

**计算机网络课程设计总结报告**



**设计题目：基于CNN和的植物病害叶片图像识别系统 ——识叶**

**学生姓名：易洋**

**系 别：生命科学学院**

**专 业：计算生物学**

**班 级：计算生物学班级 2020240601**

**学 号：2020141240137**

**指导教师：宋万忠**

2022年 6 月 11 日

**目录**

[1. 开发背景 3](#_Toc15008)

[2. 数据集介绍 3](#_Toc25030)

[3. 环境配置说明 3](#_Toc12933)

[4. 功能描述 4](#_Toc8417)

[5. 业务流程分析 4](#_Toc23159)

[5.1 识别叶片病害流程分析： 4](#_Toc27747)

[5.2 查询植物病害流程分析： 5](#_Toc21638)

[5.3 数据库维护流程分析： 6](#_Toc29848)

[6. 概念模型设计 7](#_Toc13408)

[7. 逻辑模型设计和优化 7](#_Toc22720)

[8. 应用层协议设计 8](#_Toc13671)

[9. 神经网络设计 9](#_Toc9038)

[10. 应用前端设计 10](#_Toc27252)

[11. 课程设计心得体会 12](#_Toc19744)

[12. 参考文献及其他资料 13](#_Toc31920)

1. **开发背景**

在自然情况下，园林植物的叶部最易受各种生物性因素和非生物性因素的危害，所以，叶部病害最为普遍，种类也最多。植物叶片病害影响叶片的光合作用效能，成为制约农作物高产、优质的主要障碍，影响作物的经济效益，药用植物品质、产量，观赏植物的欣赏价值等[3,4,5]。

如今，病虫害已成为各类作物减产的重要原因之一，而且随着引起植物病害的病毒、细菌等的变异，害虫抗药性的增强，农作物病虫害的发病率越来越高,具体的病害也越来越复杂。同时，传统的依靠农民自身经验或是靠专家指导的人工方法主观性强、准确率低、无时效性。使用更加快捷、准确的信息化，智能化病虫害识别、防治方法已迫在眉睫[8]。

为了提升对病害的高效率、高准确性识别、应对与预防能力，利用人工智能技术识别植物叶片的具体情况，并可以结合数据库中存储的相关信息给予病害相关的细节信息及应对措施建议。该系统的最终目的是允许用户通过植物叶片图片识别便捷地识别植物病害，并获取植物病害的细节信息及应对措施。

1. **数据集介绍**

项目所用数据集为来自飞桨AI studio平台的开源数据集Plant Village。Plant village数据集含14科植物叶片的图片，囊括了诸如蔷薇科植物等重要经济作物，禾本科植物等重要粮食作物的相关叶片常见病害状况，具有一定代表性和实际意义。

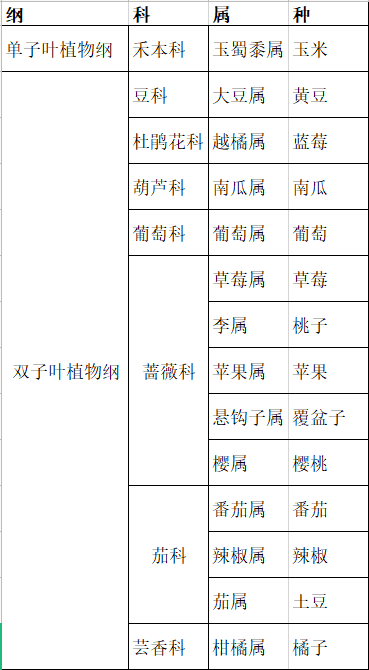
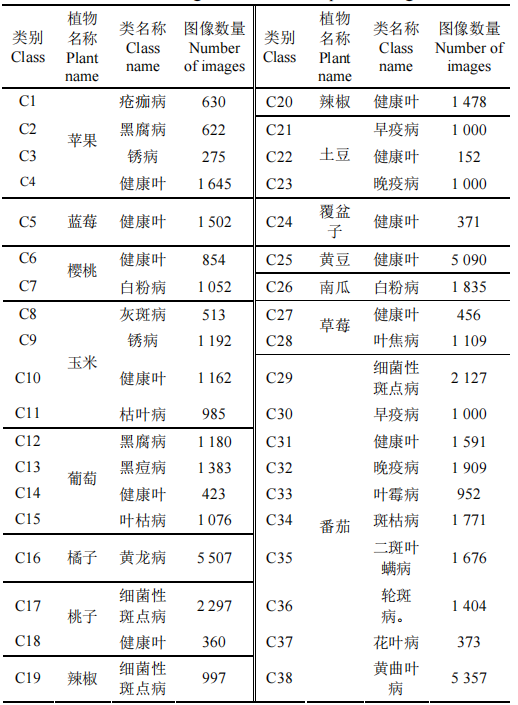


