2304，分别占据了全部数据集的 62%和 36%。

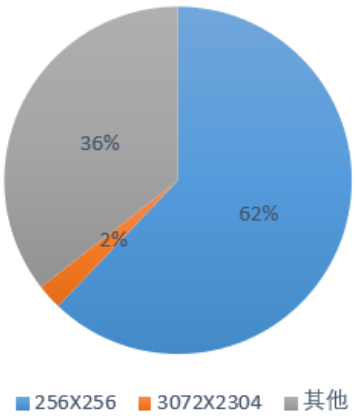


图 3.2 原始图像尺寸分布图

3.2.2 图像标准化

1）图片命名规范化。图片使用“字符串+下划线+数字”的形式重新命名，其中 Hl，Hi，Hfl，Fli，Hfr，Fri，Hfr，Fri，分别是叶片正常（Healthy leaf）、叶片异常（Leaf infected）、花朵正常（Healthy flower）、花朵异常（Flower infected）、果实正常（Healthy fruit）以及果实异常（Fruit infected）六个类别，下划线只是一个分割符，数字是一个图象的号码，除此之外，随着数据的加强，图像的名字也会根据这个命名格式扩展。

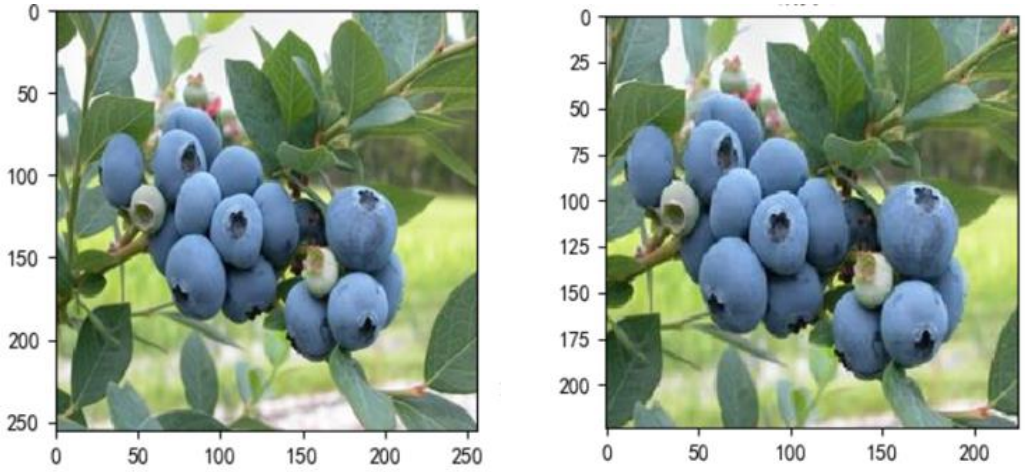
表 3.2 命名符号与分类类别对应说明

命名符号	分类类别
Hl	叶片正常
Hi	叶片异常
Hfl	花朵正常
Fli	花朵异常
Hfr	果实正常
Fri	果实异常

在图 3.3 中，我们可以看到，图片的名字从混乱变成了有秩序，并且从图片的名字中可以看到更多的有用的信息。

图 3.3 图像规范化命名

2）图像尺寸标准化。由于原始影像都是以蓝莓开花结果为背景，大部分影像的有效影像位于原始影像的中心，而原始影像的周围则是一些无效影像，在剔除某些无效资料的同时，对原始影像进行中心切割，得到的影像尺寸为 224x224，并显示出图 3.4 所示的剪裁效果。



(a)图像裁剪前　　(b) 图像裁剪后

图 3.4 图像尺寸标准化前后对比图

3.3 图像增强

本试验所用的原始资料共有 424 张彩色图片，包含了蓝莓植株的花、叶、果三大器官的正常或病害的 6 种类型。在此基础上，卷积神经网络在建立卷积神经网络模型时，首先要注意的是：首先，卷积神经网络是利用大数据和校准监控机制来学习模型，充分的训练数据是实现深度学习和数据挖掘能力的先决条件。但是，由于采集蓝莓的花、叶、果数据的采集费时、费力、费用昂贵，仅靠采集原始数据进行数据采集是不现实的。目前所掌握的资料数量还不够大，无法应用于大规模的卷积神经网络。第二，在多分类问题中，卷积神经网络的训练结果与训练结果之间存在着显著的因果关系。本试验所用的原始资料中，出现了资料不平衡的情况，如以 123 张叶为异常类型，只有 45 张叶正常分类，样本比例不平衡，容易造成模型训练偏差。在此基础上，对不同类型的影像进行了数据强化，采用了修正后的数据存储概率，以解决样本数据分配的不均匀性，从而实现了各个类型的样本数量的均衡。

由于训练样本的缺乏，无法满足卷积神经网络的训练数据，而且数据集中的样本数目也不平衡，迫切需要扩大模型的训练样本，但是手工收集样本的难度很大，所以必须利用现有的数据进行增强和扩充，以确保模型的推广。尽管有很多方法可以提高数据的性能，但在实际应用中，数据增强的算法选择要与实际情况相适应，选择适当的方法可以起到事半功倍的作用。本文在对不同数据增强方法进行脱机强化后，在图像频域、色彩和几何空间等方面进行了改进。

图 3.5 图像增强方式导图

3.3.1 高斯模糊

高斯模糊技术是一种图像模糊技术，利用该技术对图像进行处理，可以有效地降低图像中的噪声，并能有效地减小图像中的细节等级。高斯模糊算法的原理与图像卷积运算相似，每个像素点的值都是由象素点本身和它的区域内的象素值相乘而得，就像是一个正常分布的卷积核，它是以一个像素点为中心的卷积核，它的像素值是由它自己和它的范围内的像素值决定的。

高斯模糊技术是目前图像降噪中常用的一种方法，可以对正态分布造成的图像噪声进行抑制或消除。高斯模糊的卷积核尺寸，也就是它的邻居尺寸，它可以控制图像的模糊程度，并通过不断地调节卷积核的尺寸，最后用高斯模糊来加强实验数据。高斯模糊效果图显示在图 3.7 所示。

3.3.2 直方图正规化

直方图标准化，它是用一种技术方法调整图像的灰度直方图，使得它的分布更加均匀和标准化。一般情况下，图像的灰度直方图具有非均匀性，而直方图具有非线性特性，将高频段的灰度降低，提高了低频的灰度值，这样可以使灰度分布更均匀，从直方图上看，直方图上的直方图更平坦，从数据上来说，这是一种非线性的拉长转换。

由于直方图正规化可以使低频的灰阶数量增加，从而使整个图像的对比度得到改善，从而使图像的视觉质量得到显著的改善。直方图规范化可以有针对性地增强图像的局部和全局特性，可以加强特定的有效区域，抑制低反差的区域，从而可以利用直方图规范化，改善图像的细节表达。直方图正规化效果显示在图 3.7 所示。

3.3.3 伽马变换

伽玛变换是对图像进行强化和改善的。伽马变换本质上是将图像中每个象素的数字相乘，其数学表达如下：

$$V_{out} = AV_{in}^{\gamma} \tag{3.1}$$

式中，V_{out}——图像输出像素值

A——变换系数（常量）

V_{in}^{γ}——归一化后的图像输入像素值$V_{in} \in [0,1]$

事实上，伽马变换可以改善灰度较高的区域，从而实现对图像的校正，这一点可以从伽马曲线中得到更好的解释。

如图 3.6 所示，在，伽马转换不会提高图像，转换后的图像与原来的图像一样，并将其作为边界，此时伽马变换可以有效地改善图像的灰度较低的部分，且随着数值的减小，对低灰度部分的改进能力也更大，如图 3.6 所示，几乎所有的灰度都得到了显著的提高；在那个时候，伽马变换可以提高图像的灰度，而且随着灰度的增加，图像的灰度也会得到很大的提高。利用不同的 γ 值进行伽玛转换，可以提高图像的高灰度和低灰度，提高的程度可以得到控制。

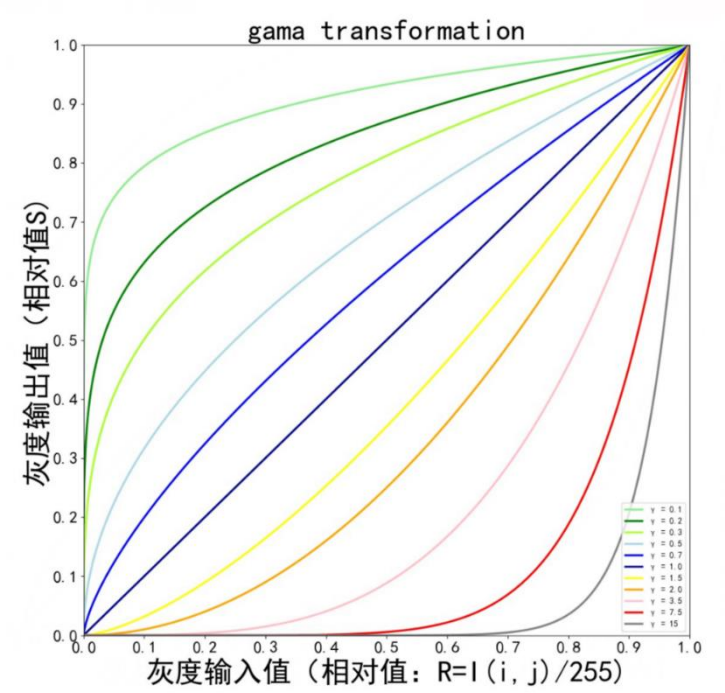


图 3.6 0~1 范围上的伽马变换示意图

经过多次实验，最后用$\gamma = 2$的伽马变换对各种类型的图像进行了增强。伽马转换后的图像的效果见图 3.7。

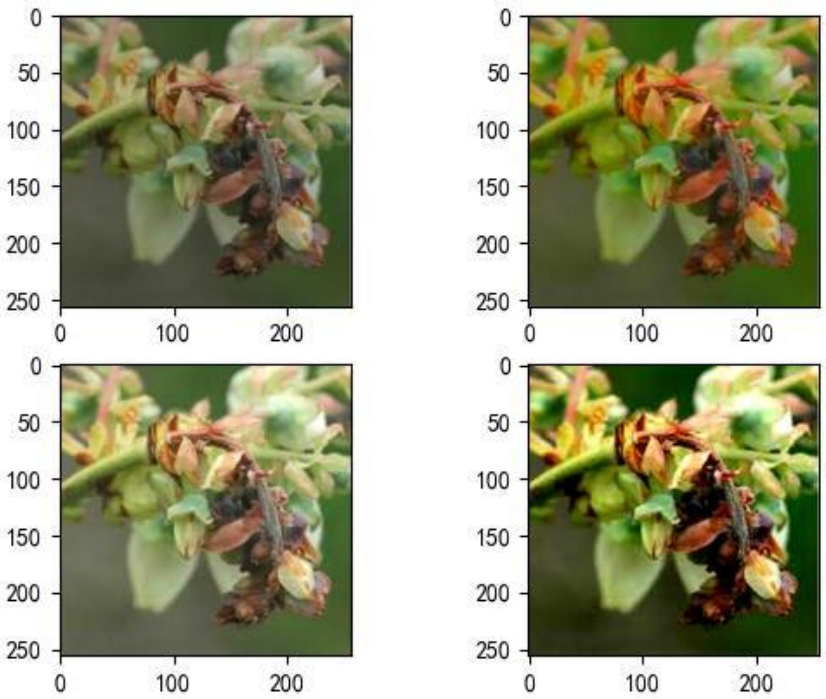


图 3.7 图像变换效果图

3.3.4 彩色增强

颜色增强是将 HSI 空间中的 HSI 空间由红、绿、绿、蓝三种 RGB 空间转换成色调（H）、饱和度（S）、亮度（I)，并通过 HSI 空间的灰度增强方法，对各个成分进行转换，以增强色调、饱和度、亮度，最终达到由 HIS 空间向 RGB 空间转换的目的。

