Vrije Universiteit Amsterdam

Universiteit van Amsterdam

Master Thesis

Designing a plant disease detection system for Chinese and Global Agriculture - an AI4D research project

Author: Ling Jin (2757276)

1st supervisor:	Anna Bon
daily supervisor:	Anna Bon
2nd reader:	Hans Akkermans

A thesis submitted in fulfillment of the requirements for the joint UvA-VU Master of Science degree in Computer Science

June 26, 2024

"科学种田，五谷丰登。"

"With scientific farming, we will have abundant harvests."

Abstract

Context. In recent years, due to climate change and environmental degradation, the problem of crop pests has become increasingly severe. These pest problems threaten food security and severely impact farmers' economic conditions, especially in remote and resource-limited areas. In China's remote regions, many farmers lack sufficient resources and expertise to effectively detect and manage crop pests and diseases, leading to frequent crop losses, exacerbating poverty, and food insecurity in these areas. Therefore, developing a system that can timely and accurately identify and manage crop pests is crucial for ensuring agricultural production and improving farmers' living standards

Goal. The goal of this study is to develop a plant pest detection and analysis system based on machine learning and artificial intelligence technology. Farmers or plant growers can take or upload images of diseased plant leaves for the system to analyze. The system model will return the pest results and use AI to provide reasonable solutions. This system will be particularly suitable for resource-limited areas that lack professional knowledge and tools.

Method. This study developed a plant pest detection and analysis system using the following main methods: Data Collection, collecting a large number of detailed examples of crop diseases leaves, and labeling the names and severity of various pests and diseases; Model Training, using the collected data to train a highly accurate pest and disease recognition model through efficient model training; System Development, developing the system using Uniapp and PyCharm to call artificial intelligence technologies, enabling the system to be packaged as an application for users to download and use.

Results. The plant disease detection system consists of a ResNet-50 model, an AI model, and a front-end user interface mobile application. The trained model achieves an overall accuracy of 87%. This indicates that the model training is effective, with the model's performance continuously improving and remaining stable and accurate by the end of the training. The AI model also demonstrates a certain level of stability and usability when invoked. Additionally, the front-end design meets user needs and has received positive evaluation and validation from three reviewers.

Conclusions. The plant pest detection and analysis system developed in this study provides farmers with scientific and effective pest control solutions through modern technology, ensuring food security and promoting sustainable agricultural development in China and globally. The application of this system not only improves crop yield and quality but also significantly enhances the economic conditions and living standards of farmers. Future research and optimization will focus on expanding the system's applicability and improving the accuracy of the recognition model to address more complex agricultural pest and disease problems.

Contents

List of Figures

List of Tables

1

Introduction

The world population is projected to reach 9.8 billion in 2050, and 11.2 billion in 2100,(1), and the demand for food production is experiencing exponential growth. China, as the most populous country in the world, has a crucial role in agricultural production to ensure national food security. However, in recent years, due to climate change and environmental degradation, crop pest issues have become increasingly severe. From 1970 to 2016, the national average CPD[1] incidence rate increased fourfold (from 53% to 218%) (2). This presents a significant challenge to China's agricultural production. Extreme weather events, such as frequent floods, droughts, and irregular rainfall, not only affect crop growth cycles and yields but also make crops more susceptible to pests and diseases. Pest problems not only threaten food security but also severely impact farmers' economic conditions, especially in remote and resource-limited areas.

In China's remote regions, many farmers lack adequate resources and expertise to effectively detect and manage pests and diseases in crops. This leads to frequent crop losses, exacerbating poverty and food insecurity in these areas. Therefore, developing a system capable of timely and accurately identifying and managing crop pests is crucial for ensuring agricultural production and improving farmers' living standards.

To address this issue, this paper introduces a plant pest detection and analysis system based on machine learning and artificial intelligence technologies, developed using Uniapp for web development, with the capability to be packaged as an application system for download and use. The system uses a large number of detailed crop disease leaf pictures for model training, with labels recording the names and severity of various pests and diseases, providing rich data resources for model training. Through efficient algorithm optimization and model training, this system not only improves the accuracy of pest and

[1] Crop pests and diseases

disease identification but also significantly reduces the time required for identification. This is particularly important for resource-limited areas, which often lack sufficient expertise and tools to combat pests and diseases.

This system enhances the efficiency and accuracy of pest detection, greatly reduces the time needed for identification, helps farmers and agricultural managers respond to pest problems promptly, reduces pesticide use, lowers environmental pollution, and improves the sustainability of agricultural production. The development of this project aims to provide farmers with more scientific and effective pest control solutions through modern technological means, ensuring food security and promoting sustainable agriculture in China and globally.

2

Background

With global climate change and the continuous evolution of agricultural production methods, plant pest problems are becoming increasingly serious, posing a huge challenge to agricultural production. Pests not only directly damage crops and lead to reduced yields, but also cause plant diseases by spreading pathogens, further affecting agricultural output and quality. According to statistics, China's agricultural losses caused by pests amount to hundreds of millions of yuan each year, which has brought a heavy economic burden to farmers and agricultural enterprises. The amount of farmland hit by crop pests and diseases in China has quadrupled in the past 50 years, according to a recent study. Climate change was found to be responsible for more than a fifth of this change – an ominous warning for the agricultural industry in a warming world.(3)

The impact of climate change-induced abiotic stress on ecosystems is expected to detrimentally affect pest diversity and abundance. Diverse populations of organisms play a crucial role in naturally regulating pest and disease outbreaks through various mechanisms (4). However, the decline in specific species undermines these natural controls, leading to increased vulnerability of crops to pest infestations and diseases. For instance, the devastating locust outbreak in East Africa in 2020 (5)severely impacted countries like Kenya, Ethiopia, and Somalia, resulting in extensive agricultural damage and jeopardizing food security for millions. The reduction in beneficial insect populations exacerbates the risk of pest outbreaks while decreasing plant diversity limits crop genetic variability. This genetic uniformity heightens crop susceptibility to emerging diseases and pests, hampering their ability to withstand environmental stresses and adapt effectively.

In China, climate change has increased the frequency and severity of pest outbreaks, making Chinese agriculture highly vulnerable. Rising temperatures and changes in precipitation patterns create favorable conditions for pests, potentially leading to crop death and reduced yields. In such situations, farmers use pesticides to kill these pests, despite the significant environmental damage, harm to biodiversity, and risks to human health. Particularly in remote areas of China, some small farms still need to use pesticides extensively. As shown in Figure 2.1, in January 2024, a farmer in a remote rural area of Zhejiang Province manually sprayed pesticides.

Not only in China, but globally, more and more pesticides are being used in agriculture. According to data from the Food and Agriculture Organization of the United Nations, the amount of pesticides used rose by about 60% between 1990 and 2020.(6)

Figure 2.1: A photo of farmers manually spraying pesticides

Traditional pest detection methods among small farmers involve manual inspection and empirical judgment, which are time-consuming and labor-intensive. These methods are also limited by inspectors' expertise, making large-scale, rapid, and accurate pest monitoring challenging. Moreover, these methods often fail to promptly respond to sudden pest outbreaks, resulting in increased damage.

Advances in modern technology offer solutions for more effective pest detection. AI-based pest detection systems use high-resolution cameras and deep learning algorithms to swiftly and accurately identify pest species and assess damage levels from image data. This improves detection efficiency and accuracy, enabling farmers and agricultural managers to quickly address pest issues, reduce pesticide use, minimize environmental impact, and enhance agricultural sustainability.

3

Related Work

In the field of plant pest and disease identification and control, extensive research has been conducted, particularly focusing on advancements in China and globally. This section delves into the related inventions and technological developments in this area, highlighting notable studies and their contributions.

Domestic Research in China

In China, significant progress has been made in the identification and control of plant pests and diseases.

One prominent study by Yiwen Liu(7) and colleagues proposed a method for crop pest and disease identification based on an improved transfer learning network. They constructed and fine-tuned an image analysis model based on the VGG16(8) and Inception-ResNet-v2(9) networks. By integrating these networks through ensemble algorithms, they effectively enhanced the performance of the pest and disease identification model, ensuring both completeness and accuracy.

Fangyan Zhang et al.(2021)(10) developed a method using ResNet-50 and transfer learning to detect diseased Takifugu rubripes[1]. They pre-trained ResNet-50 on the ImageNet dataset, transferred the weights to the model, and added a deconvolution layer to improve detail detection. Data augmentation techniques were used to increase sample diversity. The method achieved 99% accuracy, outperforming other ResNet models(ResNet-18, ResNet-34, ResNet-101, and ResNet-152 by 10.7%, 6.6%, 6.2%, and 5.6% respectively.) and showing a 0.4% improvement over ResNet50 without deconvolution. This approach effectively addresses the challenges of small sample sizes and low accuracy, providing a reference for future research in fish disease detection and image classification.

[1] https://en.wikipedia.org/wiki/Takifugu$_{r}ubripes$

Chen Bingcai(11) used the Plant Village[1] dataset to evaluate the performance of three CNN models (ResNet-50, Google Net(12), VGG16) in plant disease detection in their study, and tested feature extraction and transfer learning in combination with SVM and KNN[2] classifiers. The results show that SVM performs better than KNN in feature extraction, while in transfer learning, the accuracy of VGG16 is 97.82%, the accuracy of ResNet-50 is 95.3%, and the accuracy of Google Net is 95.3%, compared with traditional feature extraction methods, deep feature extraction and transfer learning significantly improve the classification accuracy, indicating that deep learning technology has broad application prospects in the field of plant disease detection.

Zhibin Wang and his team(13) highlight that climate change and agricultural system changes have exacerbated the impact of pests and diseases. While conventional pesticide treatments are effective, concerns over environmental impact and resistance have arisen. Therefore, they have developed a green crop pest management system based on IoT technology, integrating ozone disinfection and light induction techniques. They have designed two devices suitable for both controlled environment and field production, equipped with sensors for light intensity, temperature, humidity, and monitoring cameras. This system enables real-time data collection and monitoring of environmental conditions, aiming to reduce chemical pesticide usage and promote sustainable agricultural development.

International Research

Globally, there have been parallel advancements in this field.