图1,2 plant village 数据集内容概要[2]

1. **环境配置说明**
   1. **硬件环境**

电脑型号：Lenovo Legion Y7000 2020

CPU型号：Intel CORE i5 10th gen @2.5GHz；

* 1. **软件环境**

操作系统：Windows10 10.0.19043

开发环境为：anaconda 2.1.2 spyder 4.2.5

Python版本：Python 3.8

主要包及其版本：

keras 2.8.0

mysql-connector-python 8.0.23

Pillow 8.2.0

PyMySQL 1.0.2

scikit-image 0.18.1

scikit-learn 0.24.1

scipy 1.6.2

tensorflow 2.8.0

其他软件环境（与神经网络正常载入与运行有关）：

NVIDIA CUDA v11.6

nvcc: NVIDIA (R) Cuda compiler driver

Copyright (c) 2005-2022 NVIDIA Corporation

Built on Thu\_Feb\_10\_19:03:51\_Pacific\_Standard\_Time\_2022

Cuda compilation tools, release 11.6, V11.6.112

Build cuda\_11.6.r11.6/compiler.30978841\_0

Cudnn 8.4.0

1. **功能描述**

经过设计与分析，“识叶”系统主要完成以下功能：

1. 植物信息存储：主要实现对信息的存储，包括按科——目——种——病害的植物分类存储以及各病害相关图片的存储。
2. 识别叶片病害：用户可以通过客户端打开植物叶片病害图像，并将图像上传至服务器，服务器端的神经网络计算识别后，将识别结果及附加信息返回客户端。
3. 提供病害细节：服务器返回对用户上传的图像识别结果之前，先将识别结果形成SQL语句，向数据库查询对应病害的细节信息及应对措施，作为提供给用户的“附加信息”之一。
4. 查询植物病害：主要支持用户对数据库的访问，允许用户查询数据库中的植物病害相关信息及应对措施。
5. 关键字查找：用户可以通过输入某关键字来查找数据库中存在的相关信息及关键字，如输入“苹”可以得到“苹果——疮痂病”“苹果——黑腐病”“苹果——健康”“苹果——锈病”等信息，便于查找。
6. 植物信息维护：管理员可以通过登录管理员界面以对数据库内容进行增、删和改，允许管理员修改植物分类信息，更新数据库中图片等。
7. **业务流程分析**
   1. **识别叶片病害流程分析：**

用户通过客户端打开本地图像后，客户端系统将检查网络状况。若网络不佳，则利用客户端本地神经网络进行识别，并将图像对应植物病害（识别结果）显示在客户端上，完成一次查询。若网络状态良好，则将图片发送至服务器端，服务器端神经网络接收上传数据作为输入后进行预测，将预测结果优先形成SQL语句访问数据库，随后数据库的查询结果（植物病害及应对措施）与服务器的识别结果一同返回至客户端，完成一次查询，如图3所示。

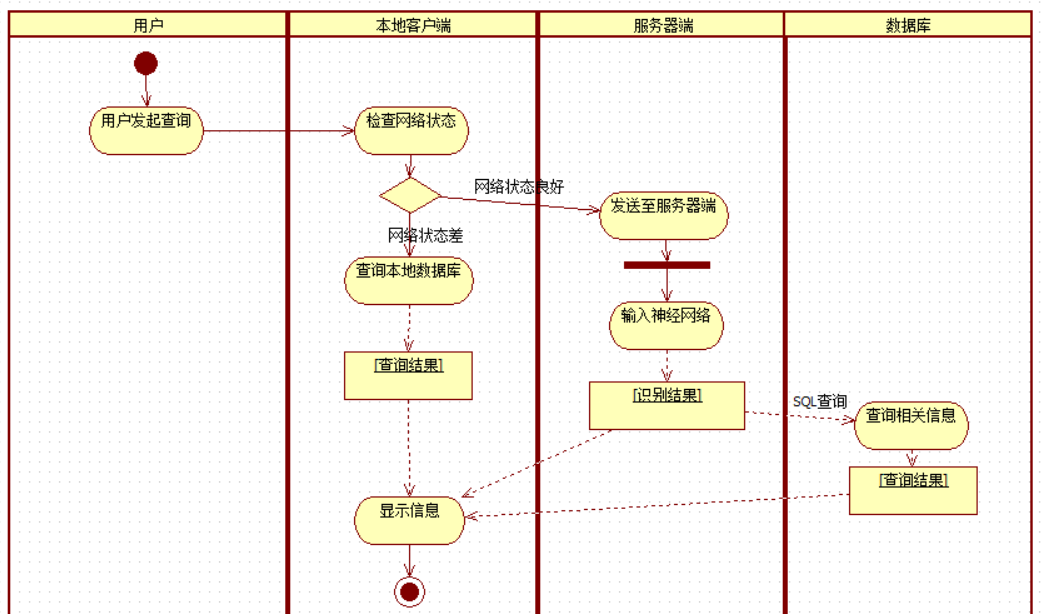


图3 图像识别业务流程分析图（活动图）

* 1. **查询植物病害流程分析：**

用户进入客户端查询页面后，根据界面的下拉菜单提示依次选择目的植物的纲、科、种（根据克朗奎斯特系统分类法）和待查询的植物病害后点击“查询”，客户端将输入内容形成MySQL语句后访问数据库，将查询结果返回客户端，如图4所示。

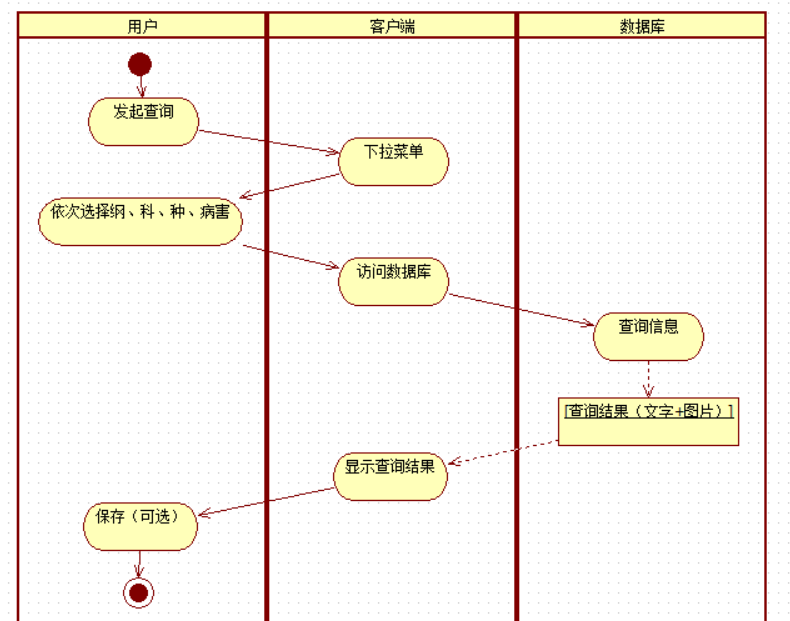


图4 信息查询业务流程分析图（活动图）

* 1. **数据库维护流程分析：**

管理员可以在查询界面的“病害”输入框中输入密码并点击五次“查询”进入管理员界面，在管理员界面下可以通过图形界面的形式或直接通过输入MySQL语句对数据库的信息进行增加、删除、修改或查询，如图5所示。

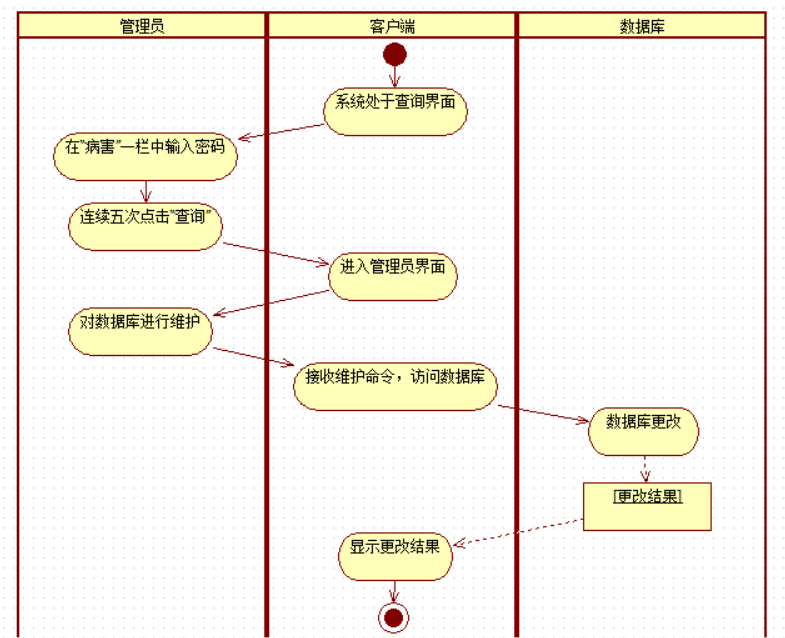


图5 数据库维护业务流程分析图（活动图）

1. **概念模型设计**

根据需求分析及数据字典，概念模型设计如图7示意。

说明：**由于数据库中实体及联系较多（纲——科——种——病害，表格数量指数增长），因此仅对单子叶植物纲进行较为完整的绘图，且用线条代替联系集的菱形图标，其余表格仅作示意。**其中，纲——科联系集、科——种联系集、种——病害联系集均为1-n的联系，用线条的形状而非单独的联系集符号表示出来。

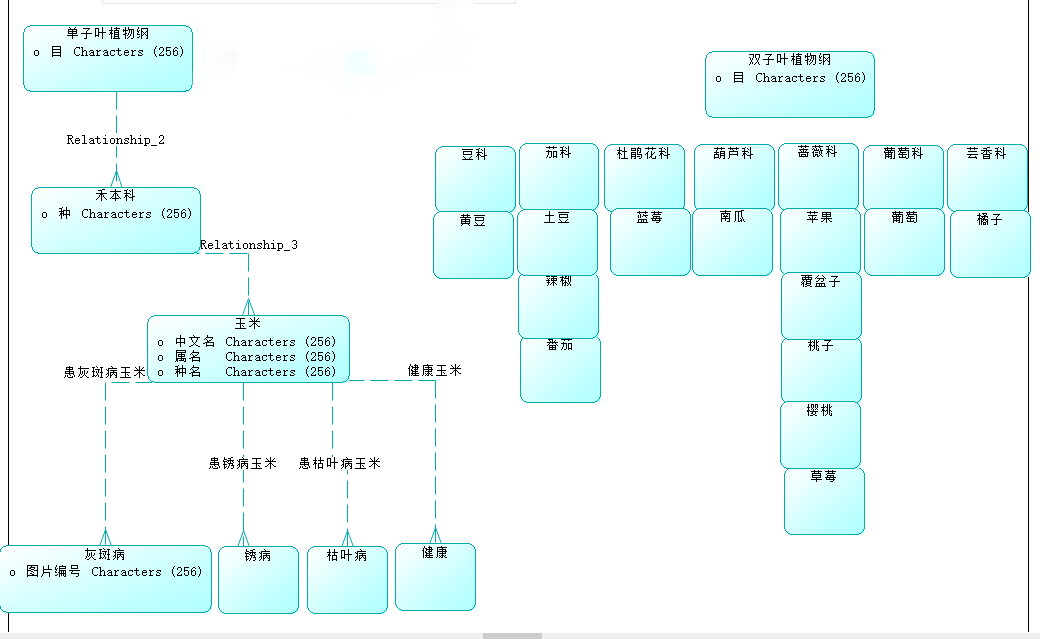


图7 总体概念模型E-R示意图

1. **逻辑模型设计和优化**

根据系统的总体概念设计模型、E-R图向关系模式的转化规则和数据库的范式理论，得到系统优化后的逻辑模型，如表1～表5所示。

括号中的单词为该级别的表格名称，如纲&科关系表（'Monocotyledons','Dicotyledoneae'）表示在纲与科的这一层关系中建立了两张表格，Monocotyledons（单子叶植物纲）和Dicotyledoneae（双子叶植物纲）。这样设计可能导致表格数量较多，但是考虑到每个表格，尤其是第四级（病害图片）表格中数据量较大，每个表格可能包含上千条数据（plant village数据集中图片有五万余张），如果同一级别只有一个表格，则执行SQL语句将极为缓慢，因此按照某种分类形式将同一级别的表格“拆分”开来。

例如，如果病害叶片图片表格不按照病害分为38个表格，而是将表格合并，因为表格中含有blob大文件，故在五万余条数据中执行一次查询语句可能耗费数秒时间甚至更久，这对于用户而言是难以忍受的，也将不必要的消耗数据库的服务，因此采取了更零散却更具性能的设计方式。