文章介绍了一种增强图像颜色、亮度、对比度的新方法。增强后的图像更加明亮，颜色更加丰富；亮度提升是指整个画面亮度的提升；而增强反差，则能使目标影像更加清楚，减少影像中的无效资讯。图 3.8 显示了图像的色彩强化。

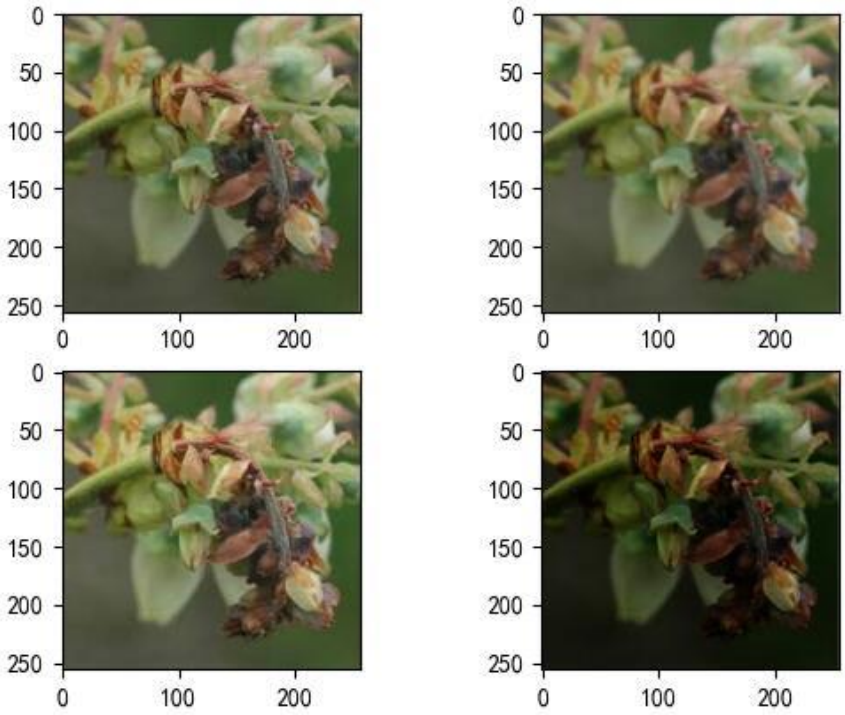


图 3.8 图像彩色增强效果图

3.3.5 随机几何变换

图像的几何转换主要是通过改变现有图像的空间关系和空间位置来实现，一般包括旋转、平移、镜像、转置、图像伸缩等，通过几何变换可以得到更多的信息，

虽然可以解决数据不平衡的问题，但也会造成模型过拟合的问题。因此，在对数据进行强化的基础上，引入了随机几何转换方法，可以有效地防止模型过拟合。

在图像的几何转换中，仅使用了“旋转+转置（沿着副对角线交换）”的方法进行数据的扩展，并且根据概率存储了旋转和转置后的图像，其概率 P 由以下公式来表示：

$$P = (C \div L - 1) \div A \tag{3.1}$$

式中，L——几何转换之前，当前分类图象的总数量

C——经过几何转换，目前的分类图像数据总数（这里用 1000 个常数）

A——几何转换类型（A=7：原图形的转置和旋转 90°、180°、270°及其转置）

图 3.9 显示了图像的几何转换效果。

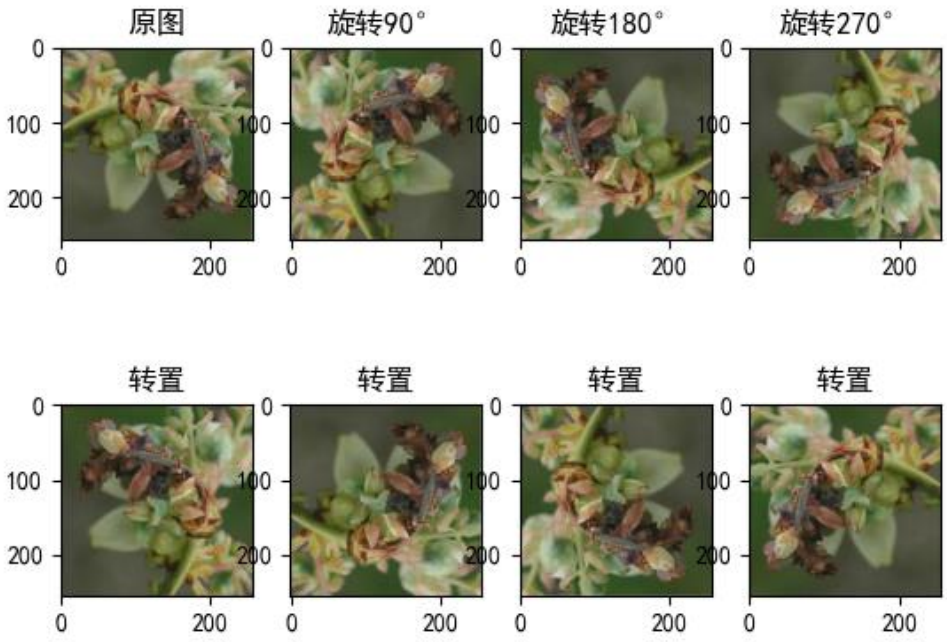
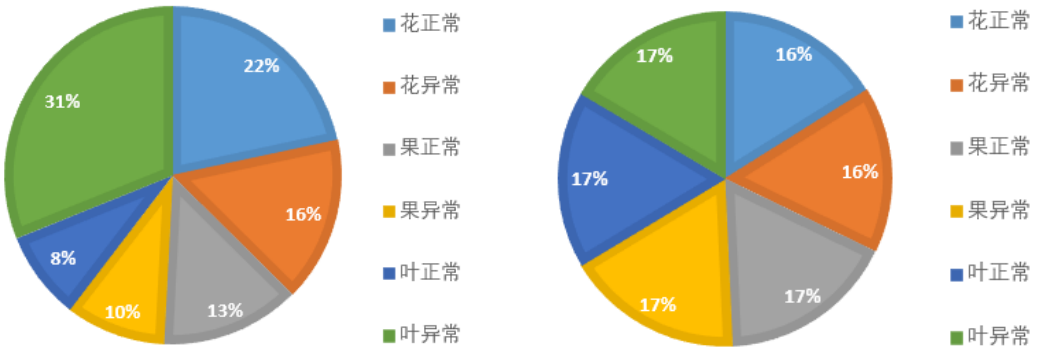


图 3.9 图像几何变换效果图

利用上述的图像处理技术对数据集进行了改进，每个分类所包含的数据量变得更均匀，见图 3.10，各种不同的图像数据量见表 3.3



(a) 原始数据各类别图像数量占比情况　　(b) 增强后各类别图像数量占比情况

图 3.10 数据增强前后各类别图像占比情况对比图

表 3.3 数据增强前后各类别数据量

类别	原始数据量	增强后数据量
叶片正常	45	1040
叶片异常	123	1031
花朵正常	83	1112
花朵异常	76	1113
果实正常	56	1118
果实异常	41	1057

3.4 数据划分

经预处理后的数据集经过数据强化，每一类的数据都包含 1050 张左右的图像，最后选取 1000 张作为模型的训练集和验证集数据，并将其分割成 7:3。在图 3.11 中显示了特定的分割流程。

因为在分割数据之前，首先要获得图像的名字，其次，在分割时，图像的名字是按照顺序排列的，而图像的名字是按照顺序排列的，而图像的名字则是按照顺序排列的，如果将列表中的前 700 个图片作为训练样本，那么后面的 300 个图片作为验证样本，将会造成样本之间的差异性不够，而训练样本的识别和分类效率较低，因此，在分割之前，随机地将采集到的图片名称进行了排序，并将其存储在训练集合或测试集合中，并且每隔 10 轮，重新将图像名称列表随机地打乱，从而可以有效地避免训练样本差异性不够造成的问题。划分效果如图 3.12。

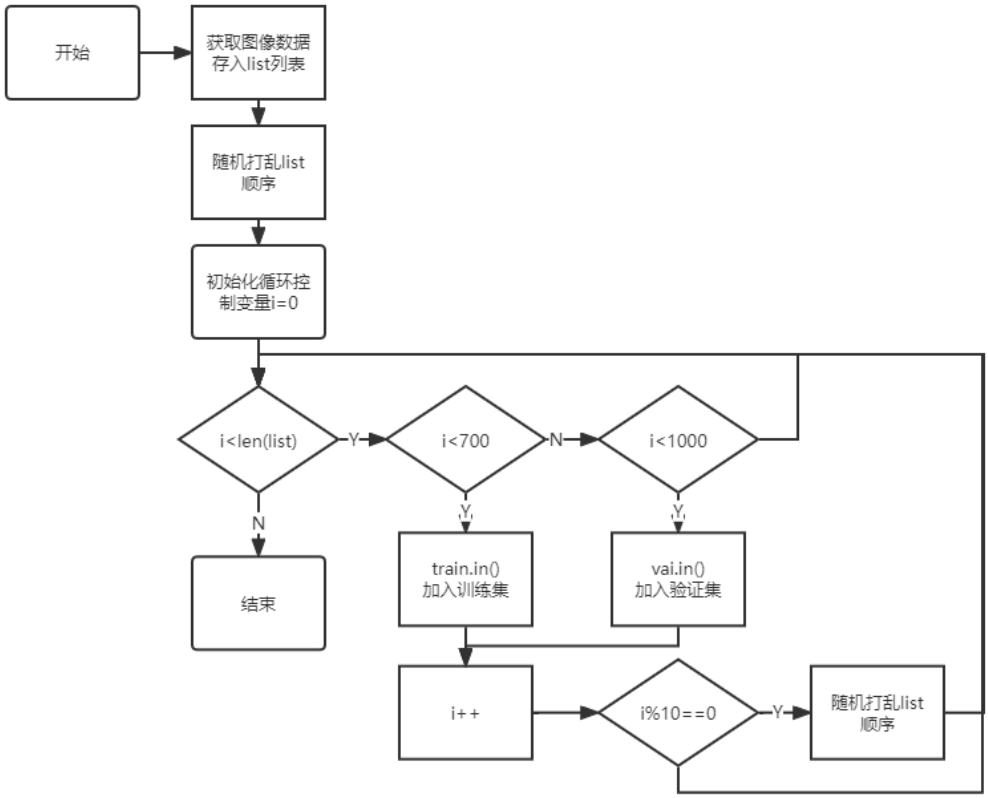


图 3.11 模型训练集与测试集数据划分流程图

（a）测试集样本

（b）验证集样本

图 3.12 模型训练集与验证集划分效果图

3.5 本章小结

首先介绍和分析了原始数据，然后进行压缩、裁剪、标准化命名等预处理。通过对图像进行高斯模糊、伽马变换、色彩增强、对比增强、图像随机旋转、转置等方法，对图像进行了扩展，然后随机地随机排序，然后选取预处理完成后的 1000 张图像，将训练和验证的数据进行 7:3 的分割。