"Machine learning is changing or will change every industry, and leaders need to understand the fundamentals, potential, and limitations,"(14) said Aleksander Madry, a professor of computer science at MIT. As AI technologies advance, their application in agriculture is becoming more widespread and is expected to be adopted on a large scale in the next few years. Agricultural technology companies such as Croptracker[3], Aibono[4], and Hortau[5] provide farmers with efficient solutions and improve the efficiency and profitability of agricultural production by developing and applying advanced AI and machine learning technologies. technology to optimize crop quality management.

For instance, Aitor Gutierrez(15) conducted a study in 2019 focusing on the major pests in greenhouse-grown tomatoes and peppers. He developed and compared two automated visual detection and identification methods for pests: one utilizing computer vision and

[1] Plant village
[2] SVM VS KNN in Machine Learning
[3] Croptracker
[4] Aibono
[5] Hortau

machine learning techniques, and the other employing deep learning technologies. The research concluded that deep learning algorithms excelled in image-processing tasks. However, their "black-box"[1] nature made it challenging to understand the reasoning behind their predictions. Conversely, computer vision and machine learning approaches require less data and training time, offering a more transparent methodology.

Maria Eloisa Mignoni et al.(2024)(16) reviewed the advances in pest control using AI tools and image technology. Their study focused on pest classification, insect identification, UAV image capture, and the application of deep learning (DL) and convolutional neural networks (CNN). They conducted searches across 5 databases, reviewed 71 articles through rigorous screening, and analyzed 47 articles. The findings demonstrate that DL and CNN tools using real images show significant potential in insect control, enhancing efficiency and accuracy. These advanced technologies provide important insights and guidance for the future application of AI in agriculture.

In summary, both domestically and internationally, research on plant pest and disease identification and control based on image technology and artificial intelligence has achieved important results, providing solid technical support and direction for the future development of agriculture.

[1] In deep learning, "black box" means that these complex models (especially neural networks) are difficult for users to understand internally, even though they can see the input and output.

4

Design

4.1 Model Training

4.1.1 Data Collection

I have collected and organized a large dataset of plant disease leaf images. These datasets are the core of the system, as machine learning models require a substantial number of samples for training to improve recognition accuracy. Based on my research and review of literature, I obtained the data from the AI Challenger[1], a platform for open datasets and programming competitions for artificial intelligence (AI) talents around the world. The data[2], sourced from the 2018[3] competition resources, includes 61 categories (classified by "species-disease-severity"), 10 species, and 27 diseases (of which 24 diseases have both mild and severe degrees), and 10 healthy classification, as shown in Figure 4.1, an image of plant disease. The specific category classifications are shown in Table 4.1 (which lists the crop disease categories included in the dataset). The total number of training images is 31,718, the total number of validation images is 4,540, and the total number of test validation images is 4514, as seen in Table 4.2. The specific quantities for each category in the train dataset and validation dataset are detailed in Tables 4.3 and 4.4. From Table 4.4, it can be concluded that category 45 is missing in the validation set. Hence I deleted this label and kept the last 60 categories to do the training.

---

[1] AI Challenger
[2] Dataset
[3] Competition content

Figure 4.1: A plant with general symptoms of Apple Scab disease from validation dataset

Table 4.1: Crop Disease Categories of 10 Species

Label	Label Name	Label	Label Name
0	Apple healthy	31	Pepper Scab general
1	Apple Scab general	32	Pepper Scab serious
2	Apple Scab serious	33	Potato healthy
3	Apple Frogeye Spot	34	Potato Early Blight Fungus general
4	Cedar Apple Rust general	35	Potato Early Blight Fungus serious
5	Cedar Apple Rust serious	36	Potato Late Blight Fungus general
6	Cherry healthy	37	Potato Late Blight Fungus serious
7	Cherry Powdery Mildew general	38	Strawberry healthy
8	Cherry Powdery Mildew serious	39	Strawberry Scorch general
9	Corn healthy	40	Strawberry Scorch serious
10	Cercospora zeae-maydis general	41	Tomato healthy
11	Cercospora zeae-maydis serious	42	Tomato Powdery Mildew general
12	Puccinia polysora general	43	Tomato Powdery Mildew serious
13	Puccinia polysora serious	44	Tomato Bacterial Spot general
14	Corn Curvularia leaf spot fungus general	45	Tomato Bacterial Spot serious
15	Corn Curvularia leaf spot fungus serious	46	Tomato Early Blight Fungus general
16	Maize dwarf mosaic virus	47	Tomato Early Blight Fungus serious
17	Grape healthy	48	Tomato Late Blight Water Mold general
18	Grape Black Rot Fungus general	49	Tomato Late Blight Water Mold serious
19	Grape Black Rot Fungus serious	50	Tomato Leaf Mold Fungus general
20	Grape Black Measles Fungus general	51	Tomato Leaf Mold Fungus serious
21	Grape Black Measles Fungus serious	52	Tomato Target Spot Bacteria general
22	Grape Leaf Blight Fungus general	53	Tomato Target Spot Bacteria serious
23	Grape Leaf Blight Fungus serious	54	Tomato Septoria Leaf Spot Fungus general
24	Citrus healthy	55	Tomato Septoria Leaf Spot Fungus serious
25	Citrus Greening Disease general	56	Tomato Spider Mite Damage general
26	Citrus Greening Disease serious	57	Tomato Spider Mite Damage serious
27	Peach healthy	58	Tomato Yellow Leaf Curl Virus general
28	Peach Bacterial Spot general	59	Tomato Yellow Leaf Curl Virus serious
29	Peach Bacterial Spot serious	60	Tomato Yellow Leaf Curl Virus
30	Pepper healthy		

Table 4.2: Sample Counts of Datasets

Dataset	Sample Count
Training	31,718
Validation	4,540
Test	4,514

Table 4.3: Number of images of each label in the training set

Label	Count	Label	Count	Label	Count
0	1185	21	419	41	1208
1	211	22	61	42	319
2	152	23	630	43	966
3	427	24	367	44	1
4	142	25	1828	45	1
5	40	26	1799	46	251
6	598	27	251	47	442
7	116	28	857	48	264
8	110	29	770	49	1109
9	376	30	1025	50	325
10	191	31	287	51	336
11	167	32	377	52	43
12	483	33	1430	53	22
13	355	34	203	54	421
14	208	35	510	55	807
15	498	36	251	56	542
16	815	37	446	57	271
17	294	38	242	58	1414
18	381	39	192	59	2473
19	462	40	583	60	261
20	503				

Table 4.4: Number of images of each label in the validation set

Label	Count	Label	Count	Label	Count
0	169	20	74	40	83
1	30	21	59	41	173
2	22	22	9	42	46
3	61	23	90	43	138
4	20	24	52	44	1
5	6	25	269	46	36
6	85	26	262	47	63
7	12	27	36	48	38
8	18	28	122	49	158
9	54	29	110	50	46
10	27	30	147	51	48
11	24	31	40	52	4
12	69	32	54	53	5
13	51	33	204	54	60
14	29	34	29	55	115
15	71	35	73	56	77
16	116	36	36	57	39
17	42	37	64	58	202
18	54	38	35	59	353
19	66	39	27	60	37

4.1.2 Data preprocessing

Due to the collected raw data having varying sizes and formats, it is not possible to use them in a unified model for training. To ensure consistency in specifications, dimensions, and formats, data processing is required. For the training set, data augmentation operations will be applied; for the test and validation sets, only basic preprocessing will be conducted.

Training Set: Images will be standardized so that pixel values conform to the statistical properties of the pre-trained model. Image size will be adjusted to 224x224 pixels. Random augmentations will include rotation (between -30 and +30 degrees), horizontal flip, vertical flip, affine transformations (up to 45 degrees rotation), converting images from PIL format to PyTorch tensors, adding Gaussian noise, and randomly altering image brightness. An example of data preprocessing results is shown in Figure 4.2.

Validation and Test Sets: Image size will be adjusted to 224x224 pixels, converted to tensors, and standardized.

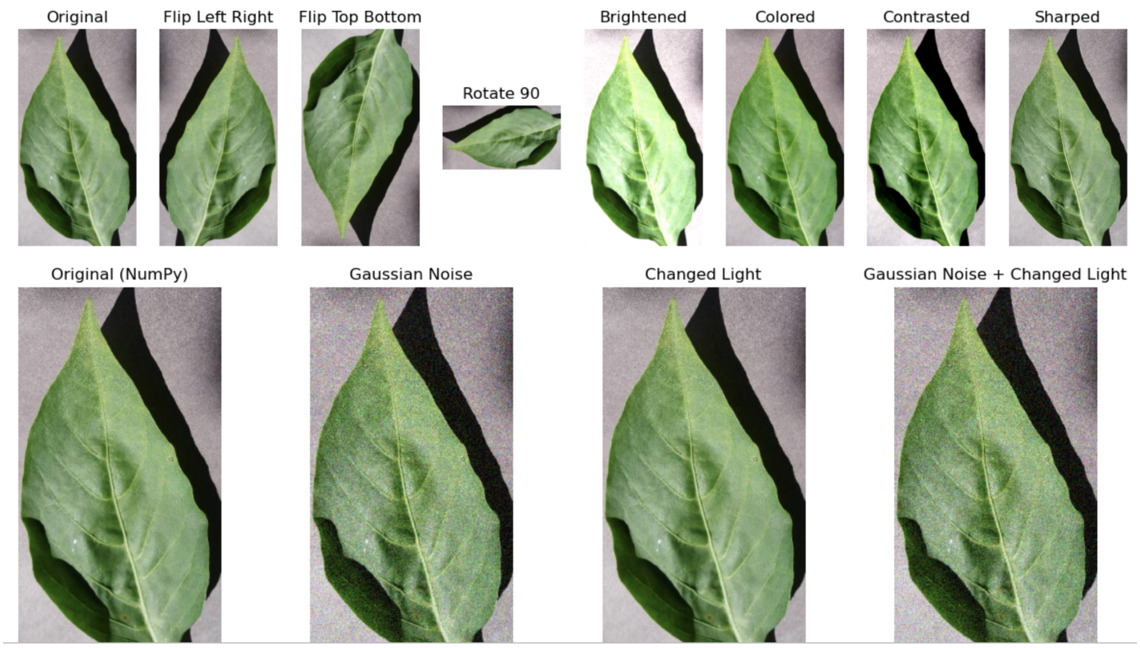


Figure 4.2: Partial preprocessing images

By implementing a custom dataset class and preprocessing functions, we aim to load, preprocess, and augment the image data effectively to enhance the training and generalization capabilities of the image classification model.