表1 纲&科关系表（'Monocotyledons','Dicotyledoneae'）

|  |  |
| --- | --- |
| 科 | 纲 |
|  |  |

表2 科&种关系表（'Gramineae','Fabaceae','Ericaceae','Cucurbitaceae', 'Vitaceae','Rosaceae','Solanaceae','Rutaceae'）

|  |  |
| --- | --- |
| 种 | 科 |
|  |  |

表3 病害&种关系表（'Corn','Soybean','Blueberry','Squash','Grape', 'Strawberry','Peach','Apple','Raspberry','Cherry', 'Tomato','Pepper','Potato','Orange'）

|  |  |  |  |
| --- | --- | --- | --- |
| 病害名称 | 种 | 细节信息 | 应对措施 |
|  |  |  |  |
|  |  |  |  |

表4病害叶片图片表

（'Apple\_\_\_Apple\_scab', 'Apple\_\_\_Black\_rot', 'Apple\_\_\_Cedar\_apple\_rust', 'Apple\_\_\_healthy', 'Blueberry\_\_\_healthy', 'Cherry\_\_\_healthy', 'Cherry\_\_\_Powdery\_mildew', 'Corn\_\_\_Cercospora\_leaf\_spot\_Gray\_leaf\_spot', 'Corn\_\_\_Common\_rust\_', 'Corn\_\_\_healthy', 'Corn\_\_\_Northern\_Leaf\_Blight', 'Grape\_\_\_Black\_rot', 'Grape\_\_\_Esca', 'Grape\_\_\_healthy', 'Grape\_\_\_Leaf\_blight', 'Orange\_\_\_Haunglongbing', 'Peach\_\_\_Bacterial\_spot', 'Peach\_\_\_healthy', 'Pepper\_\_bell\_\_\_Bacterial\_spot', 'Pepper\_\_bell\_\_\_healthy', 'Potato\_\_\_Early\_blight', 'Potato\_\_\_healthy', 'Potato\_\_\_Late\_blight', 'Raspberry\_\_\_healthy', 'Soybean\_\_\_healthy', 'Squash\_\_\_Powdery\_mildew', 'Strawberry\_\_\_healthy', 'Strawberry\_\_\_Leaf\_scorch', 'Tomato\_\_\_Bacterial\_spot', 'Tomato\_\_\_Early\_blight', 'Tomato\_\_\_healthy', 'Tomato\_\_\_Late\_blight', 'Tomato\_\_\_Leaf\_Mold', 'Tomato\_\_\_Septoria\_leaf\_spot', 'Tomato\_\_\_Spider\_mites\_Two\_spotted\_spider\_mite', 'Tomato\_\_\_Target\_Spot', 'Tomato\_\_\_Tomato\_mosaic\_virus', 'Tomato\_\_\_Tomato\_Yellow\_Leaf\_Curl\_Virus'）

|  |  |  |
| --- | --- | --- |
| 图片ID | 种 | 病害图片（PNG,BLOB） |
|  |  |  |

1. **应用层协议设计**

本系统通过自定义协议在客户端与服务器间传递消息（主要为传递图片和查询结果等），数据库与服务器、数据库与客户端的直接交流通过调用pymysql包完成。协议设计如下：

该自定义协议底层协议为TCP/IP，该协议在应用层上运行

该协议定义如下：

1 [total packet amount]: varies from 1 to 197 (4B)

2 [host IP]: in the form like '192.168.1.1' (15B at most)

3 [time stap]: current time, in the form like 2022-05-26 13:37:40 (19B at most)

4 [message]: at most 1000 bytes per packet (1000B at most)

5 'QueryTail' : query message has ended, else ‘Continued’ (9B)

第1部分为包（packet）的总数量。考虑到IP协议存在最大报文长度，因此该协议通过控制数据部分最大为1000字节来保证包长度不超过允许的范围；

第2部分为客户端地址；

第3部分为客户端创建该报文并加上头部的时间点；

第4部分为消息块，可以容纳最多1000字节的消息，若消息总长度大于1000则分片；

第5部分显示“是否为最后一个包”，考虑到TCP可靠数据传输会保证包到达的顺序与发送顺序一一致，因此若当前包第五部分为‘querytail’信息则表明本次数据传输结束，可以断开TCP链接。

客户端在发送图片（或其他文件前）按照最大字节数对应用层数据进行分片，装上报文头部后递交给TCP进行传输，服务器收到报文后按照对应协议格式分别提取出请求的类型、时间、正文（报文载荷）等信息后，按照情况进行数据库访问或神经网络查询，随后以相同的报文格式将数据返还客户端，告知用户识别与查询结果。