第 4 章 基于卷积神经网络的蓝莓僵果病识别系统实现

由于不同的图像资料存在着差异，传统的卷积神经网络对不同的图像进行分类和识别的能力也存在着很大的差别。本文采用 yolov5 算法，针对识别对象蓝莓植株的特征及类别，对经典 yolov5 网络进行部分修剪，使其具有更好的适应性，构建的模型更加轻量化，从而实现蓝莓僵果病识别系统。

4.1 实验环境

本文的试验软件环境为 Windows11 64 位系统，采用 yolov5 算法、Pytorch 深度学习开源框架，选用 Python3 作为编程语言，编程 IDE 为 PyCharm 2022.1 x64，训练平台为 Google Colab 平台。计算机内存(RAM)大小为 16GB，搭载 12th Gen lntel(R) Core(TM) i7-12700H 处理器，如表 4.1。

表 4.1 实验软硬件环境

配置	内容
操作系统	Windows11 64 位
中央处理器	12th Gen lntel(R) Core(TM) i7-12700H
内存(RAM)	16GB
深度学习框架	Pytorch 1.4.0+cpu
编程语言	Python3
编程平台	PyCharm 2022.1
训练平台	Google Colab

4.2 Yolov5s 神经网络模型

在传统的卷积神经网络中，一般都是通过增大网络的层次（深度）来提高训练的效率，而加入网络层级的方法虽然可以提高训练的效率，但也会造成大量的参数过多、梯度消失、过度拟合、计算复杂等负面影响。本文使用深度较浅的轻量级的 YOLOv5s 网络模型。

YOLOv5 是一种基于 YOLOo 系列的单级目标检测方法，它是一种基于 YOLOv5 的高效实现方法。YOLOv5 网络模型包括 YOLOv5s、YOLOv5m、YOLOv5l 和 YOLOv5x 四种， YOLOv5s 是 YOLOv5 中最短的一个，后面三种模式则是 YOLOv5s 进一步深化和拓展的产物。由于网络模型体积较小，对系统的性能要求较低，便于部署，所以本文基于 YOLOv5s 的网络模型，对其进行了修剪优化。YOLOv5s 的结构可划分为 Input（输入）、Backbone（骨干网络）、Neck（多尺度特征融合模块）、Prediction（预测端）四大模块，其网络结构见图 4.1。

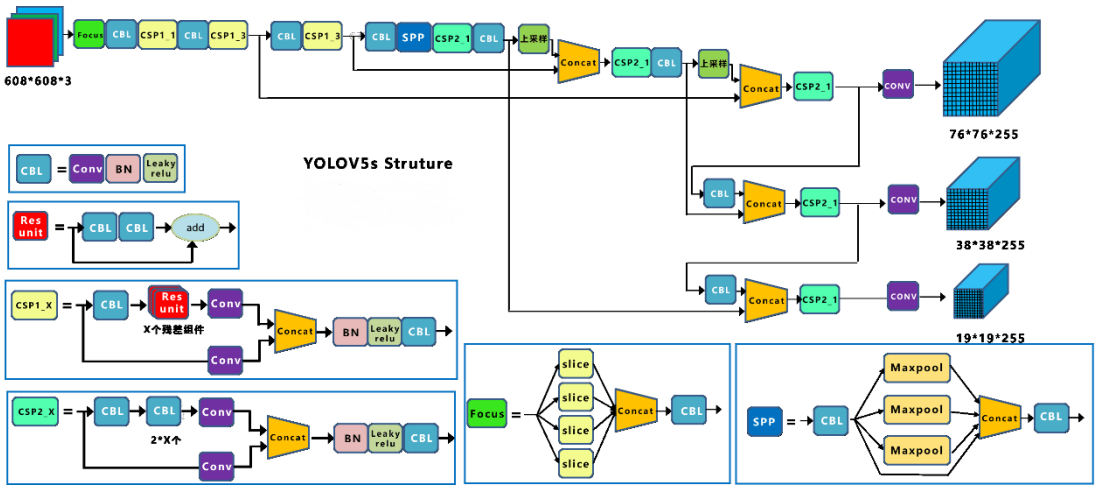


图 4.1 YOLOv5s 网络结构图

在输入部分，YOLOv5s 首先利用 IMlosaic 的数据增强技术，实现了图像的任意剪切、任意缩放、任意排列，从而丰富了对象被探测对象的背景，同时在大规模标准化运算时，一次四幅图像的数据处理，利用单一 GPU 可以获得更好的结果，提高了网络的普适性。其次，利用自适应的锚框结构来确定初始的锚框参数，然后根据不同的数据，由网络自动地求出相应的先验框参数。最后，利用图像的自适应缩放来实现图像的统一缩放或放大。

YOLOv5s 相对于 YOLO 系列的网络模式，加入了 Focus 架构的骨干网络，以分割图像为中心。后来 YOLOv5 采用了类似于 YOLOv4 的 CSP（交叉局部网）架构，YOLOv4 只在 Backone 中采用 CSP 结构，YOLOv5 则将两种 CSP 用于 Backbone 和 Neck。在 Backone 中，采用了具有残差结构的 CSP1_X，由于 Backone 网络深度较深，增加了残差结构，从而提高了层间的梯度，从而可以有效地避免了由于网络深度而导致的梯度消失，从而获得了更加精细的特征颗粒。在 Neck 中采用模块

CSP2_X，将主干网的输出分为两个部分，然后进行合并，增强了网络的特征融合能力，同时保持了更多的特征信息。

多尺度特征融合是目标识别的一个重要步骤，早期的 Neck 采用了上下两个取样块，其特点是不需要像 SSD 那样进行特征层聚集运算，而是直接跟踪多层特征。目前应用最广泛的是 FPN，PAN，ASFF，BiFPN 等。YOLOv5s 的 Neck 采用 FPN（特征金字塔网络）+PAN（金字塔注意力网络）的存储结构。

4.3 数据集特征标注

在进行模型训练前，必须对所收集的数据进行标记。本文利用 Make Sense 图象标记技术来标记数据集合。特征标记的重点在于对数据图象进行特殊的标记，其处理步骤见图 4.2。

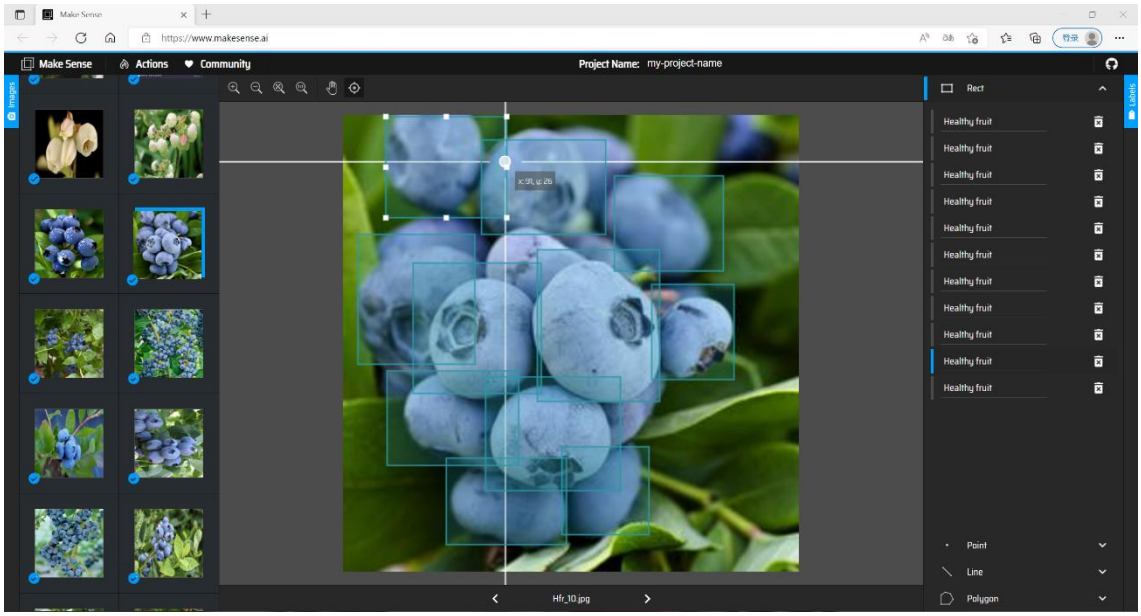


图 4.2 Make Sense 特征标注图

4.4 实验模型训练

本文实验采用的深度学习框架为 Pytorch，网络模型为 YOLOv5s，训练平台为 Google Cloab，训练与测试使用的图像大小均为224×224，每个训练集合的图片 700 张，总计 4200 张；校验集包括 300 张图片，总计 1800 张。

