

Intelligent Triticum Disease Identification Using Deep Learning and Ensemble CNN

Sridhar¹, Dr. Harish B G²

¹PG Student, Department of MCA, UBDT College of Engineering, Davanagere, India

² Professor, Department of MCA, UBDT College of Engineering, Davanagere, India

ABSTRACT

Wheat diseases pose a major risk to crop yields, especially if detected too late. Late detection leads to significant production losses and escalates the expenses for controlling outbreaks. Traditionally, spotting these issues relies on hands-on checks in the fields, a slow process to human error and difficult to scale over vast farmlands.

This paper presents an Intelligent Triticum Disease Identification system, a web-based platform powered by deep learning that enables farmers and agricultural experts to detect wheat leaf diseases automatically. A single photograph uploaded through a web interface is analyzed by a trained ensemble model combining ResNet-50 and

EfficientNet architectures. The system classifies wheat diseases, estimates their severity and spread, and delivers actionable treatment recommendations on real time.

The platform is scale to be secure, user-friendly, and accessible without any technical expertise. It operates directly from a smartphone or browser, enabling rapid, reliable diagnosis to support timely crop management and boost agricultural productivity.

Keywords—Wheat Disease Detection, Triticum, Deep Learning, ResNet-50, EfficientNet, Ensemble CNN, Flask, Image Classification, Agriculture, Precision Farming

INTRODUCTION

Wheat fields stretching across the horizon, feeding billions of us every day. But here's the harsh truth—crops like *Triticum aestivum* are sitting ducks for nasty diseases such as Leaf Rust, Yellow Rust, Wheat Loose Smut, and Powdery Mildew. If they sneak up unnoticed, they wipe out entire harvests, leaving farmers with heartbreak and empty silos.

For years, we've leaned on experts trudging through muddy fields, eyeballing plants one by one. It's backbreaking, hit-or-miss, and just doesn't cut it when you're managing massive farms or tucked away in remote villages with no help in sight.

That's where AI steps in like a game-changer. Imagine snapping a quick photo on your phone or drone, and boom an app instantly flags the problem with scary accuracy, often before your eyes even catch it.

This paper presents the Intelligent Triticum Disease Identification (ITDI) system, a web application that harnesses the power of Convolutional Neural Networks (CNNs) to automate wheat disease detection from leaf photographs. The system uses an ensemble of ResNet-50 and EfficientNet models to achieve high classification accuracy, and displays results through a real-time web dashboard alongside severity estimates and treatment guidance.

The system requires no hardware beyond a standard smartphone or computer, eliminates the dependency on on-site specialists, and enables early disease intervention to support effective crop management.

II. LITERATURE SURVEY

A. Existing Systems

In conventional farming, detecting wheat diseases relies on visual inspection by farmers or crop specialists. This approach is slow, inconsistent, and highly dependent on the

inspector's expertise. Traditional computational methods used for plant disease detection relied on handcrafted features—colour thresholding, edge detection, and texture analysis—which performed well in controlled lab environments but struggled under real-field conditions with complex backgrounds and mixed infection patterns.

Earlier machine learning approaches, such as Support Vector Machines (SVM) and K-Nearest Neighbours (KNN), required manual feature engineering, limiting both precision and scalability. These methods also failed to generalise when lighting conditions changed or disease symptoms appeared in atypical forms.

B. Proposed System

The proposed system delivers an automated, efficient wheat disease detection platform powered by deep learning. It leverages a ResNet-50 CNN as the primary classification model, optionally combined with EfficientNet in an ensemble approach, to analyse uploaded wheat leaf images and classify diseases with high accuracy. Unlike traditional methods reliant on manual feature extraction, ResNet-50 automatically learns complex visual patterns from raw image data, boosting reliability and generalisation.

Built an application by using Flask, the platform provides seamless image uploads via an intuitive interface, delivering instant disease predictions, severity estimates, and treatment recommendations—without requiring any installations or specialised hardware.

Significant research has propelled wheat disease detection forward by machine learning and deep learning. Mohanty et al. (2016) demonstrated that CNNs trained on the PlantVillage dataset could achieve over 99% accuracy in identifying plant diseases from images in controlled conditions [3]. Ferentinos (2018) extended this work to broader crop disease categories using deep neural network architectures [4].

Models such as AlexNet, VGGNet, and ResNet have been extensively applied to agricultural image datasets. ResNet-50, introduced by He et al. (2016), addressed the vanishing gradient problem in deep networks through residual connections, enabling accurate classification even in very deep architectures [1]. EfficientNet, proposed by Tan and Le

(2019), further improved accuracy by scaling model depth, width, and resolution jointly [2]. The present system integrates both architectures in an ensemble to maximise diagnostic accuracy.