1. **神经网络设计**

“识叶”系统采取的神经网络为卷积神经网络（CNN），其架构如图8所示：

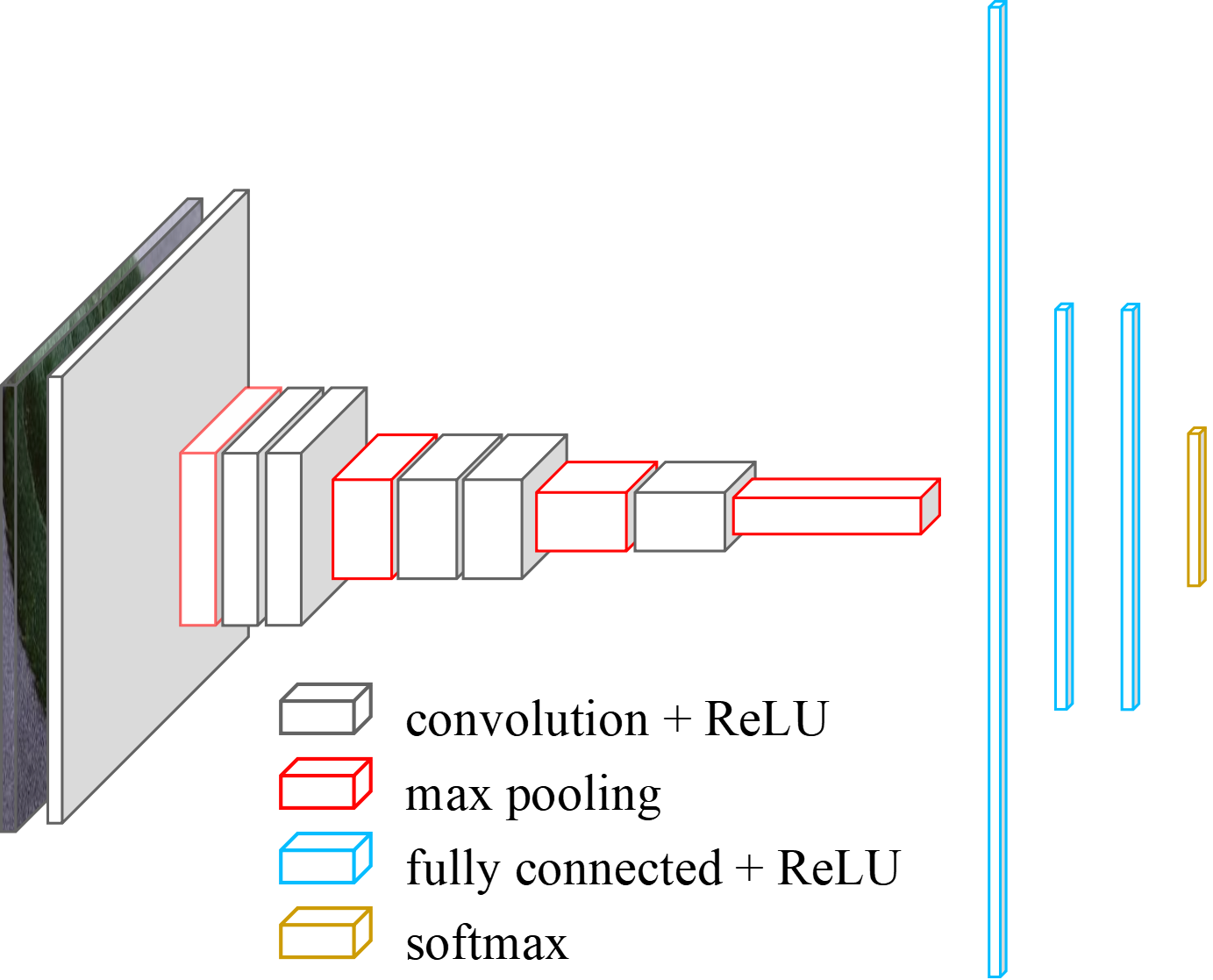


图8 本系统CNN结构示意图

该架构模仿VGG-16网络架构，考虑到硬件设备性能与时间等删除了部分卷积层与池化层，并简化了全连接层。该卷积神经网络通过卷积层与最大池化层交替提取图片特征信息，通过全连接层对信息进行进一步处理，最终通过softmax函数进行38分类，多次训练后以实现对植物叶片病害的准确识别。