以下为整个实验流程图：

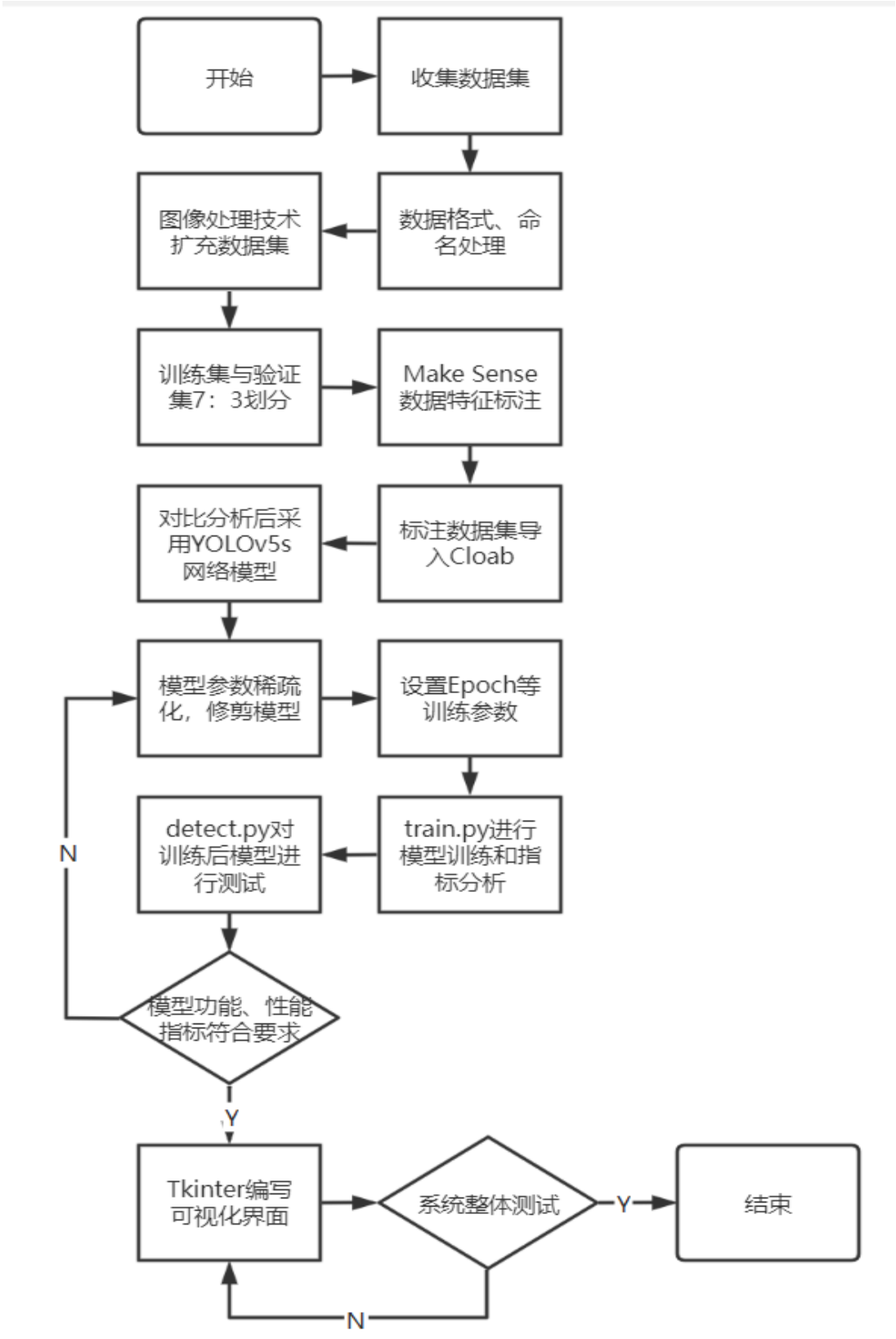
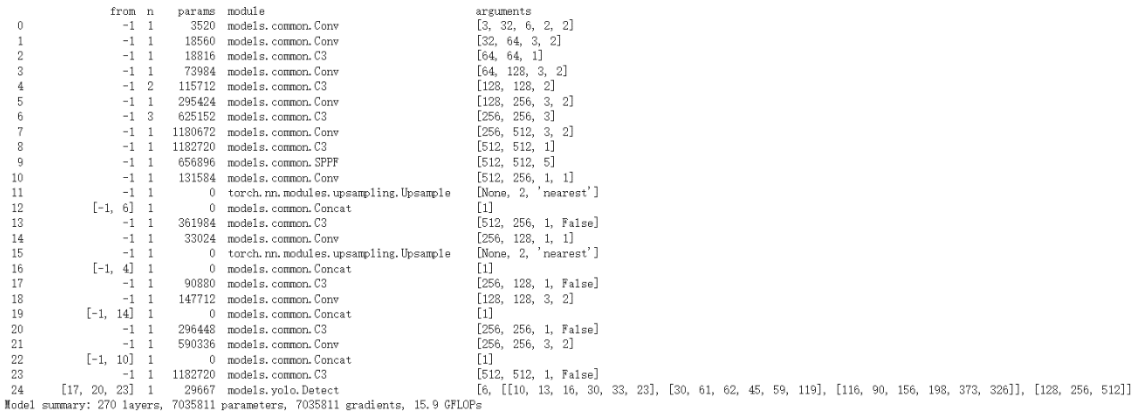


图 4.3 本文实验流程图

Google Colab 是谷歌研究小组研发的一个产品。在 Colab 里，每个人都能用自己的浏览器写出并运行任何的 Python 代码。特别适用于机器学习，数据分析和教学。技术上来说，Colab 是一个主机型的 Jupyter 笔记本电脑。不需要任何设定，就可以直接使用，并且可以免费使用 GPU 等计算资源。

本文通过在 val.py 文件中使用 torch_utils.prine（）函数对 YOLOv5s 模型进行修剪，实现 30%的稀疏性，即 30%的模型在层中的权重参数等于 0，修剪过程如图 4.4，修剪后模型如图 4.5。

图 4.4 模型修剪图



	from	n	params	module	arguments
0	-1	1	3520	models.common.Conv	[3, 32, 6, 2, 2]
1	-1	1	18560	models.common.Conv	[32, 64, 3, 2]
2	-1	1	18816	models.common.C3	[64, 64, 1]
3	-1	1	73984	models.common.Conv	[64, 128, 3, 2]
4	-1	2	115712	models.common.C3	[128, 128, 2]
5	-1	1	295424	models.common.Conv	[128, 256, 3, 2]
6	-1	3	625152	models.common.C3	[256, 256, 3]
7	-1	1	1180672	models.common.Conv	[256, 512, 3, 2]
8	-1	1	1182720	models.common.C3	[512, 512, 1]
9	-1	1	656896	models.common.SPPF	[512, 512, 5]
10	-1	1	131584	models.common.Conv	[512, 256, 1, 1]
11	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
12	[-1, 6]	1	0	models.common.Concat	[1]
13	-1	1	361984	models.common.C3	[512, 256, 1, False]
14	-1	1	33024	models.common.Conv	[256, 128, 1, 1]
15	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
16	[-1, 4]	1	0	models.common.Concat	[1]
17	-1	1	90880	models.common.C3	[256, 128, 1, False]
18	-1	1	147712	models.common.Conv	[128, 128, 3, 2]
19	[-1, 14]	1	0	models.common.Concat	[1]
20	-1	1	296448	models.common.C3	[256, 256, 1, False]
21	-1	1	590336	models.common.Conv	[256, 256, 3, 2]
22	[-1, 10]	1	0	models.common.Concat	[1]
23	-1	1	1182720	models.common.C3	[512, 512, 1, False]
24	[17, 20, 23]	1	29667	models.yolo.Detect	[6, [[10, 13, 16, 30, 33, 23], [30, 61, 62, 45, 59, 119], [116, 90, 156, 198, 373, 326]], [128, 256, 512]]

Model summary: 270 layers, 7035811 parameters, 7035811 gradients, 15.9 GFLOPs

图 4.5 修剪后网络模型图

在本文的实验中，batchsize=16。在对该模型进行预训练时，发现该模型的收敛性更好，因此将该模型的训练循环设定为 100 个最小值。在优化算法方面，我们使用了一种较为常见的随机梯度下降算法（SGD）。在模型损耗的计算中，采用

了交叉熵损失函数，在激励函数方面，各模型都使用 ReLU，从而降低了数据的维数和计算量。

```
Epoch   gpu_mem       box       obj       cls    labels  img_size
96/99     4.64G   0.02777   0.02901  0.002697        38       640: 100% 8/8 [00:01<00:00,  4.57it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 2/2 [00:00<00:00,  5.14it/s]
            all         50        141      0.971      0.889      0.959      0.706

Epoch   gpu_mem       box       obj       cls    labels  img_size
97/99     4.64G   0.02749    0.0309  0.002246        55       640: 100% 8/8 [00:01<00:00,  4.52it/s]
          Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 2/2 [00:00<00:00,  5.81it/s]
            all         50        141      0.957      0.897       0.96      0.703
```

图 4.6 模型训练图

4.5 系统可视化设计

本论文主要研究了一种简单的人机交互应用系统，并将训练好的卷积神经网络在系统中进行了集成。目的是为了简化系统，减少操作的困难。

4.5.1 需求分析

本系统采用人工神经网络技术，对蓝莓花、叶、果病害进行了分级，并给出了可视界面的需求：

1）可视化系统需要对模型进行自动集成。

2）该系统能正常工作，能离开。

3）可以选择自己的图像进行识别。

4）本系统能够完成图像的自动识别和显示。

5）系统界面简洁，操作简单。

4.5.2 模块划分

采用模块化的方法，可以加快开发进度，方便后期的维护。本系统的主要模块及功能划分如下：

1）初始化系统。装入卷积神经网路，可以设定界面尺寸，初始位置等。

2）影像处理组件。通过对所需识别的图像进行处理，方便了模型的辨识。

3）影像辨识模组。对所处理的影像进行了辨识。

4）影像显示模块。显示处理后的影像结果。

4.5.3 系统实现

该系统采用 Tkinter 图形接口库设计和开发，在初始化过程中，将训练好的模型装入到类的初始化中，设定界面的大小和位置，并定义了不同的功能，从而设计和开发了不同的接口和模块。在模型识别图像时，该模型从测试集中读取经过处理的图像，然后把图像输入到模型中进行辨识，并根据输出的最大值与每个标记相对应，从而输出一个预测结果。在图象显示方面，利用 Label 控件的图片属性来设定图片，为了界面的美感，将图片重新之后才能显示。

使用文本控制来显示所显示的结果，并且在每次显示之前都会删除现有的数据。为了使该系统具有一定的稳定性，通过对输入的文件名后缀进行判定，在无图像格式的文档中可以自动地显示出错误信息，而在图像文档中，通过数字将该图像处理后送到该模型中进行识别，并将该图像与识别结果一起输出。图 4.7 为系统可视化界面。

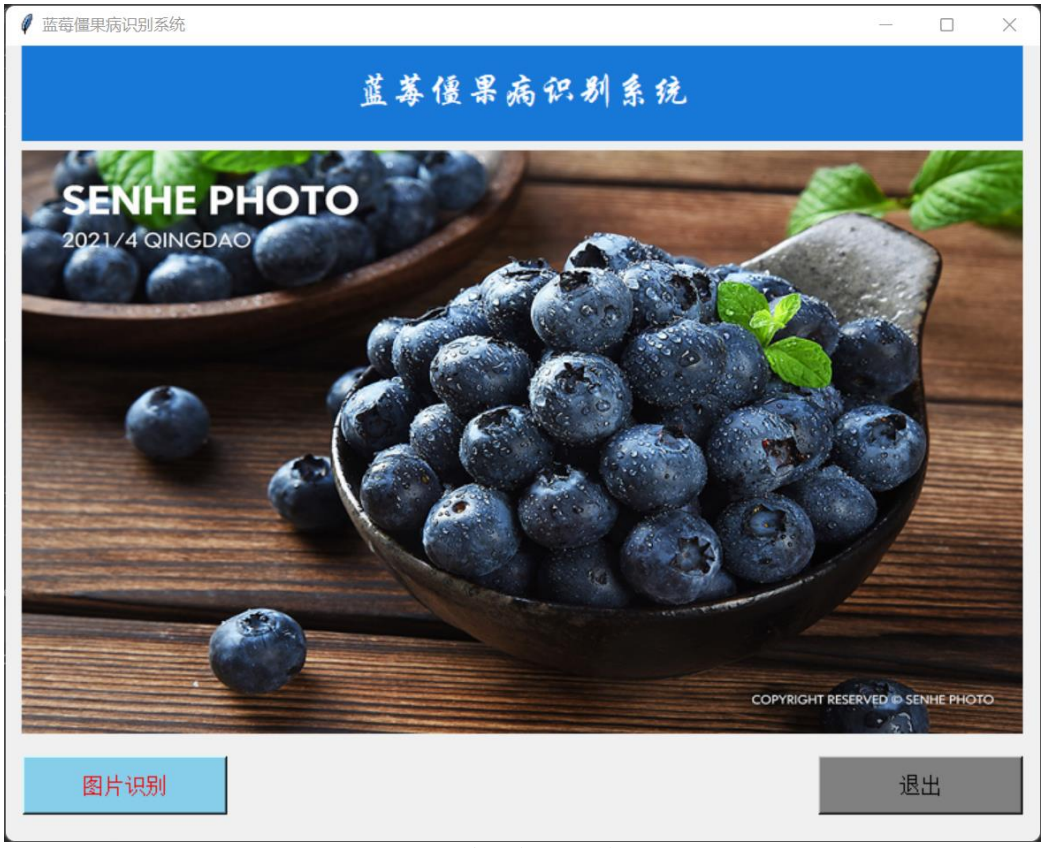


图 4.7 系统启动开始界面

可视化系统工作流程图如图 4.8 所示。

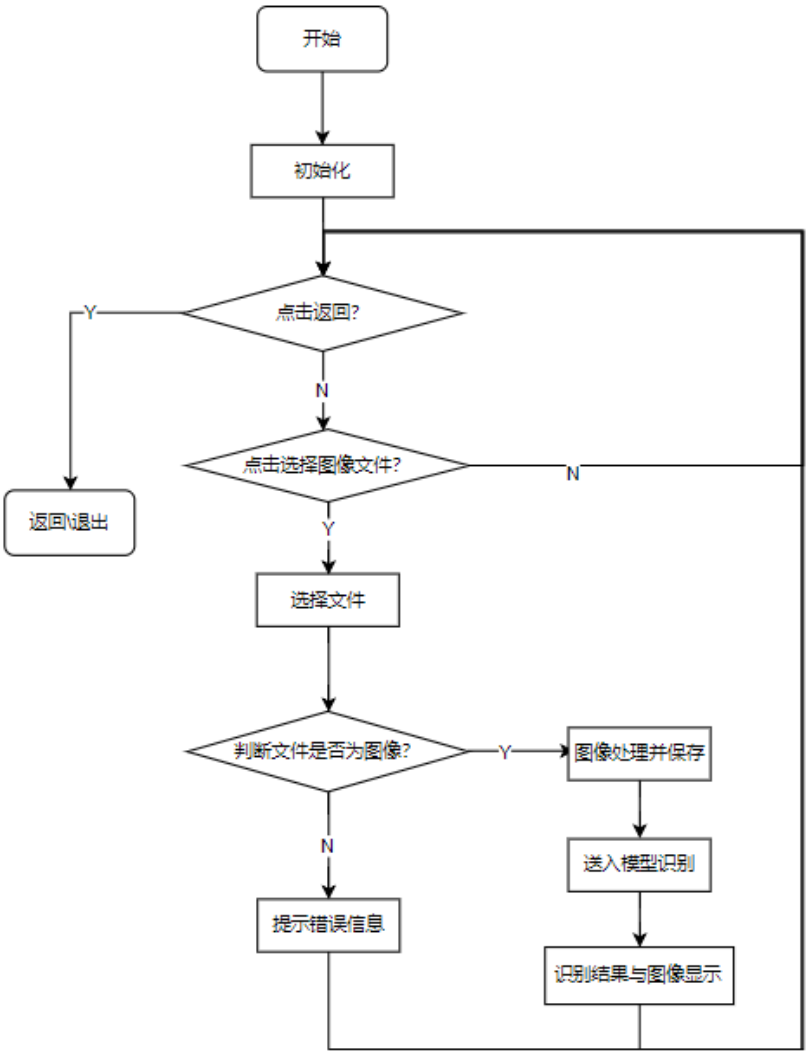


图 4.8 系统可视化工作流程图

4.6 本章小结

本章首先对本课题的实验环境作了简单的介绍，然后对 YOLOv5s 的网络进行了详实的描述，并对 YOLOv5s 网络进行了裁剪，使得它更符合本试验的要求。接着，本文介绍了该数据集的特征标注流程，并详细地描述了该方法的参数设定，并详细地描述了该模型的训练过程，并详细地描述了该系统的设计。

第 5 章 实验结果及模型评价

在模型的训练和训练中，我们所关心的是训练的损失和辨识的精确度，但是，评估一个模型的好坏并不能完全的依赖于两个因素，为了更好的检验该模型的有效性，还需要从多个方面来评估。这一章将会展示试验的结果，以及有关的模型评估指标，并根据相应的指标计算方法，对模型的各项指标进行了验证。

5.1 实验结果

模型的分类评价准确率 Acc 计算公式为：

$$Acc = \frac{1}{N_{c1}} \sum_{i=1}^{N_{c1}} \frac{N_{ii}}{N_i} \tag{5.1}$$

式中，N_{c1}——样本类别总数（N_{c1} =6）

i——类别标签（1~6）

N_i——类别i样本总量

N_{ii}——类别i预测为第i类的样本数 （即预测正确的个数）

在 100 个 Epoch 的训练下，该模型的平均精度达到了 96.6%，符合了模型的要求，该模型可以根据测试集的预测和相关的测试结果进行检验，并根据相应的评估系统提供了具体的数据，具体的评估指标见后文，模型的各项总体评价指标见图 5.1。

```
Validating runs/train/exp5/weights/best.pt...
Fusing layers...
Model summary: 213 layers, 7026307 parameters, 0 gradients, 15.8 GFLOPs
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 2/2 [00:00<00:00,  2.41it/s]
                 all         50        141      0.958      0.946      0.966      0.713
     Flower infected         50         12          1      0.981      0.995      0.772
      Fruit infected         50         25      0.965       0.96      0.957      0.704
      Healthy flower         50         23          1       0.88      0.926      0.682
       Healthy fruit         50         50      0.889       0.96      0.956      0.634
        Healthy leaf         50         20          1      0.898      0.983      0.741
       Leaf infected         50         11      0.893          1       0.98      0.743
Results saved to runs/train/exp5
```

图 5.1 模型的各项总体评价指标

该模型的平均准确率达到了 96.6%，花异常准确率达到 99.5%，果异常准确率达到 95.7%，花正常准确率达到 92.6%，果正常准确率达到 95.6%，叶正常准确率达到 98.3%，叶异常准确率达到 98.0%。

该模型在训练迭代时，通过对模型的损失函数进行了数值变化分析，并通过对试验数据的预测精度进行了检验，从而分析了模型在训练期间的表现。相关变化图像如图 5.2 所示。

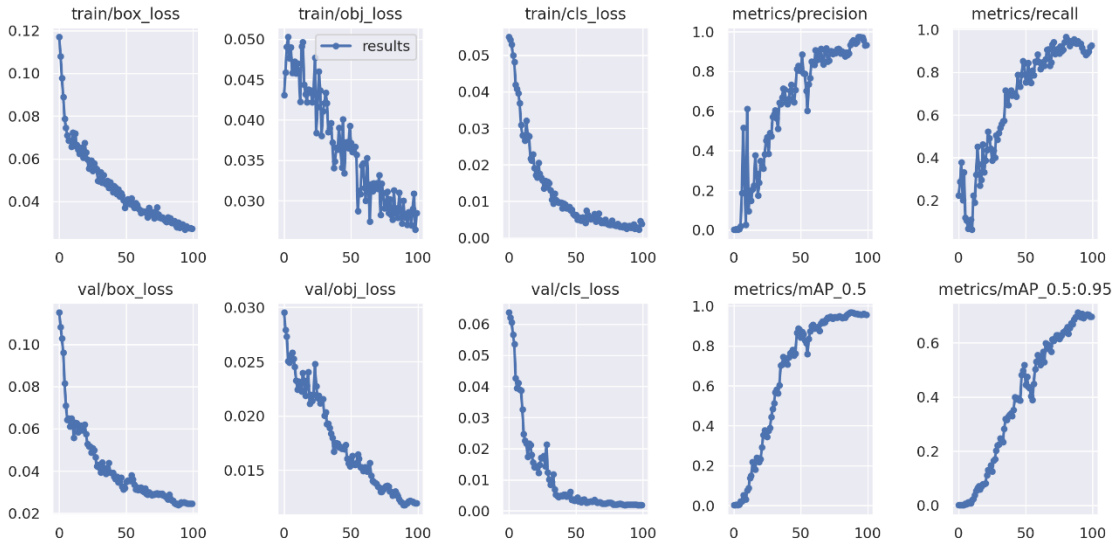


图 5.2 模型训练轮次损失函数值以及测试平均准确率曲线图

如图 5.2 所示，在模型的训练中，损失函数的数值整体上呈下降的趋势，经过 70 轮的训练，其损失值几乎为 0。同时，在 20 轮的训练中，模型的预测精度总体上都有提高，经过 80 轮的模型训练，其平均精度在 0.96 左右。训练损失值的降低和测试的精确度都保持了恒定，这说明在反复的训练和参数的更新中，模型的性能得到了提高，并且在训练循环-损耗函数和训练循环-试验的平均精确度两条曲线最后趋于水平，这说明该模型并没有发生过拟合的问题。

5.2 模型评价体系

5.2.1 评价体系讲解

在机器学习等领域，有很多方法可以用来测量模型的好坏，比如准确率（accuracy）、精确率（precision）、召回率（recall）以及 ROC 曲线下的 AUC 面积（ROC_AUC）等，其中，前四个评估指标是由混沌矩阵直接计算出来的，而 ROC

曲线的纵轴坐标与混沌矩阵有着密切的关系，所以在检验该模型的正确性前，需要对其进行深入的研究。

在二分类问题中，模型所预测的每一资料均存在正确（T）及误差（F)，而所预测的资料亦分为正（P）与负值（N)，其中以矩阵所示的情况如表 5.1 所示。

表 5.1 混淆矩阵法表示样本分类情况表

类别	预测结果为 T	预测结果为 F
数据为正样本（P）	TP	FN
数据为负样本（N）	FP	TN

在表格中，TP-模式把真实的正样（P）划分成正样（P）的数量，也就是说，该模式的预测值与实际值一致（T）。

模型将真实的正样（P）划分成负值（N)，也就是说，该模型的预测值与实际值不符（F）

FP-模式将真实的负值（N）归入正态（P）的数量，也就是说，该模式的预测值与实际值不一致（F）

该模型将真实的负样值（N）划分成负样值（N)，也就是说，该模型的预测值与实际值一致（T）

实际上，TN、FN、FP 和 TN 四个分类法分别代表了该模式的预测值，P/N 代表了该模式的分类类型。

该方法的正确性是指在一个特定的试验数据集合中，该模型预测的分类样本型与实际样本型的样本量与试验集样本型的比例。也就是，在混乱矩阵中，主对角上的数据之和与混乱矩阵中的全部数据之和。因此，正确率可由以下公式来表示：

$$accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{5.2}$$

而所谓的“回收率”，就是在一个特定的试验集合中，由该模型精确地预测出的正态样本数量与试验集中的正样本数量的比例。

所以，可以用以下公式来表示：

$$recall = \frac{TP}{TP + FN} \tag{5.3}$$

精确率的定义是：在一个特定的测试集合中，当一个真实的样本为正的样品时，被模型精确地预测为正的样品数量和被模型预测为正的样品数量的比例。所以，准确度的计算公式是：

$$precision = \frac{TP}{TP + FP} \tag{5.4}$$

ROC (receiver operating characteristic curve）是一条以 TPR 为纵轴线、假正比（FPR）为横坐标的曲线。真实率是指召回率，假设率（1-特异性）计算公式为：

$$FPR = \frac{FP}{TP + TN} \tag{5.5}$$

从 ROC 曲线可以看出一个模型的好坏，正确率越高，“越上凸”的曲线越好。

而 ROC 曲线和 FPR=0、FPR=1、TPR=0 的区域就是 AUC 的面积。AUC 的面积按“以常代变”的思路进行计算，如图 5.3 所示，将其下面的区域分成许多等宽度（步长Δx）的矩形，其长度为 TPR 值$R(\varepsilon_i)$，而矩形区域的和就是 AUC 区域。其公式如下：

$$AUC = \sum_{i=1}^{n} R(\varepsilon_i) \cdot \Delta x_i \tag{5.6}$$

AUC 值愈趋 1，则表示该模式的分类与辨识能力愈强。

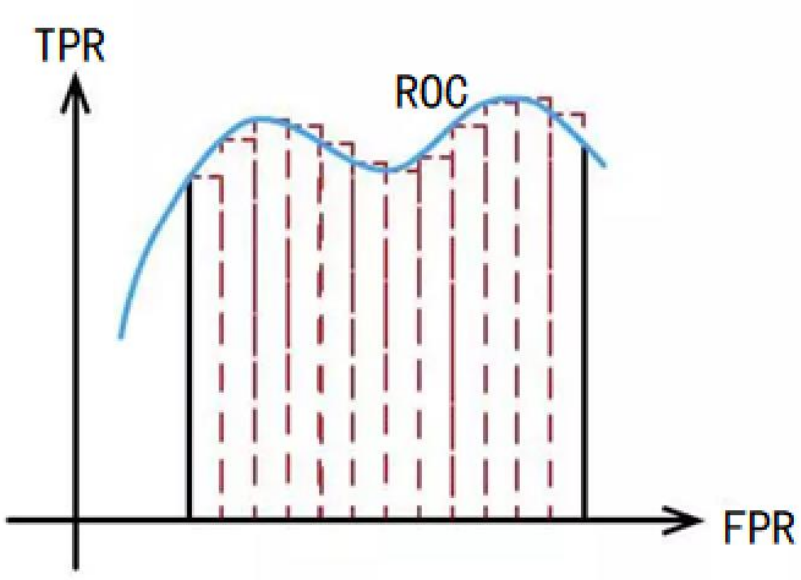


图 5.3 ROC 曲线下 AUC 面积计算演示图

5.2.2 评价指标计算

表 5.2，图 5.4 显示了由卷积神经网络模型所得到的模糊矩阵。

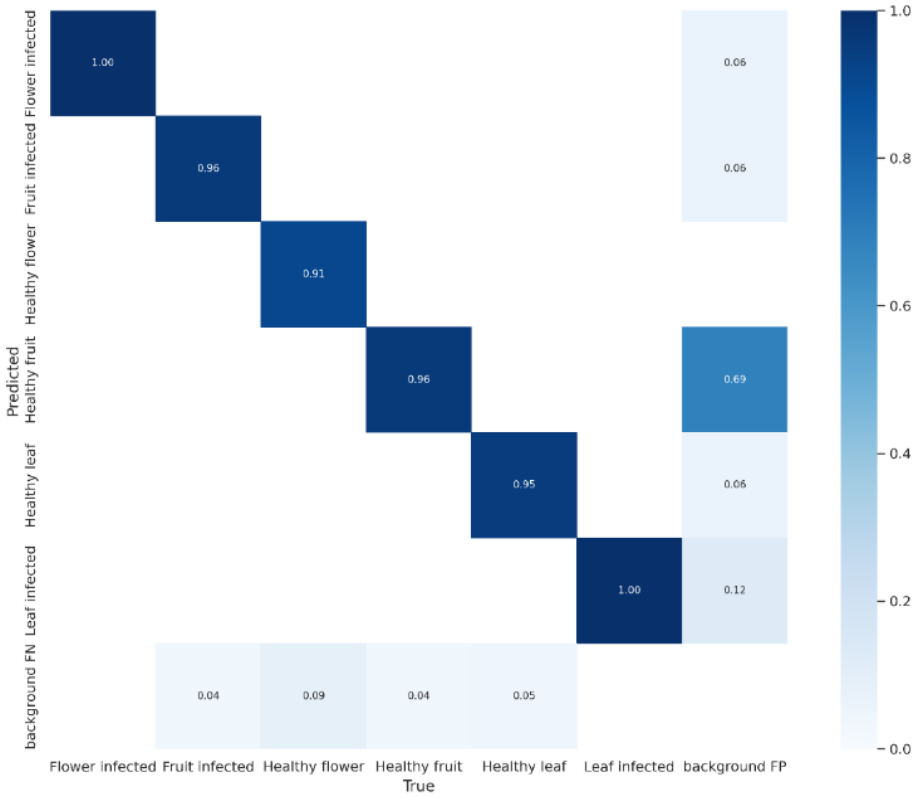


图 5.4 模型对验证集混淆矩阵图

表 5.2 模型对验证集混淆矩阵表

	预测类别						
实际类别	花异常	果异常	花正常	果正常	叶正常	叶异常	总量
花异常	275	1	14	1	1	7	300
果异常	1	279	5	15	0	0	300
花正常	8	7	283	0	1	1	300
果正常	2	6	2	288	0	2	300
叶正常	3	1	0	1	291	4	300
叶异常	10	3	2	2	2	281	300
总量	300	300	300	300	300	300	1800