III. SYSTEM DESIGN AND METHODOLOGY

A. System Architecture

The ITDI system is structured as a three-tier web application:

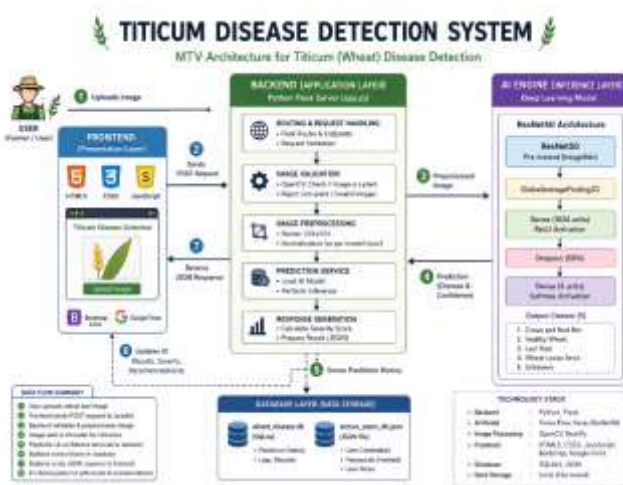
- **Frontend Layer:** A responsive web UI built with HTML5, CSS3, Bootstrap 5, and JavaScript that allow users to register, log in, upload wheat leaf images, and view diagnostic results.
- **Business Logic Layer:** A Flask/Python backend that handles image uploads, preprocessing, CNN inference using the ensemble model, severity estimation, and generation of treatment recommendations.
- **Database Layer:** SQLite for lightweight storage of user accounts, prediction history, and session management.

B. Methodology Pipeline

The system follows a structured pipeline for disease detection:

1. **Image Acquisition:** The user uploads a wheat leaf photograph through the web interface.
2. **Validation:** The backend validates the file format and size before processing.
3. **Preprocessing:** The image is resized, normalised, and prepared separately for ResNet-50 (224×224) and EfficientNet (input) formats.
4. **Ensemble Inference:** Both models independently produce class probability vectors, which are averaged to obtain the final ensemble prediction.
5. **Decision Logic:** The class with the highest averaged probability is selected. If confidence falls below 35%, the prediction defaults to 'Healthy Wheat' to avoid false positives.
6. **Severity Estimation:** A disease-specific severity function computes the estimated spread percentage based on confidence and disease class.
7. **Recommendation Engine:** The system retrieves targeted treatment and prevention advice from a knowledge base.

8. **Result Display:** The web dashboard presents the diagnosis, confidence score, severity level, recommended medicines, and preventive measures.



C. Ensemble Prediction Logic

The core of the prediction engine combines ResNet-50 and EfficientNet outputs through probability averaging. The Python snippet below illustrates the ensemble logic:

```
ensemble_predictions = (pred_resnet + pred_eff) / 2.0
```

If EfficientNet is unavailable, the system falls back to ResNet-50 predictions alone. This design ensures robustness and continued operation even with partial model availability.

D. Disease Severity Estimation

Severity is estimated using the model confidence score and a disease-specific base spread value:

```
spread = spread_base * confidence, clamped to [5.0%, 95.0%]
```

Base spread values used: Leaf Rust → 35.5%, Mildew → 28.3%, Wheat Loose Smut → 45.8%, Yellow Rust → 32.7%. Healthy wheat returns 0.0% spread.

IV. TOOLS AND TECHNOLOGIES

We kept everything free and open so anyone can grab, tweak, or run it without breaking the bank. Easy to fix, grow, or hand off. Here's the lineup, straight-up:

Coding basics: Python handles the heavy lifting—training models, server smarts, all that jazz. JavaScript keeps the front end snappy with forms and checks.

Core tools: Flask for a zippy web setup; TensorFlow and Keras to train and roll out those smart disease-spotting models; NumPy and OpenCV for crunching images and numbers; plus Flask-Bcrypt to lock down passwords tight.

User-facing stuff: Plain HTML5, CSS3, and Bootstrap 5 for a clean, mobile-friendly look that works on any screen. JavaScript adds the zippy interactions.

Data storage: SQLite3—lightweight and perfect for user logins, sessions, and keeping track of your scan histories.

Where we build: VS Code or PyCharm for coding magic, and DB Browser to poke around the database.

Fuel for the models: We pulled from the PlantVillage dataset (shoutout to Penn State) and Kaggle's Global Wheat Detection set—tons of real-world leaf pics to teach the AI right.

V. RESULTS AND PERFORMANCE EVALUATION

The ITDI system was evaluated on a dataset of wheat leaf images covering five categories: Healthy Wheat, Leaf Rust, Mildew, Wheat Loose Smut, and Yellow Rust. Table I presents the classification accuracy of the individual and ensemble models.