训练采取Python中的TensorFlow.keras包，训练过程中参数设置如下：batch size = 16, epoch = 11, 激活函数除最后一层均为ReLU函数，卷积层padding = ‘same’。经过约55小时的训练，结果如图9所示：

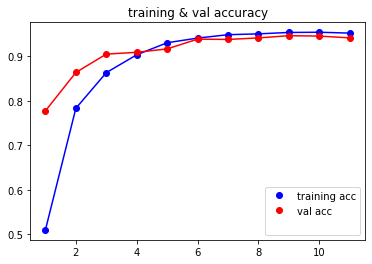


图9 模型训练性能变化曲线

模型训练集:测试集=7:3，由上图可知在大约第6批次时模型在训练集与测试集上的准确率都相对稳定，最终模型在测试集上的准确率为94.10%，模型预测性能较令人满意。

1. **应用前端设计**

在设计之初，本系统期望运行端为手机端，目前尚未实现手机端的开发，因此在电脑上模仿手机端的屏幕比例进行界面设计。



图10 系统主界面（主菜单）

图10为系统的主界面。在上方搜索框输入关键词，可以搜索系统数据库已记录的相关病害名称，便于用户快捷查找期望查询的病害内容；点击中间的相机按钮可以进入视图界面，识别用户上传的照片；点击查询可以进入查询界面，通过图形界面的操作实现对数据库的访问与查询。

图11,12 识图界面（选择图片前与选择图片后）

图11，图12展示系统的查询界面。点击“选择图片”可以唤起文件资源管理器，选择图片后会显示在中央，点击识别图片后，系统将向服务器和数据库发起请求，并返回查询结果界面。



图13 识图结果界面

图13展示系统显示查询结果的界面。左上角为缩放后的上传图片，右上角为物种名与病害名，下方文字框显示植物病害细节及应对措施。其中字体与格式尚有待改进。



图14,15,16 查询界面

图14,15,16展示系统查询界面，用户可以通过四个下拉菜单一次选择目标病害植物的纲、科、种和病害名称，从而实现对数据库的访问。点击查询按钮后，系统将在左下角显示一张具有代表性的示意图，右下角显示细节与对策。

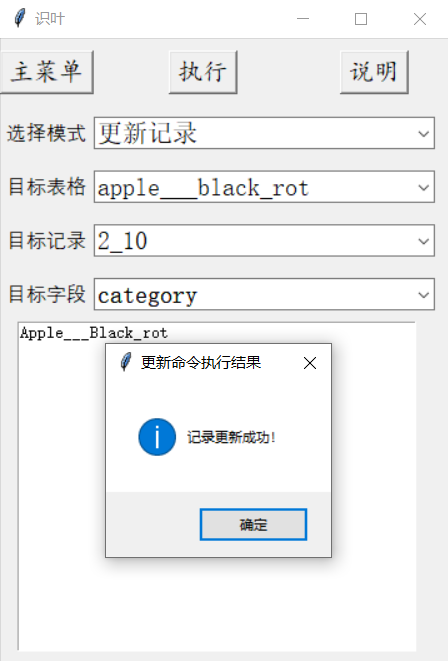
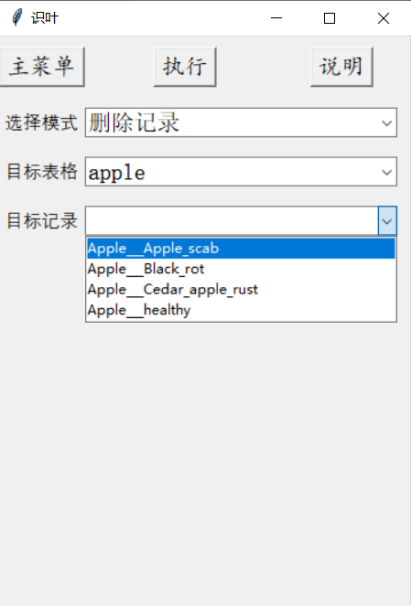


图17 进入管理员界面 图18,19 管理员界面

在病害情况一栏中输入管理员密码，并点击五次查询按钮将进入管理员界面，如图17所示。管理员可以通过此界面进行数据库的维护与更新，包括对数据库的内容进行增加、删除与修改，如图18，19所示。

1. **课程设计心得体会**

本次项目开发是我第一次尝试开发一个比较大型，功能相对丰富的系统，包括需求、设计、编码、测试等软件工程的全环节，是一次崭新的体验，也是一次勇敢但没有把握没有自信的尝试。

在本次项目开发中，我可以用上许多课程中学到的知识，包括但不限于植物生物学、花卉学、人工智能、程序设计基础、数据库、计算机网络、软件设计与体系结构等等，将两年来学到的许多生命科学和计算机科学、软件工程的知识都加以实践，是一件激动人心的事情。

当一个个功能实现——当神经网络训练完成，当服务器与客户端可以正常通过计算机网络交流，当软件可以连通并访问数据库，当图形界面与功能正确交互时，无论效果如何，总会有一种成就感，即便结果可能确实“功能不完备，质量不完善”。

然而这次项目开发的不足之处还是显而易见的。因为需求只是个人的需求，只需要理清思路即可，没有遇到太大的阻碍。在设计阶段，问题便接踵而至——不管是应用层协议的设计，还是软件架构的设计，或是图形界面的设计都让人有些头疼，并且自己之前也没有任何经验可以参考。尤其是软件架构非常不合理，一边编码，一边眼睁睁地看着自己代码的耦合度越来越高，并且后续单元测试时一旦发现bug，修改一个模块可能牵扯到不止一个类的时候，真切地感受到自己设计的能力实在不足。此外简陋的图形界面也表明了能力上的缺乏与经验的不足。

因此，这次开发是一次重要的经历，也有不少的经验教训供我总结。设计与编码能力的提高需要继续大量学习，也需要通过实践来获取，而数据库、神经网络的设计与搭建则需要更加细化的专业知识支持。此外，开发方法也有待改进，也可以尝试诸如“图形界面驱动的开发”“测试驱动的开发”等方式，也许可以弥补一些设计能力上的不足。

此外，本项目设计之初期望其功能可以真正用于现实生活中，但是当神经网络训练一段时间后就发现有难度。就方法而言，没有采取任何的数据预处理，也没有采取一些数据增强的手段，导致本模型难以识别一些不同光照环境下拍摄的植物叶片；就数据集而言，plant village数据集中皆为256x256像素的RGB图像，清晰度及其有限，且均为精致摆放的单片叶子。上述方法与数据源的不足决定了本项目只能是一次尝试，无法用于实际生活当中。

这个项目也暴露出个人的创造性不足，与生命科学专业素养的不足。在需求阶段，我尝试思考人工智能，数据库与生命科学能有哪些具有创新性的交互，但是却没有什么高可行性的结果，因此最终只是设计了一个以人工智能识图和数据库访问为主要功能的系统，创新能力不足，对生物的认识也非常局限，进步空间仍然极大。

1. **参考文献及其他资料**

[1] 史冰莹,李佳琦,张磊,等. 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(6):89-96. DOI:10.15888/j.cnki.csa.007471.

[2] 刘洋,冯全,王书志. 基于轻量级CNN的植物病害识别方法及移动端应用[J]. 农业工程学报,2019,35(17):194-204. DOI:10.11975/j.issn.1002-6819.2019.17.024.

[3] 周红梅. 园林植物叶部病害的症状鉴别方法及综合防治技术[J]. 城市建设理论研究（电子版）,2015(23):3104-3105.

[4] 韩行成. 植物叶片病害及其防治[J]. 养殖技术顾问,2012(1):248-248. DOI:10.3969/j.issn.1673-1921.2012.01.242.

[5] 李世东. 植物医学:我国农业现代化进程中的病害治理挑战、机遇和创新[J]. 植物保护,2022,48(2):1-8,15. DOI:10.16688/j.zwbh.2021595.

[6] https://mp.weixin.qq.com/s/n-yd7cN9Ycc9jXvcidq3QQ从MobileNet看轻量级神经网络的发展

[7] Fei-Fei Li’s TED talk

[8] 基于深度学习的农作物病虫害图像识别技术研究进展，贾少鹏 高红菊 杭潇，2019年7月，农业机械学报.

[9] 《数据库系统概念》 Abraham Silberschatz, Henry F.Korth著，机械工业出版社

[10] 《计算机网络 自顶向下方法》 James F.Kurose, Keith W.Ross著，机械工业出版社

[11] 四川大学生命科学学院、计算机/软件学院相关课程：植物生物学、花卉学、人工智能导论、面向对象程序设计导论、数据库系统和信息管理、计算机网络和分布式系统等