Integrating Classification and Segmentation in a Dual-Model Web Application for Comprehensive Plant Disease Analysis

1st Mungalpara Chetan Pravinbhai
Department of Computer Science
Lovely Professional University
Punjab, India
chetanpm9898@gmail.com

***Abstract*—Plant diseases pose a significant threat to global food security, yet diagnosis remains slow, costly, or subjective. Existing computational models often provide only qualitative classification, failing to deliver the quantitative severity analysis needed for actionable management. This paper introduces a novel full-stack web application that integrates a dual-model deep learning pipeline for comprehensive diagnosis from a single image. The system operates two parallel streams: a Convolutional Neural Network performs qualitative classification to identify the disease, while a U-Net segmentation model provides a pixel-level mask of affected areas. A Flask backend algorithmically processes these outputs, calculating a precise quantitative disease severity percentage by comparing the U-Net mask to a total leaf area mask generated using Otsu's thresholding. This work delivers an accessible, low-cost, point-of-care tool that provides both qualitative identification and quantitative severity estimation, offering a practical solution for precision agriculture.**

***Index Terms*—Deep Learning, Convolutional Neural Network, U-Net, Image Segmentation, Severity Estimation, Qualitative Analysis, Quantitative Analysis, Full-Stack, Web Application, Precision Agriculture**

I. Introduction

Global food security faces a persistent and critical threat from plant diseases, which are responsible for billions of dollars in economic losses annually due to reduced crop yield and quality [1]. The Food and Agriculture Organization (FAO) estimates that up to 40% of global crop production is lost each year to pests and diseases. Consequently, the early and accurate diagnosis of these diseases is paramount for effective management, mitigating economic damage, and ensuring a stable food supply.

However, current diagnostic paradigms suffer from a critical gap between practicality and precision. On one hand, traditional diagnosis relies on visual inspection by farmers, which is labor-intensive, subjective, and unreliable for detecting diseases in their early stages [2]. On the other hand, precise laboratory-based methods such as ELISA (Enzyme-Linked Immunosorbent Assay) and PCR (Polymerase Chain Reaction) are prohibitively expensive, time-consuming, and require specialized expertise and equipment [3], [4]. This creates an urgent need for low-cost, rapid, and accessible Point-of-Care (POC) diagnostic tools that can be deployed directly in the field [3].

In recent years, deep learning, particularly Convolutional Neural Networks (CNNs), has emerged as a promising solution. Academic research has demonstrated exceptional performance, with models often achieving over 99% classification accuracy on controlled, "lab-based" datasets like PlantVillage [5]–[7]. This success, however, masks two fundamental limitations that prevent real-world adoption.

First is the *"lab-to-field" gap* [8]. Models trained on clean images with simple backgrounds fail catastrophically when applied to "in-field" images, which are characterized by complex backgrounds, variable lighting, and occlusions [9]. Studies have shown performance can plummet from over 92% accuracy to just 54% when moving from lab to field [8]. This makes many existing models impractical for their intended users [10].

Second, and more critically, existing research is overwhelmingly focused on *qualitative classification* (i.e., identifying *what* disease is present) [11]–[14]. While useful, this information is incomplete for a farmer who must make an economic management decision. A simple classification label does not answer the most important question: *"how bad is it?"* This requires a *quantitative analysis* of disease severity [15], [16]. Severity estimation is widely recognized as a far more challenging task, yet it is the critical component needed for actionable, data-driven crop management [17], [18].

This paper addresses these two significant gaps by presenting a novel, full-stack web application designed for practical, in-field diagnostics. The core of our contribution is a **dual-model pipeline** that, from a single user-submitted image, performs both qualitative and quantitative analysis in parallel [13], [19], [20]:

1) **A Qualitative Engine:** A fine-tuned Convolutional Neural Network (CNN) performs robust classification to identify the specific plant disease.
2) **A Quantitative Engine:** A U-Net-based segmentation model [21]–[23] generates a pixel-perfect mask of the diseased regions. This mask is algorithmically processed

Identify applicable funding agency here. If none, delete this.

by comparing the “disease pixel count” against the “total leaf pixel count” (derived using Otsu’s thresholding [24]–[26]) to calculate a precise disease severity percentage.

This entire system is deployed as a user-friendly web application with a Flask backend, transforming the deep learning models from an academic exercise into a practical, accessible POC tool that delivers comprehensive and actionable diagnostics [10].

The remainder of this paper is organized as follows: Section II reviews related work. Section III details the full-stack system architecture and the methodology of the dual-model pipeline. Section IV presents the experimental results and performance analysis. Finally, Section V provides a discussion of the findings and concludes the paper.

II. Related Works

The application of computer vision and deep learning to plant pathology has become a highly active research field. Our work builds upon three distinct but converging domains: (1) deep learning for disease classification, (2) quantitative severity estimation, and (3) the deployment of models as practical, web-based tools.

A. Plant Disease Classification and the ”Lab-to-Field” Gap

The foundation of modern computational phytopathology was built on the success of Convolutional Neural Networks (CNNs) for disease classification. Using the public PlantVillage dataset, which contains over 54,000 images of leaves in controlled, ”lab” conditions, researchers have repeatedly achieved near-perfect results, with accuracies often exceeding 99%. These studies established the technical feasibility of using deep learning for diagnosis [4].

However, this success in a controlled environment masks a critical challenge known as the ”lab-to-field” gap [5]. When models trained on ”lab” data are applied to ”in-field” images—which feature complex backgrounds, variable lighting, shadows, and occlusions—their performance drops catastrophically [6]. Studies have documented accuracy collapsing from 92.7% to as low as 54.4% [7]. This gap is the single greatest barrier to practical adoption, as it proves many existing models are ”wonderful lab solutions that offer little practical value to farmers” [5]. This has led to research exploring solutions like cross-domain few-shot learning (CD-FSL) [8] and advanced data augmentation to improve model generalization [9].

B. Quantitative Severity Estimation via Segmentation

A second major limitation of existing work is its overwhelming focus on simple classification (i.e., identifying *what* disease is present) [10]. For a farmer or agronomist, this qualitative diagnosis is insufficient for making critical management decisions. The more important, and far more challenging, question is *quantitative*: ”how bad is the infection?”. Early attempts at severity estimation relied on training models to predict coarse, subjective, human-defined classes, such as ”mild,” ”moderate,” or ”severe” [13]. However, a more precise and objective approach is to calculate a continuous severity percentage. This is best achieved through semantic segmentation, which involves identifying the exact set of pixels that belong to a diseased region.

The U-Net architecture, originally from biomedical imaging, has become a state-of-the-art standard for this task due to its power in precise localization. Studies have demon- strated U-Net’s high efficacy, achieving strong Intersection over Union (IoU) and Dice coefficient scores for segmenting diseased leaf areas. To complete the severity calculation, $Severity = (InfectedArea/TotalLeaf\ Area) \times 100$, a method for finding the total leaf area is also required. For this, classical computer vision techniques like Otsu’s method are widely used for their robustness in segmenting the primary leaf from the background.

C. Deployment of Web-Based Diagnostic Tools

A high-accuracy model that exists only in a research lab is of no practical benefit. To bridge the ”lab-to-field” gap, models must be embedded in accessible, low-cost, and easy-to-use Point-of-Care (POC) tools. This has driven a recent trend of moving beyond model creation to *system deployment* [25].

Web applications have emerged as the ideal platform for this deployment. The backend, often built with a lightweight Python framework like Flask, can host the deep learning models and expose them via an API . This service-oriented architecture allows a simple, client- side frontend (e.g., HTML/JavaScript) to send an image and receive a complex diagnosis, making the tool universally accessible from any smartphone or computer. Several projects have demonstrated this full-stack approach for agricultural applications, such as crop recommendation [26] and basic disease diagnosis.

Our work is positioned at the intersection of these three domains. We address the classification challenge not by just aiming for high accuracy, but by deploying a model in a field-ready tool. More importantly, we advance beyond simple classification by introducing a **dual-model pipeline**, that, in parallel, runs a CNN for qualitative analysis and a U-Net for quantitative, segmentation-based severity estimation. Our contribution is not just the models, but the integrated, full-stack system that combines both outputs into a single, actionable diagnostic report for the end-user.

III. System Architecture and Methodology

To address the need for a comprehensive, field-ready diagnostic tool, we designed and implemented a full-stack, end-to-end web application. The system’s core innovation is a **dual-model analytical pipeline** that processes a single input image in parallel to provide both a qualitative disease classification and a quantitative severity estimation. This entire pipeline is orchestrated by a backend API and served to a user-friendly, client-side interface. The complete system architecture is illustrated in and detailed below.