在模型的训练中，还对模型的 Precision(精确率)、Recall(召回率)、PR 图、AUC 面积指标进行了分析显示，见图 5.5-5.8。

1）Precision(精确率)，模型训练精确率见图 5.5。

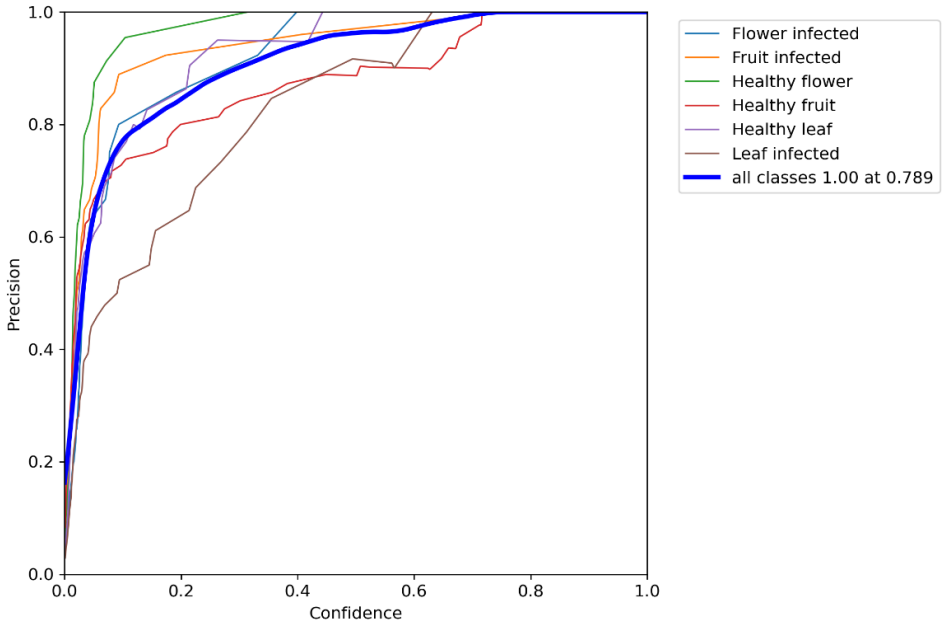


图 5.5 模型训练精确率图

2）Recall(召回率)，模型训练召回率图见图 5.6。

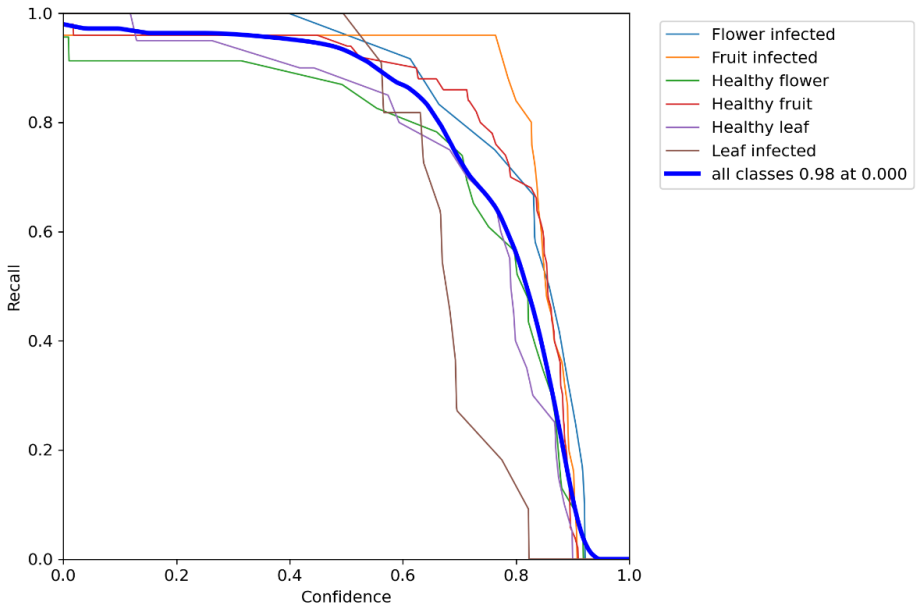


图 5.6 模型训练召回率图

3）PR 图，横坐标是 R 值，纵坐标是 P 值，曲线表示当召回率为 R 时，精确率 P 的大小，模型训练 PR 图见图 5.7。

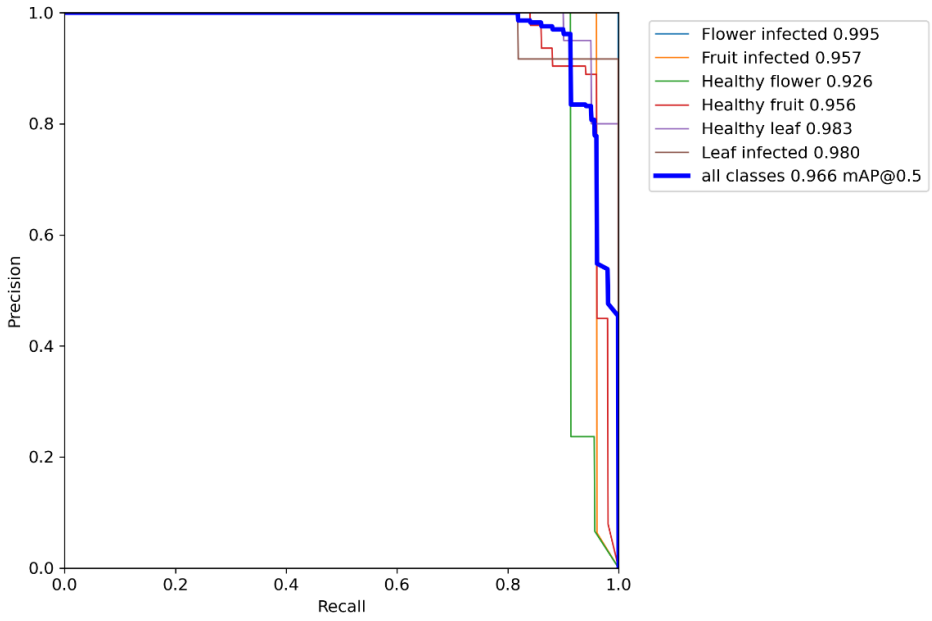


图 5.7 模型训练 PR 图

4）ROC 曲线绘制与 AUC 面积

通过计算各类别的 FPR以及TPR绘制得到的 ROC 曲线如图 5.8 所示：

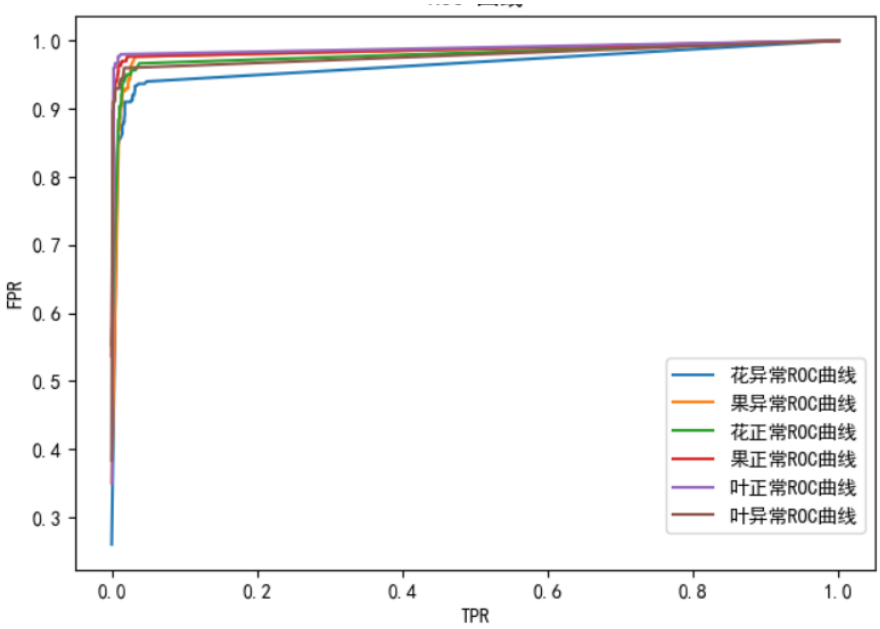


图 5.8 各分类类别的 ROC 曲线图

通过公式 5.6 计算得到各个类别的 ROC 曲线下的 AUC 面积如表 5.3 所示：

表 5.3 各分类类别 ROC 曲线下的 AUC 面积

分类类别	AUC 值
花异常	0.923
果异常	0.924
花正常	0.941
果正常	0.966
叶正常	0.972
叶异常	0.943

从 ROC 曲线 5.5 和 AUC 面积表可以看出，该模型在不同类型的 ROC 预测中都有显著的上升趋势，AUC 都在 0.9 以上，这表明该模型能够很好地预报出测验集合。

5.3 本章小结

本章对所构造的卷积神经网络在测试集合中的应用进行了预测，并对其进行了相应的评估。首先，给出了该模型在测试集上的试验结果，并给出了该模型在测试集上的应用。

在实验中，对 1800 张图片进行了分类，其正确率达 96.6%。其次，利用训练循环-损耗函数的数值和训练循环-试验的平均精度曲线，对该模型进行了评估。

在此基础上，给出了该系统的视觉效果，实验结果显示，该界面可以正确地识别出该图像符合设计要求的病害状况。

接下来，对模型的准确率、准确率、召回率、 ROC 曲线等进行了详细的说明，并给出了相应的计算公式，方便了后面的数据统计。最后，利用该模型对 6 组共 1800 张图像进行了模糊矩阵的分析，并对其进行了多个方面的验证。

最后得出结论：卷积网络分类法是一种很好的识别与分类方法。

第 6 章 总结与展望

6.1 主要工作与创新点

本文从频域、色彩和几何空间等多个方面扩展了图像的数据，将不充分、不均衡的图像集扩展到充分和均衡。对 YOLOv5s 网络模型进行针对识别对象蓝莓植株的特征及类别的部分修剪，使其具有更好的适应性，构建的模型更加轻量化，从而实现蓝莓僵果病识别系统。

本文通过修剪后的 YOLOv5s 网络模型，在蓝莓植株的花、叶、果的病害鉴定中，经过 100 轮的训练，该模型的识别精度达 96.6%，对所有类型的识别精度和召回率都在 92.6%以上。

上述结果表明，本论文所建立的卷积神经网络可以用于蓝莓植株的花、叶、果病害的鉴别，并提出了相应的改进方法。另外，利用卷积神经网络进行农业病害的识别也是一种有效的途径。

中国是一个拥有大量耕地和大量农作物的国家，在这种情况下，卷积神经网络在疾病诊断方面的应用将会非常广泛。

6.2 后续研究工作展望

本文设计卷积神经网络进行蓝莓的僵果病识别，其不足有二。

一是所使用数据集数量不足，即使通过图像增强技术进行了扩充数据集，仍就会有数据集特征相似的问题，导致模型训练会出现略微的过拟合现象。

二是所使用的神经网络比较复杂，可以根据实际情况对神经网络进行改进和裁剪，以适配实际应用场景。

后续可以收集更多的数据集，并对卷积神经网络模型进行改进和裁剪来优化模型，以达到更好的识别效果。

参考文献

[1] 孙中才. 准确把握农业和农村发展在国家经济中的地位[N]. 中国社会科学报, 2019-02-12(001)

[2] 史冰莹, 李佳琦, 张磊. 基于 CNN 的农作物病虫害图像识别模型[J]. 计算机系统应用, 2020, 29(06): 89-96.

[3] 孙奥, 吴冬燕, 吴阳江. 深度学习在农作物病虫害识别中应用初探[J]. 电子测试, 2019(06): 68-70.

[4] 李勤斌. 农业病虫害防治现状与方法研究[J]. 种子科技, 2019, 37(04): 118+121.

[5] 边柯橙, 杨海军, 路永华. 深度学习在农业病虫害检测识别中的应用综述[J]. 软件导刊, 2021, 20(3): 8.

[6] Meyer G E, Davison D A. An electronic image plant growth measurement system[J]. Trans ASAE, 1987, 30(1): 165-208.

[7] Whittaker D, Miles G E, Mitchell O R, et al. Fruit location in a partially occluded image[J]. Transactions of the ASAE, 1987, 30(3): 0591-0596.

[8] Slaughter D C, Harrell R C. Discriminating fruit for robotic harvest using color in natural outdoor scenes[J]. Transactions of the ASAE, 1989, 32(2): 0757-0763.

[9] Malthus T J, Madeira A C. High resolution spectroradiometry: spectral reflectance of field bean leaves infected by botrytis fabae[J]. Remote Sensing of Environment, 1993, 45(1): 107-116.

[10] Zayas IY, lnna. Detection of insects in bulks wheat samples with machine vision. transactions of the ASAE, 1998, 6(3): 883-888.

[11] Delwiche S R, Kim M S, Deshazer J A, et al. Hyperspectral imaging for detection of scab in wheat[J]. Proceedings of SPIE - The International Society for Optical Engineering, 2000, 4203: 13-20.

[12] Habib. A soft computing approach for classification of insects in agricultural ecosystems [D]. Mexico:New Mexico Sate University, 2000.

[13] Kulkarni A H, Patil A. Applying image processing technique to detect plant diseases[J]. International Journal of Modern Engineering Research, 2012, 2(5): 3661-3664.

[14]Montalvo M, Guijarro M, Guerrero J M, et al. Identification of plant textures in agricultural images by principal component analysis[C]// International Conference on Hybrid Artificial Intelligence Systems. Springer International Publishing, 2016.

[15]Mohanty S P, Hughes D P, Salathe M. Using deep learning for image-based plant disease detection[J]. Frontiers in Plant Science, 2016, 7.

[16]陆玉荣，韩光杰，祁建杭．辣椒蚜虫高效防治药剂筛选及田间应用效果[J]. 2022(8).

[17]师红宇．计算机视觉技术在棉花异性纤维检测中的应用综述[J]．科技创新导报, 2018, 15(24): 2.

[18]梁子安，刘飞，赵秋红．数学形态学在昆虫总科阶元分类学上的应用研究[J]. 动物分类学报, 2007, 32(01): 147-152.

[19]Fang Y, Zhang X, Zhou D, et al. Improve inter-day hand gesture recognition via convolutional neural network-based feature fusion[J]. International Journal of Humanoid Robotics, 2021.

[20]张芳，李晓辉，杨洪伟．复杂背景下植物叶片病害的图像特征提取与识别技术研究[J]．辽宁大学学报(自然科学版), 2016, 43(04): 311-318.

[21]曹英丽，江凯伦，于正鑫．基于深度卷积神经网络的水稻纹枯病检测识别[J]. 沈阳农业大学学报, 2020, 51(5): 8.

[22]马浚诚，杜克明，郑飞翔．基于卷积神经网络的温室黄瓜病害识别系统[J]．农业工程学报, 2018, 34(12): 186-192.

[23]刘阗宇，冯全，杨森．基于卷积神经网络的葡萄叶片病害检测方法[J]．东北农业大学学报, 2018, 49(03): 73-83.

[24]尹显东，郭竞，刘琪芬．一种基于深度学习的农作物病虫害智能检测方法[P]. 四川省:CN109919239A, 2019-06-21.

[25]贾少鹏，高红菊，杭潇．基于深度学习的农作物病虫害图像识别技术研究进展[J]．农业机械学报, 2019, 50(S1): 313-317.

[26]胡明越．基于深度学习的树种识别算法研究[D]．浙江农林大学, 2019.

[27]刘志勇，张丽秀，钟婷婷．基于改进 leNet-5 的番茄病虫害识别的研究[J]．赣南师范大学学报, 2020, 41(06): 70-74.

[28]高德民, 史东旭, 薛卫. 基于物联网与低空遥感的农业病虫害监测技术研究[J]. 东北农业科学, 2021, 46(01): 108-113.

[29]秦彩杰, 李勇. 作物病虫害自动识别技术研究—基于视频监控和无人机平台[J]. 农机化研究, 2021, 43(10): 42-45+50.

[30]肖禹, 王景中, 王宝成. 基于深度学习的中文文本分类方法[J]. 2021.

[31]盖荣丽, 蔡建荣, 王诗宇. 卷积神经网络在图像识别中的应用研究综述[J]. 小型微型计算机系统, 2021, 42(9): 5.

[32]Alvarez I, Finlayson N J, Ei S, et al. Heritable functional architecture in human visual cortex[J]. NeuroImage, 2021, 239(Part A): 118286.

[33]Ansari M A, Crampton A, Garrard R, et al. A convolutional neural network (CNN) classification to identify the presence of pores in powder bed fusion i mages[J]. The International Journal of Advanced Manufacturing Technology, 2022, 120(7): 5133-5150.

[34]许德刚, 李凡. 现代信息技术在储粮害虫检测中的应用[J]. 中国粮油学报, 2021.

[35]Gadjimuradov F, Benkert T, MD Nickel, et al. Robust partial fourier reconstruction for diffusion-weighted imaging using a recurrent convolutional neural network[J]. Magnetic Resonance in Medicine, 2021.

[36]Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[37]Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 1-9.

[38]Zhao H, Li Y, Wang J. A convolutional neural network and graph convolutional network-based method for predicting the classification of anatomical therapeutic chemicals[J]. Bioinformatics, 2021(18): 18.

致谢

四年的读书生活在这个季节即将划上一个句号，而于我的人生却只是一个逗号，我将面对又一次征程的开始。四年的求学生涯在师长、亲友的大力支持下，走得辛苦却也收获满囊，在论文即将付梓之际，思绪万千，心情久久不能平静。伟人、名人为我所崇拜，可是我更急切地要把我的敬意和赞美献给我的导师。我不是您最出色的学生，而您却是我最尊敬的老师。您治学严谨，学识渊博，思想深邃,视野雄阔，为我营造了一种良好的精神氛围。授人以鱼不如授人以渔，置身其间，耳濡目染，潜移默化，使我不仅接受了全新的思想观念,树立了宏伟的学术目标，领会了基本的思考方式，从论文题目的选定到论文写作的指导，经由您悉心的点拨，再经思考后的领悟，常常让我有“山重水复疑无路,柳暗花明又一村”之感。

感谢我的爸爸妈妈，焉得谖草，言树之背，养育之恩,无以回报，你们永远健康快乐是我最大的心愿。在论文即将完成之际，我的心情无法平静，从开始进入课题到论文的顺利完成，有多少可敬的师长、同学、朋友给了我无言的帮助，在这里请接受我诚挚谢意!

同时也感谢学院为我提供良好的做毕业设计的环境。

最后再一次感谢所有在毕业设计中曾经帮助过我的良师益友和同学,以吸在设计中被我引用或参考的论著的作者。

附录 A 源程序

受篇幅限制，本文只展示部分关键代码。

A_EnhanceData.py

```
# -*- coding: UTF-8 -*-
'''
@Time    : 2022/4/20 22:26
@Author : Ran hao hong
@File    : A_EnhanceData.py
@desc    ： 图像数据增强
'''

# 读取文件用
import os
import glob
# openCV 的库
import cv2
import cv2 as cv
import numpy as np
# PIL 的库
from PIL import Image
from PIL import ImageFilter
from PIL import ImageEnhance
# 显示图片用
import matplotlib.pyplot as plt
# random 库
import random

# time 库
import time

# plt 绘图配置显示中文标签
plt.rcParams['font.sans-serif'] = ['Simhei']
plt.rcParams['axes.unicode_minus'] = False
```

```
# Image 库打开图像
def openImg(rootpath, imgname):
    """
    打开图像
    :param rootpath:  图片地址
    :param imgname:  图片名称
    :return:  打开为 PIL 格式的图像
    """
    image = Image.open(os.path.join(rootpath, imgname))
    return image


#  图片翻转
def flip(image):
    """
    图片翻转
    :param image: PIL 格式的图像
    :return:  翻转后的图片
    """
    filp_img = image.transpose(Image.FLIP_LEFT_RIGHT)
    return filp_img


#  图片旋转
def rotation(image, rotation_num):
    """
    图片旋转
    :param image: PIL 格式的图像
    :param rotation_num:  旋转度数
    :return:  旋转后的图片
    """
    rotation_img = image.rotate(rotation_num)   #  旋转角度   random.choice([-20,-10,10,20])
    return rotation_img
```

```
# 图像进行转置(相当于顺时针旋转 90°)
def transpose(image):
    """
    图像进行转置(相当于顺时针旋转 90°)
    :param img: PIL 格式的图像
    :return: 转置后的图片
    """
    image_transpose = image.transpose(Image.TRANSPOSE)

    return image_transpose


# 图像进行转置,再水平翻转
def transverse(image):
    """
    图像进行转置,再水平翻转
    :param img: PIL 格式的图像
    :return: 转置,再水平翻转后的图片
    """
    image_transverse = image.transpose(Image.TRANSVERSE)
    return image_transverse


# 拼接文件地址
def makeNewPath(save_path, new_filename):
    """
    拼接文件地址
    :param savepath:
    :param new_filename:
    :return: 返回拼接的文件地址
    """
    # 获取文件名称
    new_filename = os.path.basename(new_filename)
    # 拼接需要保存的地址
    new_filename = os.path.join(save_path, new_filename)
```

```
    # 获取新的文件路径
    file_dir_name = os.path.dirname(new_filename)
    # 判断文件夹是否存在
    if not os.path.exists(file_dir_name):
        # 新建文件夹
        os.makedirs(file_dir_name)

    return new_filename


# Image 库对图片进行数据增强
def ImageEnhancementImage(img_path):
    """
    Image 库对图片进行数据增强
    :param img_path: 图片的地址
    :return: 图像与名称列表 img_Lists, img_Names
    """
    # print(img_path) #输出图片的路径
    # 读取图像
    im = Image.open(img_path)
    # 图像数据列表
    img_Lists = []
    # 图像名称列表
    img_Names = []

    # 原图
    img_Lists.append(im)
    img_Names.append("原图")

    # 第一部分：调整图像的颜色增强
    om = ImageEnhance.Color(im).enhance(2)
    img_Lists.append(om)
    img_Names.append("颜色增强")

    # 第二部分：调整图像的亮度，enhance 中的参数可变（建议在 0.3~2 的范围内；0.3，0.7，
1.5）
```

```
    om = ImageEnhance.Brightness(im).enhance(1.3)
    img_Lists.append(om)
    img_Names.append("亮度增强")

    # 第三部分：调整图像的对比度，enhance 中的参数可变（建议在-2~4 的范围内；-2，-1，0.5，1.5）
    om = ImageEnhance.Contrast(im).enhance(2)
    img_Lists.append(om)
    img_Names.append("对比度增强")

    return img_Lists, img_Names


# Opencv 库对图片进行数据增强
def ImageEnhancementOpencv(img_path, plt_flag=False, save_flag=False):
    """
    Opencv 库对图片进行数据增强
    :param img_path: 图片的地址
    :param plt_flag: 是否使用 plt 库绘制显示
    :param save_flag: 是否会使用 cv2.imwrite()函数保存图片
    :return: 图像与名称列表 img_Lists, img_Names
    """
    img = cv2.imread(img_path)
    if plt_flag:
        # 色彩转换
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    # 图像数据列表
    img_Lists = []
    # 图像名称列表
    img_Names = []

    # 第五部分：高斯模糊
    om = cv2.GaussianBlur(img, (5, 5), 0)
    img_Lists.append(om)
    img_Names.append("高斯模糊")
```

```
    # 第六部分：直方图正规化
    om = np.zeros(img.shape, np.uint8)
    cv.normalize(img, om, 255, 0, cv.NORM_MINMAX, cv.CV_8U)
    img_Lists.append(om)
    img_Names.append("直方图正规化")

    # 第七部分：伽马变换
    # 图像归一化
    fi = img / 255.0
    # 伽马变换
    gamma = 2
    om = np.power(fi, gamma)
    if save_flag:
        # 像素还原    否则保存为全黑图像
        om = om * 255.0
    img_Lists.append(om)
    img_Names.append("伽马变换")

    return img_Lists, img_Names


# 数据增强第一步
def enchanceDataOne(dataset_dir, save_path):
    """
    图片增强第一步
    :param dataset_dir: 图片数据所在的根目录
    :param save_path: 保存的文件目录
    :return:
    """
    for root, dirs, files in os.walk(dataset_dir):
        '''
        root 所指的是当前正在遍历的这个文件夹的本身的地址
        dirs 是一个 list ， 内容是该文件夹中所有的目录的名字(不包括子目录)
        files 同样是 list , 内容是该文件夹中所有的文件(不包括子目录)
        '''
```

```
        for sDir in dirs:
            # 当前文件夹下子目录的绝对路径
            imgsList = glob.glob(os.path.join(root, sDir) + '/*.jpg')
            # 拼接地址
            savePath = os.path.join(save_path, sDir)
            for imgPath in imgsList:
                # 图像增强与保存
                img_Lists, img_Names = ImageEnhancementImage(imgPath)
                for i in range(len(img_Lists)):
                    new_filename = imgPath[:-4] + '_%d_0.png' % (i)
                    new_filename = makeNewPath(savePath, new_filename)
                    img_Lists[i].save(new_filename)
                img_Lists, img_Names = ImageEnhancementOpencv(imgPath, save_flag=True)
                for i in range(len(img_Lists)):
                    new_filename = imgPath[:-4] + '_%d_0.png' % (i + 4)
                    new_filename = makeNewPath(savePath, new_filename)
                    cv2.imwrite(new_filename, img_Lists[i])


# 图片增强第二步
def enchanceDataTwo(dataset_dir):
    """
    图片增强第二步
    :param dataset_dir: 图片数据所在的根目录
    :return:
    """
    print(dataset_dir)
    for root, dirs, files in os.walk(dataset_dir):
        '''
        root 所指的是当前正在遍历的这个文件夹的本身的地址
        dirs 是一个 list ，内容是该文件夹中所有的目录的名字(不包括子目录)
        files 同样是 list , 内容是该文件夹中所有的文件(不包括子目录)
        '''
        for sDir in dirs:
            # 获取文件夹路径
```

```
        dir_path = os.path.join(dataset_dir, sDir)
        # 获取文件列表
        imgs_list = os.listdir(dir_path)

        # 所需生成各个分类的数据总量
        count = 1050
        # 计算图像保存概率 count=(1+7*rate)*len(imgs_list)
        save_rate = (count / len(imgs_list) - 1) / 7
        for img_name in imgs_list:
            # 原图
            img = openImg(dir_path, img_name)

            im_list = []
            # 随机操作
            # 原图转置
            if random.random() < save_rate:
                im_list.append(transverse(img))
            for i in range(3):
                # 旋转
                if random.random() < save_rate:
                    im_list.append(rotation(img, 90 * (i + 1)))
                # 转置
                if random.random() < save_rate:
                    img01 = rotation(img, 90 * (i + 1))
                    im_list.append(transverse(img01))

            # 图像保存
            for i in range(len(im_list)):
                mystr = "_%d.png" % (i + 1)
                new_filename = img_name.replace("_0.png", mystr)
                new_filename = makeNewPath(dir_path, new_filename)
                im_list[i].save(new_filename)


# 图片数据增强主函数
def enhanceDataMain():
```

```
    # 图片数据所在的根目录
    dataset_dir = 'D:\CNN\Location of infection'
    # 保存的文件目录
    save_path = "D:\CNN\Location of infection\enhance data"

    # #调用函数
    enchanceDataOne(dataset_dir, save_path)
    enchanceDataTwo(save_path)

if __name__ == '__main__':
    # 更换当前工作目录
    DIR = os.getcwd()
    os.chdir(DIR + "\..")

    start = time.time()
    print("START:", time.ctime())
    # 图片数据增强主函数
    # enhanceDataMain()



    print("END:", time.ctime())
    print("USE TIME:", time.time() - start)
```

B_Rename.py

```
# -*- coding:utf8 -*-

import os

class BatchRename():
    '''
    批量重命名文件夹中的图片文件
    '''

    def __init__(self):
        self.path = 'D:\CNN\Location of infection\Healthy fruit'  # 存放图片的文件夹路径
```

```
    def rename(self):
        filelist = os.listdir(self.path)
        total_num = len(filelist)
        i = 1
        for item in filelist:
            if item.endswith('.jpg'):  # 图片格式为 jpg

                src = os.path.join(os.path.abspath(self.path), item)
                dst = os.path.join(os.path.abspath(self.path), 'Hfr_' + str(i) + '.jpg')  # 设置新的
图片名称
                try:
                    os.rename(src, dst)
                    print("converting %s to %s ..." % (src, dst))
                    i = i + 1
                except:
                    continue

        print("total %d to rename & converted %d jpgs" % (total_num, i))

if __name__ == '__main__':
    demo = BatchRename()
    demo.rename()
```