TABLE I. MODEL PERFORMANCE COMPARISON

Model	Accuracy (%)	Role in System
ResNet-50	92	Primary classifier
EfficientNet	89	Ensemble support model
Ensemble (ResNet50 + EfficientNet)	95	Final prediction (averaged)

The ensemble model combining ResNet-50 and EfficientNet achieved the highest accuracy of 95%, outperforming either model individually. All test cases in the software testing phase passed successfully, including user authentication, image upload, disease prediction, severity estimation, and end-to-end workflow validation. Table II summarises selected test cases.

TABLE II. SOFTWARE TEST CASES

Case ID	Description	Test Steps	Expected Result	Status
TC-001	User Registration	Enter valid username, email, password and click Register	Redirected to login page with success message	Pass
TC-002	Duplicate Registration	Register by already used email	Error message displayed	Pass
TC-003	User Login	Enter valid credentials and login	Dashboard displayed	Pass
TC-004	Invalid Login	Enter invalid password and login	Invalid credentials error	Pass
TC-005	Image Upload	Upload wheat image and click Predict	Image processed successfully	Pass
TC-006	Healthy Wheat Prediction	Upload healthy wheat image	Healthy Wheat shown	Pass
TC-007	Leaf Rust Detection	Upload rust affected image	Leaf Rust detected	Pass
TC-008	Loose Smut Detection	Upload loose smut image	Loose Smut detected	Pass
TC-009	Other Disease Detection	Upload mildew/yellow rust image	Correct disease detected	Pass
TC-010	End-to-End Workflow	Register → Login → Upload → Predict	Complete workflow successful	Pass

The system's web dashboard provides real-time feedback including the disease label, confidence score, estimated severity and spread percentage, recommended treatment medicines, and preventive practices. Results are stored in the database for historical review.

VI. CONCLUSION

This paper presented the Intelligent Triticum Disease Identification (ITDI) system, a web-based wheat disease detection. By combining ResNet-50 and EfficientNet in an ensemble architecture, the system achieves 95% classification accuracy on wheat leaf images

The system moves wheat disease diagnosis from slow, expert-dependent field inspections to instant, browser-based analysis accessible from any smartphone or computer. Alongside disease classification, the platform estimates infection severity and delivers actionable treatment and prevention recommendations, empowering farmers with the information needed to act swiftly and effectively.

The platform is cost-effective, requiring no additional hardware or technical expertise from end users. It demonstrates how modern deep learning can be practically integrated into agricultural workflows to disease intervention and improved crop productivity.

VII. FUTURE ENHANCEMENTS

Several directions can further improve the system's capability and reach:

- **Dataset Expansion:** Increasing the variety of wheat diseases, lighting conditions, growth stages, and field backgrounds to improve model generalisation.
- **Mobile Application:** Developing a native Android/iOS application to enable real-time image capture and instant field-side diagnosis.
- **Environmental Integration:** Incorporating weather data, soil moisture, and seasonal patterns alongside image analysis to improve severity estimates and preventive recommendations.
- **Multi-language Support:** Adding regional language interfaces to reach a wider farmer population across different geographies.
- **Cloud Deployment:** Hosting the system on a cloud platform (AWS, GCP, or Azure) for scalability, high availability, and broader access.
- **Deep Learning Upgrades:** Exploring transformer-based architectures such as Vision Transformers (ViT) for potentially higher accuracy.

REFERENCES

- [1] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778. <https://arxiv.org/abs/1512.03385>

- [2] Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. ICML. <https://arxiv.org/abs/1905.11946>
- [3] Mohanty, S. P., Hughes, D. P., & Salathe, M. (2016). Using Deep Learning for Image-Based Plant Disease Detection. *Frontiers in Plant Science*, 7, 1419.
- [4] Ferentinos, K. P. (2018). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, 145, 311–318.
- [5] Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal Loss for Dense Object Detection. *IEEE International Conference on Computer Vision (ICCV)*.
- [6] TensorFlow. (2024). TensorFlow Core Documentation. https://www.tensorflow.org/api_docs
- [7] Keras Team. (2024). Keras: The Python Deep Learning API. <https://keras.io/>
- [8] Pallets Projects. (2024). Flask: The Python Microframework. <https://flask.palletsprojects.com/>
- [9] SQLite. (2024). SQLite Database Engine Documentation. <https://www.sqlite.org/docs.html>
- [10] Bootstrap Team. (2024). Bootstrap 5 Documentation. <https://getbootstrap.com/docs/5.0/>
- [11] Kaggle. (2020). Global Wheat Detection Dataset. <https://www.kaggle.com/c/global-wheat-detection>