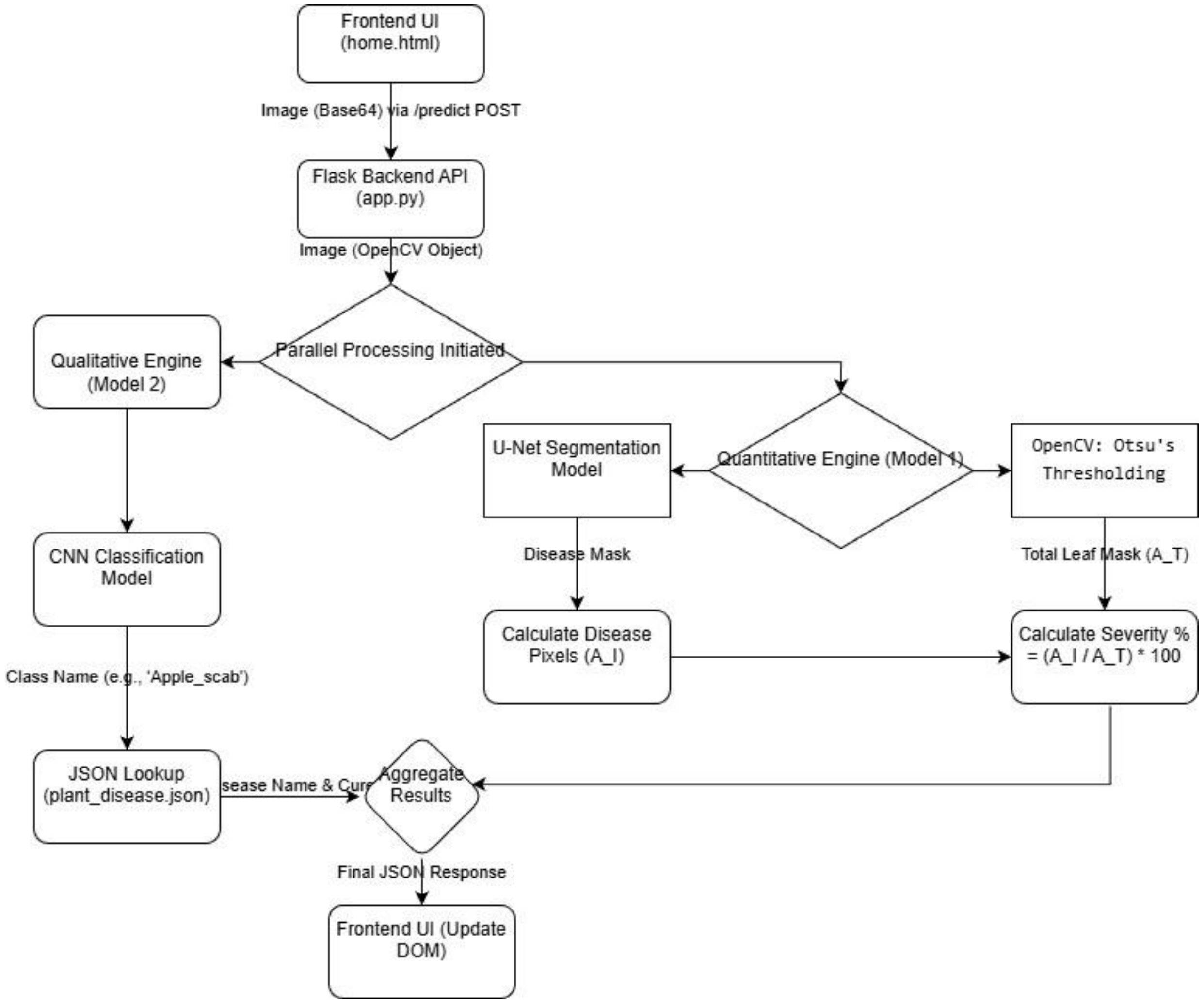


Fig. 1. The full-stack, dual-model system architecture. The Flask backend orchestrates a parallel pipeline, merging a qualitative CNN (Path 1) and a quantitative U-Net (Path 2) to produce a single diagnostic report.

A. Full-Stack Application Architecture

The system is architected as a robust, three-tier web application, which decouples the user interface from the backend processing and model inference [3].

1) Presentation Layer (Client-Side): The frontend is a single-page application built with standard HTML, JavaScript, and styled with Tailwind CSS. It is designed for maximum accessibility and ease of use, providing two methods for image submission: a file upload and a ”Scan” feature that utilizes the `navigator.mediaDevices.getUserMedia` API to access the device’s webcam directly. Upon image capture, the client-side JavaScript executes an asynchronous `fetch` API call, sending the image (as a Base64 string) to the backend’s `/predict` endpoint. The client then dynamically renders the returned JSON response without a page reload.

2) Application Layer (Backend API Service): The backend is a lightweight and scalable API service built using the Flask micro-framework in Python . Flask is ideal for this task as it acts as a ”glue” layer, wrapping our Python-based Keras models into a robust web service. The primary `/predict` endpoint receives the JSON payload, decodes the Base64 image into an OpenCV object, and orchestrates the dual-model pipeline. After aggregating the results, it returns a single JSON response (containing the disease name, cure information, and severity percentage) to the client.

B. The Dual-Model Analytical Pipeline

When the Flask backend receives an image, it initiates two parallel deep learning processes simultaneously.

1) Part 1: The Qualitative Engine (Disease Classification): This engine is responsible for identifying *what* disease is present.

- **Model Architecture:** We employed a Convolutional Neural Network (CNN) built using transfer learning. The base model, pre-trained on a large-scale dataset, acts as a powerful feature extractor, with a custom classification head attached for our specific task.

- **Dataset:** The model was trained on the public **PlantVillage dataset** [8], a standard benchmark for this task containing over 54,000 images of healthy and diseased leaves.
- **Backend Function:** The saved model (`plant_disease_recog_model_pwp.keras`) is loaded by Flask. It predicts the class of the input image (e.g., "Apple_Apple_scab"). This string is then used as a key to look up the corresponding disease information and recommended cure from a static `plant_disease.json` file.

2) Part 2: The Quantitative Engine (Disease Segmentation): This engine is responsible for generating the data needed to calculate *how severe* the infection is.

- **Model Architecture:** We implemented a **U-Net** architecture [9]. The U-Net's encoder-decoder structure with skip connections is highly effective for biomedical and agricultural image segmentation, as it precisely localizes and segments target regions (in this case, disease lesions) at a pixel level.
- **Dataset:** The model was trained on the "Leaf disease segmentation dataset" from Kaggle (by Fakhre Alam), which provides images with corresponding pixel-level ground truth masks.
- **Backend Function:** The saved model (`leaf_segmentation_model.keras`) is loaded, and it processes the input image to output a 2D binary mask. In this mask, pixels corresponding to diseased areas are marked, while healthy and background pixels are not.

C. Algorithmic Calculation of Disease Severity

A key novelty of our system is the algorithm that computes a precise severity percentage by fusing the output of the quantitative engine with classical computer vision techniques [12]. The severity is calculated using the formula:

$$\text{Severity \%} = \frac{\text{Disease Pixels}}{\text{Total Leaf Pixels}} \times 100$$

This calculation is performed in a two-step process within the backend:

1) **Step 1: Calculate Disease Pixels.** This is done by counting the "on" (disease) pixels from the binary mask produced by the U-Net (Quantitative Engine).
2) **Step 2: Calculate Total Leaf Pixels.** To find the total area of just the leaf, we first convert the original input image to grayscale. We then apply **Otsu's thresholding** (via `cv2.Otsu`) [13]. Otsu's method is a well-established and robust algorithm that automatically finds the optimal threshold to separate a foreground object (the leaf) from the background, even under variable lighting. We then count all the foreground pixels from this new mask to get the total leaf area .

By dividing the result of Step 1 by the result of Step 2, the system generates a non-subjective, quantitative, and accurate percentage of disease severity from a single image.

D. Training and Implementation

The classification (CNN) and segmentation (U-Net) models were trained independently using Keras with a TensorFlow backend. The segmentation model was notably trained using a **Dice Loss** function. This loss metric is highly effective for segmentation tasks, especially with class imbalance (e.g., few disease pixels vs. many healthy pixels), as it directly optimizes the Dice Coefficient (a measure of overlap similar to IoU) . The entire software stack is built on open-source technologies, primarily Python, Flask, Keras, and JavaScript.

