République Algérienne Démocratique et Populaire
الجمهورية الجزائرية الديمقراطية الشعبية
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
وزارة التعليم العالي والبحث العلمي

المدرسة الوطنية للإعلام الآلي
(المعهد الوطني للتكوين في الإعلام الآلي سابقا)
École nationale Supérieure d'Informatique
ex. INI (Institut National de formation en Informatique)

**Final Year Project**

**In partial fulfillment of the requirements for the degree of Computer Engineering**

**Option: Computer Systems and Software (SIL)**

# Implementation of Deep Learning Techniques for Smart Agriculture - Detection and Classification of Wheat Diseases.

*Written By:*
ABBACI Zoulikha
kz_abbaci@esi.dz

BELLALI Amira
ka_bellali@esi.dz

*Supervised By:*
Dr. BESSAH Naima
n_bessah@esi.dz

Dr. SEHAD Abdenour
a_sehad@esi.dz

2024 / 2025

# إهـداء

إلى أبي وأمي، صاحبي الفضل علي بعد الله،

إلى أخي وأختيّ وجميع أهلي،

إلى شوقي، أعز أصدقائي،

إلى أولئك الذين سألت عونهم كلما أشكل عليّ أمر

وشاركوني فرحة كل نجاح،

إليكم جميعا، أهدي هذا العمل. أنتم من شاركتموني تعبه وسهره وهمه، وليس من أحد أحق منكم أن يشاركني فرحته.

# شكر و عرفان

الحمد لله أولا أن وفقني لهذا وماكنت لأبلغه لولا أن وفقني، وأعانني على إتمامه وما كنت لأتمه لولا أن أعانني. أحمدك اللهم ملء السماوات والأرض وملء مابينهما وملء ما شئت دون ذلك، لا أحصي ثناء عليك، أنت كما أثنيت على نفسك.

أشكر شكرا جزيلًا الهيئة المشرفة على هذا المشروع، الدكتورة زكريا شهناز، البروفيسور سماعيلي كمال والدكتور Langlois David ، الذين لولا نصحهم وتوجيههم، ما كان هذا العمل قد نجح. أشكر صبركم على أخطائي وأشكر تصويبكم إياها، وأشكركم فوق كل شيء، على ما تعلمت منكم.

أشكر أيضا والدي الذين حملا قلق هذا العمل وتعبه بقدر ما حملتهما أنا. وأخي وأختيّ الذين أسندوني ساعة تعبي وأضحكوني ساعة حزني وغضبي.

الشكر كذلك لأولئك الذين ساعدوني ولم يبخلوا علي بمشورة أو رأي، لا لشيء سوى كرم أنفسهم وطيبة معدنهم. نزيم ونهى وبلال ونسيم وزكريا وماسينيسا وبرهان ومليك وهيثم والدكتور إسماعيل عدون ويحيى وأحمد وولام وسيلينا.

والشكر لله آخرًا كما له الشكر أولا، هو الذي أنعم علي بهؤلاء جميعا، وبهذا كله فالحمد له ملء ما أنعم، والحمد له ملء ما شاء

**Abstract**

here you can make your abstract

**Keywords —** Smart Agriculture, Deep Learning, Computer Vision, Wheat Diseases, Pest Detection.

**Résumé**

here make your summery in french

**Mots clés —** Agriculture intelligente, apprentissage profond, vision par ordinateur, maladies du blé, détection des parasites.

# ملخص

ملخصك باللغة العربية

---

**الكلمات المفتاحية —** الزراعة الذكية، التعلم العميق، الرؤية الحاسوبية، أمراض القمح، كشف الآفات الحشرية.

---

# Contents

# List of Figures

# Contents

# General introduction

The introduction goes here.

# Part I

# State of Art

# Chapter 1

# Threats to Wheat Production: Disease Identification and the Shift Toward Smart Agriculture

## 1.1 Introduction

Wheat is a crucial global crop, but its production is threatened by various diseases and insect pests, leading to significant yield losses. Traditional detection and control methods are often ineffective, highlighting the need for improved solutions.

This chapter explores common wheat diseases and pests, their impact on agriculture, and the importance of timely detection. It also discusses strategies for enhancing disease control and the challenges of implementing smart agricultural technologies.

## 1.2 Motivation for the Protection of Wheat Crops

Wheat is an ancient and vital food crop that provides energy and feeds billions of people around the world (see Figure 3.7). Its demand is growing quickly because it's used in many affordable food products and plays a big role in global food security. The FAO (Food and Agriculture Organization) estimates that by 2050, the world will need about 840 million tonnes of wheat, up from 642 million tonnes today [38]. This doesn't even include the extra needs like animal feed or the impact of climate change.

To meet this growing demand, developing countries need to increase wheat production by 77%, mostly by improving how much wheat is grown on the same land [38]. But this is becoming harder, as wheat productivity is slowing down and diseases are becoming a bigger problem. If we don't manage pests and diseases properly, wheat production could fall short of what the world needs.

That's why it's important to invest in research, use better farming methods, and grow disease-resistant wheat. Protecting this essential crop is key to making sure we have enough food for the future.

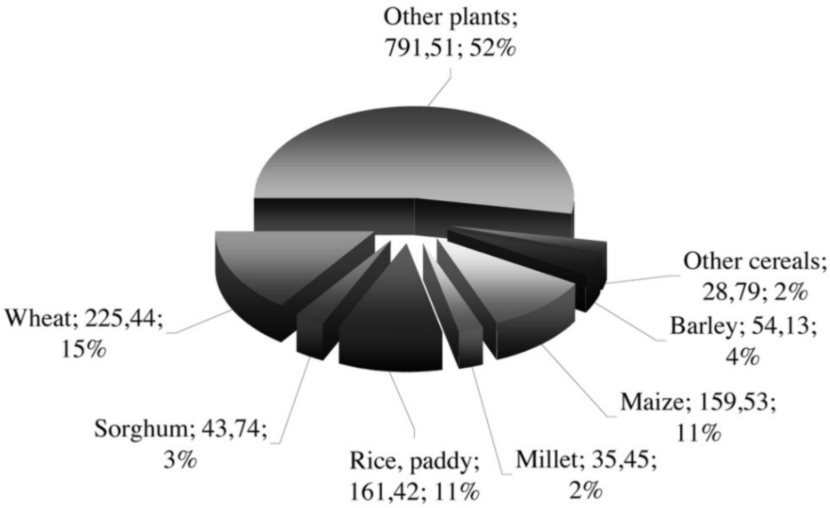



Figure 1.1: Division of sowing area in the world in 2009 (in million hectares and percentage) Source: FAO, 2011

## 1.3 Wheat growth stages

Wheat growth follows a series of distinct developmental stages (Figure 1.2), each with specific environmental needs and a direct impact on crop health and yield [37]. The key stages are:

- **Seeding:** The initial stage where seeds germinate and seedlings emerge. Adequate soil moisture and suitable temperatures are essential for successful crop establishment.
- **Tillering:** The plant produces side shoots (tillers), which increase potential yield. This stage depends on sufficient nutrients and water.
- **Booting:** The wheat head develops inside the leaf sheath. Stress at this stage can reduce the number of spikelets and affect grain development.
- **Heading:** The spike emerges and flowering occurs. This is a sensitive period where drought or heat can severely impact pollination and grain set.
- **Ripening:** Grains mature and the plant loses its green color. Proper conditions are needed for effective grain filling and harvest readiness.

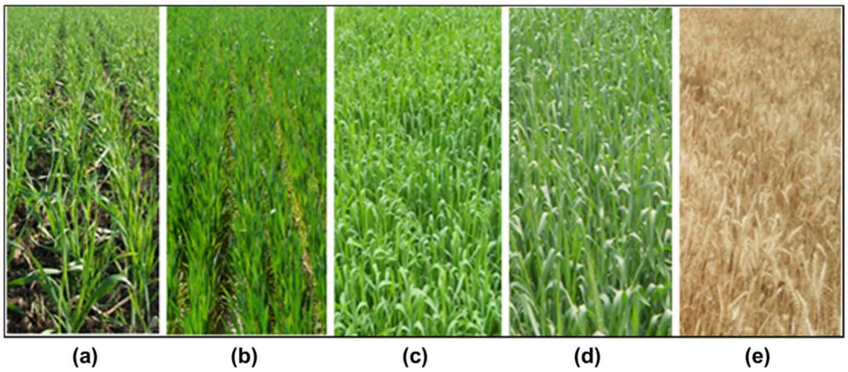



Figure 1.2: wheat crop growth stages: (a) Tillering; (b) Jointing; (c) Booting; (d) Heading; and (e) Ripening [26]

## 1.4 Wheat Diseases: Types and Impacts

Wheat diseases are influenced by several factors, including plant resistance, spore density, temperature, and environmental conditions, especially the presence of moisture on plant surfaces, which facilitates infection. While some fungi are host-specific, others can infect a wide range of plants. Symptoms can differ greatly, making accurate identification essential. Researchers primarily rely on fungal morphology for diagnosis. A clear understanding of these diseases is key to effective management and control. The following classification (Figure 1.3) outlines the major wheat diseases, grouped by their causes and the plant parts they affect.

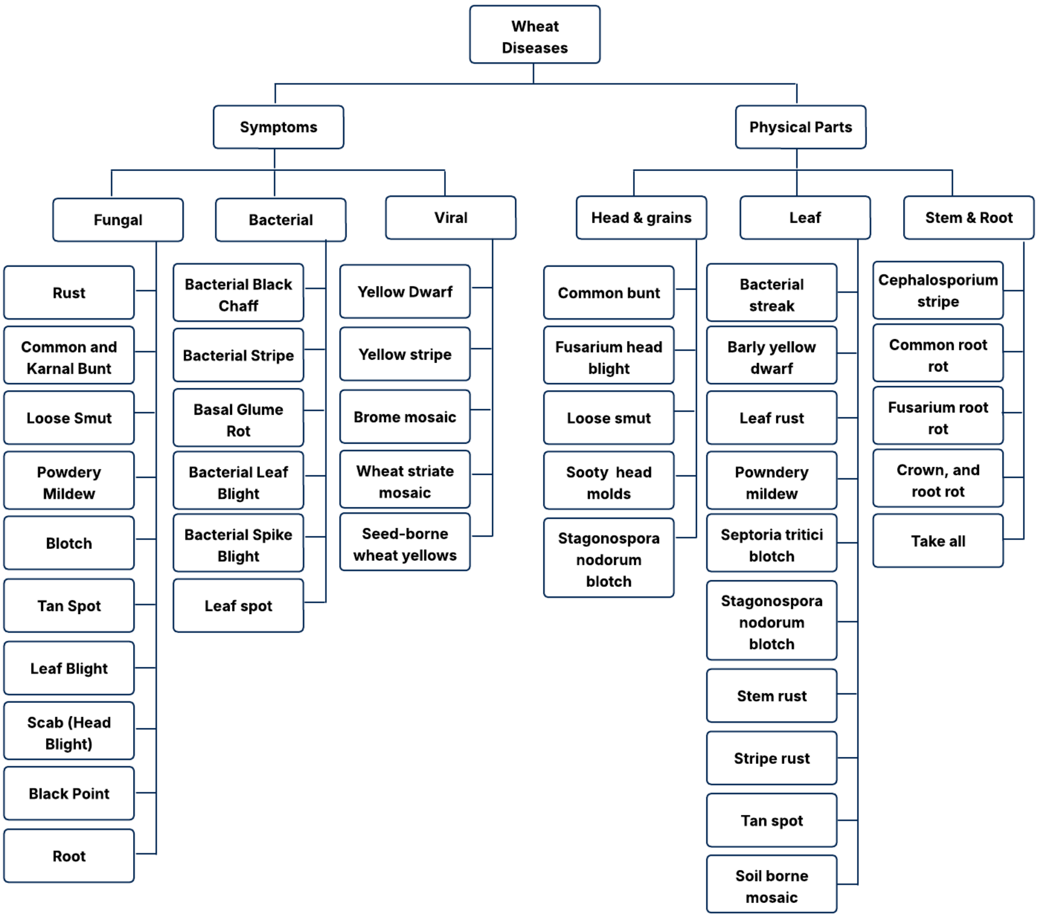



Figure 1.3: Taxonomy of wheat diseases [15]

### 1.4.1 Leaf Rust (Brown Rust)

Leaf rust, caused by *Puccinia triticina*, appears as small, circular, orange to brown pustules on the upper surfaces of leaves and leaf sheaths. It spreads through wind-borne spores and develops quickly in moist conditions at around 20°C. New spores form every 10–14 days if conditions are favorable. As plants mature or conditions worsen, black spores may appear (as shown in Figure 1.4).
This disease affects wheat, triticale, and related grasses and is common in temperate cereal-growing regions. Severe infections reduce grain yield, kernel number, weight, and quality [9].

Figure 1.4: Leaf rust with Brown spores (1), Leaf rust with Black spores (2) [9]

### 1.4.2 Stem Rust (Black Rust)

Stem rust, caused by *Puccinia graminis*, appears as dark reddish-brown pustules on leaves, stems, and spikes (as observed in Figure 1.5). Light infections show scattered pustules, while severe cases cause them to merge. Before pustules form, small flecks may appear, and infected areas feel rough. The disease spreads through wind-borne spores and develops quickly in moist conditions with temperatures around 20°C. New spores can form in 10–15 days. It affects wheat, barley, triticale, and related grasses and is common in temperate cereal regions. Severe infections can reduce grain weight and quality and, in extreme cases, lead to total crop loss [9].

Figure 1.5: Stem rust [9]

### 1.4.3 Stripe Rust (Yellow Rust)

Stripe rust, caused by *Puccinia striiformis*, appears as yellow to orange-yellow pustules forming narrow stripes on leaves, leaf sheaths, necks, and glumes (as seen in Figure 1.6). It spreads through wind-borne spores and develops quickly in moist conditions at temperatures between 10–20°C but slows down above 25°C. Severe infections reduce grain yield, kernel number, weight, and quality [9].

Figure 1.6: Stripe rust [9]

### 1.4.4 Blotch Diseases

The blotch diseases, which include Septoria tritici blotch (STB), Septoria nodorum blotch (SNB), and tan spot (TS) (as presented in Figure 1.7), are caused by the *Ascomycete fungi Zymoseptoria tritici*, *Parastagonospora nodorum*, and *Pyrenophora tritici-repentis*, respectively [12].

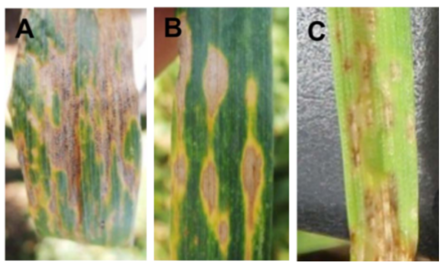



Figure 1.7: Symptoms of foliar blotch diseases. (A) Septoria tritici blotch. (B) Tan spot. (C) Septoria nodorum blotch [12]

### 1.4.5 Fusarium Head Blight (FHB)

Fusarium head blight (FHB), also known as wheat scab or ear blight, is a major disease of wheat caused primarily by the Ascomycete fungus *Fusarium graminearum* (Fg). It can also be caused by other regional *Fusarium* species [12].
Fusarium head blight appears as dark, oily florets with pinkish spores (as seen in Figure 1.8). Infected kernels may be covered in white fungal growth. The disease spreads in warm, humid conditions (10–28°C), infecting spikes during flowering and spreading between florets. It affects all small grain cereals and is found in most soils and crop residues. Severe infections can reduce yields by over 50% and lower grain quality. Contaminated grain may contain harmful mycotoxins, making it unsafe for humans and animals [9].

Figure 1.8: Symptoms of Fusarium head blight/scab: The left image shows early infection signs manifested as a partially bleached wheat head, while the right image illustrates advanced infection of *Fusarium graminearum* [12].

### 1.4.6 Loose Smut

Loose smut, caused by *Ustilago tritici*, replaces wheat spikes with black fungal spores (as observed in Figure 1.9), which are later dispersed by wind. The fungus infects wheat flowers and stays dormant in kernels until germination. It then grows with the plant, destroying floral parts at flowering. The disease thrives in cool, humid conditions and is found wherever wheat is grown. Yield losses depend on infection levels, usually below 1% but sometimes reaching 30% [9].

Figure 1.9: Loose smut [9]

### 1.4.7 Powdery mildew

Powdery mildew (Figure 1.10), caused by *Blumeria graminis f. sp. tritici*, affects wheat globally, particularly in cool, dry climates, and can cause yield losses ranging from 10% to 40%, with severe cases leading to seedling or tiller death [39].

Figure 1.10: Powdery mildew from Kaggle dataset ‘Wheat Plant Diseases.’

### 1.4.8 Common Root Rot

Common root rot, caused by *Cochliobolus sativus*, *Fusarium spp*, and *Pythium spp*, darkens and weakens wheat roots, crowns, and stems (as illustrated by Figure 1.11), sometimes leading to plant lodging and white spikes before maturity. Early infections can cause seedling death. The disease spreads from infected crop debris, thriving in different soil conditions: *C. sativus* in warm, dry soils and *Fusarium* and *Pythium* in cool, moist soils. Found in temperate regions, it rarely causes major outbreaks but can lead to localized losses due to reduced plant growth and yield [9].

Figure 1.11: Common root rot from Kaggle dataset 'Wheat Plant Diseases.'

## 1.5 Common Insect Pests in Wheat Cultivation

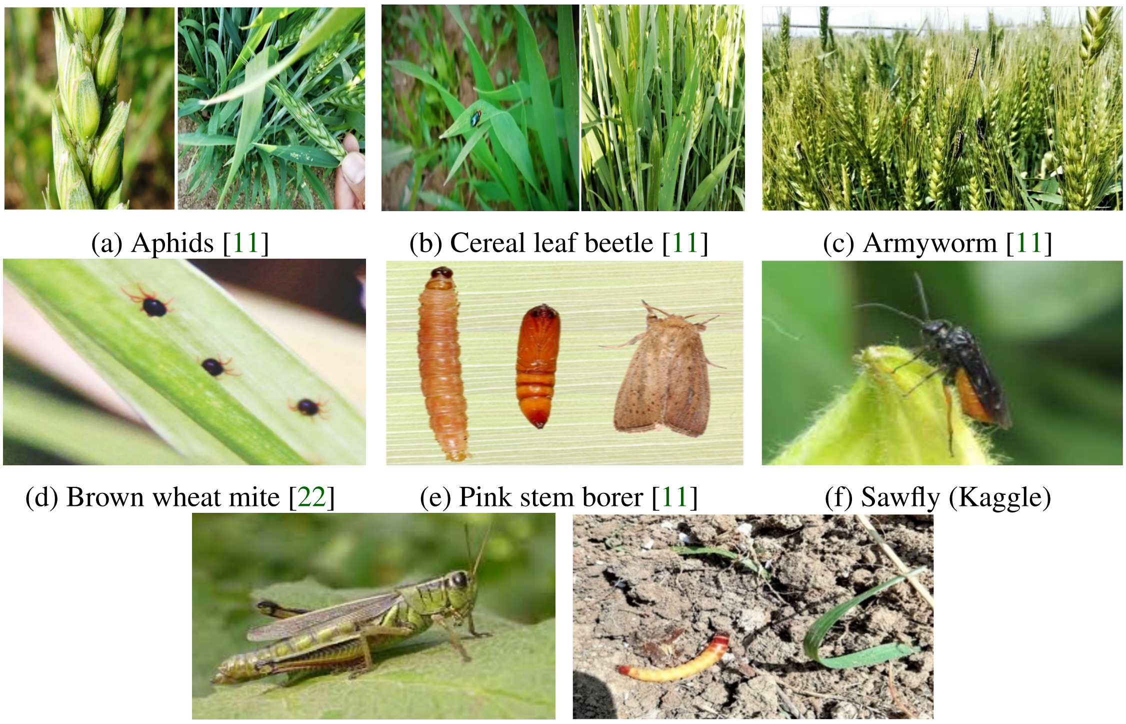
Wheat is affected by several insect pests that can seriously reduce yield and quality. Below are some of the most common pests and their impacts (see Figure 1.12).

- **Aphids:** Aphids are soft-bodied, transparent insects that feed on wheat leaves and grain heads, causing yellowing, leaf rolling, and poor pollination, especially during early growth. Their sap-sucking damages crops, and the honeydew they excrete promotes black sooty mold, reducing photosynthesis and leading to yield losses of 20–80% [9, 11].
- **Cereal leaf beetle:** Adult cereal leaf beetles are 4–5 mm long with a black head, light brown thorax, and shiny blue-green wings. Larvae are initially yellow but turn into a black mass due to accumulated fecal material. The main symptom of infestation is distinct longitudinal stripes on leaves caused by the feeding of both adults and larvae. Infestations can lead to yield losses of 14% to over 25% in winter and fall-sown spring wheat [9].
- **Armyworm:** The armyworm (Mythimna separata) is a wheat pest. Adult moths are stout and pale brown, while larvae have orange, white, and brown stripes, along with black spots on their prolegs. Caterpillars cause significant damage by swarming from field to field, feeding on seedling leaves and ear heads, which halts plant growth [11].
- **Brown wheat mite:** The brown wheat mite, found in rainfed wheat areas, has only females that lay red eggs in winter and white-covered eggs in summer. They damage crops by sucking sap, causing silvery flecks, yellowing leaves, and reduced grain quality. Mites are active in daylight and do not form webs. Infestations start in December–January and last until maturity, with winter rains limiting their spread [22].
- **Pink stem borer:** The pink stem borer (Sesamia inferens) is an oriental pest that originally affected rice but has adapted to wheat in North-Western India. Its larvae feed inside wheat stems, causing "dead hearts" and "white heads," leading to yield losses over 11%. Damage symptoms in wheat are similar to those in rice [11].
- **Sawfly:** Sawflies produce one generation per year, with larvae overwintering in straw. The legless white larvae bore into wheat stems, weakening plants, causing poor head

development, and making them prone to lodging. While infestations are patchy, the wheat stem sawfly (Cephus cinctus) can cause severe localized yield losses [9].

- **Slugs, Snails, Grasshoppers, and Crickets:** These are widespread pests affecting wheat and other plants. They damage crops by chewing leaves, causing a frayed appearance in mature plants. While their presence is often localized, large infestations can significantly impact plant health and yield worldwide.

- **Wireworm:** Wireworms are yellow to brown larvae with six short legs that feed on wheat kernels, consuming the endosperm and leaving only the seed coat. They attack young seedlings, causing "damping off" symptoms and damaging crops early on. Their presence can significantly affect wheat growth and yield, making timely identification and control crucial [11].



(a) Aphids [11] (b) Cereal leaf beetle [11] (c) Armyworm [11]

(d) Brown wheat mite [22] (e) Pink stem borer [11] (f) Sawfly (Kaggle)

(g) Grasshoppers (Kaggle) (h) Wireworm [11]

Figure 1.12: Common pests affecting wheat crops.

## 1.6 Enhancing Wheat Disease Control Strategies

Effective wheat disease management combines traditional farming practices with modern technologies. Together, they offer a balanced approach to reducing disease impact and improving crop health.

### 1.6.1 Usual Instruments (Classic Methods)

These are long-standing approaches that provide the foundation for managing wheat diseases [29]. The following practices have been widely used to reduce disease pressure and support healthy crop development.

- **Crop Rotation:** Rotating wheat with non-host crops reduces the buildup of soil-borne pathogens and interrupts disease cycles.
- **Tillage:** Tillage affects disease development by influencing residue decomposition and soil pathogen levels; conservation tillage can increase some necrotrophic diseases.
- **Healthy Seeds:** Clean, pathogen-free seeds minimize seed-borne disease transmission and ensure strong early crop establishment.
- **Soil Management:** Managing soil pH, structure, and nutrient balance helps prevent stress-related susceptibility and supports healthier root systems.
- **Fertilizer Use:** Balanced fertilization strengthens plant defense mechanisms, while over- or under-fertilization can predispose plants to infection.
- **Diversification of Cultivars and Sowing Dates:** Altering cultivars and planting schedules helps reduce the uniformity that pathogens exploit and spreads risk across environments.
- **Use of Resistant Cultivars:** Cultivars bred for specific, partial, or generalized resistance can significantly reduce disease severity, especially when tailored to local pathogen races.
- **Alternative Eco-Friendly Practices:** Methods like field sanitation, residue management, and proper spacing contribute to reducing pathogen survival and disease spread.

### 1.6.2 Technological Tools

Complementing traditional methods, the following tools, rooted in smart agriculture, offer modern solutions to enhance the effectiveness and precision of wheat disease management.

- **Remote Sensing:** Remote sensing using unmanned aerial vehicle-mounted multispectral sensors enables high-resolution monitoring of wheat canopy characteristics across different growth stages. By analyzing spectral bands (Green, Red, Red Edge, Near Infrared), the system captures critical indicators of plant health, canopy structure, and stress conditions, supporting precise and timely crop management [42].
- **Disease Forecast Modeling:** Weather-based and biological models are used to predict disease outbreaks and support timely interventions [29].
- **Computer Vision:** Enables automated analysis of crop images for monitoring wheat growth, detecting diseases, and assessing yield. It plays a crucial role in real-time decision-making by processing visual data from the field [13].

- **AI and Machine Learning Algorithms:** Used to interpret complex image data, these algorithms support tasks like disease classification, crop health prediction, and optimizing farm operations by learning from patterns in large agricultural datasets [13].
- **Autonomous Robotic Platforms and Drones:** Facilitate efficient field data collection, spraying, and crop monitoring. These tools reduce manual labor and enable precise, targeted interventions across large wheat fields [13].
- **Precision Agriculture Systems:** Precision agriculture systems integrate technologies like the Global Positioning System (GPS) and the Internet of Things (IOT) to manage field variability. These technologies help optimize the use of inputs (e.g., water, fertilizer) and support sustainable, data-driven wheat farming [13].

## 1.7 Challenges Facing the Integration of Smart Agricultural Systems

Fully automated smart farming faces both technical and practical challenges. A major obstacle is the generalization of computer vision models across diverse field conditions like lighting, weather, soil, and crop types, which complicates real-time deployment. Robust decision-making in unpredictable outdoor environments also remains difficult, and integrating the full pipeline from image capture to treatment is still under development [13].

On the technical side, communication protocols often support only short distances, limiting scalability. Many devices rely on batteries, reducing operational time. Additionally, processing the large volumes of data generated introduces computational bottlenecks, alongside concerns about privacy, trust, and security in data handling [18].

## 1.8 Conclusion

This chapter provided an overview of the major wheat diseases and pests, along with the key challenges in their detection and management. These challenges underscore the urgent need for more effective and intelligent solutions. In the following chapter, we delve into the domain of deep learning—exploring essential concepts, popular architectures, and their practical applications in enhancing wheat disease classification and pest detection.

# Chapter 2

# Deep Learning for Smart Agriculture

## 2.1 Introduction

Deep learning and computer vision have become indispensable tools in modern agriculture, enabling automated, high-precision analysis of crop health, disease detection, and pest management. By leveraging convolutional neural networks (CNNs), these technologies can process vast amounts of visual data from drone imagery to ground-based cameras, identifying subtle patterns indicative of wheat diseases. This shift from manual scouting to AI-driven monitoring improves scalability, reduces human error, and supports timely interventions, ultimately enhancing yield and sustainability.

In this chapter, we establish the foundational concepts of deep learning and computer vision as applied to agricultural challenges. We begin by examining the core principles of CNNs and their role in feature extraction from crop imagery. Next, we explore transfer learning techniques, which allow pre-trained models to adapt to agricultural datasets with limited labeled examples. The chapter then discusses object detection methods critical for localizing diseases and pests in field conditions. Finally, we address the practical challenges of implementing these solutions, including data variability, model scalability, and real-world deployment constraints in agricultural settings.

## 2.2 Distinctions Between Deep Learning and Machine Learning

Deep learning is a subset of machine learning that uses deep neural networks to automatically learn complex patterns from large, unstructured data like images, text, and audio. Unlike traditional machine learning, which requires manual feature extraction and works best with smaller, structured datasets, deep learning eliminates this need and can handle large datasets with significant computational power, often using GPUs. While machine learning is suitable for tasks like predictive analytics, deep learning excels in applications such as image recognition, speech processing, and natural language understanding [3].

## 2.3 Deep Learning Fundamentals

In this section, we introduce the core principles of deep learning, focusing on deep neural networks (DNNs) and their role in learning complex patterns from data. We explore the structure and components of neural networks, the learning process, optimization techniques, and model evaluation methods. Additionally, we highlight regularization strategies to prevent overfitting.

### 2.3.1 Deep Neural Network Basics (DNN)

Neural networks form the backbone of deep learning, enabling machines to learn patterns and make predictions from data. Inspired by the structure of the human brain, these networks consist of interconnected layers of artificial neurons that hierarchically process information.

#### Definition of DNN

Before defining deep neural networks, we first need to understand two essential components:

- **Artificial neuron**: An artificial neuron is the basic building block of artificial neural networks, designed based on the structure and functionality of biological neurons. It receives weighted inputs, processes them through a transfer function, and outputs the result. The artificial neuron model simplifies the biological process where information is received through dendrites, processed in the soma, and transmitted via the axon, as shown in Figure 2.1 [25].
- **Layer**: A layer in a neural network is a set of neurons that perform a specific operation on the data [6].

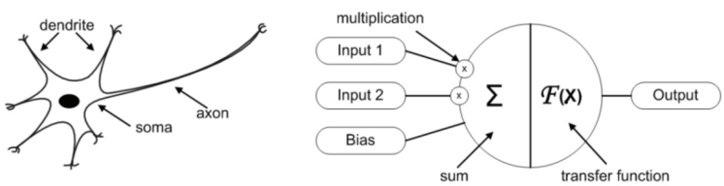



Figure 2.1: Biological Neuron Structure and Its Mathematical Model Representation [25].

By combining multiple layers of interconnected artificial neurons, we arrive at the concept of a Deep Neural Network (DNN). A DNN is a neural network that contains multiple hidden layers between the input and output layers. These additional layers enable the network to learn complex patterns and high-level features from data. Each layer transforms its input into a more abstract representation, improving the network's ability to recognize intricate structures and relationships [27].

#### Structure of DNN

A neural network consists of three main layers [36]:

**The input layer:** Represents the features of the input data, such as pixel values in an image, denoted as a vector

$$X = [x_1, x_2, \ldots, x_n].$$

**The hidden layers:** Process this input using weighted connections and biases, computed as:

$$z = W \cdot X + b_z \tag{1}$$

$$F(z) = a \tag{2}$$

where:

- $W$ is the weight matrix,
- $b$ is the bias vector,
- $z$ is the pre-activation value,
- $F(z)$ is the activation function applied to $z$.

Each neuron in the hidden layers applies an activation function to capture complex patterns.

**The output layer:** Generates the network's prediction, with the number of neurons corresponding to the specific task, such as one neuron for binary classification or multiple neurons for multiclass classification.

#### Activation Functions

An activation function (AF) is a mathematical function applied to a neuron's output in a neural network to introduce non-linearity. Without an activation function, a neural network with multiple layers would behave like a single-layer perceptron, limiting its ability to model complex relationships [8]. Activation functions decide whether a neuron should be activated based on its input.

Table 2.1 summarizes the most commonly used activation functions, their formulas, ranges, and typical use cases in deep neural networks.

### 2.3.2 Learning Process in Deep Neural Networks

In the information processing flow within an artificial neuron, several elements, known as parameters, are learned from the training data. These parameters include [23]:

Table 2.1: Common Activation Functions in Deep Learning [8]

| **Activation Function** | **Formula** | **Range** | **Usage** |
|---|---|---|---|
| Sigmoid | $\text{Sigmoid}(x) = \frac{1}{1+e^{-x}}$ | [0, 1] | Commonly used in binary classification problems, especially in the output layer of models predicting probabilities. |
| Tanh (Hyperbolic Tangent) | $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ | [-1, 1] | Often used in hidden layers of neural networks, as it outputs values centered around zero, which helps in reducing bias during optimization. |
| ReLU (Rectified Linear Unit) | $\text{ReLU}(x) = \max(0, x)$ | [0, ∞) | Widely used in hidden layers of deep neural networks due to its simplicity and effectiveness in handling the vanishing gradient problem. |
| Softmax | $\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$, where $x_i$: current element, $x_j$: all elements in the vector. | (0, 1) (outputs sum to 1) | Used in the output layer for multi-class classification. |

- **Weights**: Weights control the amount of each input feature that passes through the neuron. They represent the coefficients of the connections between neurons in the layers of a neural network. Weights are essential for determining the influence of each input on the output.
- **Biases**: Biases are values added to the outputs of the neurons before applying the activation function. They allow the network to shift the activation function, providing more flexibility to the model.

The learning process in neural networks involves training the model to map input data to desired outputs. This is achieved through the mechanisms of forward propagation and backward propagation, along with optimization techniques that refine the network's parameters (weights and biases) to minimize the error [23]:

- **Forward Propagation**: In forward propagation, the input data passes through the network layer by layer. Each neuron in a layer performs a weighted sum of its inputs, applies an activation function, and passes the result to the next layer. The process continues until the output layer is reached and a prediction is made.
- **Backward Propagation**: Backward propagation (or backpropagation) is used to update the network's weights. After calculating the error (the difference between the predicted and actual output), the error is propagated back through the network. The weights are adjusted based on the gradient of the error with respect to each weight, using optimization algorithms like gradient descent.

Figure 2.2 below provides a visual representation of the learning process in a neural network, illustrating the flow of information from the input layer through multiple hidden layers to the output layer.

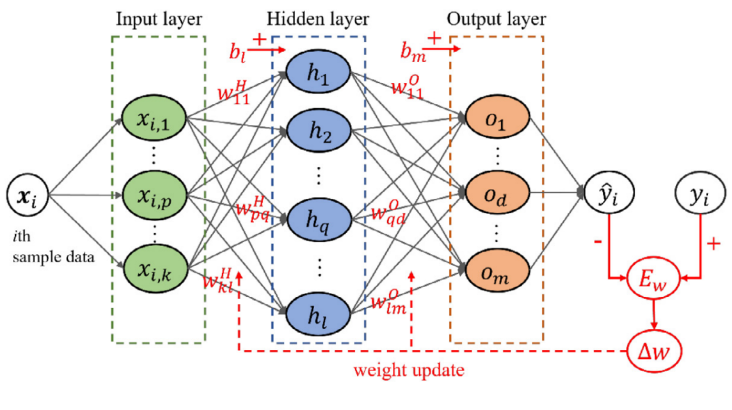



Figure 2.2: The structure of the DNN [36].

### 2.3.3 Model Training Concepts

Effective model training ensures that neural networks learn patterns in data while minimizing errors. This section highlights essential components of the training process.

**Loss Functions** Loss functions quantify the error between predicted and actual values, guiding weight updates during optimization [4]. Common loss functions include:

- **Cross-Entropy Loss**: Used for classification tasks to measure the divergence between predicted probabilities and true labels.
- **Mean Squared Error (MSE)**: Applied in regression, computing the average squared difference between predicted and actual values.

**Overfitting Prevention** Overfitting occurs when a model learns noise from training data, reducing generalization to unseen data. Common techniques to mitigate overfitting include:

- **Dropout**: Randomly deactivate neurons during training to enhance robustness.
- **Data Augmentation**: Transform training samples (e.g., rotation, scaling) to increase dataset diversity.

**Batch Normalization** Batch normalization stabilizes training by normalizing inputs across a mini-batch, reducing internal covariate shift and accelerating convergence.

### 2.3.4 Optimization Methods and Strategies

Optimization techniques in deep learning are methods used to minimize the loss function during training, improving the model's accuracy. These techniques adjust the model's weights and biases iteratively to find the optimal set of parameters that reduces the error between predicted and actual values [4].

Different optimizers are used to update model weights efficiently:

- **Stochastic Gradient Descent (SGD):** Updates weights based on a small subset (batch) of training data, improving computational efficiency.
- **Adam (Adaptive Moment Estimation):** Combines momentum and adaptive learning rates for faster and more stable convergence.
- **RMSprop:** Uses an adaptive learning rate to prevent oscillations and improve performance on non-stationary objectives.

However, an important consideration in optimization is how to set the learning rate throughout training. While optimizers control how weights are updated, learning rate scheduling adjusts the learning rate over time to further optimize training.

Different scheduling strategies can be used in conjunction with optimizers [43]:

- **Step Decay:** Reduces the learning rate at fixed intervals, allowing for more stable training as the model reaches its optimal solution.
- **Exponential Decay:** Gradually decreases the learning rate across epochs, promoting finer adjustments to the model parameters as the training progresses.
- **Cyclic Learning Rates:** Alternates between a minimum and maximum learning rate, which helps the model escape local minima and enhances exploration of the parameter space.

By combining these scheduling strategies with optimization techniques, the training process can become more efficient and effective, leading to faster and more reliable convergence.

### 2.3.5 Model Evaluation and Validation

Once a model is trained, it is crucial to assess its performance and ensure that it generalizes well to unseen data. This process is known as **model evaluation and validation**. The primary goal is to measure how well the model performs and to identify any potential overfitting or underfitting issues.

To evaluate a model's performance, several metrics can be used depending on the type of task (classification, regression, etc.) [24]:

- **Accuracy:** For classification tasks, accuracy measures the proportion of correct predictions out of all predictions made. However, accuracy alone can be misleading in imbalanced datasets.
- **Precision and Recall:** In imbalanced classification problems, precision (the proportion of true positive results among all positive predictions) and recall (the proportion of true positive results among all actual positives) are often used in conjunction to provide a clearer view of the model's performance.
- **F1-Score:** The harmonic mean of precision and recall; F1-score balances the two metrics and is especially useful when dealing with imbalanced datasets.
- **Mean Squared Error (MSE):** For regression tasks, MSE calculates the average of the squared differences between predicted and actual values. It penalizes large errors more significantly than smaller ones.
- **R-squared ($R^2$):** For regression, this metric indicates how well the model explains the variability in the data, with a value closer to 1 suggesting a better fit.

## 2.4 Deep Learning Approaches

Deep learning encompasses a variety of learning paradigms that enable models to extract complex patterns from large volumes of data. These approaches differ based on the nature of the data, the availability of labels, and the interaction mechanisms with the environment. This section presents the four main categories of deep learning methods [4]:

- **Deep Supervised Learning:** This approach involves training a neural network using a labeled dataset, where each input image (e.g., leaf with visible symptoms) is paired with its correct label (e.g., disease type). The model learns to minimize the error between its predictions and the true labels using backpropagation and optimization algorithms. Techniques include Convolutional Neural Networks (CNNs) for spatial feature extraction, Deep Neural Networks (DNNs), and Recurrent Neural Networks (RNNs) like LSTMs and GRUs when dealing with sequential data.
- **Deep Semi-Supervised Learning:** Semi-supervised learning combines a small labeled dataset with a larger pool of unlabeled data, which is common in agricultural settings where annotating plant diseases is costly and time-consuming. Methods like Generative Adversarial Networks (GANs) can generate synthetic labeled images, while RNNs, LSTMs, and Deep Reinforcement Learning (DRL) can help model complex data behavior.
- **Deep Unsupervised Learning:** Unsupervised learning aims to extract meaningful patterns from unlabeled data, such as grouping similar plant images or identifying features without predefined classes. Popular methods include Autoencoders for dimensionality reduction, Restricted Boltzmann Machines, and clustering algorithms. These are useful in exploratory stages or feature extraction before applying a classifier.
- **Deep Reinforcement Learning (DRL):** In DRL, models learn optimal actions through trial and error by interacting with an environment. This could include real-time monitoring systems in agriculture, like automated disease response tools or robotic weeders. Unlike supervised learning, DRL does not require labeled data but instead uses reward signals to adjust its strategy over time.

## 2.5 Convolutional Neural Networks (CNNs) for Plant Disease Classification

Convolutional Neural Networks (CNNs) have revolutionized image-based plant disease classification by automatically extracting hierarchical features from agricultural images. Unlike traditional machine learning approaches that rely on handcrafted features, CNNs learn spatial patterns directly from raw images, improving classification accuracy. Their ability to recognize disease symptoms from leaf textures and color variations makes them particularly effective in precision agriculture. This section explores the fundamental components of CNNs, their advantages in agricultural applications, and their limitations when applied independently.

### 2.5.1 Fundamentals of CNNs

Convolutional Neural Networks (CNNs) consist of multiple layers designed to process and learn spatial hierarchies from image data (see Figure 2.3). Their architecture typically includes [4]:

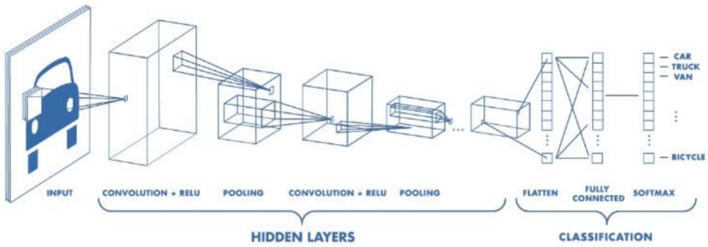



Figure 2.3: Architecture of CNN [32].

- **Convolutional Layers:** Extract features using small filters (kernels) that detect edges, textures, and patterns. A kernel is a grid of values (weights) initialized randomly at the start of training and adjusted through learning to identify important features.
  - **Input and Kernel Dimensions:** In a CNN layer, each input $x$ is structured in three dimensions: height, width, and depth. The depth corresponds to the number of channels (e.g., an RGB image has three channels). Similarly, the kernels are also three-dimensional, with spatial dimensions (height and width) and a depth matching the input channels. Each kernel has shared parameters—a set of weights and a bias. When applied to the input, these kernels generate a corresponding set of feature maps. These kernels establish local connections by interacting only with small regions of the input at a time, allowing the network to extract patterns such as edges and textures by computing dot products across these regions.
  - **Convolutional Operation:** The convolutional process begins by sliding the kernel across the input image in both horizontal and vertical directions. At each location, the dot product between the kernel and the overlapping region of the input is computed, producing a scalar value that becomes part of the resulting feature map. As this process is repeated across the image, a full feature map is constructed, highlighting areas where the kernel detects specific patterns. Parameters such as stride (controlling how far the kernel moves at each step) and padding (adding borders to the input to preserve edge information) affect the size and coverage of the output feature map. These concepts are illustrated in Figure 2.4.

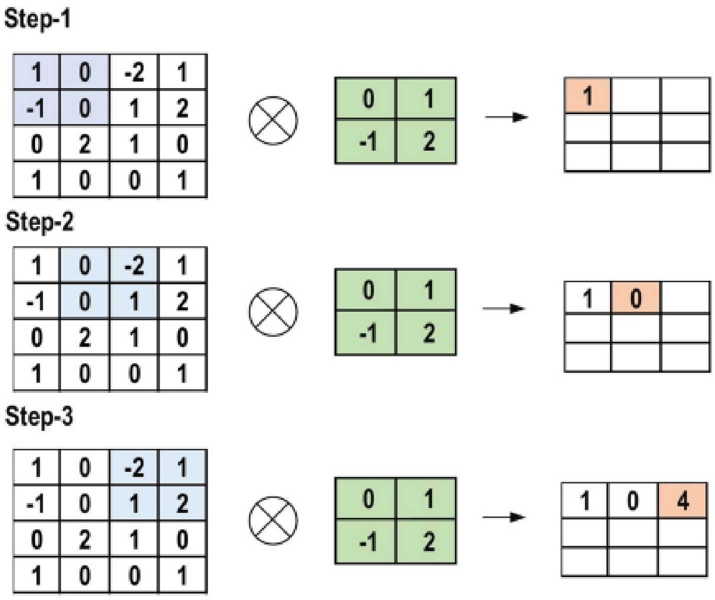



Figure 2.4: The primary calculations executed at each step of the convolutional layer [4].

- **Pooling Layers:** Pooling layers are used to reduce the spatial dimensions of feature maps while preserving the most important information. This reduction helps lower the computational cost and minimizes the risk of overfitting by simplifying the data representation. The pooling operation works by sliding a small filter over the feature map and applying a summary function within each local region (Figure 2.5). Common types of pooling include:

  – **Max Pooling:** Selects the maximum value in each region.

  – **Average Pooling:** Calculates the average value of the region.

  – **Global Average Pooling:** Computes the average across the entire feature map.

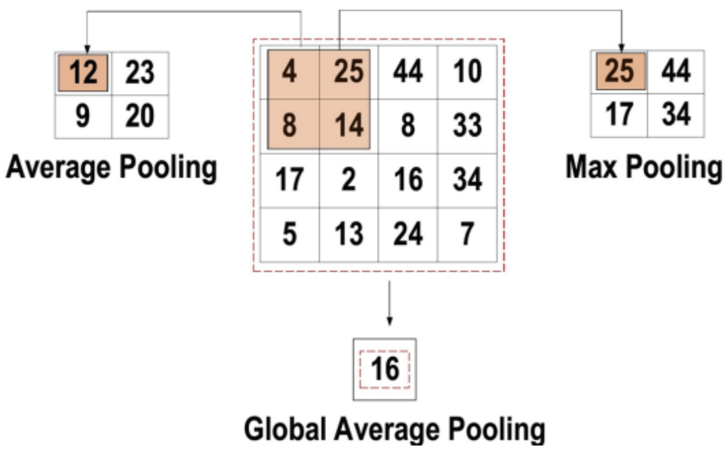



Figure 2.5: Three types of pooling operations [4].

- **Activation Functions:** Introduce non-linearity to help the network learn complex patterns. They must also be differentiable to enable backpropagation during training. CNNs commonly utilize the following activation functions: ReLU (Rectified Linear Unit), Sigmoid, Softmax, and Tanh.

- **Fully Connected Layers:** The Fully Connected (FC) layer is typically found at the end of a CNN architecture and serves as the classifier. In this layer, each neuron is connected to all neurons from the previous layer, following the fully connected approach. It operates similarly to a conventional multi-layer perceptron (MLP) network, which is a type of feed-forward artificial neural network (ANN). The input to the FC layer is a vector created from the feature maps after flattening, which comes from the last pooling or convolutional layer. The output of the FC layer represents the final result of the classification task.

Figure 2.6 below illustrates the general structure of a Convolutional Neural Network (CNN), highlighting its layers, which work together to extract and classify features from input images.

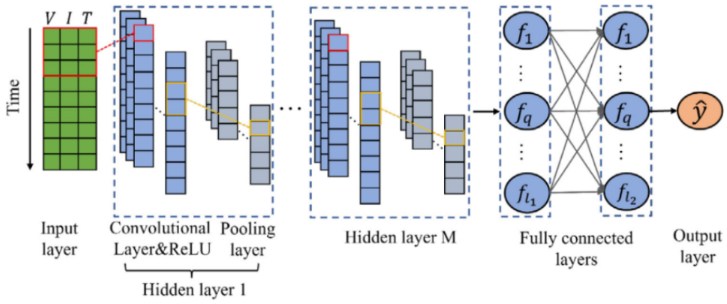



Figure 2.6: The structure of the CNN [7].

### 2.5.2 CNN Architectures for Image Classification

Over the past decade, numerous Convolutional Neural Network (CNN) architectures have been developed, each introducing unique design principles to improve accuracy, efficiency, and scalability. In the context of image classification, especially for tasks such as plant disease detection, the choice of architecture can significantly influence performance depending on the dataset size, complexity, and computational constraints. This section presents an overview of some of the most influential and widely used CNN architectures:

#### Visual geometry group network (VGGNet)

Proposed by Simonyan and Zisserman, VGGNet is a convolutional neural network (CNN) architecture widely recognized for its simplicity and strong performance in image recognition tasks.

VGG is characterized by its deep architecture, typically comprising 16 to 19 layers, which significantly enhances its representational power compared to earlier models like ZFNet and AlexNet. One of its key innovations is the replacement of large convolutional filters (such as 11x11 or 5x5) with multiple stacked 3x3 filters. This strategy maintains an equivalent receptive field while reducing the number of parameters and improving computational efficiency.

In addition, VGG uses 1 x 1 convolutions to control the model's complexity and includes max pooling layers to progressively reduce the spatial dimensions of the feature maps, as illustrated in Figure 2.7.

Despite its effectiveness, a major drawback of VGGNet is its high computational cost, with around 140 million parameters [4]. Nonetheless, its reliable feature extraction capabilities have made it a popular choice in applications like plant disease classification, especially for detecting early-stage or visually subtle symptoms.

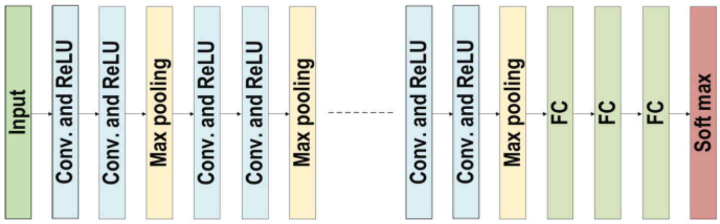



Figure 2.7: The architecture of VGG [4].

### Inception Net (GoogLeNet)

Inception Net, introduced by Szegedy et al., uses Inception modules that apply multiple convolutional filters (1 x 1, 3 x 3, 5 x 5) in parallel, followed by concatenation (see Figure 2.8). This design captures multi-scale information efficiently while reducing computational cost. The architecture has been successfully applied in plant phenotyping and classification of complex disease patterns.

It replaced standard convolutional layers with micro-neural networks and regulated computation through 1 x 1 convolutions as bottleneck layers. Sparse connections addressed redundant information by selectively connecting input and output channels, while the global average pooling (GAP) layer reduced parameters from 40 million to just 5 million, enhancing efficiency. Additional features included the RmsProp optimizer, batch normalization, and auxiliary learners to accelerate convergence [4].

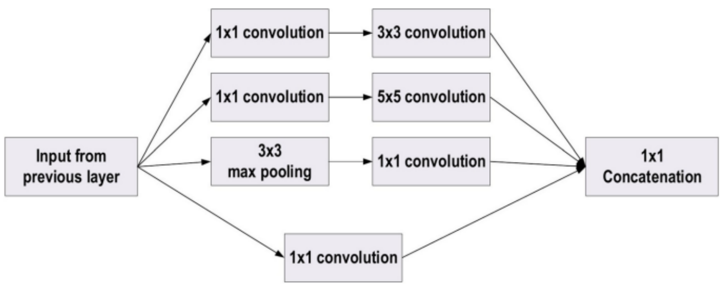



Figure 2.8: The basic structure of Google Block [4].

### Xception Net

The Xception model is an extension of the Inception architecture that replaces standard convolutions with depth-wise separable convolutions, significantly improving efficiency. It has been shown to outperform Inception in many tasks, especially with high-resolution agricultural images, by learning spatial and cross-channel correlations separately.

The core concept behind Xception is the modification of the traditional Inception block by making it wider and replacing the standard 3×3 convolution followed by a 1×1 convolution with depthwise separable convolutions, which reduces computational complexity while enhancing performance [4]. An illustration of the basic Xception block structure is presented in Figure 2.9.

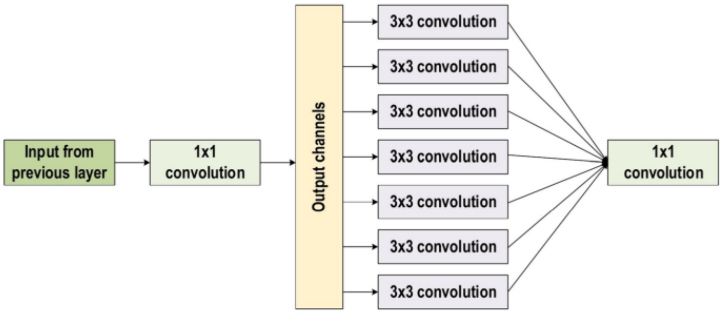



Figure 2.9: The basic block diagram for the Xception block architecture [4].

### Residual Networks (ResNet)

ResNet is a deep convolutional neural network architecture developed to facilitate the training of very deep networks, ranging from 18 to 152 layers, by introducing the concept of residual learning through shortcut (or skip) connections that bypass one or more layers.

These residual connections address the degradation problem that often occurs in deeper networks, where increasing depth leads to performance saturation or even degradation. By enabling gradients to flow more efficiently, ResNet makes it easier for the network to learn identity mappings or residual functions, simplifying the overall training process.

The output of a residual block is defined as:

$$H(x) = F(x) + x \tag{3}$$

Where:

- $H(x)$: the output of the residual block,
- $F(x)$: the residual function to be learned,
- $x$: the input passed through the shortcut connection.

This formulation (3) makes it easier to optimize deep networks by learning the difference (residual) between the desired mapping and the identity. Additionally, it helps increase the rank of the weight matrices, enhancing the network's expressiveness and preventing performance degradation.

An illustration of the residual module structure is provided in Figure 2.10.

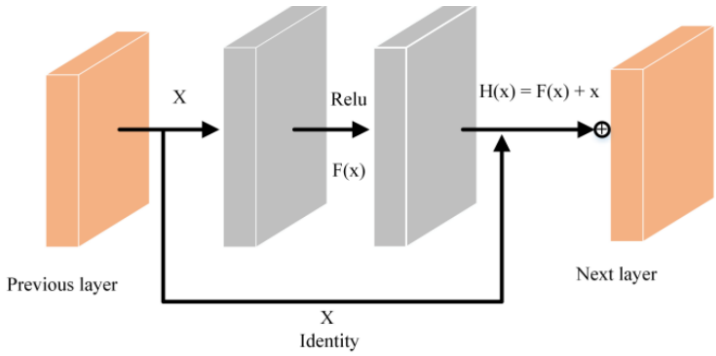



Figure 2.10: Residual module diagram [10].

### DenseNet

DenseNet is a convolutional neural network architecture that introduces the concept of dense connectivity, in which each layer is directly connected to every other layer in a feed-forward fashion, as illustrated in Figure 2.11. This unique design enables feature reuse, enhances gradient flow, and significantly reduces the number of parameters compared to traditional CNNs.

Inspired by ResNet and Highway Networks, DenseNet addresses a key limitation in ResNet, where each layer maintains isolated weights and where certain transformations contribute minimal new information. In contrast, DenseNet concatenates the outputs of all preceding layers and feeds them as input to each subsequent layer.

In a DenseNet with $l$ layers, the number of direct connections between layers is given by:

$$\text{Number of connections} = \frac{l(l+1)}{2} \tag{4}$$

This connectivity pattern facilitates richer feature propagation, introduces a regularization effect, and helps mitigate the vanishing gradient problem, thus improving training efficiency and model generalization [4].

Despite the computational cost introduced by the accumulation of feature maps, DenseNet has shown exceptional performance in fine-grained classification tasks, such as distinguishing between subtly different plant disease symptoms, particularly in scenarios with limited training data.

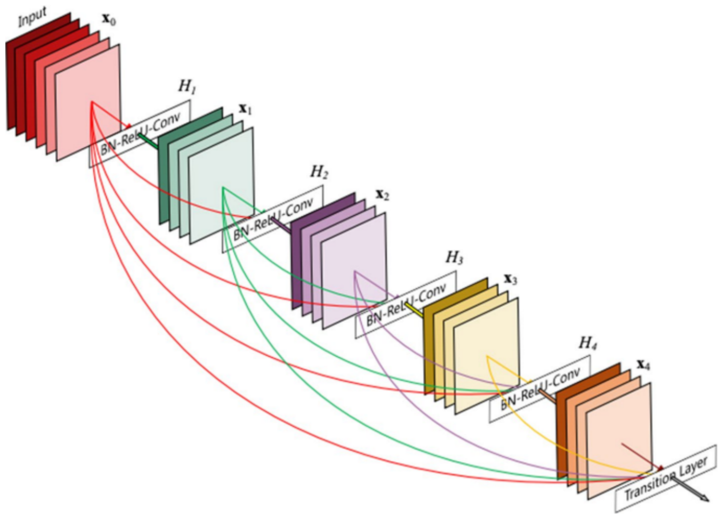



Figure 2.11: The architecture of DenseNet Network [4].

### EfficientNet

EfficientNet is a family of convolutional neural networks developed by Google AI that introduces a compound scaling method to efficiently scale deep learning models. Traditional approaches often scale models arbitrarily in one of three dimensions: depth (number of layers), width (number of channels), or input resolution. However, EfficientNet proposes a more balanced and systematic strategy, where all three dimensions are scaled simultaneously and proportionally using a fixed set of scaling coefficients. This compound approach maintains model efficiency while significantly boosting accuracy.

The baseline model, EfficientNet-B0 (see Figure 2.12), is built using Neural Architecture Search (NAS) to optimize both performance and efficiency. Larger variants (B1 to B7) are derived by uniformly scaling the baseline model using the compound scaling principle. This results in models that achieve state-of-the-art performance on image classification tasks with dramatically fewer parameters and lower computational cost compared to earlier architectures like ResNet or Inception.

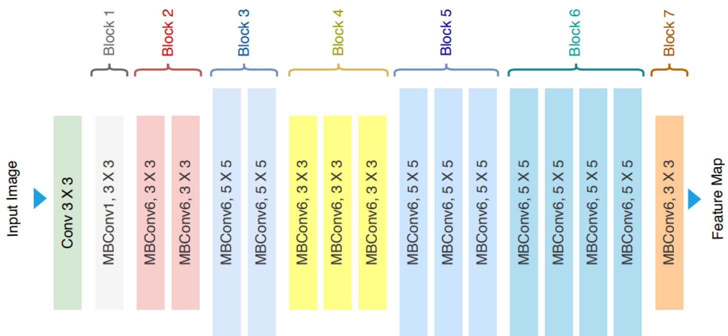



Figure 2.12: Architecture of EfficientNet-B0 with MBConv as Basic building blocks [1].

**Lightweight and Specialized CNN Models**

Several CNN architectures, such as MobileNet, NASNet, SqueezeNet, and ShuffleNet, have been specifically developed to meet constraints related to speed, model size, and power efficiency, making them suitable for deployment on mobile or edge devices. Although not explored in depth within the main text, a comparative overview of their architectures, key characteristics, and potential applications in smart agriculture is presented in 3.5 for further reference.

## 2.6 Transfer Learning and Pretrained Models

In deep learning, training models from scratch often requires large labeled datasets and significant computational resources. Transfer learning offers a powerful alternative by leveraging models pre-trained on large benchmark datasets such as **ImageNet**. These pre-trained models capture rich and generalizable features in their initial layers, which can then be adapted to new, often smaller, target datasets with minimal additional training. This section explores the concept of transfer learning, introduces popular pre-trained CNN architectures relevant to agricultural applications, and outlines fine-tuning strategies suitable for small-scale plant datasets.

### 2.6.1 Concept of Transfer Learning

Transfer learning is a machine learning technique where a model trained on one task is repurposed for a different but related task. In the context of deep learning, it typically involves taking a neural network pre-trained on a large dataset such as ImageNet, which contains over 14 million labeled images, and adapting it to a specific task that may lack sufficient labeled data.

To formalize this, consider a target learning task $T_t$ based on a domain $D_t$; transfer learning allows for assistance from a different domain $D_s$ for the learning task $T_s$. The goal of transfer learning is to improve the performance of the predictive function $f_{T_t}(\cdot)$ for the task $T_t$ by discovering and transferring latent knowledge from $D_s$ and $T_s$, where generally $D_s = D_t$ and/or $T_s = T_t$. Furthermore, it is often the case that the size of $D_s$ is much larger than that of $D_t$ [40].

This process is illustrated in Figure 2.13, which demonstrates how knowledge from a source task and domain can be transferred to a target task with limited data.

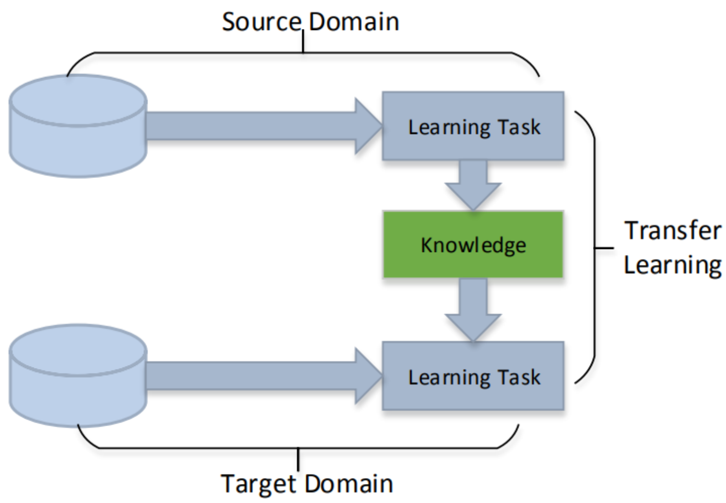



Figure 2.13: Learning process of transfer learning [40].

The two primary approaches to transfer learning are [40]:

- **Feature Extraction:** The pre-trained model is used as a fixed feature extractor. All convolutional layers are kept frozen, and only the final fully connected layer(s) are trained on the new dataset.
- **Fine-Tuning:** Some layers of the pretrained model are unfrozen and retrained on the new dataset. This allows the model to slightly adjust its learned features to better suit the new domain.

This concept is illustrated in Figure 2.14, which demonstrates the integration of a custom CNN with transfer learning networks.

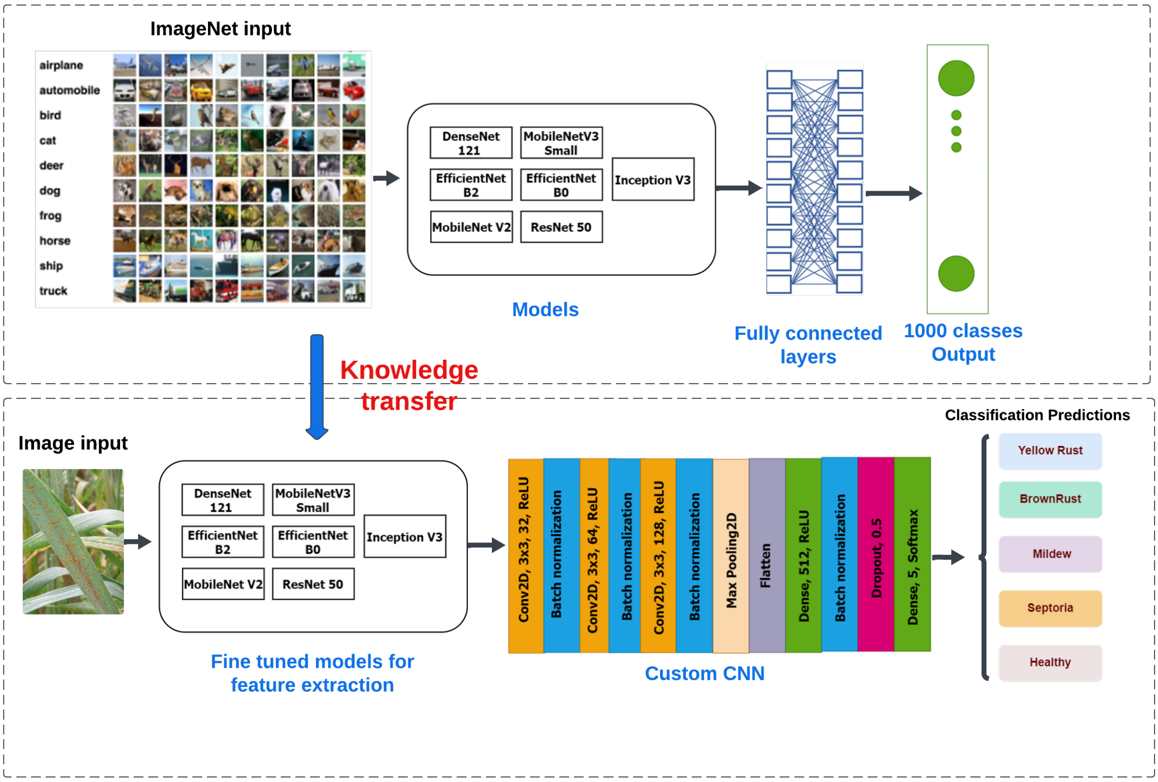



Figure 2.14: Integration of the custom CNN with transfer learning networks [21].

Transfer learning is especially valuable in agriculture, where acquiring large annotated datasets is difficult. By leveraging pre-trained models, researchers and practitioners can build effective models for plant disease detection with limited data and reduced computational cost

### 2.6.2 Fine-Tuning Strategies for Agricultural Data

While basic fine-tuning involves unfreezing and retraining a subset of layers in a pretrained model, fine-tuning strategies can be further optimized when dealing with agricultural data, which often presents unique challenges such as class imbalance, limited samples, and high intra-class variability (e.g., similar symptoms across different plant diseases).

In this context, several fine-tuning strategies can be applied to improve model generalization and performance:

- **Gradual Unfreezing:** Instead of unfreezing all layers at once, layers are incrementally unfrozen, starting from the top (fully connected layers) and moving backward. This helps prevent the model from losing previously learned useful features during early training stages [17].
- **Discriminative Learning Rates:** Assigning different learning rates to different layers, lower for earlier layers and higher for later ones, ensures that the more generic features are preserved while higher-level representations are fine-tuned for the new task [17].

- **Early Stopping and Regularization:** Due to limited data, it's essential to prevent overfitting during fine-tuning. Techniques like early stopping, dropout, and weight decay can help maintain model robustness [28].

- **Data Augmentation and Balancing:** Supplementing fine-tuning with targeted data augmentation (e.g., rotation, brightness adjustments, zoom) helps the model generalize better. In addition, synthetic oversampling techniques like SMOTE can address class imbalance issues [5].

These strategies, when integrated thoughtfully, allow researchers to tailor fine-tuning to the specific nature of agricultural datasets, maximizing the benefits of transfer learning even in data-constrained environments.

## 2.7 Object Detection in Smart Agriculture

In the context of precision agriculture, object detection has emerged as a critical computer vision technique for automating the assessment of crop health. It facilitates the identification and localization of plant diseases, insect pests, nutrient deficiencies, and weeds, supporting more efficient and timely interventions across large-scale farming environments. This approach goes beyond simple image classification by offering spatial information about multiple objects of interest within a single image, enabling actionable insights for decision-making.

### 2.7.1 Key Concepts in Object Detection

Understanding object detection requires familiarity with its key components, including the problem definition, how annotated data is structured, and the metrics used to evaluate model performance.

#### Definition of Object Detection

At its core, object detection involves predicting both the category and precise location of objects within an image, thus combining classification with localization. Traditional approaches consist of stages such as region proposal, feature extraction, and object classification. Over time, detection methods have evolved to address increasing demands for accuracy and speed, giving rise to both two-stage and one-stage detection frameworks [44].

Here are more details about the concepts of object detection [44]:

- **Informative Region Selection:** It is used to identify specific areas within an image where objects are likely to appear. This step helps reduce computational load by focusing only on promising regions instead of scanning the entire image at all scales and positions. In agricultural imagery, objects like plants or pests may vary in size, shape, and location. Early methods used multiscale sliding windows to generate candidate regions, but this approach was computationally expensive and often produced redundant proposals. To improve efficiency, modern techniques now use region proposal algorithms or attention mechanisms to better focus on meaningful areas while avoiding irrelevant ones.

- **Feature Extraction:** It is the process of identifying and isolating relevant visual attributes from an image that can effectively represent objects within it. Object recognition involves detecting characteristics that are both robust and semantically significant. Traditional methods such as Scale-Invariant Feature Transform (SIFT), Histograms of Oriented Gradients (HOG), and Haar-like features were designed to mimic human vision by emphasizing edges, textures, and patterns. However, these handcrafted techniques often struggled to maintain performance due to challenges like changes in object appearance, lighting variations, and cluttered backgrounds, leading to their limitations in complex scenarios.
- **Classification and Localization:** Classification and localization refer to the process of both identifying the object in an image and determining its precise location through bounding boxes. With the rise of deep learning, this step underwent a significant transformation. Models such as R-CNN and its subsequent versions—Fast R-CNN, Faster R-CNN, and YOLO—automated the feature extraction process and seamlessly integrated classification with bounding box regression. These advancements led to substantial improvements in both detection accuracy and processing speed, enabling real-time applications in fields like crop monitoring and pest detection.

### Annotation of Objects

Annotation refers to the process of labeling the objects (such as pests, diseases, or damaged crops) in the images by drawing bounding boxes around them and assigning class labels. This is a critical step for supervised learning, where the model learns to identify patterns based on labeled data. Several annotation techniques can be used:

- **Bounding Boxes:** The most common annotation method in object detection. Each object is enclosed in a rectangular box, and the class of the object is assigned to it (e.g., "rust", "aphid").
- **Polygons:** For more precise object delineation, especially when objects have irregular shapes (e.g., plant leaves affected by disease), polygons are used instead of bounding boxes.
- **Semantic Segmentation:** In cases where the task involves classifying each pixel in the image, semantic segmentation labels each pixel to indicate which class it belongs to (e.g., diseased or healthy tissue in a leaf).

### Evaluation Metrics in Object Detection

In object detection, several metrics are used to assess model performance:

- **Mean Average Precision (mAP):** Measures the average precision across all object classes, balancing precision and recall to evaluate overall model performance.

- **Intersection over Union (IoU):** Calculates the overlap between predicted and ground truth bounding boxes, indicating localization accuracy. Higher IoU means better localization.
- **Precision:** Measures the proportion of true positive detections out of all predicted objects.
- **Recall:** Measures the proportion of true positive detections out of all actual objects.
- **F1-Score:** The harmonic mean of precision and recall, providing a balance between both.
- **Average Recall (AR):** Evaluates recall at different IoU thresholds, useful for detecting small objects or handling occlusions.
- **Confusion Matrix:** Summarizes true positives, false positives, true negatives, and false negatives, providing insights into model errors.
- **Speed Metrics (FPS, Latency):** Assess real-time performance, essential for time-sensitive applications like precision agriculture.
- **AP at Specific IoU Thresholds:** Measures precision at different IoU levels to understand performance under stricter conditions.

### 2.7.2 Key Architectures

Many architectures are designed to efficiently and accurately detect objects in images, even in complex agricultural environments. Below, we discuss four of the most widely used and effective object detection models: YOLO (You Only Look Once), R-CNN, Faster R-CNN, and SSD (Single Shot Multibox Detector).

#### Yolo

YOLO is a fast and efficient object detection framework that predicts both object confidences and bounding boxes (BBs) using the entire topmost feature map. The image is divided into a $S \times S$ grid, where each grid cell is responsible for predicting objects centered within it. Each cell predicts multiple bounding boxes and their corresponding confidence scores, which reflect the likelihood of an object being present and how well the predicted box overlaps with the ground truth (IoU) (see figure 2.15).

At test time, class-specific confidence scores are computed by multiplying the box confidence with conditional class probabilities. YOLO optimizes a loss function during training to fine-tune predictions and improve detection accuracy [44].

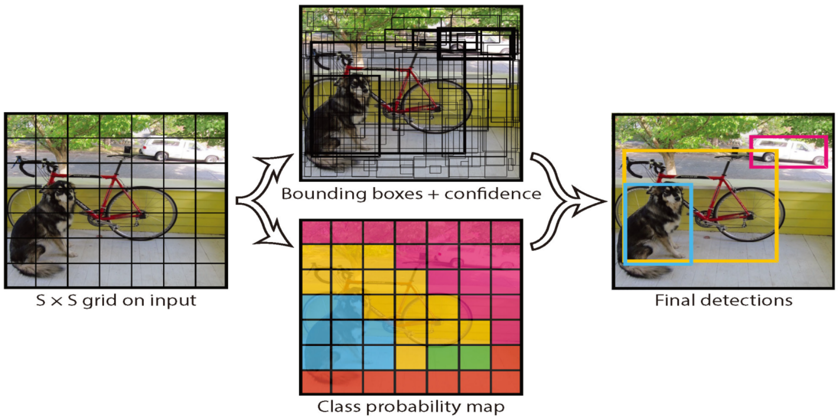



Figure 2.15: Main idea of YOLO [44].

#### R-CNN

R-CNN is a significant advancement in object detection, improving the quality of candidate bounding boxes (BBs) and utilizing deep architecture for high-level feature extraction [44]. It consists of three main stages, as presented in figure 2.16:

**Region Proposal Generation:** R-CNN uses selective search to generate about 2000 region proposals per image, improving candidate box accuracy and reducing the search space.

**CNN-Based Feature Extraction:** Each region proposal is resized and passed through a CNN to extract a 4096-dimensional feature, creating a high-level, robust representation of the object.

**Classification and Localization:** Region proposals are classified using pre-trained linear SVMs, and bounding box regression is applied. Non-maximum suppression (NMS) is used to eliminate redundant boxes and finalize object detections.

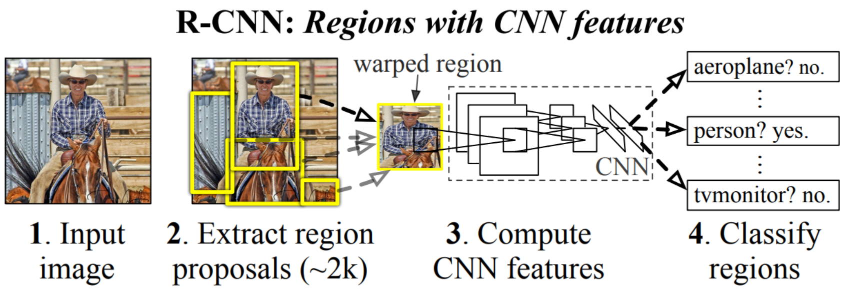



Figure 2.16: Flowchart of R-CNN [44].

Despite its success, R-CNN has drawbacks, including slow inference due to CNN computation for each region, time-consuming multistage training, high memory and storage requirements for storing region features, and redundant region proposals from selective search that slow down the process [44].

### Faster R-CNN

Faster R-CNN improves upon earlier object detection models by introducing a Region Proposal Network (RPN), a deep learning-based method for generating object proposals, which shares convolutional features with the detection network to generate object proposals efficiently, eliminating the need for methods like selective search. The RPN uses a fully convolutional network (FCN) (as presented in Figure 2.17) to predict bounding boxes and object scores simultaneously. The system uses anchors of multiple scales and aspect ratios and is trained end-to-end with a multitask loss function. While Faster R-CNN achieves state-of-the-art accuracy and high-speed processing, it is limited by its alternate training algorithm, which is time-consuming and struggles with extreme object scales and shapes [44].

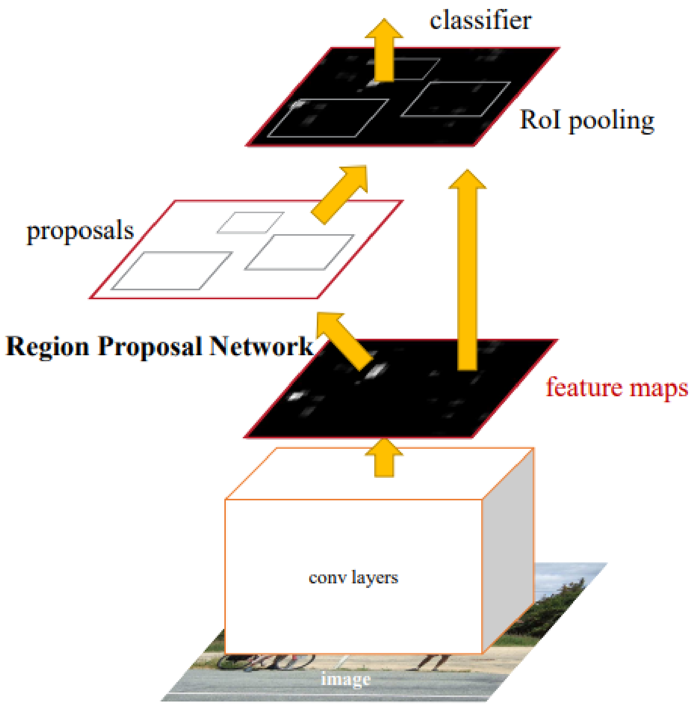



Figure 2.17: An illustration of the Faster R-CNN model [35].

### SSD

SSD was introduced to address the limitations of YOLO, particularly in handling small objects and objects with unusual aspect ratios. Unlike YOLO's fixed grid approach, SSD uses default anchor boxes of various aspect ratios and scales to better handle objects of different sizes.

It integrates predictions from multiple feature maps with different resolutions and uses a VGG16 backbone architecture with additional layers for bounding box predictions. SSD is trained with a combination of localization and confidence losses and refines detections using non-maximum suppression (NMS). It outperforms Faster R-CNN in accuracy on PASCAL VOC and COCO while being three times faster, running at 59 fps with an input size of 300 × 300. However, SSD still struggles with small objects, which can be improved with better feature extractors and network modifications [44].

Below is the architecture of SSD (Figure 2.18), illustrating its key components, including the VGG-16 backbone, extra feature layers, and classifier convolutions.

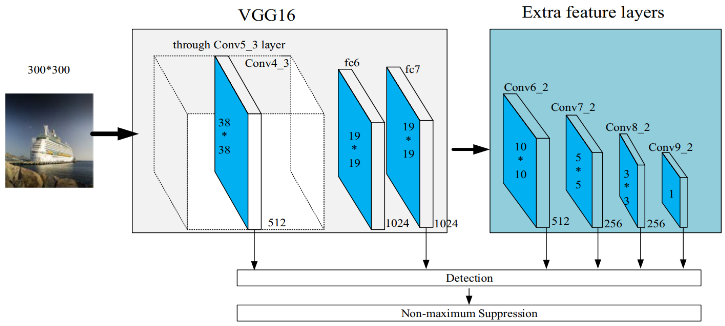



Figure 2.18: Architecture of SSD [27].

## 2.8 Challenges of Deep Learning in Agricultural Contexts

Despite the promising performance of deep learning and object detection in various fields, their application in agriculture presents a range of specific challenges. One of the primary issues is the limited availability of large, annotated agricultural datasets, which hampers the training of robust and generalizable models. Unlike natural image datasets like ImageNet, agricultural datasets often suffer from class imbalance, incomplete labeling, and domain specificity (e.g., crop types, diseases, and environmental conditions) [4].

Another challenge is the high intra-class variability and low inter-class variability found in agricultural images. For instance, symptoms of different diseases may look visually similar, or the same disease may appear differently across plant species and growth stages. Additionally, variations in lighting, occlusion by leaves, overlapping plants, and background clutter significantly affect detection performance [4].

The seasonal and geographic diversity further complicates the generalization of models trained on limited datasets. Moreover, the deployment of deep learning models in real agricultural environments must consider limited computing resources, especially for edge or mobile devices used in fields.

Lastly, ensuring the interpretability and trustworthiness of AI decisions is crucial in agricultural contexts, as these technologies directly impact yield, resource use, and farmers' livelihoods. Addressing these challenges requires collaborative efforts in data collection, annotation, model adaptation, and efficient deployment strategies [4].

## 2.9 Conclusion

In conclusion, this chapter explored the integration of deep learning techniques, especially convolutional neural networks (CNNs), into agricultural contexts, focusing on their role in plant disease classification and object detection. We introduced key CNN architectures such as VG-

GNet, ResNet, DenseNet, and EfficientNet, each contributing unique design principles and performance improvements. Transfer learning was also discussed as an effective strategy to overcome data scarcity by adapting pretrained models to agricultural datasets with minimal resources.

We then highlighted the concept of object detection and its significance in precision agriculture for tasks like identifying pests, diseases, and weeds. