Discover Artificial Intelligence

RESEARCH Open Access

# A three-tier deep learning framework with mobile application integration for multi-crop disease diagnosis

Aarti Kochhar[1†], Gagan Narang[2*†], Shashikant Patel[1], Harleen Kaur[3], Chetan Singla[4], Brijendra Pateriya[1] and Alessandro Galdelli[2*]

[†]Aarti Kochhar and Gagan Narang contributed equally to this work.

*Correspondence:
Gagan Narang
g.narang@pm.univpm.it
Alessandro Galdelli
a.galdelli@univpm.it
[1]Punjab Remote Sensing Centre, Punjab, India
[2]Department of Information Engineering, Università Politecnica delle Marche, Ancona, Italy
[3]Guru Nanak Dev Engineering College, Ludhiana, India
[4]Punjab Agricultural University, Punjab, India

## Abstract

Crop diseases remain a critical threat to global food security, contributing to substantial yield losses and reduced farmer incomes. Timely and accurate identification of these diseases is essential to mitigate their impact. Traditional diagnostic methods, dependent on expert visual inspection, are labour-intensive, time-consuming, and prone to judgment errors. Accurate and timely detection of crop diseases supports sustainable agricultural management and contributes to achieving global objectives under the United Nations Sustainable Development Goal 2 on Zero Hunger. This study proposes a three step framework that relies on pattern recognition and classification of visual disease symptoms to deliver reliable, field-applicable diagnostics. The approach combines image acquisition through smartphone camera with a structured processing pipeline that includes feature extraction, classification, and result delivery via a mobile application built on a three-tier architecture. Convolutional Neural Networks and an optimized VGG-16 model form the core classification engine, trained to recognize 19 leaf based diseases across wheat, rice, fodder, maize, and sugarcane. The models were trained and evaluated on a dataset comprising both field-collected and publicly available images using repeated stratified k-fold cross-validation. The framework achieves accuracies of 84.61% for wheat, 44.15% for rice, 85.71% for fodder, 95.23% for maize, and 64.28% for sugarcane (testing accuracy of the best-performing model per crop, where VGG-16 demonstrated superior generalization). The framework is able to support farmers, by integrating a technically robust backend with a simple and oriented interface, with diagnosis of multiple crops from a single platform, offering a scalable solution for precision agriculture and sustainable crop protection.

**Keywords** Crop disease detection, Classification, Precision agriculture, Mobile-based diagnosis, Sustainable agriculture

## 1 Introduction

Over the years, the global population has experienced continuous growth and stress on production output has consistently remained an open area of research. According to the United Nations' World Population Prospects 2024 report, global population is anticipated to reach approximately 10.3 billion by the mid-2080s, up from 8.2 billion in 2024 [1]. This rapid increase in population places a substantial strain on the agricultural sector, intensifying the demand for food production. As illustrated in Fig. 1, even though roughly 30,000 plant species worldwide are classified as edible, global cultivation patterns continue to concentrate on only a very small subset of them, amounting to approximately 4% of all edible species that are actually grown for food production.

Furthermore, within this limited selection and reliance on just nine crops serve as the primary source of 75% of the global food supply. Even more strikingly, only three of these nine staple crops, rice, wheat, and maize account for half of the total food consumed by humans [2, 3]. This heavy reliance on a narrow range of crops raises concerns about food security, biodiversity loss, and the resilience of the global food system in the face of climate change and other challenges. Though, farm operations and management are being improved for more precision use of resources [4, 5], however it may face unforeseen challenges to progress such as phytopathology issue. Crop diseases have been identified as one of the most significant threat to global food security, leading to substantial yield losses and economic impacts for farmers worldwide [6–8]. As an example, crops such as cocoa have played a direct role in global inflation in the prices of products like chocolates, since cocoa prices have skyrocketed in recent years and reached historic highs of nearly $10,000 per metric ton in 2024. Although prices have softened slightly with market adjustments, they continue to remain high because of persistent supply side pressures, and studies show that this situation may continue, given the ongoing crop health crisis and the long gestation period required for cocoa production [9].

Early detection of plant diseases therefore is very crucial for minimizing losses and protecting crops and inflation. The earlier a disease is identified, the more effectively it can be controlled. Traditional methods of disease diagnosis, often reliant on visual inspection by experts [10], are time-consuming, subjective, dependent on availability of expert and prone to errors [11]. Moreover, limited access to expert knowledge and diagnostic facilities in rural areas, where many smallholder farmers are located, further

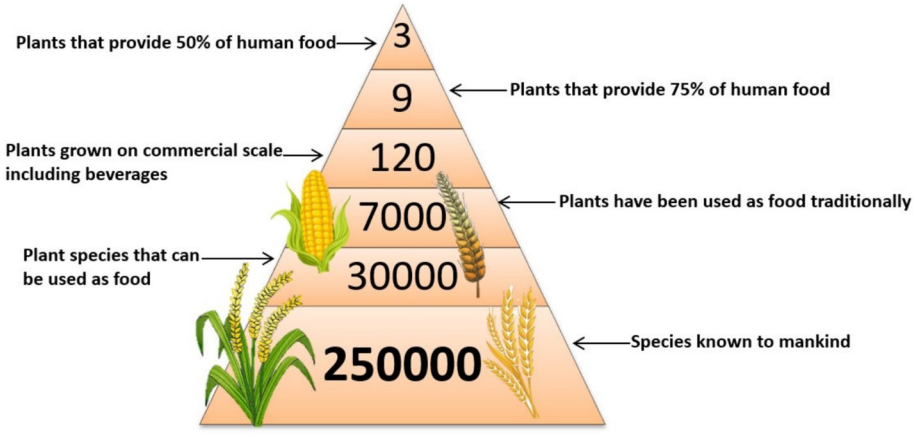



**Fig. 1** Consumption of plant species by human beings [2]

exacerbates these challenges [12]. To overcome, modern technology offers automated solutions that make plant disease detection more accurate and efficient. Recent advancements in deep learning technology have presented new opportunities for addressing challenges in agriculture, particularly in the domain of crop disease detection and management [13–15]. The extensive integration of Machine Learning (ML) in agricultural practices [16] has led to the emergence of terms like precision agriculture and smart farming. ML operates on the principle of learning from experience, enabling models to successfully perform new and unseen tasks.

The process of acquiring experience is known as the learning phase, while applying the trained model to make predictions is referred to as the testing phase [17]. Expecting farmers with limited digital literacy to master complex technologies in order to reap their benefits is unrealistic, and one of the main adoption barrier of the technology. This creates a challenge in making advanced agricultural solutions accessible and usable [18]. Now, envision a smart, mobile-based application that enables farmers to accurately identify plant diseases in real-time. Such a tool could empower both small and large-scale farmers to make informed decisions regarding fertilizer use and disease management. Though there exist model based solutions that are emerging recently [19], however the actual usability through ubiquitous computing paradigms, is not tested. The motivation behind this research arises from the pressing need to support farmers in the early detection and management of crop diseases, a challenge that significantly impacts agricultural productivity and food security. Traditional disease detection methods are often time-consuming as they rely on expert intervention. Although deep learning models have demonstrated strong potential in this domain, their practical accessibility to farmers remains limited. To overcome this gap, the study integrates deep learning techniques with an Android application to automate plant disease identification across multiple crops. The key contribution of this paper lies in developing and evaluating a scalable, cross-crop disease prediction framework that merges the precision of deep learning with real-world applicability. The proposed model consists of three core components: (i) a deep learning backend, (ii) an API (Application Programming Interface) middleware, and (iii) an Android-based frontend, forming an end-to-end, user-friendly solution. The framework operates for five different crops within a single integrated platform. The system enables farmers to capture, analyze, and obtain disease predictions through a mobile interface, effectively bridging the gap between advanced AI research and practical agricultural implementation. Leveraging the computational power of deep learning algorithms, specifically Convolutional Neural Networks (CNNs), the proposed system aims to enable farmers to accurately identify and classify various types of crop diseases directly from images captured by smartphone cameras.

## 2 Related works

Deep learning has emerged as a powerful tool for diagnosing plant diseases, improving the accuracy and efficiency of crop disease identification. One widely adopted technique in this domain is transfer learning. Among the prominent architectures, VGG-16, developed by the Visual Geometry Group at the University of Oxford stands out. Named for its 16 weight layers (including 13 convolutional and 3 fully connected layers), VGG-16 has become a go-to model in transfer learning applications [20]. A lot of other approaches have been emerging in literature as summarized in Table 1.

**Table 1** Summary of key studies on plant leaf disease detection using machine learning and deep learning

| Author and Year | Focus | Model Used/Methodology | Dataset | Key Findings / Accuracy | Remarks |
|---|---|---|---|---|---|
| Ahmad et al. (2023) [21] | Comprehensive review of image processing, ML and DL in plant disease diagnosis | Reviewed 70 studies, compared DL and ML approaches | Various public datasets | – | Identified key factors like dataset requirements, accessibility, data collection methods, deep learning strategies, model generalization, disease severity estimation, early detection and automated systems |
| Rautaray et al. (2020) [22] | Paddy disease classification (Hispa, Brown Spot, Leaf Blast) | Transfer learning using VGG-16 | Paddy leaf images | 92% training, 90% testing accuracy | Demonstrated efficiency of pretrained VGG-16 for crop disease detection |
| Agarwal et al. (2020) [23] | Lightweight CNN for plant disease classification | CNN with eight hidden layers | PlantVillage dataset (Kaggle) | Proposed CNN: 98.4%, k-NN: 94.9%, VGG-16: 93.5% | Outperformed pretrained models, strong generalization beyond PlantVillage, benefited from preprocessing |
| Li et al. (2021) [24] | Review of DL for leaf disease identification | Review of CNNs, data augmentation and transfer learning | Various public datasets | – | Highlighted limited labeled data and importance of transfer learning |
| Harakannanavar et al. (2022) [25] | Tomato leaf disease detection | SVM, CNN and k-NN | PlantVillage dataset (Kaggle) | SVM: 88%, k-NN: 97%, CNN: 99.6% | Showed DL superiority for complex leaf features, benefited from feature extraction |
| Sujatha et al. (2021) [26] | Comparative study for citrus disease detection | Compared SVM, RF, SGD with Inception-v3, VGG-16, VGG-19 | Citrus leaf dataset | VGG-16: 89.5%, Inception-v3: 89%, VGG-19: 87.4%, SVM: 87%, SGD: 86.5%, RF: 76.8% | DL models outperformed ML models |
| Salam et al. (2024) [27] | Early disease detection in mulberry leaves (healthy, rust, spot) | ResNet50, VGG19, MobileNetV3Small | 6,000 images (augmented from 1,091) | MobileNetV3Small: 96.4%, VGG-19: 94.2%, ResNet50: 94.4% | Integrated with an Android application |
| Iftikhar et al. (2024) [28] | Early leaf disease detection in apple, maize and potato | Enhanced CNN | PlantVillage dataset (Kaggle) | Enhanced CNN: 98.17% | Integrated with an Android application |
| Yilmaz et al. (2025) [29] | Investigation of leaf disease detection across five crop types | Analysis of 19,838 papers (198 selected) | Various public datasets | – | Emphasized need for user accessibility options via mobile applications |

Ahmad et al. [21] provided a comprehensive review of 70 studies involving image processing, machine learning, and deep learning techniques for plant disease diagnosis. The review summarized several publicly available datasets and discussed various prevalent plant diseases. The authors organized their findings around seven key considerations, including dataset requirements, accessibility and usability, data collection methods, deep learning strategies, model generalization, disease severity estimation, comparisons between deep learning and human accuracy, and open research questions. They emphasized the value of automated systems for early disease detection and severity assessment, which are crucial for effective plant disease management. The demonstrations highlight Rautaray et al. [22], where they employed a transfer learning-based approach using VGG-16 to classify diseases in paddy plants, specifically targeting three common conditions: hispa, brown spot, and leaf blast. Their model achieved impressive results, with 92% training accuracy and 90% testing accuracy, showcasing the potential of pre-trained models for crop disease detection.

In another comparitive study, Agarwal et al. [23] proposed a lightweight CNN architecture with eight hidden layers, using the publicly available Plant Village dataset. Their model outperformed both traditional machine learning methods and pre-trained models, achieving an accuracy of 98.4%. While traditional methods peaked at 94.9% and VGG-16 reached 93.5%, the proposed model benefited from preprocessing techniques such as image augmentation and random brightness adjustment. It also demonstrated strong generalization by achieving 98.7% accuracy on datasets beyond Plant Village. Several studies have also concentrated on disease detection in leaf-based crops, recognizing the critical role of leaf health in early-stage diagnosis. These efforts typically leverage deep learning models to analyze visual symptoms manifested on leaves, often using high-resolution imagery and enhanced feature extraction techniques. This focus aligns with the broader trend of utilizing leaf characteristics as primary indicators for identifying and classifying crop diseases. Li et al. [24] provided a comprehensive review of recent developments in deep learning applications for crop leaf disease identification. The study identified key trends such as the integration of advanced imaging modalities and the growing reliance on data augmentation and transfer learning to boost model performance. Emphasis was placed on the challenges of working with limited labeled data and the importance of building large, diverse datasets. The review also highlighted the utility of CNN activation map visualization for improving model interpretability and accuracy [30]. While similar explainable pipelines are emerging in other domains such as maritime, there is limited research in agriculture [31].

Additionally, the potential of hyperspectral imaging for early disease detection was discussed, particularly in scenarios involving small sample sizes. Harakannanavar et al. [25] focused on the detection of disorders in tomato leaf samples using a hybrid approach combining image preprocessing, feature extraction, and multiple classification techniques. The input images were resized to $256 \times 256$ pixels and enhanced using histogram equalization. K-means clustering was applied to partition the image space into Voronoi cells, and leaf boundaries were extracted via contour tracing. Feature extraction involved Discrete Wavelet Transform (DWT), principal component analysis, and gray level co-occurrence matrix. These features were then classified using Support Vector Machines (SVM), CNN, and K-Nearest Neighbor (k-NN). The model achieved classification

accuracies of 88% with SVM, 97% with k-NN, and 99.6% with CNN, indicating the superior performance of deep learning models in complex leaf disorder classification tasks.