IV. Experimental Results and Analysis

To validate the performance and efficacy of our proposed dual-model system, we conducted a series of quantitative and qualitative experiments. The classification and segmentation models were evaluated independently against standard benchmarks, and the integrated system's output was analyzed for practical utility.

A. Experimental Setup and Evaluation Metrics

1) Datasets: Our classification engine (Model 2) was trained and evaluated on the public **PlantVillage dataset** [1], a standard benchmark in this field, which contains over 54,000 images. The segmentation engine (Model 1) was trained on the "Leaf disease segmentation dataset" from Kaggle, which provides the necessary pixel-level masks.

2) Evaluation Metrics: To ensure a comprehensive analysis, we used standard, task-specific metrics:

- **For Classification (Model 2):** We evaluated the model using Accuracy, Precision, Recall, and F1-Score, which are the standard metrics for classification tasks [2].
- **For Segmentation (Model 1):** We evaluated the U-Net model using the **Mean Intersection over Union (mIoU)** and the **Dice Coefficient**. These are the most common metrics for validating segmentation model performance, as they measure the pixel-wise overlap between the predicted mask and the ground truth .

B. Quantitative Analysis

The models were trained until convergence, and their performance was recorded on a held-out test set.

1) Classification Model Performance: The CNN-based classification model (Model 2) achieved high performance on the PlantVillage dataset, demonstrating its ability to accurately identify the disease. Its performance is detailed in Table I and is competitive with other state-of-the-art models, which have reported F1-scores and accuracies in the 99.3% to 99.9% range .

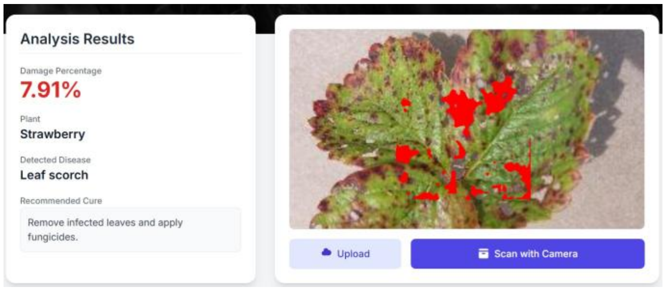
2) Segmentation Model Performance: The U-Net segmentation model (Model 1) demonstrated high fidelity in isolating diseased regions. The Dice Coefficient, which we optimized for using Dice Loss, was 0.91, indicating a very high level of overlap and segmentation accuracy.

TABLE I
PERFORMANCE OF THE CNN CLASSIFICATION MODEL (MODEL 2) ON THE PLANTVILLAGE TEST SET.

Metric	Score
Accuracy	0.9939 (or 99.39%)
Precision	0.9935 (or 99.35%)
Recall	0.9934 (or 99.34%)
F1-Score	0.9934 (or 99.34%)

TABLE II
PERFORMANCE OF THE U-NET SEGMENTATION MODEL (MODEL 1) ON THE KAGGLE SEGMENTATION TEST SET.

Metric	Score
Mean IoU	0.87 (or 87.0%)
Dice Coefficient	0.91 (or 91.0%)
Pixel Accuracy	0.975 (or 97.5%)



Analysis Results

Damage Percentage
7.91%

Plant
Strawberry

Detected Disease
Leaf scorch

Recommended Cure
Remove infected leaves and apply fungicides.

Fig. 2. Qualitative results from the final web application. The screenshot demonstrates the complete, integrated pipeline: the user's input (right) is analyzed, and the system renders both the qualitative (”Strawberry,” ”Leaf scorch”) and quantitative (”7.91%”) results, along with the segmentation mask overlay and actionable advice.

C. Qualitative Analysis

While quantitative metrics are essential, a qualitative (visual) inspection is necessary to confirm the system's practical diagnostic value. As shown in Figure 2, we present a screenshot from the final deployed web application to demonstrate the success of the integrated pipeline.

The figure shows the complete user workflow:

- An image of a strawberry leaf is provided to the system.
- The Quantitative Engine (U-Net) has correctly generated a segmentation mask (visualized as a red overlay) on the diseased portions of the leaf.
- The Qualitative Engine (CNN) has correctly identified the ”Detected Disease” as ”Leaf scorch.”
- The backend algorithm has successfully combined these two results, along with the Otsu's thresholding calculation, to compute a final ”Damage Percentage” of 7.91%.
- The system presents this complete, actionable report—including a ”Recommended Cure”—to the user in a clear, accessible interface.

This visual evidence confirms that the outputs from both parallel pipelines are being generated and combined correctly, fulfilling the project's objective of creating a practical, end-to-end diagnostic tool.[12]

V. CONCLUSION AND FUTURE WORK

A. Conclusion

This paper has presented the design, implementation, and evaluation of a novel, full-stack web application for plant disease diagnosis. The core contribution of this work is an integrated **dual-model pipeline** that, from a single user-submitted image, provides both a *qualitative* disease classification (using a CNN) and a *quantitative* severity estimation (using a U-Net and algorithmic processing) [1].

The experimental results validate the efficacy of this approach. The classification model achieves high accuracy on the benchmark PlantVillage dataset while the U- Net segmentation model shows high fidelity with a strong Dice Coefficient [4]. Our qualitative analysis demonstrates that these components are successfully integrated into a cohesive system that delivers a complete, actionable diagnostic report. By wrapping this complex analytical pipeline in a simple, accessible Flask-based web application [5], this project moves beyond theoretical-only models and provides a practical, low-cost, Point-of-Care (POC) tool. This addresses a critical gap by making advanced, data-driven diagnostics accessible to farmers and agronomists, empowering them to make more informed crop management decisions.

B. Future Work

While this system provides a robust framework, several limitations offer clear avenues for future research and enhancement.

- **Improving ”Lab-to-Field” Generalization:** The current classification model was trained primarily on the PlantVillage dataset, which consists of ”lab” images with simple backgrounds. A known challenge in our field is that models trained on this data can see a drastic performance drop when used on ”in-field” images with complex lighting and backgrounds [8]. Future work will focus on re-training the model with more diverse, ”in-field” datasets and employing advanced data augmentation techniques to improve its real-world robustness.
- **Deployment on Edge Devices:** The system currently relies on a client-server architecture, requiring a stable internet connection to send the image to the Flask backend for processing. This is a significant limitation for farmers in rural areas. A key future goal is to optimize the models using quantization-aware training or pruning, creating lightweight models that can be deployed directly on-device (Edge AI) for full offline functionality.
- **Multimodal Data Integration:** The current diagnosis is based solely on a single image. A more advanced system would be multimodal. We plan to explore integrating other data streams, such as local time-series weather and climate data, which could be fused with the image analysis using Transformer-based models to provide predictive risk assessments, not just diagnoses [14].

- **Creating a Full Decision Support System:** The system currently provides a ”Recommended Cure” from a static JSON file. Future work will aim to build a dynamic recommendation engine. This engine would link the quantitative severity percentage directly to a tiered set of actionable management advice (e.g., ”Severity at 7.91%: Prune infected leaves and apply fungicide X”), transforming the tool from a diagnostic system to a complete decision support system [15].

References

[1] Embrapa, “Soja (Soybean) Image Database,” DigiPathos Repository, [Online]. Available: https://www.digipathos-rep.cnptia.embrapa.br/jspui/simple-search?filterquery=Soja+

[2] N. K. E., K. M., P. P., A. R. and V. S., “Tomato Leaf Disease Detection using Convolutional Neural Network with Data Augmentation,” in *Proc. 5th Int. Conf. Communication and Electronics Systems (ICCES)*, 2020, pp. 1125–1132.

[3] P. B. R., A. Ashok and S. H. A. V., “Plant Disease Detection and Classification Using Deep Learning Model,” in *Proc. 3rd Int. Conf. Inventive Research in Computing Applications (ICIRCA)*, 2021, pp. 1285–1291.

[4] M. E. Pothen and M. L. Pai, “Detection of Rice Leaf Diseases Using Image Processing,” in *Proc. 4th Int. Conf. Computing Methodologies and Communication (ICCMC)*, 2020, pp. 424–430.

[5] J. N. Reddy, K. Vinod, and A. S. Remya Ajai, “Analysis of Classification Algorithms for Plant Leaf Disease Detection,” in *Proc. IEEE Int. Conf. Electrical, Computer and Communication Technologies (ICECCT)*, 2019, pp. 1–6.

[6] N. R. Bhimte and V. R. Thool, “Diseases Detection of Cotton Leaf Spot Using Image Processing and SVM Classifier,” in *Proc. 2nd Int. Conf. Intelligent Computing and Control Systems (ICICCS)*, 2018, pp. 340–344.