4.1.3 Model Selection

I employed the ResNet-50[1] model. ResNet-50 is CNN architecture that belongs to the ResNet (Residual Networks) family, a series of models designed to address the challenges associated with training deep neural networks.(17) ResNet-50 is widely used in image classification, object detection, image segmentation, and other computer vision tasks, making it a foundational model for many deep learning applications. It is a convolutional neural network with a depth of 50 layers, and its core innovation lies in the introduction of residual blocks. For the specific requirements of this project, I processed a large amount of data and used it to train the ResNet-50 model, establishing the foundational architecture for the plant disease analysis system, the following is a detailed description of the project functions and the ResNet-50 architecture. The specific structure can be seen in Figure 4.2.

[1] ResNet50

- **Input Layer: Zero Padding:**Input plant leaf images, add padding to maintain dimensions.
- **Stage 1: Layers:** Convolutional layer, Batch Normalization, ReLU activation, and Max Pooling. **Function:** Extract low-level features like edges and textures, reduce spatial dimensions, and prepare for subsequent stages.
- **Stage 2: Layers:** One convolutional block and two identity blocks. **Function:** Extract more complex local features, and identify simple pest spot shapes and structures.
- **Stage 3: Layers:** One convolutional block and three identity blocks. **Function:** Extract mid-level features, and deepen the model's learning capacity.
- **Stage 4: Layers:** One convolutional block and five identity blocks. **Function:** Capture high-level features and patterns, recognize overall disease structures, and distinguish different pest types.
- **Stage 5: Layers:** One convolutional block and two identity blocks. **Function:** Extract global features, and prepare for global average pooling and final classification.
- **Output Layer: Global Average Pooling:** Generate a 1D feature vector. **Flattening:** Convert to a format suitable for the fully connected layer. **Fully Connected Layer:** Output probability distribution of disease categories.

Through the above structure, ResNet-50 can gradually extract and process the features of plant disease images, and ultimately achieve accurate classification and recognition.

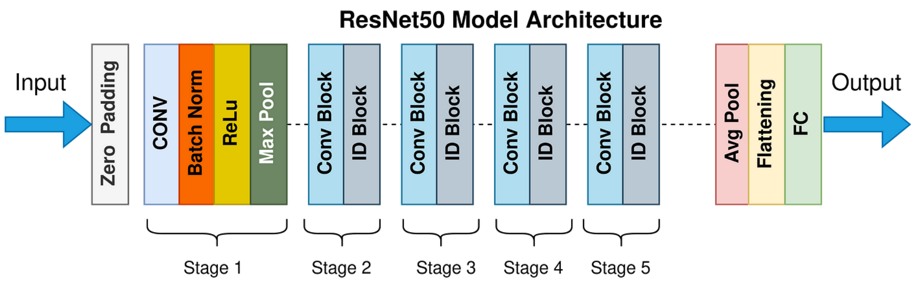


Figure 4.3: The model structure of ResNet-50

Specifically, in this project, ResNet-50 is applied to classify and identify various plant disease images. Through training on a large dataset of annotated images, ResNet-50 can accurately distinguish different types of plant diseases, providing high-quality analysis results. By combining the natural language processing capabilities of AI system with the image recognition capabilities of ResNet-50, this project has built a powerful, user-friendly plant disease diagnosis system.

4.2 AI Technology

In the agricultural field, early detection and treatment of plant diseases are key to ensuring healthy and high-yield crops. However, most farmers tend to rely on their experience when dealing with plant diseases, which may lead to plant diseases not being discovered and treated in time. To solve these problems, I introduced Zhipu AI[1] into the project, which uses advanced artificial intelligence technology to retrieve plant diseases and provide treatment recommendations. Even though there are many excellent AI systems on the market, for example, GPT-4(18)and Google Assistant(19), I chose Zhipu AI for the following reasons:

- **Outstanding Chinese and Cultural Understanding:** Based on the ChatGLM(20) model, Zhipu AI has excellent Chinese comprehension and adaptation to Chinese culture, making it very suitable for applications aimed at Chinese users.
- **Bilingual Support in Chinese and English:** Zhipu AI supports bilingual dialogues in Chinese and English and has been optimized for Chinese QA and conversations. This capability meets the needs of both Chinese and international users.
- **Open Source and Cost-Effective:** Zhipu AI is an open-source platform, that allows users to make free calls within certain limits. This provides economic support for project development.

4.3 System Structure Design

4.3.1 Front-End Development

Using HBuilderX[2] to Create Uniapp(21) Front Structure

- **Page Layout and Function Implementation:**

[1]Zhipu AI
[2]HBuilderX

- Design and implement a user interface that includes an image selection feature, an image upload button, and an area to display the prediction results.
- Utilize Uniapp's Plugin **uni-file-picker** to enable image selection, allowing users to choose images via their mobile phone camera or gallery for upload. And send the picture to the Back-end via **uni.uolioadFile** plugin.

- **Communication with the Back-end:**
  - Configure the server address within the front-end code.
  - Use Uniapp's network request functionality to upload the selected images to the back-end server for processing.
  - Receive and process the prediction results returned by the server, displaying them on the front-end interface.

4.3.2 Back-End Development

Using PyCharm[1] for Model Training, Data Processing, and API Creation

- **Model Training:**
  - Load the pre-trained ResNet-50 model using PyTorch[2] and fine-tune it for the task of plant disease analysis.
  - Train and evaluate the model using training, validation, and test datasets.
  - Save the trained model as a file for subsequent deployment and invocation.
- **Data Preprocessing:**
  - Save the photos transferred from the front end and perform data preprocessing on the photos, including image resizing, normalization, and data augmentation, to enhance the performance of the model when analyzing pictures.
- **API Creation:**
  - Create a web server using the Flask framework, providing API endpoints for image upload and prediction result retrieval.
  - Handle the uploaded images, preprocess them, and utilize the trained ResNet-50 model for pest identification.

[1] PyCharm
[2] PyTorch

- Return the prediction results to the front-end application.
- Call the Zhipu AI API to search for prediction results and obtain plant disease treatment methods to return to the front-end.

In order to create a functional flowchart of the front-end and back-end development tasks, I will visually outline each step in Figure 4.4 to clearly illustrate the system structure. The flowchart will cover both front-end and back-end activities, highlight the interactions between components, and visually show the workflow from front-end UI design to back-end model training, API creation, and finally display the prediction results in the front-end interface.

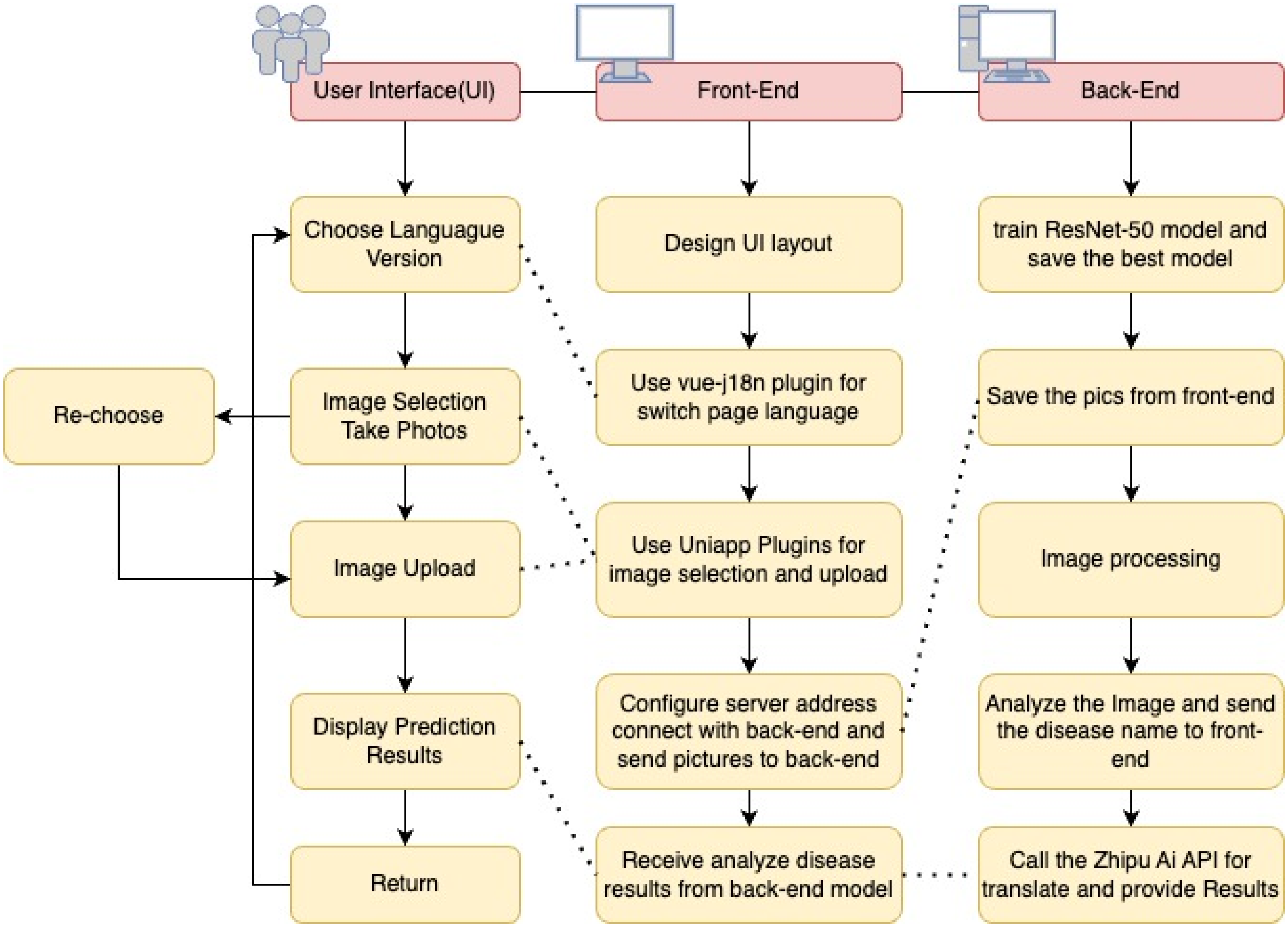


Figure 4.4: The structure of the whole system

5

Results

5.1 Model training results

This project trains and tests the ResNet-50 model using the PyTorch framework. The steps are as follows:

- **Data Import:** Use the `get_files` function to load the training, validation, and test datasets.
- **Data Preparation:** Use `DataLoader` to batch process the data.
- **Model and Optimizer Initialization:** Get the model through `get_net()`, use the Adam optimizer and cross-entropy loss function.
- **Training Process:**
  - Train and validate for each epoch.
  - Record the training and validation loss and accuracy in the training log as shown in section 5.1.1.
- **Model Fine-tuning:** Use the `StepLR` scheduler to adjust the learning rate, improve model performance, and save the best model based on validation results.
- **Testing Process:** Use the best model to predict the test set and record the analysis results as shown in section 5.1.2.

5.1.1 Training Log

The training log recorded the training process of the deep learning model. Each epoch recorded the loss values and accuracy (top-1 accuracy indicates the accuracy of predicting the correct category, and top-2 accuracy indicates the accuracy of the correct category being in the top two prediction results) on the validation set and training set. Detailed data can be seen in Table 5.1 and Table 5.2. The overall analysis is as follows:

- **Initial Stage:** In the first epoch, the validation set loss was 0.607, top-1 accuracy was 80.456%, and top-2 accuracy was 96.286%. The training set loss was 1.018, top-1 accuracy was 68.025%, and top-2 accuracy was 86.305%.

  This indicates that in the initial state, the model performed better on the validation set than on the training set, possibly because the training set was more complex or contained more noise.

- **Middle Stage:** By the 10th epoch, the validation set loss had dropped to 0.390, top-1 accuracy had increased to 86.100%, and top-2 accuracy was 98.548%. The training set loss also significantly decreased to 0.455, top-1 accuracy was 82.809%, and top-2 accuracy was 97.015%.

  This indicates that the model is continuously being optimized, gradually improving its accuracy on both the training set and the validation set.

- **Later Stage:** From the 21th epoch to the 39th epoch, the learning rate was set to 0.0000, likely to stabilize the model performance in the later stages of training and prevent overfitting. In the 39th epoch, the validation set loss was 0.406, top-1 accuracy was 86.413%, and top-2 accuracy was 98.437%. The training set loss was 0.291, top-1 accuracy was 88.600%, and top-2 accuracy was 99.500%.

Conclusion: The model performance continuously improved, as seen from the loss values and accuracy on both the validation set and training set, indicating that the model is gradually converging. The best top-1 accuracy on the validation set reached 86.578%, and the best top-2 accuracy was 98.584%. The best top-1 accuracy on the training set reached 88.600%, and the best top-2 accuracy was 99.500%. Overall, this indicates that the model training was effective, with the model's performance continuously improving and remaining stable and accurate by the end of the training.

Table 5.1: Validation Results

epoch	Loss	Top-1	Top-2
0.0	0.607	80.456	96.286
1.0	0.650	80.566	97.095
2.0	0.567	81.394	97.555
3.0	0.509	82.497	97.518
4.0	0.468	84.041	97.941
5.0	0.524	83.343	98.014
6.0	0.579	82.975	97.610
7.0	0.464	83.618	98.069
8.0	0.611	82.111	97.573
9.0	0.566	81.375	97.683
10.0	0.390	86.100	98.548
11.0	0.389	86.064	98.548
12.0	0.398	86.321	98.584
13.0	0.390	86.431	98.548
14.0	0.382	86.119	98.548
15.0	0.385	86.045	98.658
16.0	0.389	86.266	98.621
17.0	0.395	86.413	98.529
18.0	0.413	85.825	98.529
19.0	0.397	86.100	98.492
20.0	0.383	86.431	98.603
21.0	0.387	86.450	98.529
22.0	0.395	86.486	98.548
23.0	0.400	86.284	98.364
24.0	0.399	86.321	98.511
25.0	0.404	86.431	98.419
26.0	0.400	86.468	98.437
27.0	0.404	86.505	98.364
28.0	0.400	86.395	98.345
29.0	0.390	86.578	98.584
30.0	0.404	86.192	98.566
31.0	0.399	86.542	98.382
32.0	0.397	86.468	98.437
33.0	0.401	86.211	98.492
34.0	0.412	86.284	98.548
35.0	0.392	86.395	98.548
36.0	0.399	86.376	98.345
37.0	0.393	86.597	98.492
38.0	0.398	86.431	98.492
39.0	0.406	86.413	98.437

Table 5.2: Training Results

epoch	Loss	Top-1	Top-2
0.0	1.018	68.025	86.305
1.0	0.776	73.888	91.318
2.0	0.673	76.528	93.306
3.0	0.612	78.179	94.411
4.0	0.571	79.342	95.126
5.0	0.541	80.212	95.649
6.0	0.518	80.889	96.038
7.0	0.500	81.417	96.343
8.0	0.484	81.863	96.601
9.0	0.472	82.248	96.797
10.0	0.455	82.809	97.015
11.0	0.439	83.312	97.212
12.0	0.426	83.764	97.380
13.0	0.414	84.166	97.532
14.0	0.403	84.508	97.663
15.0	0.394	84.809	97.777
16.0	0.385	85.092	97.879
17.0	0.378	85.355	97.972
18.0	0.370	85.588	98.058
19.0	0.364	85.812	98.135
20.0	0.357	86.034	98.208
21.0	0.351	86.244	98.274
22.0	0.346	86.444	98.385
23.0	0.341	86.625	98.493
24.0	0.336	86.796	98.592
25.0	0.331	86.971	98.684
26.0	0.327	87.146	98.771
27.0	0.323	87.301	98.850
28.0	0.320	87.446	98.924
29.0	0.316	87.584	98.994
30.0	0.313	87.712	99.059
31.0	0.310	87.833	99.120
32.0	0.307	87.950	99.179
33.0	0.304	88.059	99.234
34.0	0.302	88.162	99.284
35.0	0.300	88.258	99.333
36.0	0.297	88.352	99.378
37.0	0.295	88.439	99.421
38.0	0.293	88.523	99.462
39.0	0.291	88.600	99.500

5.1.2 Model Test Results

Use the best model to predict the test set and record the precision, recall, and F1 score[1]for each disease label, the details as seen in Table 5.3[2].

- **Precision:** The ratio of correctly predicted positive samples to all predicted positive samples, which measures the accuracy of the prediction.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5.1}$$

- **Recall:** The ratio of correctly predicted positive samples to all actual positive samples, which measures the ability of the model to capture positive samples.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{5.2}$$

- **F1-Score:** The harmonic mean of precision and recall, a single metric that balances the two metrics.

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.3}$$

[1] Precision, Recall and F1-Score in Machining Learning

[2] Note that I deleted Label 45 as mentioned in section 4.1.1

Table 5.3: Performance Metrics of each disease label

Label	Precision	Recall	F1-Score	Label	Precision	Recall	F1-Score
0	0.98	0.96	0.97	30	0.99	0.98	0.99
1	0.87	0.75	0.81	31	0.75	0.78	0.76
2	0.80	0.77	0.78	32	0.81	0.86	0.84
3	0.96	0.97	0.97	33	0.99	1.00	0.99
4	0.75	1.00	0.86	34	0.85	0.94	0.89
5	1.00	0.29	0.44	35	0.97	0.90	0.93
6	0.99	0.99	0.99	36	0.77	0.93	0.84
7	0.90	0.95	0.92	37	0.93	0.82	0.87
8	0.94	0.84	0.89	38	1.00	0.98	0.99
9	0.98	1.00	0.99	39	0.73	0.82	0.77
10	0.70	0.58	0.63	40	0.94	0.89	0.91
11	0.69	0.62	0.65	41	1.00	0.99	0.99
12	0.85	0.89	0.87	42	0.59	0.75	0.66
13	0.78	0.84	0.81	43	0.91	0.83	0.87
14	0.60	0.81	0.69	44	0.84	0.88	0.86
15	0.78	0.79	0.78	46	0.62	0.80	0.70
16	1.00	0.99	1.00	47	0.92	0.85	0.88
17	1.00	1.00	1.00	48	0.68	0.93	0.79
18	0.70	0.66	0.68	49	0.83	0.59	0.69
19	0.74	0.76	0.75	50	1.00	1.00	1.00
20	0.74	0.90	0.81	51	0.50	0.50	0.50
21	0.85	0.65	0.74	52	0.84	0.88	0.86
22	0.75	0.30	0.43	53	0.93	0.88	0.91
23	0.94	0.99	0.96	54	0.87	0.85	0.86
24	1.00	1.00	1.00	55	0.78	0.70	0.74
25	0.72	0.81	0.76	56	0.80	0.79	0.80
26	0.78	0.67	0.72	57	0.88	0.89	0.88
27	0.90	0.88	0.89	58	0.90	0.98	0.94
28	0.91	0.93	0.92	59	0.87	0.91	0.89
29	0.94	0.91	0.92	60	0.84	0.87	0.88

From Table 5.1, it is evident that the accuracy and recall for labels 0, 3, 6, 16, 17, 24, 30, 33, 38, 41, and 50 are very high, indicating that the model performs exceptionally well in predicting these categories. However, some labels, such as 5, 22, and 51, have noticeably lower accuracy or recall, indicating poor performance in these categories. For instance, label 5 has high accuracy but low recall, meaning the model's predictions for label 5 are generally correct, but it misses many actual instances of label 5. Labels 10, 11, 18, and 26 show moderate performance, with balanced but not particularly high accuracy and recall.

Overall, the model performs well on most labels, especially excelling in certain specific labels, demonstrating stable performance in handling the majority of the data.

Table 5.4 summarizes the overall performance of the model, focusing on the model's performance in terms of accuracy, macro average, and weighted average. From Table 2, it can be seen that the overall accuracy of the model is 0.87, indicating that 87% of all predictions are correct, demonstrating the model's ability to accurately classify samples in most cases. The macro-average metrics (recall 0.84, precision 0.85, F1 score 0.83) do not consider class imbalance, reflecting the model's average performance across all labels. The weighted average metrics (recall, precision, and F1 score all being 0.87) take into account the number of samples in each category, further confirming the model's overall good performance in the presence of class imbalance. Overall, the model's strong ability to identify plant pests suggests its potential to support system development.

Table 5.4: Overall performance metrics

Metric	Recall	Precision	F1-Score
Accuracy	0.87		
Macro avg	0.84	0.85	0.83
Weighted avg	0.87	0.87	0.87

5.2 Calling Zhipu AI Result

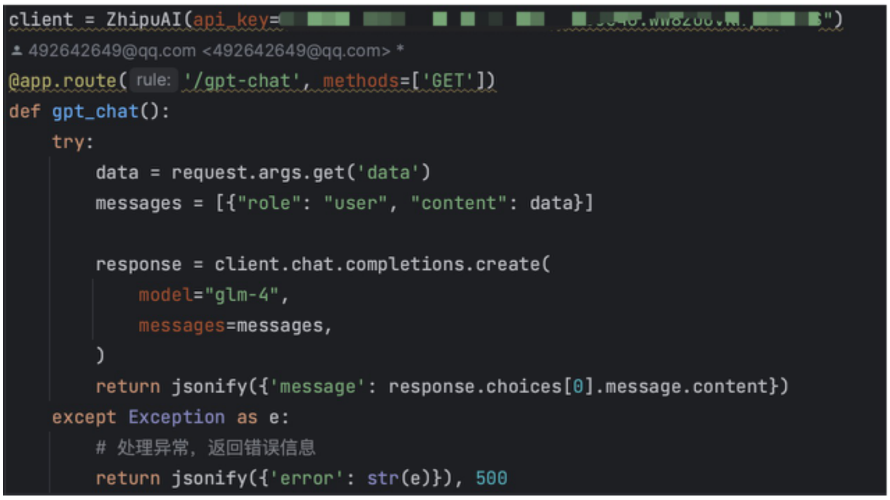
The system uses the ResNet-50 model to analyze images, return the disease name, and obtain detailed information and treatment methods through Zhipu AI, and return the results to the front end.

For Zhipu Ai, the implementation method of calling Zhipu AI is to register an account on the official website[1] and apply for an API key, which is free to use every month for a certain amount. This key is the authentication information required when calling the API.

Then, download the plugin using *'pip3 install –upgrade zhipuai'*, and set the request URL and header information (including the API key), the API will return the response data after sending the HTTP request (GET, POST, etc.).

The main calling code is shown in Figure 5.1 below.

[1] The website to register the Zhipu AI token



```
client = ZhipuAI(api_key=[redacted]")
492642649@qq.com <492642649@qq.com> *
@app.route( rule: '/gpt-chat', methods=['GET'])
def gpt_chat():
    try:
        data = request.args.get('data')
        messages = [{"role": "user", "content": data}]

        response = client.chat.completions.create(
            model="glm-4",
            messages=messages,
        )
        return jsonify({'message': response.choices[0].message.content})
    except Exception as e:
        # 处理异常，返回错误信息
        return jsonify({'error': str(e)}), 500
```

Figure 5.1: The code to call the Zhipu API into the system

5.3 Overall Back-End Result

The image is uploaded through the front-end. After the back-end receives the image, it first calls the model to analyze the image and outputs the disease label and disease name. Then it calls Zhipu AI to analyze the disease name and output the content. The specific test log entry is shown in Figure 5.2, the system log shows several successful interactions between the front-end and the back-end.

```
100%|██████████| 1/1 [00:01<00:00,  1.44s/it]
Predicted Label Index: 48
{'predicted_label': '48', 'predicted_text': 'Tomato Late Blight Water Mold general'}
10.0.0.191 - - [10/Jun/2024 21:00:36] "POST /upload HTTP/1.1" 200 -
10.0.0.191 - - [10/Jun/2024 21:01:26] "GET /gpt-chat?data=Tomato%20Late%20Blight%20Water%20Mold%20generalIntroductory%20version HTTP/1.1" 200 -
100%|██████████| 1/1 [00:01<00:00,  1.37s/it]
Predicted Label Index: 40
{'predicted_label': '40', 'predicted_text': 'Strawberry Scorch serious'}
10.0.0.191 - - [10/Jun/2024 21:19:37] "POST /upload HTTP/1.1" 200 -
10.0.0.191 - - [10/Jun/2024 21:19:55] "GET /gpt-chat?data=Strawberry%20Scorch%20seriousIntroductory%20version HTTP/1.1" 200 -
100%|██████████| 1/1 [00:01<00:00,  1.23s/it]
Predicted Label Index: 10
{'predicted_label': '10', 'predicted_text': 'Cercospora zeae-maydis general'}
10.0.0.191 - - [10/Jun/2024 21:20:42] "POST /upload HTTP/1.1" 200 -
10.0.0.191 - - [10/Jun/2024 21:21:00] "GET /gpt-chat?data=Cercospora%20zeae-maydis%20generalIntroductory%20version HTTP/1.1" 200 -
100%|██████████| 1/1 [00:01<00:00,  1.42s/it]
Predicted Label Index: 11
{'predicted_label': '11', 'predicted_text': 'Cercospora zeae-maydis serious'}
10.0.0.194 - - [10/Jun/2024 21:36:56] "POST /upload HTTP/1.1" 200 -
10.0.0.194 - - [10/Jun/2024 21:37:03] "GET /gpt-chat?data=Cercospora%20zeae-maydis%20seriousIntroductory%20version HTTP/1.1" 200 -
```

Figure 5.2: The response logs to connect the back-end

The detailed information can be analyzed from the figure, including the exact date and time of requests and responses. For instance, log entry 1 records the plant disease name "Tomato Late Blight Water Mold general" identified by the ResNet-50 model, the IP address from which the request was made, the HTTP method used (POST/GET), and the accessed endpoint. Each prediction is followed by an HTTP GET request to retrieve detailed information and treatment options from Zhishu AI. A successful response status (200) indicates seamless integration between components. Four log entries record four different plant pest and disease conditions, with ResNet-50 processing times ranging from 1.23 seconds to 1.44 seconds each time. This demonstrates the system's reliable performance and successful interactions, validating the design and implementation of the system. The above development code and implementation is available on the GitHub repository(22)

5.4 System Design results

5.4.1 Web page operation results

To align with the theme of plant pest analysis, the overall layout of the system application revolves around "green" and "plants." The page design is simple and clear, providing a refreshing experience for users. In terms of functionality, the main page features an upload window where users can upload or take photos. Additionally, the CN (EN) button in the top right corner allows for language switching between Chinese and English. The upload button "submit" is based in blue, making it clear and noticeable.

As shown in Figure 5.3, this is the front-end main page effect, running in the built-in browser of HbuilderX. The built-in browser allows developers to preview and debug the page in real-time during the coding process, ensuring the accuracy of the page design and functionality. This is also the effect of web page operation, which allows some users who have computers at home or want to use the website directly to query to use the system.

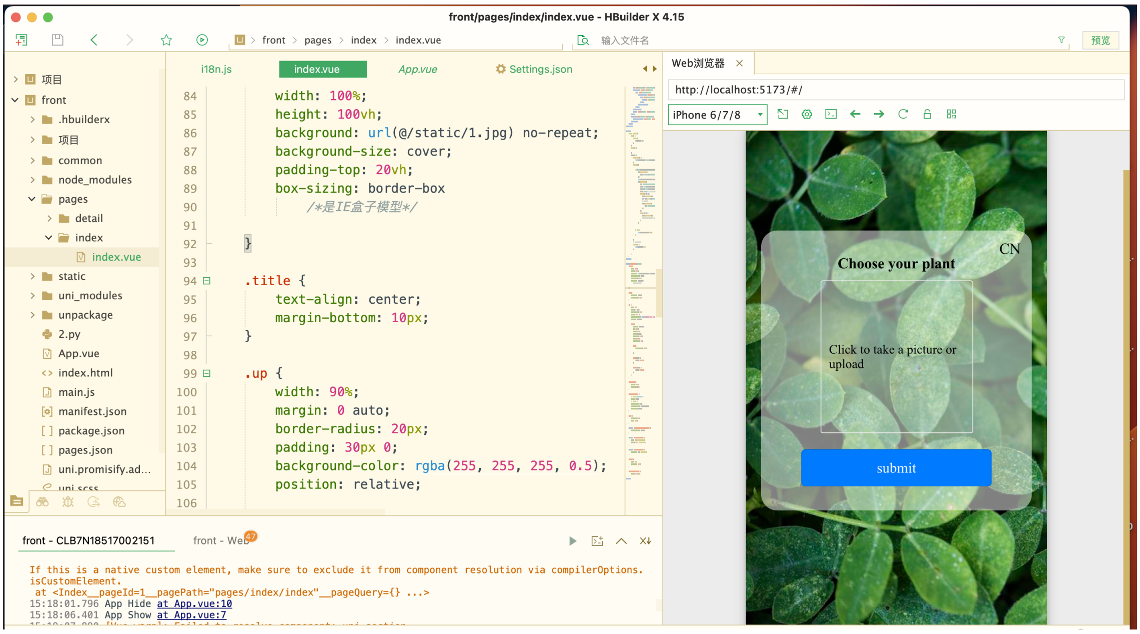


Figure 5.3: The front-end main page effect of using HbuilderX running in its built-in browser

The subpage is the results page, displaying the outcomes after uploading the image for analysis. The overall design style is consistent with the main page. The specific functional layout includes the uploaded image, the disease name identified by the back-end model, and the description and solution provided by Zhipu AI.

As shown in Figure 5.4, this is the subpage effect after calling the backend in the built-in browser, displaying an image of a corn leaf infected with Cercospora zeae-maydis[1] and the system displays the disease information and main treatment methods provided by Zhupu AI.

[1] The scientific name of the fungus that causes Gray Leaf Spot

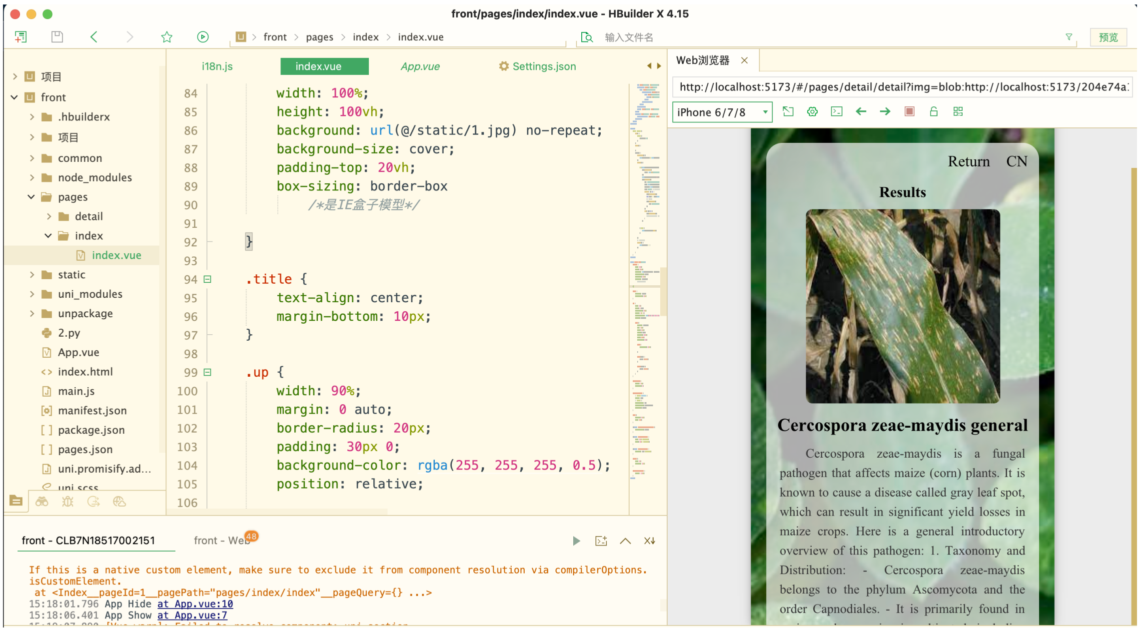


Figure 5.4: The front-end subpage effect of using HbuilderX running in its built-in browser

5.4.2 Application operation results

To make this application accessible to farmers in remote areas, especially in China, I provide multiple ways to access it. They can use it by downloading the mobile app or directly accessing and using the app through WeChat(23). This allows farmers to easily access and utilize the features of the app even when they do not have a high-speed internet connection. Figures 6.3 and 6.4 show use case images of the app being tested on an Android phone. It should be noted here that after the project is completed, it is not deployed on the server, so when testing on an Android phone, it is necessary to connect to the local development environment (connect to the computer via a data cable).

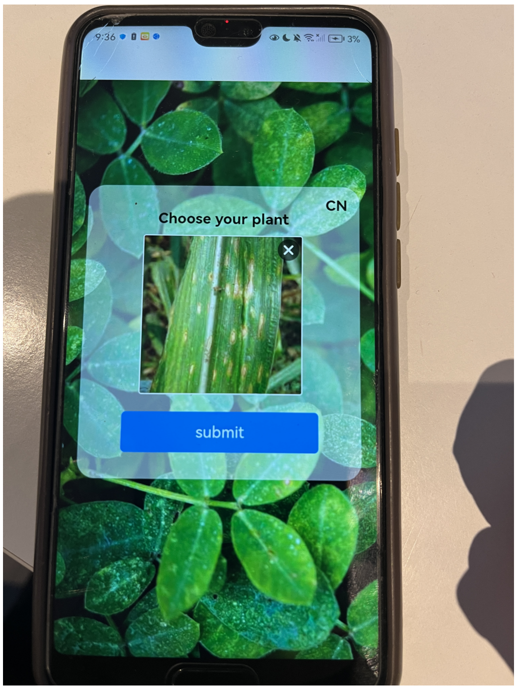


Figure 5.5: The main page of the system running in an Android Phone

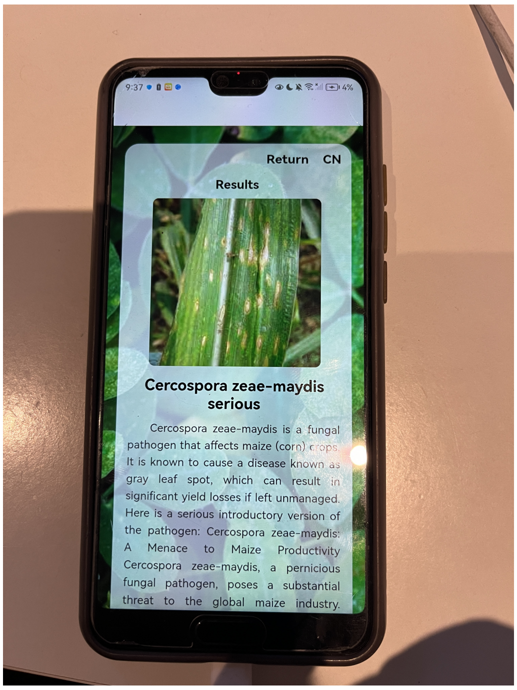


Figure 5.6: The subpage of the system running in an Android Phone

5.5 Project Logo

I have named this project "Agricare," and I have trained an AI to design a logo that aligns with this project, as shown in Figure 5.7.

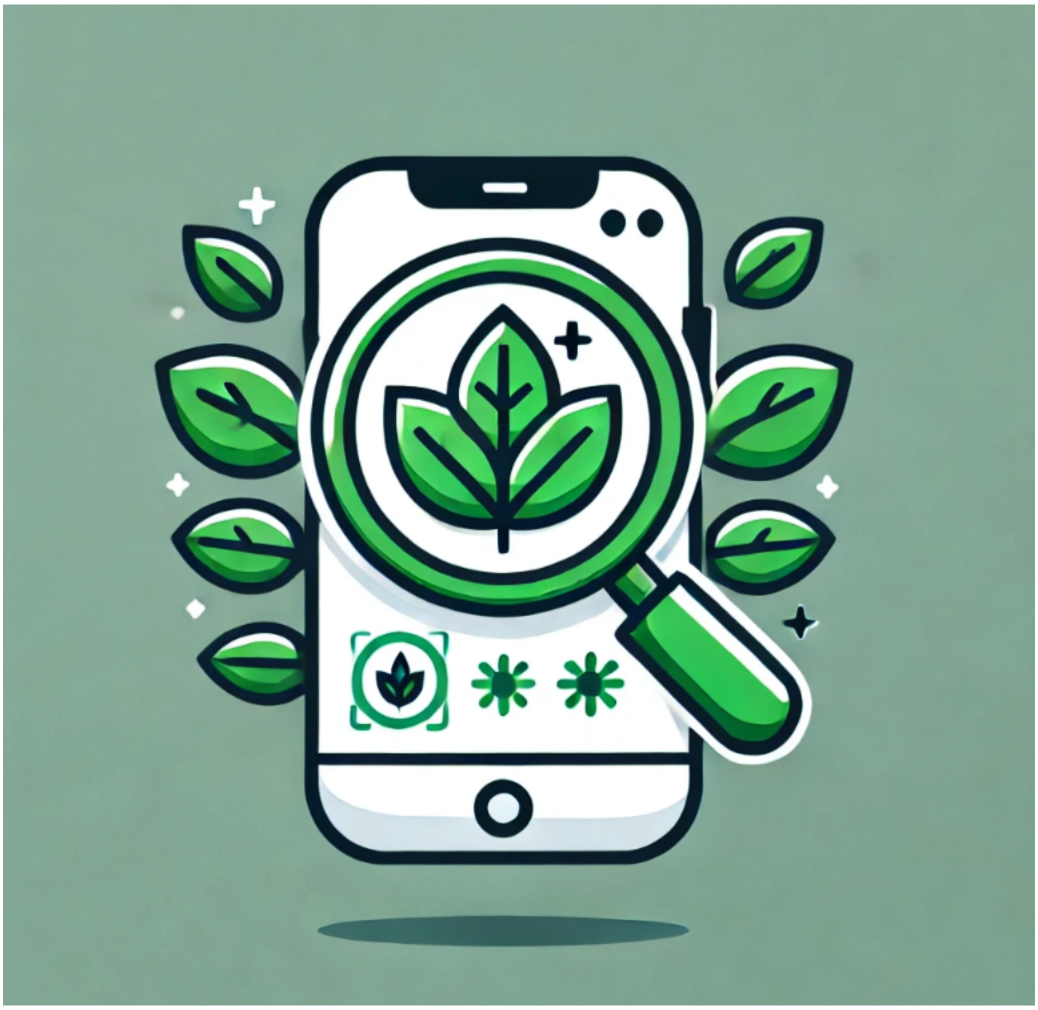


Figure 5.7: Project Logo Designed Using Chatgpt-4o

5.6 System Test results

To test the system's performance, I designed a series of test cases to verify its availability and stability.

5.6.1 Functionality Testing

- Upload 10 images[1] of each known plant disease, record each prediction result, detailed information, and treatment suggestions.

- Compare the system's output with actual labels to calculate accuracy.

- Toggle translation on the main page and sub-pages.

5.6.2 Performance Testing

- Record the time taken from uploading each image to returning the result.

5.6.3 Integration Testing

- Check the log records for each test case to ensure accuracy.

- Verify that each step's response status code is 200.

As shown in Table 5.5 is a test case of a tomato leaf spot.

[1] Test cases Dataset

Table 5.5: Test Case: Detection of Tomato Septoria Leaf Spot Fungus Disease

Testing Type	**Test Case**	**Details**
Functionality Testing	Upload 10 images of tomato leaf spot disease	Return data and test results normally, functionality completeness 100%.
	Compare system output with actual labels	The system correctly identifies 8, accuracy is 80%.
	Switch between Chinese and English	The system correctly switched, as shown in Figure 5.7& 5.8.
Performance Testing	Record processing time	The time taken from uploading each image to returning the disease result is an average of 1.5 seconds. Calling AI for disease result takes an average of 16 seconds.
Integration Testing	Verify log records	Check log records for each test case to ensure accuracy, confirm all 10 test logs are accurate.
	Verify response status codes	Ensure that the response status code for each step is 200. All 10 tests have a response status code of 200.

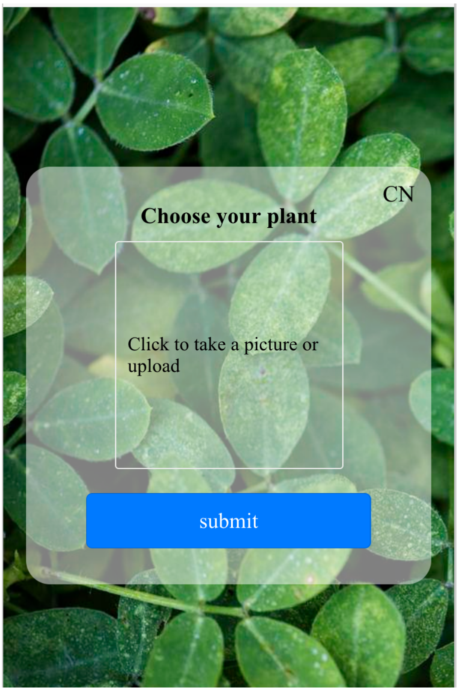


Figure 5.8: The test result of switching system language to English.

Figure 5.9: The test result of switching system language to Chinese.

5.6.4 Exception Handling Testing

- Upload abnormal image files[1] and record the system's error handling results.
- Upload photos that are correctly formatted but unrelated to plants. Record the system's error handling results.
- Upload correctly formatted plant photos that are unrelated to the system's provided range. Record the system's error handling results.
- Simulate network disconnection or timeout and check the system's returned error messages.

As shown in Table 5.6 is a test case of Exception Handling Testing.

Table 5.6: Image Upload and Processing Robustness Test Cases

Test Case	**Details**
Upload an invalid image file	The upload format is incorrect. The system prompts an image error and automatically returns to the main page, as shown in Figure 5.7.
Upload a correctly formatted but non-plant-related photo	Upload a photo of a cat. The system cannot recognize these non-plant-related photos but still analyzes and provides the most reasonable plant disease based on the model, as shown in Figure 5.8.
Upload a correctly formatted plant photo not within the system's provided range	Upload a photo of a Guiana Chestnut[2] that is not included in the system database. The system cannot recognize it but still analyzes and provides the most reasonable plant disease, as shown in Figure 5.9.
Simulate network disconnection or timeout	Intentionally disconnect the network during the upload or processing request. The system fails to reasonably respond to network disconnection or timeout, remaining on the upload page.

[1] In this system, the image file must have no special characters or spaces in its path and name to avoid errors. The file should not be damaged, must open and read normally, and have a .jpg or .jpeg extension.

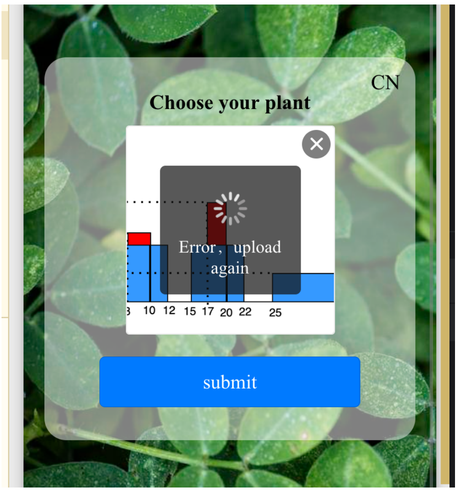


Figure 5.10: The test case result of uploading an invalid image file.

Figure 5.11: The test case result of uploading an invalid image file of a non-plant image.

Figure 5.12: The test case result of uploading a photo of a plant that meets specifications but is outside the system's range.

5.6.5 Test Results

The system demonstrates good testing capabilities in functionality, performance, and integration testing, but needs further optimization in exception handling:

- Functionality Testing Results: The system shows efficient completion and provides the ability to accurately identify various plant diseases, averaging over 80%, and offers translation, detailed information and treatment advice successfully.

- Performance Testing Results: The system returns quickly with high efficiency in disease handling, typically identifying diseases within an average of 1-1.5 seconds, but AI invocation time more than 10s which still needs optimization.

- Integration Testing Results: The system's components are well-coordinated, highly stable, and respond well at each step.

- Exception Handling Testing Results: The system can handle incorrect image uploads; however, its capability to handle other types of exceptions is limited and needs further improvement to enhance system robustness and user experience.

Through the above testing plan, the comprehensive performance of the plant disease analysis system can be fully verified to ensure its reliability and effectiveness in practical applications.

6

Validation

6.1 Self Validation

By analyzing the experimental results, I found that the ResNet-50 model achieved high accuracy and F1 score in classifying plant pest and disease images. This demonstrates significant achievements in improving the accuracy and efficiency of pest and disease recognition. These results support my decision to use ResNet-50 as the image classifier and show that the model can effectively handle the diversity and complexity of plant diseases within the specified plant range, greatly enhancing the system's practicality.

Furthermore, the effectiveness of the Zhipu AI model in returning disease information confirms the success of my study in providing practical technical support. By combining these two efficient models, my system can quickly and accurately detect plant pests and diseases and provide effective descriptions and treatment suggestions, especially benefiting farmers with limited resources.

In terms of user experience, I designed the system's homepage to be clear, simple, and visually appealing to provide a good user experience. The homepage design considers users' intuitive operation needs, ensuring that they can easily find the necessary functions and information. Additionally, I offer both Chinese and English language options to meet the needs of users from different language backgrounds. This design not only increases the system's global applicability but also enhances the convenience for users of different languages, further improving accessibility and user-friendliness.

However, the system also has many areas that need improvement. For example, the model cannot accurately identify the difference between plants and non-plants. When analyzing non-plant photos, the system still identifies the results of plant diseases based on some sensitive information such as the color of the picture. This situation shows that the system needs to be further optimized in image classification and feature extraction to ensure that plants and non-plants can be correctly distinguished in various scenarios, thereby improving the accuracy of diagnosis and the practicality of the system.

In addition, when the system processes a variety of plant disease images, it may be affected by factors such as lighting conditions, shooting angles, and image quality, which may affect the system's diagnostic ability and stability. Therefore, further data enhancement and model training are needed to improve the system's ability to identify and analyze plant diseases in different scenarios.

In summary, although the system performs well in functionality, performance, and integration testing, there is still room for improvement in exception handling, image classification accuracy, and stability. Through continuous optimization and improvement of these aspects, the overall performance and user experience of the system can be further improved to ensure its reliability and effectiveness in practical applications.

6.2 External Validation

In evaluating the system, three individuals with different backgrounds and expertise were interviewed to provide a comprehensive assessment. Their feedback highlights the strengths and areas for improvement of the system.

- Interviewee 1: Ms. Xu, 24 years old, female, student, plant enthusiast

  Ms. Xu finds the application excellent for personal plant growers. As a novice, she struggles with identifying plant pests and typically relies on extensive online searches. She often uses ChatGPT on her computer to seek help. The mobile upload feature of this system is a minor but helpful advantage. The system's ability to focus precisely on plant analysis and provide diverse results beyond a few plant types is particularly beneficial.

- Interviewee 2: Mr. Chen, 60 years old, male, grain farmer Mr. Chen appreciates the system's concept, finding it useful even for experienced farmers when trying to grow new crops. He mentions the challenge older individuals face with electronic devices. Although he learned to use the system today, he might forget how to use it in the future. He suggests providing a user manual or step-by-step prompts to aid older users.

- Interviewee 3: Mr.Mok, 37 years old, male, software engineer, interested in plants and nature Mr.Mok appreciates the system's simple UX design but notes that it lacks the visual appeal of competitors like Plantix[1] and Agrobase[2]. He values the lightweight nature of the application, which makes deployment easy without requiring high-end devices or software. He suggests improving the underlying model accuracy, particularly Resnet-50, as some images are not correctly analyzed. He points out that the response time of Zhipu AI sometimes exceeds 30 seconds, negatively impacting user experience.

Through the feedback from three interviewees, it is evident that the system has significant advantages, including the convenience of mobile uploads and a comprehensive focus on plant disease analysis. These features are highly beneficial for novice plant growers and experienced farmers looking to cultivate new crops. As a lightweight application, it is easier to deploy and use compared to some market applications like ChatGPT.

However, the system still has many areas that need improvement. Currently, the system is limited to analyzing certain specific plants and cannot provide more diversified results. Additionally, elderly users face critical usability issues. Implementing a comprehensive user manual or intuitive usage prompts will greatly enhance the system's accessibility and usability for elderly users, ensuring their continued engagement and ease of use. Furthermore, it is necessary to optimize the response time of Zhipu AI, as long response times negatively impact user experience.

Overall, the system has great potential, and through targeted improvements, user satisfaction and system performance can be further enhanced.

[1] Plantix
[2] Agrobase

7

Discussion

7.1 Challenges

This system, designed to help farmers check plant pests and improve agricultural development, has demonstrated its advantages. However, it faces many challenges in practical application.

- **Accuracy of Plant Analysis:** During model training, the different amounts of data for each label lead to poor analysis capabilities for diseases with less training data. This may result in inconsistent conclusions when encountering new images, posing a significant challenge.

- **Long AI Response Time:** The system's AI response time is currently too long, which affects user experience. Despite current technological limitations, optimizing this performance is crucial to enhancing the overall user experience.

- **Limitations on AI Resource Usage:** The free quota provided by Zhipu AI is limited each month, and exceeding this quota incurs costs. This is a challenge for large-scale or high-frequency usage. Future solutions include finding more cost-effective AI services or optimizing algorithms and models to improve efficiency and reduce the number of calls, thus saving costs.

- **AI Model Maintenance and Updates:** AI models need regular maintenance and updates to ensure their performance and accuracy. Managing and updating models effectively as data and needs change is an ongoing challenge.

- **User Acceptance and Usage Habits of AI:** Users might not trust or be familiar with AI technology, which can affect their acceptance and usage habits. Increasing user acceptance and willingness to use AI technology is also an important challenge.
- **Commercialization and Practical Application Challenges:** If the system is to move from a research prototype to a fully operational application, it will face numerous challenges, including but not limited to:
  - **Server and Infrastructure:** To support a large number of concurrent users, stable and efficient server infrastructure is required, involving both hardware investment and software optimization and maintenance.
  - **Review and Compliance:** The application must pass reviews from major app stores and comply with relevant regulations and policies, including privacy protection and data security.
  - **Media Promotion and Market Strategy:** Attracting users require extensive media promotion and marketing, demanding professional strategies and substantial funding for advertising and social media campaigns.
  - **Funding Needs:** All these steps require adequate funding. From server rentals to marketing, technical support, and maintenance, significant investment is necessary. Thus, securing investment or fundraising is a key step towards commercialization.

In conclusion, while the system shows great potential and advantages, achieving widespread application will require overcoming multiple challenges. These challenges involve not only technical advancements but also effective business strategies and sufficient financial support. Through continuous improvement and innovation, this system can play a greater role in the future and contribute more to agricultural development.

7.2 Future Research

The current system can diagnose and manage plant diseases effectively. However, future research can further enhance its usability and impact by exploring several key areas:

- **User Interface and Experience**: Improve the user interface to make it more intuitive and user-friendly, especially for different user groups like the elderly. This will enhance the system's usability.

- **Expanding the Plant Database**: Expand the plant database to include more species, allowing the system to serve a broader user base and increase its practicality and market competitiveness.
- **Improving Model Accuracy**: Use advanced machine learning and deep learning models to enhance the accuracy of plant disease analysis, improving diagnostic results and increasing user trust.
- **Cross-Platform Compatibility**: Enhance cross-platform compatibility so the system can efficiently run on various devices, such as smartphones, tablets, and computers, thus expanding the user base.
- **Functional Extensibility**: Expand system functionalities, such as adding a forum for users to share planting experiences or developing real-time monitoring and alert systems for plant diseases, allowing for quicker responses.
- **Integration with Other Agricultural Software**: Study how the system can integrate with other agricultural software to provide users with a more comprehensive toolset, increasing overall practicality and user satisfaction.
- **Sustainability and Environmental Impact**: Explore how the system can promote sustainable agricultural practices and minimize environmental impact, benefiting both users and the environment.
- **Impact on Other Software**: The advancements and findings from this system's research can positively influence other agricultural software by setting new standards in accuracy, user experience, and functionality.
- **Economic Impact Studies**: Research the economic impact of the system on agricultural efficiency and productivity to provide valuable insights, demonstrating tangible economic benefits to attract more users and stakeholders.

Exploring these areas will help further optimize the system, meet diverse user needs, and enhance its overall effectiveness and impact in agriculture.

8

Conclusion

This thesis presents the development and testing of a comprehensive plant disease analysis system, utilizing the ResNet-50 model for image classification and Zhipu AI for providing detailed disease information and treatment recommendations. The system aims to offer accurate, efficient, and user-friendly diagnostic services for a certain range of plant diseases.

By combining deep learning with AI-driven information retrieval, the system provides a powerful solution for plant disease identification and management. The integration with Zhipu AI not only enables the provision of comprehensive treatment options but also enhances its practicality. Future work can focus on expanding the dataset and optimizing the ResNet50 model to cover a wider range of plant diseases and improve accuracy. The system's framework can also be extended to integrate other AI-driven tools and services, promoting more comprehensive agricultural diagnostics and decision-making.

In summary, the development of this system provides a valuable tool for agriculture in China and globally, demonstrating the effectiveness of AI and deep learning in practical applications. The successful implementation of the system allows farmers to efficiently and accurately identify plant diseases and receive detailed information and treatment recommendations, providing significant support and assistance to farmers and agricultural experts. Additionally, the findings and insights from this project lay a solid foundation for future technological innovation and improvement, validating the immense potential of AI and deep learning in the agricultural sector and showing that these technologies can play an important role in real-world scenarios. Future research and development can build on this system to further expand and optimize it, advancing intelligent agricultural management and contributing to improved agricultural productivity and plant health management.

References

[1] UN Department of Economic and Social Affairs. **World Population Projected to Reach 9.8 Billion in 2050, and 11.2 Billion in 2100**. United Nations. (2017, June 21). https://www.un.org/en/desa/world-population-projected-reach-98-billion-2050-and-112-billion-2100, Assessed 26 June 2024. 1

[2] Chenzhi Wang, Xuhui Wang, Zhenong Jin, Christoph Müller, Thomas A. M. Pugh, Anping Chen, Tao Wang, Ling Huang, Yuan Zhang, Laurent X. Z. Li, and Shilong Piao. **Occurrence of crop pests and diseases has largely increased in China since 1970**. *Nature Food*, **3**(1):57–65, 2022. 1

[3] Lily Hess. **In China, drastic rise in crop pests and diseases due to climate change**. Think Landscape. (2022, January 7). https://thinklandscape.globallandscapesforum.org/56092/in-china-drastic-rise-in-crop-pests-and-diseases-due-to-climate-change/, Assessed 26 June 2024. 3

[4] Babasaheb Fand and Ankush Kamble. **Will climate change pose serious threat to crop pest management: A critical review?** *International Journal of Scientific and Research Publications*, **2**:1–15, 11 2012. 3

[5] Nicholas Bariyo. **Africa's Worst Locust Plague in Decades Threatens Millions**. *Wall Street Journal.* 3

[6] Komila Nabiyeva. **Agricultural pesticide use: impact and alternatives**. Helmholtz Climate Initiative. (2023, January 10). https://www.helmholtz-klima.de/en/aktuelles/agricultural-pesticide-use-impact-and-alternatives, Assessed 26 June 2024. 4

[7] Yiwen Liu, Xian Zhang, Yanxia Gao, Taiguo Qu, and Yuanquan Shi. **Improved CNN Method for Crop Pest Identification Based on Transfer Learning**. *Computational Intelligence and Neuroscience*, **2022**:1–8, 03 2022. 6

[8] Karen Simonyan and Andrew Zisserman. **Very Deep Convolutional Networks for Large-Scale Image Recognition**, 2015. 6

[9] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. **Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning**, 2016. 6

[10] ZHANG Fangyan, ZHAO Meng, ZHOU Yizhi, XU Qingwen, LI Haiqing, CHENG Siqi, WU Junfeng, and YU Hong. **Detection of Diseased Takifugu Rubripes Based on ResNet50 and Transfer Learning**. *Journal of Dalian Ocean University*, **36**(4):521–528, 2021. College of Information Engineering, Dalian Ocean University, Dalian, Liaoning, China; Key Laboratory of Environment Controlled Aquaculture, Ministry of Education, China, Dalian, Liaoning, China; Key Laboratory of Marine Information Technology of Liaoning Province, Dalian, Liaoning, China. 6

[11] Faye Mohameth, Chen Bingcai, and Kane Amath Sada. **Plant Disease Detection with Deep Learning and Feature Extraction Using Plant Village**. *Journal of Computer and Communications*, **8**(6):1–10, 2020. 7

[12] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. **Going Deeper with Convolutions**, 2014. 7

[13] Zhibin Wang, Xiaoyu Qiao, Yiming Wang, Hua Yu, and Cheng Mu. **IoT-based system of prevention and control for crop diseases and insect pests**. *Frontiers in Plant Science*, **15**:1323074, February 1 2024. 7

[14] Sara Brown. **Machine Learning Explained**. MIT Sloan. (2021, April 21). `https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained`, Assessed 26 June 2024. 7

[15] Aitor Gutierrez, Ander Ansuategi, Loreto Susperregi, Carlos Tubío, Ivan Rankić, and Libor Lenža. **A Benchmarking of Learning Strategies for Pest Detection and Identification on Tomato Plants for Autonomous**

Scouting Robots Using Internal Databases. *Journal of Sensors*, **2019**:1–15, 05 2019. 7

[16] Maria Mignoni, Emiliano Monteiro, Cesar Zagonel, and Rafael Kunst. **ARTIFICIAL INTELLIGENCE AND ITS TOOLS IN PEST CONTROL FOR AGRICULTURAL PRODUCTION: A REVIEWINTELIGENCIA ARTIFICIAL Y SUS HERRAMIENTAS EN EL CONTROL DE PLAGAS PARA LA PRODUCCIÓN AGRÍCOLA: UNA REVISIÓNARTIFICIAL INTELLIGENCE AND ITS TOOLS IN PEST CONTROL FOR AGRICULTURAL PRODUCTION: A REVIEWINTELIGÊNCIA ARTIFICIAL E SUAS FERRAMENTAS NO CONTROLE DE PRAGAS PARA PRODUÇÃO AGRÍCOLA: UMA REVISÃO**. *RECIMA21 - Revista Científica Multidisciplinar - ISSN 2675-6218*, **5**:e555277, 05 2024. 8

[17] Petru Potrimba. **What is ResNet-50?** Roboflow Blog. (2024, March 13). `https://blog.roboflow.com/what-is-resnet-50/`, Assessed 26 June 2024. 14

[18] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff

Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. **GPT-4 Technical Report**, 2024. 16

[19] Vijaya Bhosale, Deepak Raverkar, Mr Kadam, and Asst Prof. **Google Assistant**. 04 2021. 16

[20] Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. **Glm-130b: An open bilingual pre-trained model**. *arXiv preprint arXiv:2210.02414*, 2022. 16

[21] IvyPanda. **Uniapp Applications: Market and Data Analysis**. IvyPanda. (2022, August 26). https://ivypanda.com/essays/uniapp-applications-market-and-data-analysis/, Assessed 26 June 2024. 16

[22] Kim. **Master Thesis Repository**. GitHub. (2024, June 26). = https://github.com/kim1018/master-thesis/tree/master, note = Accessed: 2024-06-26. 27

[23] Christian Montag, Benjamin Becker, and Chunmei Gan. **The Multipurpose Application WeChat: A Review on Recent Research**. *Frontiers in Psychology*, **9**, 12 2018. 29