Citrus plant disease detection has also been a focus of extensive research. Sujatha et al. [26] conducted a comparative study evaluating the performance of traditional machine learning algorithms like SVM, Random Forest (RF), and Stochastic Gradient Descent (SGD) against deep learning architectures such as Inception-v3, VGG-16, and VGG-19. The findings demonstrated that deep learning models consistently outperformed their ML counterparts in classification accuracy. Among the tested models, RF yielded the lowest accuracy at 76.8%, while VGG-16 achieved the highest at 89.5%. Inception-v3 and VGG-19 followed closely with 89% and 87.4%, respectively, while SVM and SGD recorded 87% and 86.5%. These results highlight the robustness of deep learning approaches in detecting citrus diseases, and in handling visual features.

Over time, researchers have emphasized the significance of incorporating user interfaces, such as mobile or web applications, for successful implementation. Salam et al. [27] addresses the challenge of early disease detection in mulberry leaves, which are essential for silk production but prone to rapidly spreading infections. Researchers collected 1,091 leaf images from Bangladesh, categorized as healthy, rust-affected, and spot-affected, and expanded the dataset to 6,000 through augmentation. Three pretrained CNNs, ResNet50, VGG19, and MobileNetV3Small-were modified with additional convolutional layers and evaluated. The enhanced MobileNetV3Small model outperformed others, achieving 96.4% accuracy, precision, recall, and F1-score. To enable real-world deployment, the model was quantized and converted to TensorFlow Lite format for use in an Android app, ensuring faster, power-efficient inference. Iftikhar et al. [28] proposed an enhanced Convolutional Neural Network (E-CNN) for early identification of leaf diseases in apple, corn, and potato crops. The work analyzes the impact of hyperparameter tuning and data augmentation on model performance, comparing various machine learning and pre-trained deep learning models. The optimized E-CNN achieved a high accuracy of 98.17% in detecting fungal diseases. To enhance usability, the model was integrated into a mobile application that allows farmers to upload or capture images for diagnosis.

Yilmaz et al. (2025) analyzed 19,838 articles published between 2021 and 2023 from major academic databases, narrowing them down to 198 relevant studies [29]. It addresses key research questions concerning disease types, global applicability of models, contributions of computer vision, robotics, and Internet of Things (IoT) technologies, and the limitations of current datasets. The review highlights the growing role of deep learning and hybrid systems with user interface in early detection, while emphasizing persistent challenges such as limited geographic diversity, real-world data scarcity, and insufficient integration of technologies with traditional farming. A research gap and opportunity improvements exists since there is limited progress in an end-to- end applicable pipeline with modular components such that the models that are capable of adapting to diverse environmental and crop conditions, and an ease of practical deployment. This research investigates the classification of diverse leaf diseases across 5 crop types with 19 different leaf-based diseases, aiming to develop a modular pipeline which has possibilities to include robust deep learning models that are optimized for deployment through a mobile application. The primary contribution of this work lies in developing a scalable, cross-crop disease detection framework that integrates deep learning models

with an Android-based mobile application through an API middleware. While many existing studies focus on single-crop models, this work addresses the practical research gap by enabling disease detection for five different crops within a single unified platform. This integration makes advanced deep learning techniques accessible directly to farmers, thereby overcoming the limitation of limited field-level usability seen in prior research. The framework preserves crop health by supporting and informing on early disease detection. The farmers can capture leaf images in the field, receive predictions, and take timely corrective actions proactively, ultimately reducing crop damage and improving productivity. This end-to-end framework, integrating a mobile interface with crop-specific deep learning models, represents a notable advancement in practical applicability of AI and its integration with user-friendly platforms such as smartphones. Thus, this study emphasizes applicability and user accessibility, particularly for non-expert farmers, facilitating seamless integration with conventional farming methods and enhancing disease management in field conditions.

## 3 Materials and methods

### 3.1 Study site

The dataset used in this study was compiled from two primary sources: field-based image collection conducted at the study site in Punjab Agricultural University (PAU) in Ludhiana, Punjab, India (Fig. 2), and complemented by publicly available annotated Kaggle datasets. This hybrid approach was adopted to address both logistical and scientific requirements: (1) ensuring that locally prevalent diseases were represented accurately through in-situ collection, and (2) supplementing the dataset with high-quality, diverse images from trusted repositories to strengthen model generalization.

Field data were collected for fodder and maize during August, 2024 when disease symptoms were most prominent due to favorable environmental conditions for pathogen proliferation. The study site at PAU was chosen for its research-grade agricultural facilities, access to expert agronomists, and the presence of active crop disease outbreaks during the growing season. This enabled accurate visual diagnosis and labeling by specialists, ensuring data reliability. For fodder, two classes were collected: Healthy and Discoloration (22 images per class). The maize dataset consisted of three classes: Healthy, Banded Leaf and Sheath Blight (BLSB), and Fall Armyworm (FAW) (22 images per class). Images were captured under natural light using high-resolution digital cameras, ensuring visible preservation of disease characteristics such as lesion shape, color change, and patterning. Although this dataset provided a strong foundation, the model training pipeline can readily incorporate additional data from diverse global regions, enabling rapid adaptation and improved robustness across varied agro-climatic conditions.

#### *3.1.1 Datasets*

For rice, sugarcane, and wheat, datasets were sourced from publicly available repositories. This decision was made because the diseases of interest for these crops, particularly certain rice pathogens such as Rice Hispa and Narrow Brown Leaf Spot, were not observed during the field survey period in Punjab, and establishing controlled infection trials would require significant time and biosecurity measures. Using established datasets also ensured greater class diversity and representation of disease phenotypes across different agro-climatic zones, improving the robustness of the trained models. The rice

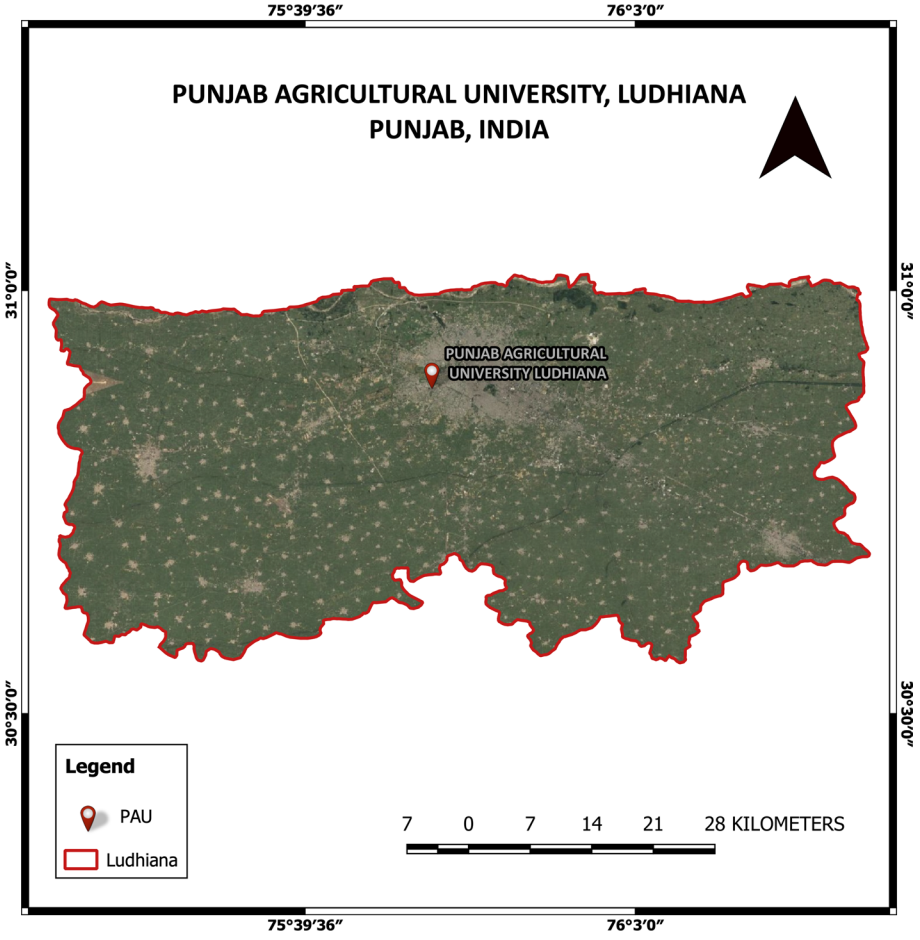



**Fig. 2** Study site located in the northern region of India

dataset covered eight distinct classes: Healthy, Bacterial Leaf Blight, Brown Spot, Leaf Blast, Leaf Scald, Narrow Brown Leaf Spot, Rice Hispa, and Sheath Blight (30 images per class) [32]. The sugarcane dataset included three classes: Healthy, Bacterial Blight, and Red Rot (30 images per class) [33].The wheat dataset comprised three classes: Healthy, Septoria, and Stripe Rust (30 images per class) [34]. Images in these datasets ranged from 1200×1600 to 4000×6000 pixels, providing high spatial resolution for effective feature extraction in CNN model. These images were subsequently standardized through a unified preprocessing pipeline for models currently used in this study and for future expansion in the framework.

### 3.2 Disease detection

This research primarily aims at disease detection for five major crop diseases across key food crops: Wheat, Rice, Fodder, Maize, and Sugarcane. Each crop has different categories of leaf diseases used to train the deep learning model. A few samples of leaves with diseases have been shown in Figs. 3 and 4. Additionally, there is a common class for healthy leaves for each crop. Table 2 represents the distribution and splitting of data for each crop. To maintain equal representation across training and testing subsets, stratification was ensured by performing the data split individually for each disease class. An

(a) Leaf Blast (Rice) (b) Rice Hispa (Rice) (c) Sheath Blight (Rice)

(d) Bacterial Blight (Rice) (e) Brown Spot (Rice) (f) Leaf Scald (Rice)

(g) Brown Spot (Rice)

**Fig. 3** Sample of rice leaf diseases

approximate of 70:30 ratio was followed for each crop and due to this stratification, the distribution remained balanced, preventing artificial inflation or deflation of accuracy.

Additionally, shuffling was incorporated during the splitting process to randomize the order of samples, ensuring that unintended sequential patterns are avoided, bias is reduced, and generalization is improved. This allows the model to learn from the underlying features rather than the data order and results in more representative training–testing sets, thereby supporting better convergence and minimizing overfitting. In total, the dataset includes a total of 19 disease classes across these five crops. To further address the limitation of small test sets and to enhance the robustness and credibility

(a) Stripe Rust (Wheat) (b) BLSB (Maize) (c) Red Rot (Sugarcane)

(d) Discoloration (Fodder) (e) FAW (Maize) (f) Septoria (Wheat)

(g) Bacterial Blight (Sugarcane)

**Fig. 4** Sample of leaf diseases in wheat, maize, sugarcane and fodder

of the evaluation, Repeated Stratified K-Fold (RSKF) cross-validation was incorporated for both CNN and VGG-16 models across all crops. In RSKF, the dataset is repeatedly partitioned into $k$ folds while stratifying the samples to preserve the original class distribution in each split. Since the classes corresponding to each crop in this study are equally distributed, stratification consistently maintains class balance across all folds. Model performance is averaged over multiple folds and repetitions, resulting in evaluation across several independent train–test partitions. For this study, 5 folds $\times$ 3 repeats

**Table 2** Dataset distribution and splitting for different crops

| Crop | Data Source | No. of Classes (including Healthy) | Images per Class | Training per Class | Testing per Class | Training | Testing |
|---|---|---|---|---|---|---|---|
| Maize | Field images | 3 | 22 | 15 | 7 | 45 | 21 |
| Fodder | Field images | 2 | 22 | 15 | 7 | 30 | 14 |
| Rice | Kaggle | 8 | 30 | 21 | 9 | 168 | 72 |
| Wheat | Kaggle | 3 | 30 | 21 | 9 | 63 | 27 |
| Sugarcane | Kaggle | 3 | 30 | 21 | 9 | 63 | 27 |
| **Total** | – | – | – | – | – | **369** | **161** |

were employed, which reduces sensitivity to individual samples and mitigates the effects of limited dataset size in our framework.

#### *3.2.1 Wheat*

In addition to healthy leaves, the wheat dataset includes the following classes of diseases:

- Septoria: the ascomycete fungus *Zymoseptoria tritici* causes septoria tritici blotch, a leaf disease that poses a significant threat to global wheat production [35]. This fungus is characterized by its ability to produce spores within a sac-like structure. It is the most common foliar disease affecting winter wheat in many Western European countries [36].
- Stripe Rust: stripe rust, or yellow rust, is a wheat disease caused by the fungus *Puccinia striiformis f. sp. tritici*. It affects wheat crops globally and can lead to significant damage. In favorable conditions, severe outbreaks may reduce yields by more than 70% [37].

The wheat dataset, comprising 30 images per class, was divided into training and testing sets, resulting in 21 images per class for training and 9 images per class for testing.

#### *3.2.2 Rice*

Rice consists of the dataset of following diseases, besides healthy leaves:

- Bacterial Leaf Blight: it primarily impacts irrigated and rainfed lowland regions. It thrives in temperatures ranging from 25–34°C, with relative humidity exceeding 70%, high nitrogen levels, strong winds, and prolonged rainfall [38].
- Brown Spot: this is identified by prominent, large spots on the leaves, which can eventually cause the entire leaf to die. When the infection spreads to the seeds, it leads to unfilled grains or seeds that appear spotted or discolored [39].
- Leaf Blast: Leaf Blast is among the most severe diseases affecting paddy, targeting multiple parts of the rice plant, such as the leaves, neck, and nodes. It can cause substantial grain yield losses, estimated between 70% and 80% [38].
- Leaf Scald: Leaf Scald usually emerges late in the growing season on mature leaves, thriving in wet conditions, high nitrogen levels, and dense plant spacing. The disease spreads more quickly on wounded leaves, which have been damaged by physical injury, pests, or environmental stress [39].
- Narrow Brown Leaf Spot: Narrow Brown Spot, caused by the fungus *Sphaerulina oryzina*, affects leaves, sheaths, and panicles. It leads to the premature death of leaves and leaf sheaths, early grain ripening, and, in severe cases, plant lodging [39].

- Rice Hispa: Rice Hispa harms rice plants by scraping the upper surface of leaf blades, leaving only the lower epidermis intact. It also tunnels through leaf tissues, and in severe infestations, the plants become less vigorous [40].
- Sheath Blight: spreads commonly during the rainy season, thriving in warm temperatures, high humidity and nitrogen fertilization in dense planting [38].

For each rice disease class, 30 images per class were incorporated and partitioned into training and testing subsets, yielding 21 training per class images and 9 images per class testing images.

#### *3.2.3 Fodder*

In addition to healthy leaves, the fodder dataset includes discoloration, which is the bacterial spot disease is especially severe in warm regions, particularly during the rainy season. It begins with short-line lesions that later expand into long, red-purple streaks measuring between 2 and 20 cm or more in length [41]. A total of 22 fodder images per class were included and divided into training and testing subsets, yielding 15 and 7 images per class respectively.

#### *3.2.4 Maize*

Maize consists of the dataset of following diseases, besides healthy leaves:

- Banded Leaf and Sheath Blight (BLSB): BLSB is a severe disease that first appears on maize leaves and sheaths as leaf sheath blight and later progresses to the ears, causing ear rot [42].
- Fall Armyworm (FAW): FAW is a major pest that severely impacts maize due to its highly destructive feeding behavior. It poses significant challenges in tropical and subtropical regions, where it spreads rapidly and causes extensive crop damage. Managing FAW is difficult due to its polyphagous nature, transboundary movement, rapid reproduction, short life cycle, and high migratory capacity. The absence of a diapause phase further complicates control efforts [43].

For both maize disease classes, 22 images per class were utilized and partitioned into training and testing subsets, resulting in 15 images for training per class and 7 images for testing per class.

#### *3.2.5 Sugarcane*

Besides healthy leaves, the dataset of sugarcane consists of the following diseases:

- Bacterial Blight: Bacterial Blight mainly targets young and middle-aged leaves, initially appearing as long, narrow, uniform, watery-green stripes near the midrib and base of the leaf blades. As the disease advances, these stripes expand across the leaf, merge, and transition from light red to dark red (necrotic) [44].
- Red Rot: Red Rot in sugarcane is a serious fungal disease caused by Colletotrichum falcatum. It deteriorates the internal tissues of the cane, forming red or darkened lesions within the stalk. This disease leads to significant yield loss and negatively impacts sugarcane quality. Common symptoms include reddish discoloration and rotting, which can spread rapidly under favorable conditions such as high humidity and warm temperatures [44, 45].

Similar to the other crops, the sugarcane dataset for each class was partitioned, resulting in 21 images per class for training and 9 images per class for testing.

## 4 Workflow

The proposed system architecture is divided into three main layers: the Deep Learning Model Layer, the Server Layer, and the Client Layer. Figure 5 shows how the different layers work together.

1. Deep Learning Model Layer (Bottom Layer): This is the core layer of the system, containing ten deep learning models (CNN and VGG-16), each designed to detect diseases in a specific crop. These models use advanced deep learning techniques to accurately identify crop diseases.
2. Server Layer (Middle Layer): The server layer acts as the backend of the system and connects the models to the user interface. It includes an API, developed entirely in Python, which processes requests from the client layer and returns the prediction results. This makes the system fast and reliable.
3. Client Layer (Top Layer): The client layer is the frontend of the system, consisting of an Android application that users interact with. The app provides a simple and easy-to-use platform for farmers to get information on disease predictions without dealing with the technical details of the system.

The models at bottom layer were implemented in Python (v3.8) using Python Integrated Development and Learning Environment (IDLE) software application. Pre-trained models were saved in *.h5* format and integrated into a mobile application (Top Layer) developed with Flutter, an open-source UI (User Interface) software development kit by Google for building natively compiled applications for mobile, web, desktop, and embedded devices from a single codebase. The Python backend uses Flask (Middle Layer), a popular lightweight API framework, to handle image preprocessing, prediction, and API responses. Key libraries include PIL for image handling, NumPy for numerical operations, Werkzeug for secure file uploads [46], Keras for deep learning models and JSON for structured output. The app allows users to upload leaf images, select the crop type,

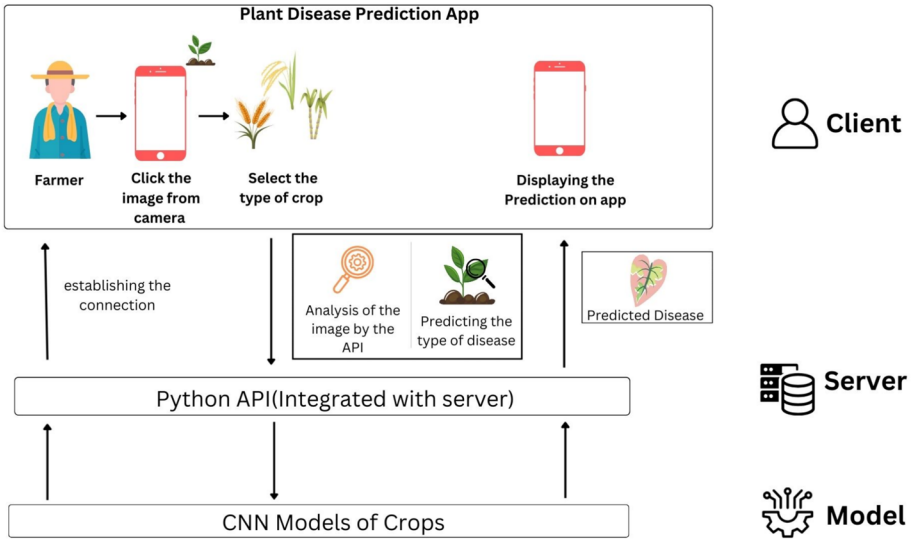



**Fig. 5** Flow diagram

and receive disease predictions. All model training and testing were conducted on a Windows server with an Intel Core i7-9700F CPU, x64-based Processor, 16 GB RAM, 1 TB HDD, running a 64-bit Windows operating system, providing sufficient computational resources for model inference and mobile application integration. The whole system is designed to give quick and accurate predictions, helping farmers make better decisions about their crops.

### 4.1 Deep learning model layer (bottom layer)

The process begins with preparing the dataset, which contains images of crop leaves affected by various diseases. These images are carefully organized into folders, with each folder representing a specific disease class. Proper organization of the dataset ensures smooth processing during model training and evaluation. Figure 6 illustrates the working of the development of deep learning model, which is the core deep learning component of the system.

The next step involves data preprocessing, which is crucial for improving model accuracy and performance. In this stage, images are converted into numerical arrays that the model can understand. Additionally, the images are resized to maintain uniform dimensions (256×256 pixels) and normalized to scale the pixel values, making the model training quicker and more effective. After normalization, the image arrays are reshaped into the format (number_of_images, 256, 256, 3) to match the expected input shape of the CNN. The class labels are also converted into one hot encoded vectors so that the model can correctly interpret them for multi class classification. After preprocessing, the CNN model is designed using the Keras Sequential API. The VGG-based model was built using the VGG-16 architecture pre-trained on ImageNet, configured with an input size of 224×224×3. Image augmentation was also applied using Keras' ImageDataGenerator [47], introducing a variety of transformations including rotations, flips, width and height shifts, zooming, brightness adjustments, shear transformations, and pixel rescaling. These augmentations helped the models better handle variations in scale, orientation, lighting, and minor distortions to improve robustness and generalization to unseen

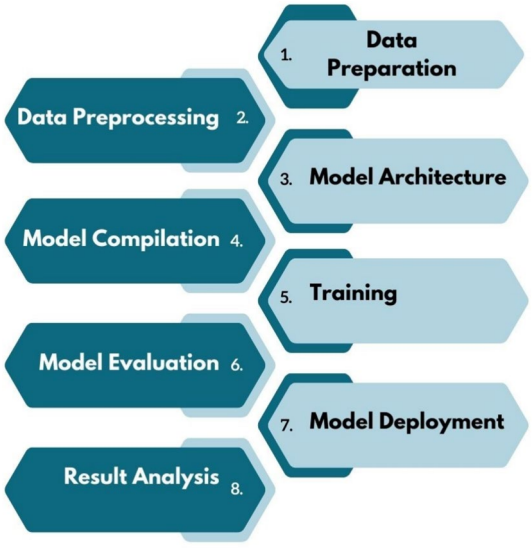



**Fig. 6** Methodology of deep learning model

data. Specifically, a rotation range of ±20°, width and height shift ranges of 0.1 (10% of image dimensions), a zoom range of 0.2 (±20% zoom), and horizontal flipping was performed. In addition, brightness adjustment was applied in the range of 0.9–1.1 (±10% brightness variation), along with shear variation of ±0.1 radians (approximately ±5.7°). Pixel rescaling was performed to normalize pixel values to the 0–1 range. Pixel filling for transformed regions was handled using the nearest-neighbour fill mode. The data augmentation was applied dynamically during model training, where randomized transformation parameters were sampled for each batch, thereby increasing data diversity across epochs without permanently expanding the dataset. A fixed random seed was specified for the augmentation generator to ensure reproducibility of transformation sampling and batch generation across runs.

Given that the plant disease detection task in this study required relatively standard feed-forward CNN and VGG-16 structures without complex branching or shared layers, the Sequential API offered a clean and readable implementation while ensuring compatibility with TensorFlow's high-performance backend. Its simplicity also streamlined model prototyping, training, and deployment to the API layer, reducing development time without sacrificing model performance. The model architecture includes convolutional layers to extract meaningful features from the images, pooling layers to reduce the spatial dimensions while maintaining essential information, and dense layers to classify the images into respective disease categories. This structure allows the model to learn patterns and features that distinguish between different crop diseases. Once the model architecture is finalized, it is compiled by selecting an appropriate loss function, optimizer, and evaluation metrics. Compilation is followed by splitting the dataset into training and test sets. The model is then trained on the training set while monitoring its performance on the testing set to avoid overfitting. During this phase, hyperparameter tuning is performed, where the number of epochs are adjusted to optimize model performance. After training, the model is evaluated on the test set to assess its accuracy and generalization ability. Metrics from both training and testing are analyzed to detect any signs of overfitting or underfitting.

The model when showcases satisfactory results, it is saved for deployment and future use. The model is exported in *.h5* format. In Python, the *.h5* format (i.e. HDF5 - Hierarchical Data Format version 5) is widely used for efficiently storing large numerical datasets. In deep learning, it is commonly utilized in TensorFlow/Keras to save and load trained models. To connect the trained model with the Android application, a Python-based API is developed to serve as the backend interface. This API facilitates communication between the deep learning models and the Android app, enabling predictions. The backend setup includes designing the API, integrating the trained models, and implementing error handling to ensure smooth operation. This comprehensive approach ensures that the deep learning model for crop leaf disease prediction is efficient, accurate, and ready for deployment.

#### *4.1.1 Model architecture*

For each crop type, two neural network architectures were implemented: a custom CNN and the VGG-16 model. This approach allowed for a direct comparison between a lightweight, task specific CNN and a deeper, pre-trained architecture. Given the five crops considered in this study i.e. fodder, maize, rice, sugarcane, and wheat, with a total of 10

models were designed, trained and tested. In this work, the proposed framework is intentionally designed as a multi-model, multi-crop deployment, rather than a single unified classifier. Separate models (CNN and VGG-16) are trained independently for each crop due to the substantial inter-crop variability in leaf morphology, disease manifestation, color distribution, and symptom patterns. Training crop-specific models enables each network to focus on a more homogeneous feature space, potentially reducing learning complexity relative to a unified model spanning multiple crops. The multi-crop unification in this work is achieved at the application and platform level rather than through a single unified classifier. Crop selection is handled explicitly through the user on system interface, after which the corresponding crop-specific trained model is dynamically invoked for inference. All models share a common deployment pipeline while allowing each model to remain specialized for its respective crop.

Each input image, representing a crop, is first subjected to normalization using Min-Max scaling. This preprocessing step ensures that pixel values are scaled to a uniform range, thereby preventing features with larger numerical magnitudes from dominating the learning process. Normalization facilitates more stable and efficient convergence of the model during training. Each image is represented in RGB color space as a three-channel input with dimensions $h \times w \times c$, where h and w denote the image height and width, respectively, and $c=3$ corresponds to the red, green, and blue color channels. The models process all three channels simultaneously to extract relevant features. The architecture of the developed CNN models is illustrated in the corresponding Fig. 7.

The CNN architecture comprises two sequential blocks, each consisting of a convolutional layer followed by a max-pooling operation. The first convolutional layer applies b1 filter size with a kernel size of $n \times n$, where b1 and n vary depending on the specific crop model. This layer extracts local features such as edges, textures, and basic patterns from the input image. A Rectified Linear Unit (ReLU) activation function is applied after convolution to introduce non-linearity, enabling the model to learn complex feature representations. Subsequently, a max-pooling layer with a kernel size of $2 \times 2$ is used to downsample the feature maps, thereby reducing computational load and capturing dominant features. The second convolutional layer follows a similar structure, employing b2 filter size with a kernel size of $m \times m$, where b2 and m are model-specific for each crop. ReLU activation is again applied, followed by a second max-pooling layer (also with kernel size $2 \times 2$). This second convolution–pooling block enables the network to learn higher-level, more abstract features from the previously extracted representations. After the second convolution–pooling block, the feature maps are flattened into a one-dimensional vector and passed into a fully connected layer. This transition from convolutional to dense layer allows the network to combine spatially extracted features into a global

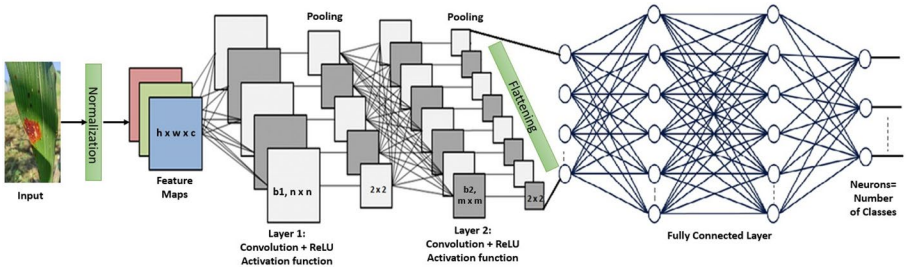



**Fig. 7** Overview of the proposed model architecture for leaf disease detection

feature vector suitable for classification. The fully connected layer then learns non-linear relationships among these features using a set of trainable weights with ReLU activation function. The final output layer, implemented using a Softmax activation function, contains a number of neurons equal to the total disease classes for the respective crop. This design ensures that each model generates class-probability predictions tailored to the specific disease categories in the respective crop dataset, thus completing the classification process.

The VGG-based classification model was developed using the VGG-16 architecture pre-trained on ImageNet, with an input resolution of 224×224×3. The original fully connected classification head was removed from network, and all convolutional base layers were frozen to preserve the pre-trained feature representations and avoid overfitting during fine-tuning on the target dataset. A custom classification head was then constructed on top of the VGG-16 feature extractor. This head included a Global Average Pooling (GAP) layer to reduce the spatial dimensions, followed by two fully connected layers of 1024 units each with ReLU activation, an additional dense layer of 512 units with ReLU activation, and a final softmax output layer with units equal to the number of disease classes. Only the custom classification head was trained, while the VGG-16 backbone remained frozen to stabilize learning and prevent overfitting on the small dataset. This transfer-learning setup enabled efficient feature extraction from the deep convolutional layers of VGG-16 while focusing training on the newly added dense layers to adapt the model to the specific crop disease classification task.

#### *4.1.2 Hyperparameter tuning*

We manually tuned key hyperparameters to balance model performance and training stability across five plant disease classification tasks: fodder, maize, rice, sugarcane, and wheat. These hyperparameters include learning rate, batch size, number of epochs, and the architecture of the convolutional layers. For all CNN and VGG-16 models, we used the Adam optimizer with a learning rate of 0.0001. This value was selected after preliminary experiments with higher rates led to unstable training and poorer testing performance. A categorical cross-entropy loss function was used due to the multi-class nature of the classification tasks. For CNN, epochs were set between 20 and 50 depending on convergence behavior. The CNN architectures were also adjusted per dataset based on early performance diagnostics. Some models incorporated dropout layers to prevent overfitting (dropout rates of 0.25 and 0.5 in the sugarcane model), while others varied in the number of convolutional layers and dense units.

For Wheat and Fodder, three convolutional layers are used, starting with 16 filters in the first layer, 32 in the second, and 16 in the third layer, all with small 2×2 kernels. Sugarcane and Maize use a simpler two-layer approach, with Sugarcane starting at 32 filters and leading to 64, and Maize from 16 to 32 filters, both using 3×3 kernels. Rice has only a single convolutional layer with 32 filters of size 3×3, and no additional layers. Following the convolutional blocks, all models employ a fully connected (dense) classification stage. The input to the fully connected network corresponds to the flattened feature vector obtained after the final convolution and pooling operations, with the number of input neurons determined by the spatial dimensions and depth of the extracted feature maps. Each crop-specific model includes one hidden fully connected layer, with the number of neurons configured as listed in Table 3 (500 for Wheat and Fodder, 128 for

**Table 3** Crop-specific CNN architecture details

| Crop | Layer Type | Filters / Neurons | Kernel Size |
|---|---|---|---|
| Wheat | Convolution-1 | 16 | $2 \times 2$ |
| | Convolution-2 | 32 | $2 \times 2$ |
| | Convolution-3 | 16 | $2 \times 2$ |
| | Fully Connected (Hidden) | 500 | – |
| | Fully Connected (Output) | 3 | – |
| Sugarcane | Convolution-1 | 32 | $3 \times 3$ |
| | Convolution-2 | 64 | $3 \times 3$ |
| | Fully Connected (Hidden) | 128 | – |
| | Fully Connected (Output) | 3 | – |
| Maize | Convolution-1 | 16 | $3 \times 3$ |
| | Convolution-2 | 32 | $3 \times 3$ |
| | Fully Connected (Hidden) | 256 | – |
| | Fully Connected (Output) | 3 | – |
| Rice | Convolution | 32 | $3 \times 3$ |
| | Fully Connected (Hidden) | 128 | – |
| | Fully Connected (Output) | 8 | – |
| Fodder | Convolution-1 | 16 | $2 \times 2$ |
| | Convolution-2 | 32 | $2 \times 2$ |
| | Convolution-3 | 16 | $2 \times 2$ |
| | Fully Connected (Hidden) | 500 | – |
| | Fully Connected (Output) | 2 | – |


**Fig. 8** Methodology of Python API

Sugarcane and Rice, and 256 for Maize). The final output layer has number of neurons equal to the number of disease classes for the respective crop (three classes for Wheat, Sugarcane, and Maize; eight for Rice; and two for Fodder). This configuration reflects a combination of crop-specific feature extraction needs, where some crops benefit from deeper architectures to capture finer details, while others require fewer layers due to simpler patterns in their images.

These design decisions were driven by tuning accuracy and visual inspection of loss and accuracy plots across epochs. The final hyperparameters were selected based on accuracy and generalization to the test set. No automated search methods were used due to computational constraints. A detailed Table 3 has been provided, consolidating the convolutional layer configurations of the CNN model. Training across all VGG-16 crop models was conducted for 10 epochs. It is noted that CNN and VGG-16 were not trained under identical preprocessing and training configurations. This design choice was intentional, as each model was optimized per crop to achieve best practical performance for deployment in the proposed mobile application. Consequently, the reported results should be interpreted as model-optimized performance in a realistic multi-crop deployment setting, rather than as a strict one-to-one architectural comparison.

### 4.2 Server layer (middle layer)

Figure 8 illustrates the working of a Python-based API, which serves as the backend for an Android application and middle layer for complete architecture. This API is developed entirely using the Flask framework and is designed to predict plant leaf diseases by

processing uploaded images. The development process of the API involves multiple key steps:

- Defining Crop Models and Classes: the first step is to establish crop-specific models along with their respective disease classification categories. This ensures that the system can accurately identify and categorize various plant diseases based on the uploaded images.
- Implementing the Prediction Function: a dedicated function is implemented to handle image preprocessing tasks. This function is responsible for resizing the image, normalizing pixel values, and ensuring the input format is compatible with the deployed CNN model used for prediction.
- Defining API Routes: the next step involves creating Flask routes to manage various HTTP (Hypertext Transfer Protocol) requests. These routes facilitate different functionalities, such as: (i) Uploading an image for disease prediction, (ii) Selecting the crop type to apply the relevant model, (iii) Receiving predictions based on the processed image.
- Error Handling Mechanisms: the API is designed to handle potential errors efficiently. This includes handling issues such as invalid image formats, missing inputs, server errors, or failed predictions, ensuring a smooth user experience.
- Deployment and Accessibility: the Flask application runs on a specified host and port, allowing seamless communication between the Android application and the backend. Clients (i.e., the Android app) can send requests to the server, which processes the image and returns disease prediction as a response. JavaScript Object Notation (JSON) is used for storing and exchanging data.

This structured API ensures a reliable and efficient system for plant disease detection, integrating deep learning with mobile-based user interaction, hence acting as a middleware.

### 4.3 Client layer (top layer)

In the final phase, an Android application was designed and developed to provide farmers with an intuitive interface for identifying crop diseases. The application allows users to capture images of crop leaves using their mobile cameras, select the type of crop from a predefined list, submit images for analysis, and receive disease predictions based on deep learning models. As previously illustrated in Fig. 5, the application workflow begins with the farmer taking a picture of the affected crop leaf using the built-in camera feature. The user then selects the crop type from a menu, ensuring the appropriate disease detection model is applied. The captured image is sent to a Flask-based Python API backend, where it undergoes preprocessing and is analyzed by a CNN model to predict possible plant diseases. Finally, the application receives the prediction results and displays the identified disease along with relevant information to assist the farmer in taking necessary actions.

#### *4.3.1 User-centric design & development*

The application was developed with a strong emphasis on user-friendliness and accessibility, ensuring that even farmers with minimal technical knowledge can use it with ease. The development process involved:

- Research & User Analysis: understanding the needs of farmers and the challenges they face in identifying plant diseases is a crucial aspect of the application's development. By recognizing these difficulties, the design process focuses on creating an intuitive interface that simplifies interactions and enhances usability. The goal is to ensure a seamless experience, allowing farmers to easily capture images, select crop types, and receive disease predictions without technical complexity.
- UI & User Experience (UX) Design: the UI and UX design focus on ensuring accessibility and ease of use. This is achieved through (i) simple navigation, where the interface follows a clean and minimalistic layout that allows users to move seamlessly between features, and (ii) integrated camera functionality, enabling users to capture images directly within the application, thereby reducing complexity.
- Continuous Testing & Quality Assurance: the application underwent rigorous testing phases to check its stability, performance, and accuracy. Issues related to image processing, API connectivity, and UI responsiveness were addressed to provide reliable experience.

The Android application is designed to be a practical tool for farmers, leveraging deep learning to provide disease detection for crops. By combining an intuitive mobile interface with powerful backend AI models, the application ensures that farmers receive quick and accurate insights to protect their crops effectively.

## 5 Results and discussion

The results present a comparative analysis of the VGG-16 and CNN models on the same dataset. A total of 10 different neural network models were designed, trained and tested, with each of the five crop types being analyzed using two distinct models. Figure 9 and Table 4 present the training accuracy results for both CNN and VGG-16 models across the five crop types. VGG-16 outperforms CNN for wheat, rice, and maize, achieving accuracies of 98.33%, 70.94%, and 95.74%, respectively, compared to CNN's 93.10%,

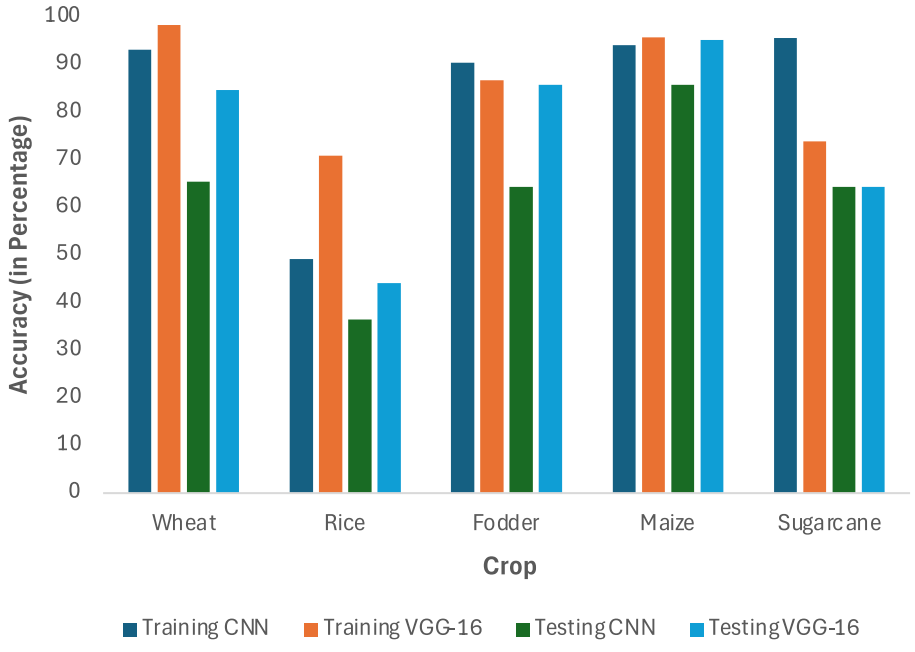



**Fig. 9** Comparison of training and testing accuracy

**Table 4** Comparison of training and testing accuracy

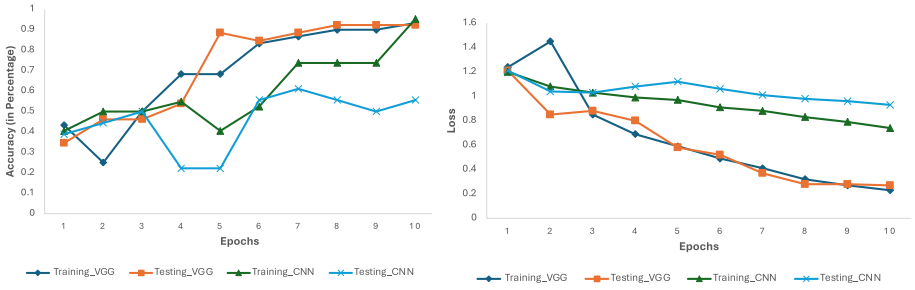
| Crop/Model | Training Accuracy (%) | | Testing Accuracy (%) | |
|---|---|---|---|---|
| | CNN | VGG-16 | CNN | VGG-16 |
| Wheat | 93.10 | 98.33 | 65.38 | 84.61 |
| Rice | 49.23 | 70.94 | 36.53 | 44.15 |
| Fodder | 90.47 | 86.66 | 64.28 | 85.71 |
| Maize | 94.11 | 95.74 | 85.71 | 95.23 |
| Sugarcane | 95.55 | 73.94 | 64.28 | 64.28 |

49.23%, and 94.11% for the same crops. In contrast, CNN demonstrates superior performance for fodder and sugarcane, achieving 90.47% and 95.55% respectively, while VGG-16 attains 86.66% and 73.84% for these crops. These results indicate that while deeper architectures such as VGG-16 excel in extracting complex features for certain crops, the lighter CNN model can sometimes achieve better fit during training, particularly for datasets with limited variation or smaller sample sizes. Additionally, Fig. 9 along with Table 4 present a comparison of the testing accuracy achieved by the CNN and VGG-16 models for each crop type. Overall, VGG-16 consistently outperforms CNN across most crops. For wheat, VGG-16 achieves an accuracy of 84.61%, compared to 65.38% for CNN. In rice classification, both models perform relatively lower than for other crops, with VGG-16 reaching 44.15% and CNN 36.53%. For fodder, VGG-16 attains 85.71%, surpassing CNN's 64.28%.

Maize classification shows high performance for both models, with VGG-16 at 95.23% and CNN at 85.71%. Sugarcane is the only crop where both models achieve the same accuracy of 64.28%. These results highlight VGG-16's advantage in most cases, particularly for crops with higher inter-class variability, while also indicating that performance for certain crops such as rice and sugarcane may benefit from further dataset expansion and model optimization. This suggests that while deeper architectures like VGG-16 generally provide better feature extraction capability, their advantage is reduced when data availability or disease feature distinctiveness is limited. The comparison of training loss between CNN and VGG-16 (presented in Appendix A) shows that VGG-16 generally achieves lower values for most crops, aligning with its higher training accuracy. For wheat, VGG-16 records a loss of 0.13 compared to CNN's 0.57, while for rice the gap is more pronounced, with VGG-16 at 0.89 versus CNN's 1.56. Similar trends appear for maize (0.12 vs. 0.15) and fodder (0.19 vs. 0.21). The only exception is sugarcane, where CNN achieves a lower training loss (0.42) than VGG-16 (0.61), indicating a better fit for this crop during training.

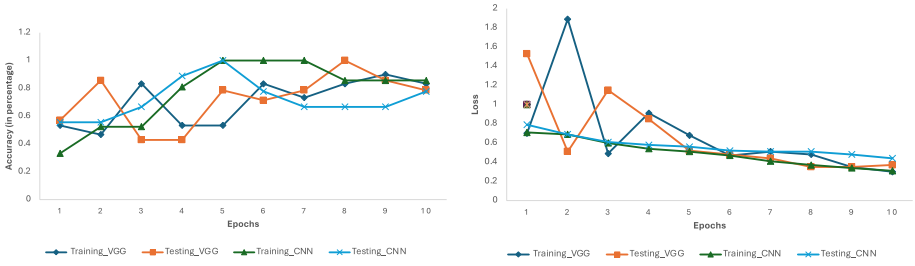
A similar pattern is observed in testing loss. VGG-16 outperforms CNN for most crops, with lower loss values for wheat (0.32 vs. 0.89), rice (1.36 vs. 1.78), fodder (0.26 vs. 0.66), and maize (0.12 vs. 0.52). Again, sugarcane is the exception, with CNN achieving a lower testing loss (0.82) compared to VGG-16 (1.09). These results suggest that while VGG-16 is generally more effective at minimizing both training and testing loss, CNN can still outperform for specific crops where the dataset characteristics or disease features are more compatible with a simpler architecture.

To gain a deeper understanding of how the CNN and VGG-16 models perform on each crop, we compare the training and testing accuracy and loss for both models. These metrics are plotted against the number of epochs to analyze the models' performance in




(a) Training and testing accuracy. (b) Training and testing loss.

**Fig. 10** Training and testing performance (wheat)

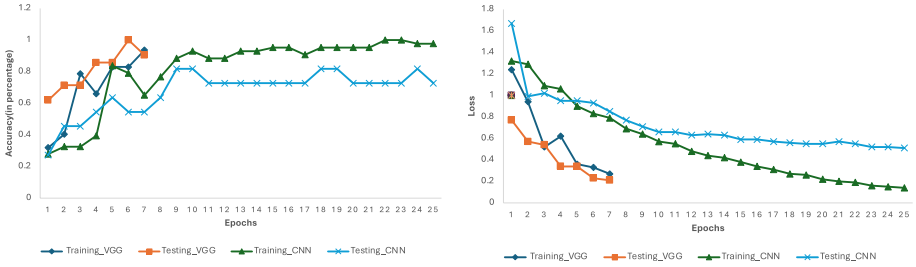



(a) Training and testing accuracy. (b) Training and testing loss.

**Fig. 11** Training and testing performance (fodder)

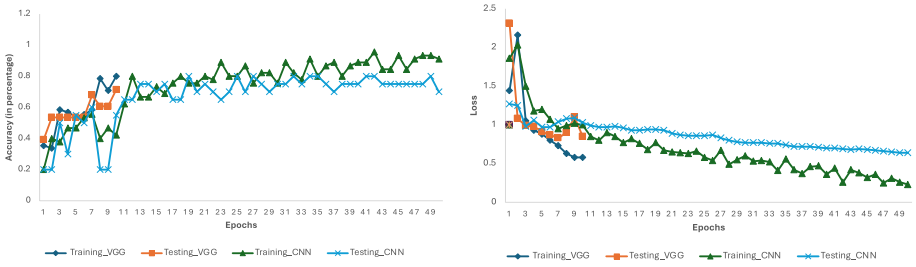
more detail. This allows not only a final accuracy comparison but also an assessment of model convergence speed, stability, and generalization patterns.

Figure 10a illustrates the comparison of training and testing accuracy for wheat across both models. The VGG-16 model shows consistent improvement over the epochs, with training accuracy increasing from 0.43 to 0.93 at epoch 10 and testing accuracy rising from 0.346 to 0.923 at epoch 8. The gap between training and testing accuracy remains relatively small for VGG-16, indicating strong generalization. In contrast, the CNN model demonstrates overall improvement in training accuracy, peaking at 0.952 at epoch 10. However, its testing accuracy fluctuates without a clear trend, indicating potential overfitting, as the model fails to generalize well to the testing dataset. Figure 10b compares the trends of training and testing loss as the number of epochs increases for wheat, across both models. In the case of the CNN model, there is a gradual decrease in both training and testing loss. However, for the VGG-16 model, the decrease is more pronounced, indicating a faster pace of learning or a higher learning rate. The VGG-16 loss curve is also smoother and more stable, whereas CNN's loss shows mild fluctuations, suggesting slightly less stable optimization. Figures 11a and b compare the training and testing accuracy and loss for fodder across both models. It is evident that, except for the training data in the CNN model, all other values fluctuate without a clear trend, which suggests possible overfitting. However, as the number of epochs increases, the trend becomes more consistent, showing a clearer pattern of improvement. The fluctuations in early epochs for VGG-16 testing accuracy point to instability in feature learning, likely due to limited dataset size for this crop. Loss curves show that VGG-16 converges faster, but CNN maintains a smaller gap between training and testing loss, indicating slightly better generalization in the earlier epochs.




(a) Training and testing accuracy. (b) Training and testing loss.

**Fig. 12** Training and testing performance (maize)



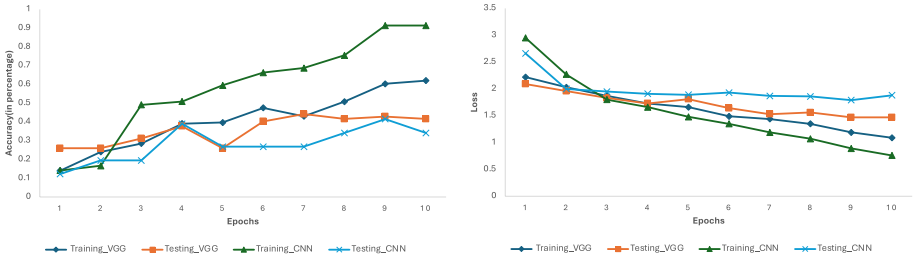

(a) Training and testing accuracy. (b) Training and testing loss.

**Fig. 13** Training and testing performance (sugarcane)

Figures 12a and b compare the training and testing accuracy and loss for maize across both models. The VGG-16 model was only trained for 7 epochs, as early stopping was implemented to prevent overfitting. This early plateau in accuracy, coupled with a consistently low loss, indicates that VGG-16 learned the maize disease features very quickly and additional epochs were unnecessary. CNN, on the other hand, required more epochs to approach similar training accuracy and showed a more gradual convergence rate.

Figures 13a and b show the comparison of training and testing accuracy and loss of Sugarcane for both the models. It can be analysed from the graphs that there is gradual increase in the values, and the model is not overfitting with testing accuracy for VGG-16 increasing from 0.393 to 0.714 at epoch 7 and for CNN 0.8 at epoch 49. The fact that CNN required 49 epochs to reach its best performance, compared to VGG-16's 7, suggests a slower learning rate but potentially better adaptation to this dataset. Loss trends confirm this CNN loss decreases steadily and remains lower than VGG-16's, supporting its competitive performance for sugarcane.

Figures 14a and b compare the training and testing accuracy and loss for rice across both models. The graphs show a gradual increase in accuracy, indicating that the models are not overfitting. For VGG-16, the testing accuracy improves from 0.26 to 0.416 by epoch 10, while for CNN, it reaches 0.341 at the same epoch. Both models show relatively low accuracies, pointing towards underfitting due to the complexity of rice disease classes and the relatively small sample size per class. Loss curves confirm a slow but steady improvement, with VGG-16 loss declining at a faster rate than CNN, though both remain relatively high compared to other crops. For all experiments, a maximum epoch limit was predefined (50 epochs for CNN and 10 epochs for VGG-16). However, the actual number of epochs completed depended on whether the model met the early stopping criterion, which was introduced to prevent overfitting and unnecessary




(a) Training and testing accuracy. (b) Training and testing loss.

**Fig. 14** Training and testing performance (rice)

**Table 5** Training configuration and convergence details

| Crop | Model | Max Epochs | Actual Epochs | Early Stopping |
|---|---|---|---|---|
| Wheat | CNN | 50 | 10 | Yes |
| | VGG-16 | 10 | 10 | No |
| Fodder | CNN | 50 | 10 | Yes |
| | VGG-16 | 10 | 10 | No |
| Maize | CNN | 50 | 25 | Yes |
| | VGG-16 | 10 | 7 | Yes |
| Sugarcane | CNN | 50 | 49 | Yes |
| | VGG-16 | 10 | 7 | Yes |
| Rice | CNN | 50 | 10 | Yes |
| | VGG-16 | 10 | 10 | No |

computation. For Wheat, Fodder, and Rice, the VGG-16 model completed the full 10 epochs, as no early stopping was triggered and performance continued to improve steadily. For Maize and Sugarcane, VGG-16 converged rapidly, therefore, early stopping terminated training at epoch 7, performance plateaued. The CNN model, having fewer parameters and slower convergence, required more epochs in several cases. For example, it trained for 49 epochs on Sugarcane and 25 epochs on Maize, whereas it converged earlier for Wheat, Fodder, and Rice. Early stopping was enabled for all CNN models and selectively for VGG-16 models where early convergence was observed. Augmentation was applied consistently across all crops and for all models. Table 5 summarizes the optimized parameters for each crop and each model.

By analyzing these factors, it is evident that in terms of training accuracy, VGG-16 outperformed CNN for wheat, rice, and maize, achieving accuracies of 98.33%, 70.94%, and 95.74%, respectively. Meanwhile, CNN performed better for fodder (90.47%) and sugarcane (95.55%). Regarding testing accuracy, VGG-16 demonstrated superior performance across all crops except sugarcane, with accuracies of 84.61% for wheat, 44.15% for rice, 85.71% for fodder, and 95.23% for maize. For sugarcane, both models achieved the same testing accuracy of 64.28%. A detailed per-class performance analysis was carried out for both Rice (Table 6) and Sugarcane (Table 7) models to better understand the factors contributing to their relatively lower top-1 VGG accuracies (Rice: 44.15%; Sugarcane: 64.28%). To gain deeper insight into class-wise behavior, Precision, Recall, F1-score, and Overall Accuracy were computed. The evaluation metrics are defined in Eqs. 1 to 4. True Positive (TP) refers to cases where the model correctly predicts a sample as belonging to a given class, while True Negative (TN) denotes instances where the model correctly identifies a sample as not belonging to that class. In contrast, False Positive (FP) occurs

**Table 6** Class-wise performance metrics for Rice (VGG-16)

| Class | Precision | Recall | F1-score | Overall Accuracy |
|---|---|---|---|---|
| Bacterial Leaf Blight | 0.25 | 0.222 | 0.235 | 0.822 |
| Brown Spot | 0.8 | 0.8 | 0.8 | 0.973 |
| Healthy | 0.667 | 0.545 | 0.6 | 0.890 |
| Leaf Blast | 0.25 | 0.083 | 0.125 | 0.808 |
| Leaf Scald | 0.222 | 0.667 | 0.333 | 0.781 |
| Narrow Brown Leaf Spot | 0 | 0 | 0 | 0.849 |
| Rice Hispa | 0.615 | 0.8 | 0.696 | 0.904 |
| Sheath Blight | 0.312 | 0.555 | 0.4 | 0.794 |
| accuracy | | | 0.411 | - |
| macro avg | 0.389 | 0.459 | 0.399 | - |

**Table 7** Class-wise performance metrics for Sugarcane (VGG-16)

| Class | Precision | Recall | F1-score | Overall Accuracy |
|---|---|---|---|---|
| Bacterial Blight | 0.75 | 0.818 | 0.783 | 0.821 |
| Healthy | 0.667 | 0.25 | 0.364 | 0.75 |
| Red Rot | 0.461 | 0.667 | 0.545 | 0.643 |
| accuracy | | | 0.607 | - |
| macro avg | 0.626 | 0.578 | 0.564 | - |

when the model incorrectly assigns a sample to a class it does not belong to, and False Negative (FN) represents situations where the model fails to identify a sample that actually belongs to the class.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1}$$

It measures how many of the samples predicted as a given class are actually correct.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

It measures how many of the actual samples belonging to a given class are correctly identified by the model.

$$\text{F1-score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

It is the harmonic mean of precision and recall.

$$\text{Overall Accuracy} = \frac{TP + TN}{FP + FN + TP + TN} \tag{4}$$

It measures how often the model is correct when considering both correct detections of the class and correct rejections of other classes. While this metric can sometimes appear artificially high in datasets containing rare classes, due to the dominant contribution of true negatives when a model fails to predict a rare class, this limitation is unlikely to affect the present study since the dataset is class-balanced across all classes.

For rice, the results show substantial variation across disease categories. Classes such as Brown Spot, Rice Hispa, and Healthy performed comparatively well. In contrast, Leaf Blast, Bacterial Leaf Blight, and Narrow Brown Leaf Spot, despite a relatively high overall accuracy, showed major confusion. The confusion primarily arises among visually

similar lesions, particularly between Leaf Blast, Leaf Scald, and Bacterial Leaf Blight. These classes share overlapping color/shape patterns (irregular spots, elongated lesions), which becomes challenging for the model under field-like variations. For sugarcane, the model performance was relatively stable but still limited by smaller class sizes. Bacterial Blight achieved the strongest performance, whereas Healthy and Red Rot remained challenging. The low recall for Healthy indicates frequent misidentification of normal discoloration or lighting artefacts as disease. Overall, the rice model suffers predominantly from inter-class visual similarity and intra-class variability, while the sugarcane model is affected by limited sample availability and subtle symptom presentation.

To further mitigate the limitation of small dataset and strengthen the robustness and credibility of the evaluation, RSKF cross-validation was implemented for both CNN and VGG-16 models across all crops. The resulting mean accuracy and standard deviation, presented in Table 8, provide a more reliable, and statistically robust estimate of model performance.

The Grad-CAM (Gradient-weighted Class Activation Mapping) visualizations in Fig. 15 provide insight into the internal feature reasoning of the VGG-16 model by qualitative visualization of the spatial regions that contribute most strongly to the final class activation. Across the samples, the heatmaps show that the model allocates high activation weights to lesion dense areas of the leaf surface, which indicates that the model has learned to associate localized perturbations, chromatic shifts, and structural discontinuities with disease classes. The highlighted regions correspond closely with the pathological patterns normally used by human experts during field diagnosis, which strengthens the reliability of the learned feature representations. For maize and fodder, the activation concentrates on sharply bordered necrotic patches and chlorotic streaks. This focused response is consistent with the high testing accuracy of VGG-16 for these crops and suggests that the network has developed discriminative mid level filters that respond to lesion geometry and color gradients. The activation patterns also show minimal spillover into background vegetation, indicating effective suppression of irrelevant context.

However, the sugarcane and rice samples exhibit broader and more diffuse activation across the leaf blade. This dispersion reflects the lower separability of disease features within these datasets, where the symptom morphology is too subtle and overlaps across the classes. The visual patterns therefore explain the reduced testing accuracy and higher loss values observed earlier, since the network attends to a wider region due to weaker class specific signal strength. The Grad-CAM outputs thus provide a qualitative confirmation of the quantitative performance differences.

**Table 8** Repeated Stratified K-Fold cross-validation results

| Crop | Model | Folds × Repeats | Mean Accuracy | Standard Deviation |
|---|---|---|---|---|
| Fodder | VGG-16 | $5 \times 3$ | 0.9546 | 0.0689 |
| | CNN | $5 \times 3$ | 0.9074 | 0.1083 |
| Maize | VGG-16 | $5 \times 3$ | 0.9044 | 0.0938 |
| | CNN | $5 \times 3$ | 0.7974 | 0.1076 |
| Wheat | VGG-16 | $5 \times 3$ | 0.9111 | 0.0667 |
| | CNN | $5 \times 3$ | 0.7630 | 0.1551 |
| Sugarcane | VGG-16 | $5 \times 3$ | 0.8370 | 0.0798 |
| | CNN | $5 \times 3$ | 0.7481 | 0.1198 |
| Rice | VGG-16 | $5 \times 3$ | 0.4500 | 0.0503 |
| | CNN | $5 \times 3$ | 0.3514 | 0.0833 |

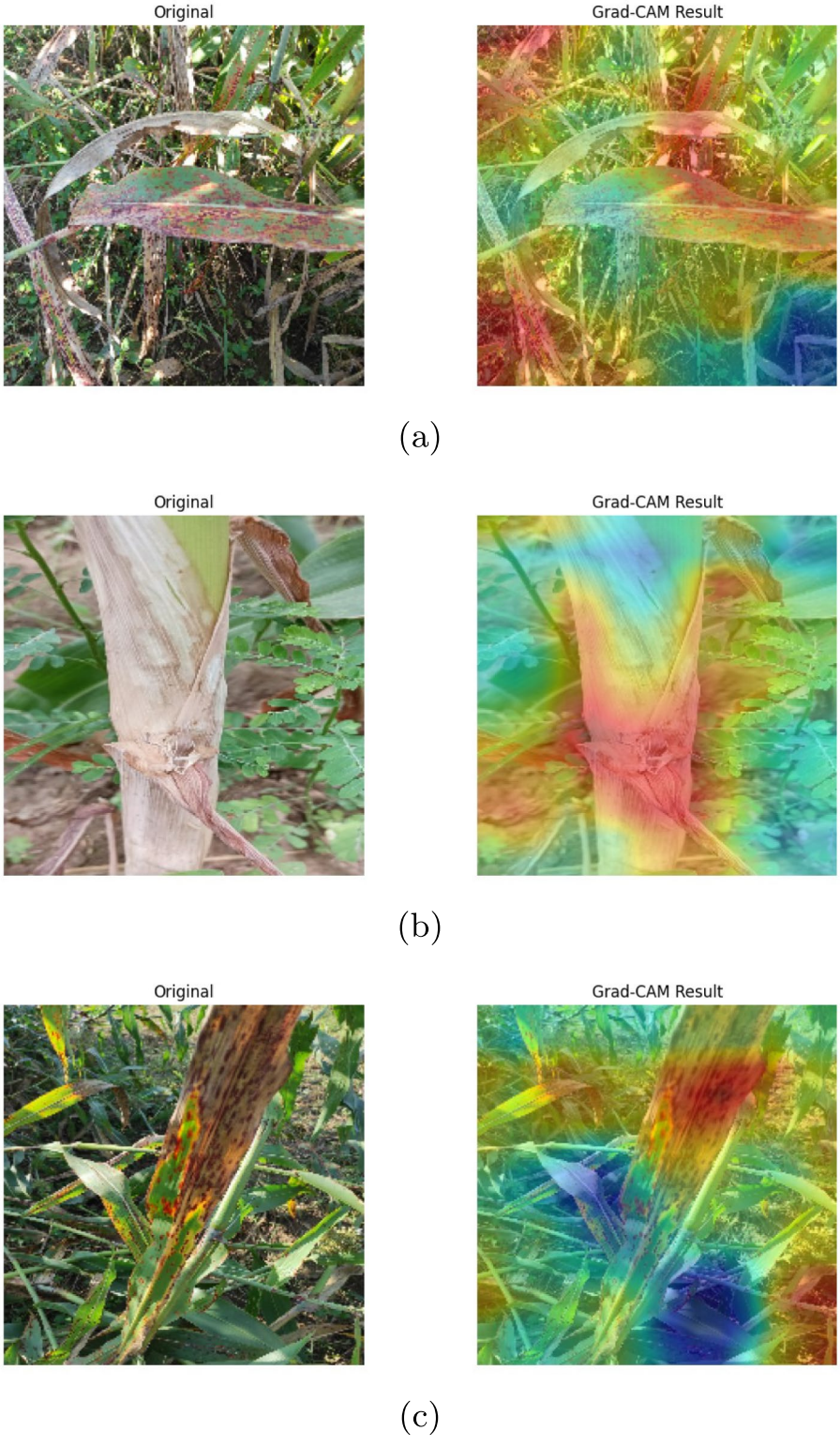
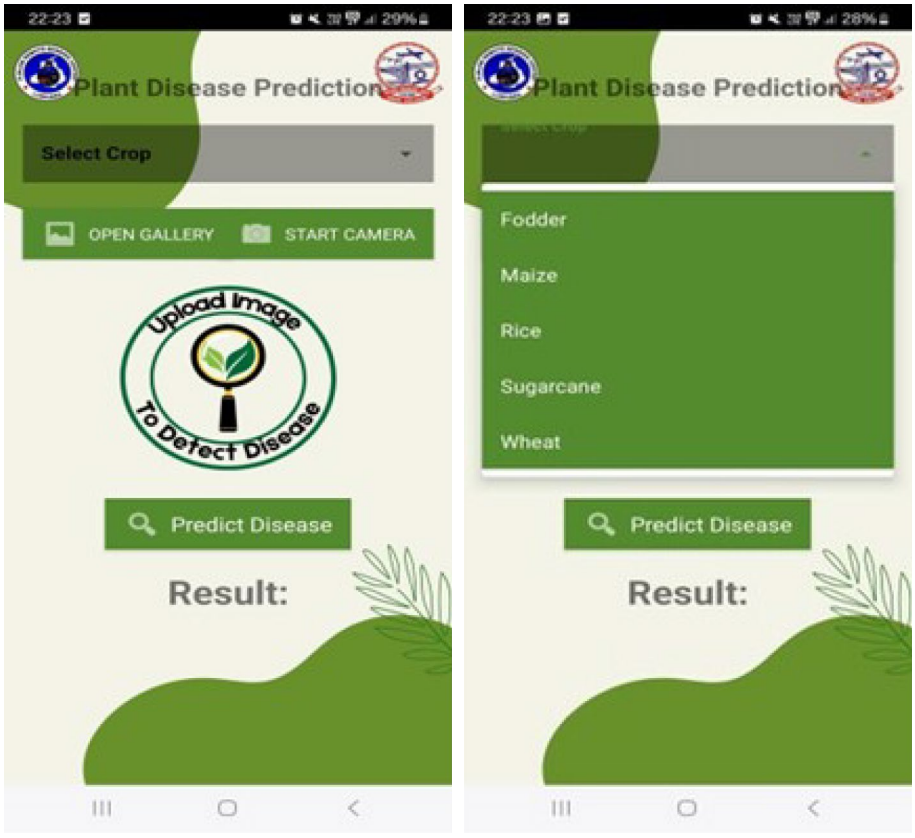



**Fig. 15** Grad-CAM visualizations showing how the model focuses on disease lesions rather than background regions

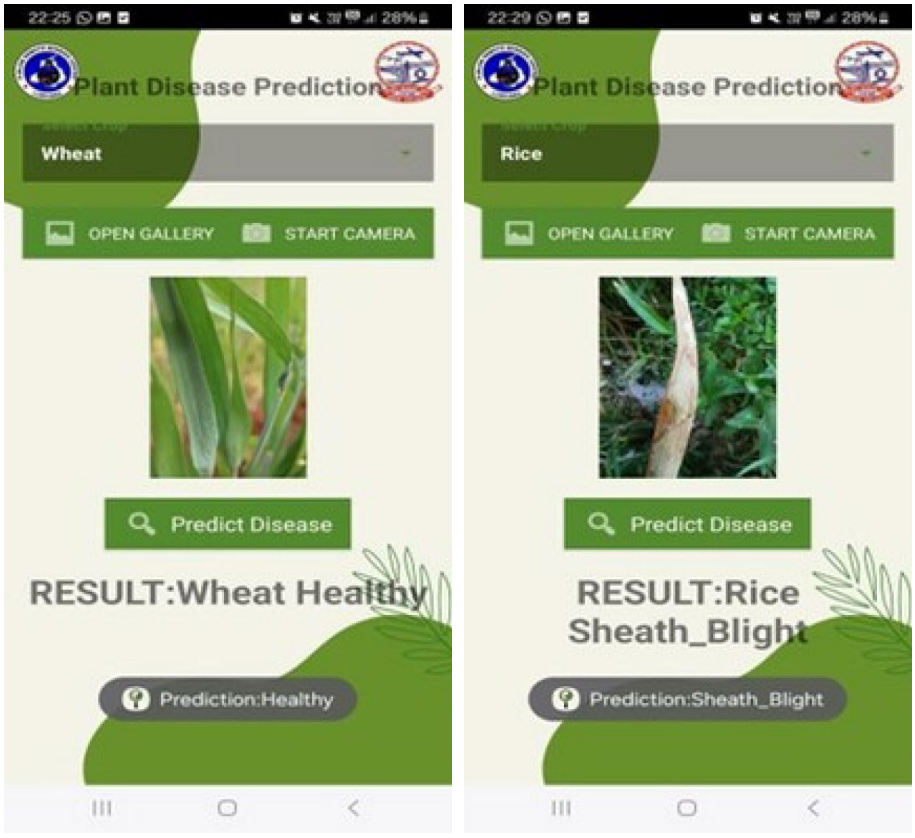
After integrating the model with the Android application using the methodology outlined in Sect. 4, the application was successfully developed and made ready for use. Figures 16a and 16b showcase the user interface of the application. Figure 16a illustrates the options available for image selection, allowing users to either choose an existing image from their device using the “Open Gallery" button or capture a real-time image using the “Start Camera" button for disease prediction. Additionally, Fig. 16b displays a drop-down menu that enables users to select the specific crop for analysis, ensuring accurate disease identification. Figure 17 illustrates the functionality of the Android application. The user uploads an image of a crop leaf and selects the corresponding crop type for analysis. Figure 17a displays the prediction for a wheat crop, where the results indicate that the crop is healthy. In contrast, Fig. 17b shows the prediction for a rice leaf, identifying the disease as sheath blight under the results section. This demonstrates the app’s ability to analyze and classify plant health conditions accurately based on the uploaded images.

Given the hardware configuration and the models deployed through a lightweight Python- Flask backend and accessed via a Flutter-based mobile application, the proposed framework processes a single image within 0.5-3 s. This duration covers image uploading, resizing, normalization, model inference using the pre-trained.h5 CNN model, and returning the prediction result. In addition, the user’s internet connection speed is a key factor influencing the total response time. Even with a CPU-based server, the effective




(a) Application Interface (b) Select Crop Drop Down

**Fig. 16** User Interface of Android application




(a) Healthy crop (b) Infected Disease

**Fig. 17** Results of prediction on application

end-to-end delay remained consistently below 5 s during testing under normal network conditions, ensuring near real-time prediction capability. The mobile application applies moderate client-side bitmap compression to images, ensuring efficient bandwidth usage while preserving sufficient image quality for accurate prediction. Multiple security measures have been implemented to safeguard user privacy. All communication between the mobile application and the server occurs over HTTPS (Hypertext Transfer Protocol Secure) with SSL (Secure Sockets Layer) encryption, ensuring secure data transmission. The backend server is further protected by an active firewall and antivirus, minimizing vulnerability to external threats.

The framework was examined through field-captured images uploaded via the mobile application. Five users were asked to test on around 2–5 field images across different crops. Testing was performed under live field conditions and across multiple sessions, therefore, the exact number of images evaluated via the application was not formally logged. These images were collected under natural field conditions with variations in lighting, background and device, providing a test environment beyond the controlled training dataset. For crops that were not in season during the survey period, validation was performed using the mobile application's "open gallery" option with previously stored field images that had been captured when those crops were in season. The models demonstrated consistent performance in this real-world setting, indicating a satisfactory level of generalization. In particular, strong performance was observed for Wheat,

Maize, and Fodder, with the Fodder model correctly identifying all tested labels. Nevertheless, the users commended the application in producing accurate results, and suggested improvement in terms of latency of the camera.

## 6 Conclusion and future scope

This research emphasizes the importance of disease detection across various crops, tackling a major challenge farmers face in detecting and managing plant diseases. The study proposed a framework which is structured into three key components that are modular: a deep learning model as the backend, an API layer as middleware, and an Android application as the frontend. The first component is for detection and classification of crop diseases from images. For our demonstration, we rely on deep learning models, that utilizes CNN and VGG-16 models to train and test datasets of crop leaf images, enabling accurate detection of plant diseases. The API layer acts as a bridge, facilitating seamless communication between the backend model and the Android application. The Android application serves as an intuitive interface, allowing users to capture crop leaf images, analyze them, and receive real time predictions. Through the evaluation of deep learning models on multiple crop types, the study assesses the models' generalization capability across distinct visual domains. The framework achieves accuracies of 84.61% for wheat, 44.15% for rice, 85.71% for fodder, 95.23% for maize, and 64.28% for sugarcane (testing accuracy of the best-performing model per crop, where VGG-16 demonstrated superior generalization).

While the framework has demonstrated its usability, few limitation of this research are that publicly available disease specific datasets for many crops are scarce, and certain diseases are less prevalent in accessible field locations. As a result, the dataset size for some classes remains small, which may limit the model's ability to generalize effectively to broader real world conditions. Though in this study we explored crossvalidation strategy, more recent models require large sizes of dataset for classification purposes. The creation of comprehensive and balanced datasets further requires substantial resources, including time, expertise, and controlled field conditions. Moreover, the practical deployment of the mobile application depends on stable internet connectivity and cellular network coverage, which can restrict its usability in remote or low-connectivity farming regions.

Therefore, future studies should work on integrating attention mechanisms that could enhance the model by focusing on key image areas for improved predictions. Furthermore, expanding the dataset using generative data augmentation [48, 49] and to include additional crops and diseases would increase the model's adaptability. In future work, a unified modelling strategy using a shared backbone with single joint crop disease classifier, can be explored to compare its effectiveness with the current multi-model deployment approach. Potential extensions, including controlled ablation studies examining the impact of different VGG-16 fine-tuning strategies such as frozen versus partial unfreezing, have been identified as important directions for future work, while the present study focuses on establishing optimal, deployment-ready model performance and minimizing overfitting risks. As a next step, the proposed model can be externally validated using datasets collected from different agricultural fields, across multiple cropping seasons, and captured using diverse imaging devices, to further strengthen evidence of real-world robustness and generalization capability. Moreover, secondary research methods should

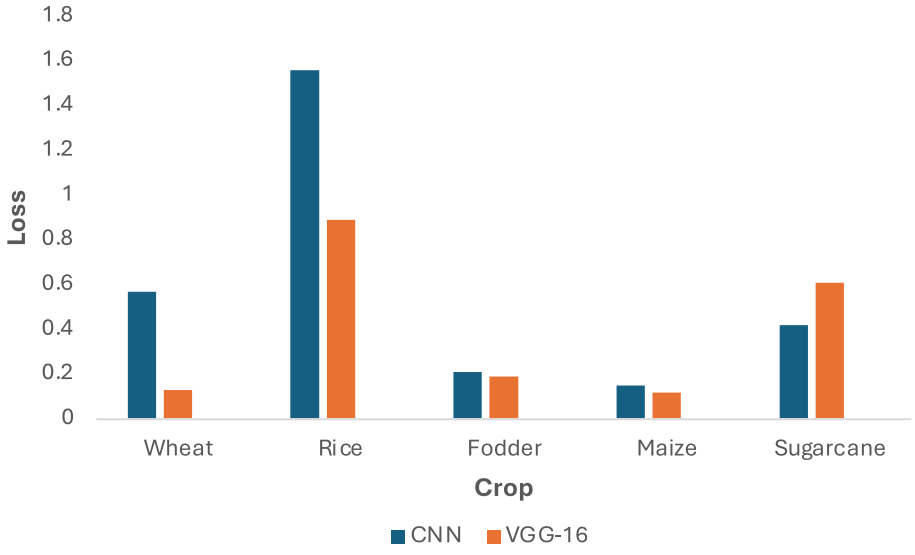



**Fig. 18** Comparison of training loss

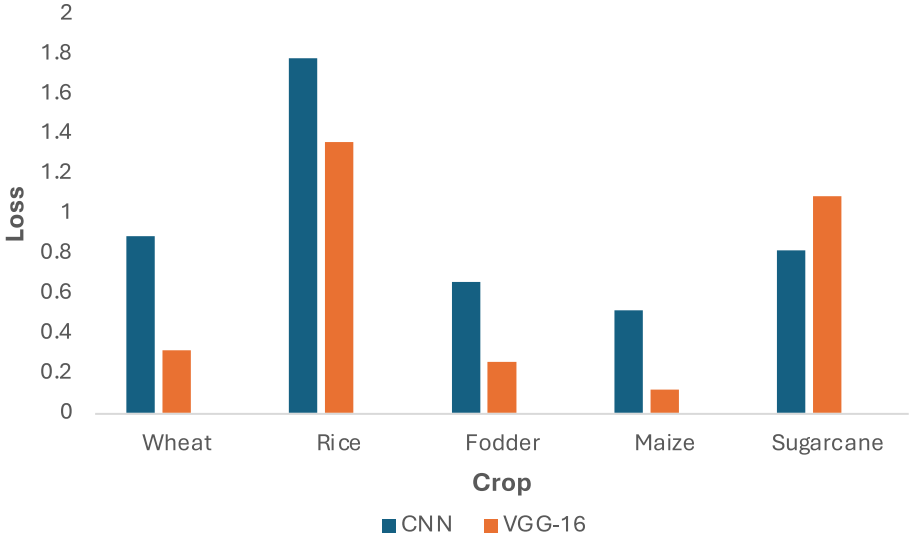



**Fig. 19** Comparison of testing loss

be explored to test the user experience of the usability of mobile application (including other mobile operating systems) enabling a more targeted benefiting for the farming community.

## Appendix A

Appendix A reports per–crop *training loss* for CNN and VGG-16, shown as a summary plot (Fig. 18) and a table with final epoch values (Table 10). The corresponding *testing loss* (Fig. 19, Table 9). Lower loss indicates better fit. Additionally, Fig. 20 presents the confusion matrices for all crops.

See Tables 9, 10, 11.

**Table 9** Testing loss

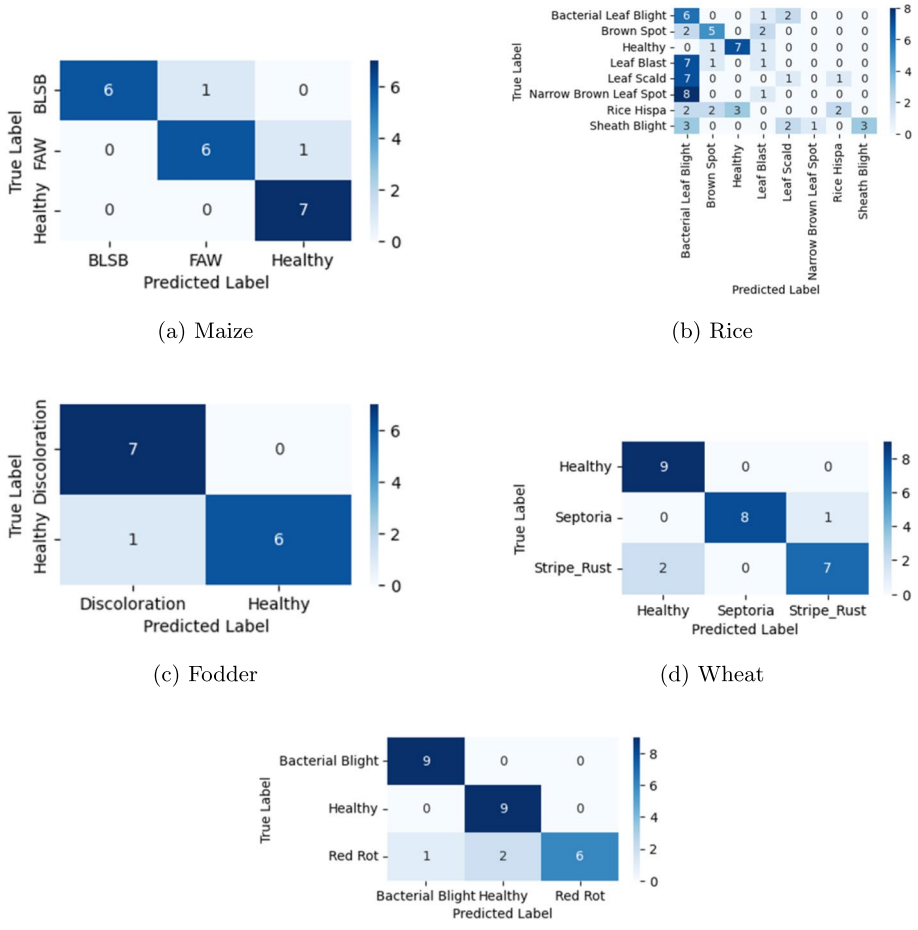
| **Testing Loss** | | |
|---|---|---|
| **Crop/Model** | **CNN** | **VGG-16** |
| Wheat | 0.89 | 0.32 |
| Rice | 1.78 | 1.36 |
| Fodder | 0.66 | 0.26 |
| Maize | 0.52 | 0.12 |
| Sugarcane | 0.82 | 1.09 |

**Table 10** Training loss

| **Training Loss** | | |
|---|---|---|
| **Crop/Model** | **CNN** | **VGG-16** |
| Wheat | 0.57 | 0.13 |
| Rice | 1.56 | 0.89 |
| Fodder | 0.21 | 0.19 |
| Maize | 0.15 | 0.12 |
| Sugarcane | 0.42 | 0.61 |

**Table 11** Classification reports for CNN and VGG across all crops (rounded to 2 decimals)

| **Crop** | **Class** | **CNN** | | | **VGG** | | |
|---|---|---|---|---|---|---|---|
| | | **precision** | **recall** | **F1 score** | **precision** | **recall** | **F1 score** |
| Maize | BLSB | 1.00 | 0.40 | 0.57 | 1.00 | 0.86 | 0.92 |
| | FAW | 1.00 | 1.00 | 1.00 | 0.86 | 1.00 | 0.92 |
| | Healthy | 0.50 | 1.00 | 0.67 | 1.00 | 1.00 | 1.00 |
| | **macro avg** | 0.83 | 0.80 | 0.75 | 0.95 | 0.95 | 0.95 |
| Sugarcane | Bacterial Blight | 0.57 | 0.67 | 0.62 | 0.75 | 0.82 | 0.78 |
| | Healthy | 1.00 | 0.70 | 0.82 | 0.67 | 0.25 | 0.36 |
| | Red Rot | 0.67 | 1.00 | 0.80 | 0.46 | 0.67 | 0.55 |
| | **macro avg** | 0.75 | 0.79 | 0.75 | 0.63 | 0.58 | 0.56 |
| Wheat | Healthy | 0.58 | 1.00 | 0.74 | 1.00 | 0.56 | 0.71 |
| | Septoria | 0.67 | 1.00 | 0.80 | 1.00 | 0.80 | 0.89 |
| | Stripe Rust | 0.00 | 0.00 | 0.00 | 0.71 | 1.00 | 0.83 |
| | **macro avg** | 0.42 | 0.67 | 0.51 | 0.90 | 0.79 | 0.81 |
| Fodder | discoloration | 0.83 | 1.00 | 0.91 | 1.00 | 0.83 | 0.91 |
| | healthy | 1.00 | 0.75 | 0.86 | 0.89 | 1.00 | 0.94 |
| | **macro avg** | 0.92 | 0.88 | 0.88 | 0.94 | 0.92 | 0.93 |
| Rice | Bacterial Leaf Blight | 0.00 | 0.00 | 0.00 | 0.25 | 0.22 | 0.24 |
| | Brown Spot | 1.00 | 0.25 | 0.40 | 0.80 | 0.80 | 0.80 |
| | Healthy | 0.43 | 0.75 | 0.55 | 0.67 | 0.55 | 0.60 |
| | Leaf Blast | 0.00 | 0.00 | 0.00 | 0.25 | 0.08 | 0.13 |
| | Leaf Scald | 0.17 | 0.33 | 0.22 | 0.22 | 0.67 | 0.33 |
| | Narrow Brown Leaf Spot | 0.18 | 0.50 | 0.27 | 0.00 | 0.00 | 0.00 |
| | Rice Hispa | 0.00 | 0.00 | 0.00 | 0.62 | 0.80 | 0.70 |
| | Sheath Blight | 1.00 | 0.17 | 0.29 | 0.31 | 0.56 | 0.40 |
| | **macro avg** | 0.35 | 0.25 | 0.22 | 0.38 | 0.46 | 0.40 |




(a) Maize (b) Rice

(c) Fodder (d) Wheat

(e) Sugarcane

**Fig. 20** Confusion matrices for different crop classes

**Acknowledgements**
The authors would like to thank Punjab Agriculture University to conduct experiments and occupy part of its land for conducting the experiments.

**Author Contributions**
Aarti Kochhar: Conceptualization, Methodology, Software development, Data curation, Writing – original draft, Formal analysis, Validation, Visualization, Writing – review & editing, Project Administration. Gagan Narang: Conceptualization, Methodology, Software development, Formal analysis, Validation, Writing – original draft, Writing – review & editing, Project Administration. Shashikant Patel: Software, Resources, Supervision and Project Administration Harleen Kaur: Formal Analysis, Investigation, Data Curation, Writing-original Draft Chetan Singla: Conceptualization, Resources Brijendra Pateriya: Resources, Supervision and Project Administration Alessandro Galdelli: Methodology, Software development, Supervision, Visualization, Writing – review & editing

**Funding**
No funding to declare.

**Availability of data and materials:**
The data that support the findings of this study are partly available from public repositories (rice, sugarcane, and wheat datasets from Kaggle, see [32–34]). Restrictions apply to the availability of the field collected fodder and maize datasets, which were generated by the authors at Punjab Agricultural University, and are therefore not publicly available. These datasets are available from the corresponding author upon reasonable request.

## Declarations

**Ethical approval**
Not applicable. The study does not involve any clinical trials or human or animal subjects.

**Consent to participate**
Not applicable.

**Consent to publish**
Not applicable.

Received: 12 September 2025 / Accepted: 18 February 2026
Published online: 16 March 2026

**References**

1. United Nations Department of Economic and Social Affairs, Population Division: World population prospects 2024: Summary of results (un desa/pop/2024/tr/no. 9). Technical report, United Nations, New York. 2024.
2. Kochhar A, Kumar N. Path loss-based distance vectored (pldv) routing protocol for greenhouse monitoring using wireless sensor networks. Int J Internet Protoc Technol. 2024;17(2):53–70.
3. Food and Agriculture Organization of the United Nations: The plants that feed the world. https://www.fao.org/resources/infographics/infographics-details/en/c/1515349/. 2025. Accessed 02 April 2025.
4. Narang G, Galdelli A, Pietrini R, Solfanelli F, Mancini A. A data collection framework for precision agriculture: Addressing data gaps and overlapping areas with iot and artificial intelligence. In: 2024 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor). 2024:580–585. IEEE.
5. Narang G, Galdelli A, Tassetti AN, Olivotto I, Zarantoniello M, Mancini A. Precision aquaculture framework for remote mussel growth monitoring with iot blockchain and cloud integration. In: 2024 IEEE International Workshop on Metrology for the Sea; Learning to Measure Sea Health Parameters (MetroSea). 2024:121–126. IEEE.
6. Savary S, Ficke A, Aubertot JN, Hollier C. Crop losses due to diseases and their implications for global food production losses and food security. Food Security. 2012;4(4):519–37. https://doi.org/10.1007/s12571-012-0200-5.
7. Mahlein A-K. Plant disease detection by imaging sensors: parallels and specific demands for precision agriculture and plant phenotyping. Plant Dis. 2016;100(2):241–51. https://doi.org/10.1094/PDIS-03-15-0340-FE.
8. Ristaino JB, Anderson PK, Bebber DP, Brauman KA. The persistent threat of emerging plant disease pandemics to global food security. Proc Natl Acad Sci. 2021;118(23):2022239118. https://doi.org/10.1073/pnas.2022239118.
9. Allen: Why are Cocoa Prices Falling? | J.P. Morgan Global Research. Technical report, J.P. Morgan Global Research (May) 2025. https://www.jpmorgan.com/insights/global-research/commodities/cocoa-prices.
10. Girmaw DW, Muluneh TW. Deep convolutional neural network model for classifying common bean leaf diseases. Discover Artificial Intelligence. 2024;4(1):96.
11. Food and Agriculture Organization of the United Nations: The State of Food and Agriculture 2019. FAO, Rome. 2019. https://doi.org/10.18356/63e608ce-en
12. Mohanty SP, Hughes D, Salathé M. Using deep learning for image-based plant disease detection. Front Plant Sci. 2016;7:1419. https://doi.org/10.3389/fpls.2016.01419.
13. Ngugi HN, Ezugwu AE, Akinyelu AA, Abualigah L. Revolutionizing crop disease detection with computational deep learning: a comprehensive review. Environ Monit Assess. 2024;196(3):302. https://doi.org/10.1007/s10661-024-12454-z.
14. Upadhyay A, Chandel NS, Singh KP, Chakraborty SK. Deep learning and computer vision in plant disease detection: A review. Artif Intell Rev. 2025;58(3):1–64. https://doi.org/10.1007/s10462-024-11100-x.
15. Bouacida I, Farou B, Djakhdjakha L, Seridi H, Kurulay MF. Innovative deep learning approach for cross-crop plant disease detection: a generalized method for identifying unhealthy leaves. Information Processing in Agriculture. 2025;12(1):54–67. https://doi.org/10.1016/j.inpa.2024.03.002.
16. Shaikh TA, Rasool T, Lone FR. Towards leveraging machine learning in agriculture and smart farming. Comput Electron Agric. 2022;198:107119. https://doi.org/10.1016/j.compag.2022.107119.
17. Mohyuddin G, Khan MA, Haseeb A, Mahpara S, Waseem M, Saleh AM. Evaluation of machine learning approaches for precision farming in smart agriculture system: a comprehensive review. IEEE Access. 2024;12:60155–84. https://doi.org/10.1109/ACCESS.2024.3390581.
18. Hasteer N, Mallik A, Nigam D, Sindhwani R, Van Belle S. Ai assurance and trustworthy ai: Implications for safety-critical and high-stakes systems in the agriculture sector. International Journal of System Assurance Engineering and Management. 2024;15(5):1841–60.
19. Ayyappan AB, Gobinath T, Kumar M, Sivaramakrishnan A. Rice plant disease detection using convolutional neural networks. Discover Artificial Intelligence. 2025;5(1):50.
20. Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2015. arXiv:1409.1556.
21. Ahmad A, Saraswat D, El Gamal A. A survey on using deep learning techniques for plant disease diagnosis and recommendations for development of appropriate tools. Smart Agricultural Technology. 2023;3:100083. https://doi.org/10.1016/j.atech.2022.100083.
22. Rautaray S, Pandey M, Gourisaria M, Sharma R, Das S. Paddy crop disease prediction- a transfer learning technique. International Journal of Recent Technology and Engineering (IJRTE). 2020;8:1490–5. https://doi.org/10.35940/ijrte.F7782.038620.
23. Agarwal M, Gupta SK, Biswas KK. Development of efficient cnn model for tomato crop disease identification. Sustainable Computing Informatics and Systems. 2020;28:100407. https://doi.org/10.1016/j.suscom.2020.100407.
24. Li L, Zhang S, Wang B. Plant disease detection and classification by deep learning-a review. IEEE Access. 2021;9:56683–98. https://doi.org/10.1109/ACCESS.2021.3069646.
25. Harakannanavar SS, Rudagi JM, Puranikmath VI, Siddiqua A, Pramodhini R. Plant leaf disease detection using computer vision and machine learning algorithms. Global Transitions Proceedings. 2022;3(1):305–10. International Conference on Intelligent Engineering Approach(ICIEA-2022). https://doi.org/10.1016/j.gltp.2022.03.016.
26. Sujatha R, Chatterjee JM, Jhanjhi N, Brohi SN. Performance of deep learning vs machine learning in plant leaf disease detection. Microprocess Microsyst. 2021;80:103615. https://doi.org/10.1016/j.micpro.2020.103615.
27. Salam A, Naznine M, Jahan N, Nahid E, Nahiduzzaman M, Chowdhury MEH. Mulberry leaf disease detection using cnn-based smart android application. IEEE Access. 2024;12:83575–88. https://doi.org/10.1109/ACCESS.2024.3407153.
28. Iftikhar M, Kandhro IA, Kausar N, Kehar A, Uddin M, Dandoush A. Plant disease management: a fine-tuned enhanced cnn approach with mobile app integration for early detection and classification. Artif Intell Rev. 2024;57(7):167. https://doi.org/10.1007/s10462-024-10809-z.

29. Yilmaz E, Bocekci SC, Safak C, Yildiz K. Advancements in smart agriculture: A systematic literature review on state-of-the-art plant disease detection with computer vision. IET Comput Vision. 2025;19(1):70004.
30. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision. 2017:618–626.
31. Galdelli A, Narang G, Pietrini R, Zazzarini M, Fiorani A, Tassetti AN. Multimodal ai-enhanced ship detection for mapping fishing vessels and informing on suspicious activities. Pattern Recogn Lett. 2025;191:15–22. https://doi.org/10.1016/j.patrec.2025.02.022.
32. Yuji I. Rice Leaf Diseases Detection - kaggle.com. https://www.kaggle.com/datasets/loki4514/rice-leaf-diseases-detection. Accessed 09 Sep 2025.
33. Prabhakaran S. Sugarcane Disease Dataset - kaggle.com. https://www.kaggle.com/datasets/prabhakaransoundar/sugarcane-disease-dataset. Accessed 09 Sep 2025.
34. Olyad G. Wheat Leaf dataset - kaggle.com. https://www.kaggle.com/datasets/olyadgetch/wheat-leaf-dataset. Accessed 09 Sep 2025.
35. Goodwin SB, Ben M'Barek S, Dhillon B, Wittenberg AHJ, Crane CF, Hane JK, et al. Finished genome of the fungal wheat pathogen mycosphaerella graminicola reveals dispensome structure, chromosome plasticity, and stealth pathogenesis. PLoS Genet. 2011;7(6):1–17. https://doi.org/10.1371/journal.pgen.1002070.
36. Mullins JGL, Parker JE, Cools HJ, Togawa RC, Lucas JA, Fraaije BA, et al. Molecular modelling of the emergence of azole resistance in mycosphaerella graminicola. PLoS ONE. 2011;6(6):1–11. https://doi.org/10.1371/journal.pone.0020973.
37. Kleczewski N, Bradley C, Chilvers M, Collins A, DeWolf E, Friskop A, Mehl H, Paul P, Salgado D, Smith D, Wise K, Young-Kelly H. An overview of stripe rust of wheat. Technical report, Crop Protection Network, United States. 2020.
38. M D. Paddy Diseases: causes, symptoms, and treatment options 2024. https://kisanvedika.bighaat.com/major-diseases-of-paddy/.
39. Mew T, Hibino H, Savary S, Vera Cruz C, Opulencia R, Hettel G. Rice diseases: Biology and selected management practices. Los Baños (Philippines): International Rice Research Institute; 2018.
40. Catindig J. Rice leaffolder PlantwisePlus Knowledge Bank. 2025. https://doi.org/10.5555/pwkb.20117800238.
41. Agriculture N, Organization FR. Diseases of forage crops. https://www.naro.affrc.go.jp/org/nilgs/diseases/contents/de4.htm.
42. Yasmin H, Shah ZA, Mumtaz S, Ilyas N, Rashid U, Alsahli AA, et al. Alleviation of banded leaf and sheath blight disease incidence in maize by bacterial volatile organic compounds and molecular docking of targeted inhibitors in rhizoctonia solani. Front Plant Sci. 2023;14:1218615.
43. Matova PM, Kamutando CN, Magorokosho C, Kutywayo D, Gutsa F, Labuschagne M. Fall-armyworm invasion, control practices and resistance breeding in sub-saharan africa. Crop Sci. 2020;60(6):2951–70.
44. Dhanusri S, Dharani K, Maheswari G. Overview of bacterial and fungal diseases of sugarcane and its effective management techniques. Fundamentals of agriculture and crop production. Bharti Publications. 2022:236–246.
45. Portal TNAU. Crop Protection. https://agritech.tnau.ac.in/crop_protection/sugarcane_diseases/sugarcane_d4.html.
46. PyPI: Werkzeug. 2025. https://pypi.org/project/Werkzeug/.
47. TensorFlow: tf.keras.preprocessing.image.ImageDataGenerator. https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator. Accessed 11 Feb 2026.
48. Khare O, Mane S, Kulkarni H, Barve N. Leafnst: an improved data augmentation method for classification of plant disease using object-based neural style transfer. Discover Artificial Intelligence. 2024;4(1):50.
49. Biancucci M, Galdelli A, Narang G, Pietrini R, Mancini A, Zingaretti P. Coigan: Controllable object inpainting through generative adversarial network for defect synthesis in data augmentation. In: 2025 IEEE International Conference on Robotics and Automation (ICRA). 2025:9046–9052. https://doi.org/10.1109/ICRA55743.2025.11127765.


## Publisher's Note