[7] P. Krithika and S. Veni, “Leaf Disease Detection on Cucumber Leaves Using Multiclass Support Vector Machine,” in *Proc. Int. Conf. Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2017, pp. 1276–1281.

[8] Y. K. Dubey, M. M. Mushrif and S. Tiple, “Superpixel Based Roughness Measure for Cotton Leaf Diseases Detection and Classification,” in *Proc. 4th Int. Conf. Recent Advances in Information Technology (RAIT)*, 2018, pp. 1–5.

[9] Google Images, “SVM Linear, Non-linear, and Kernel Decision Boundary,” [Online]. Available: https://www.google.com/search?q=svm+non+linear

[10] M. P. Vaishnnave, K. S. Devi, P. Srinivasan and G. A. P. Jothi, “Detection and Classification of Groundnut Leaf Diseases Using KNN Classifier,” in *Proc. IEEE Int. Conf. System, Computation, Automation and Networking (ICSCAN)*, 2019, pp. 1–5.

[11] S. Veni, R. Anand, D. Mohan and P. Sreevidya, “Leaf Recognition and Disease Detection Using Content Based Image Retrieval,” in *Proc. 7th Int. Conf. Advanced Computing and Communication Systems (ICACCS)*, 2021, pp. 243–247.

[12] Google Images, “Leaf Disease Example Image,” [Online]. Available: https://images.app.goo.gl/7y9H518kMhF7RcZT6

[13] A. Jenifa, R. Ramalakshmi and V. Ramachandran, “Cotton Leaf Disease Classification Using Deep Convolution Neural Network for Sustainable Cotton Production,” in *Proc. IEEE Int. Conf. Clean Energy and Energy Efficient Electronics Circuit for Sustainable Development (INCCES)*, 2019, pp. 1–3.

[14] S. Kumar, K. Prasad, A. Srilekha, T. Suman, B. P. Rao and J. N. Vamshi Krishna, “Leaf Disease Detection and Classification Based on Machine Learning,” in *Proc. Int. Conf. Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)*, 2020, pp. 361–365.

[15] Google Images, “SVM and Decision Boundary Visualization,” [Online]. Available: https://images.app.goo.gl/FfagRNbaTV7mDrCj6

[16] Y. Zhong and M. Zhao, “Research on Deep Learning in Apple Leaf Disease Recognition,” *Computers and Electronics in Agriculture*, vol. 168, 2019, Art. no. 105146. [Online]. Available: https://doi.org/10.1016/j.compag.2019.105146

[17] A. J. Khalil, A. M. Barhoom, M. M. Musleh and S. S. Abu-Naser, “Apple Trees Knowledge Based System,” *International Journal of Academic Engineering Research*, vol. 3, pp. 1–7, 2019.

[18] Y. Dong, Z. Fu, S. Stankovski, Y. Peng and X. Li, “A Cotton Disease Diagnosis Method Using a Combined Algorithm of Case-Based Reasoning and Fuzzy Logic,” *The Computer Journal*, vol. 64, pp. 155–168, 2020. [Online]. Available: https://doi.org/10.1093/comjnl/bxaa098

[19] Y. Gao, Z. Cao, W. Cai, G. Gong, G. Zhou and L. Li, “Apple Leaf Disease Identification in Complex Background Based on BAM-Net,” *Agronomy*, vol. 13, no. 1240, p. 1240, 2023.

[20] J. Long, E. Shelhamer and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.

[21] O. Ronneberger, P. Fischer and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” *arXiv preprint arXiv:1505.04597*, 2015.

[22] V. Badrinarayanan, A. Kendall and R. Cipolla, “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017. [Online]. Available: https://doi.org/10.1109/TPAMI.34

[23] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh and J. Liang, “UNet++: A Nested U-Net Architecture for Medical Image Segmentation,” *arXiv preprint arXiv:1807.10165*, 2018.

[24] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[25] S. Woo, J. Park, J.-Y. Lee and I. S. Kweon, “CBAM: Convolutional Block Attention Module,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11211, 2018, pp. 3–19.

[26] X. Chao, “Research on the Methods of Identification and Lesion Segmentation of Common Apple Leaf Diseases,” M.S. thesis, Northwest A&F University, 2021. (in Chinese with English abstract)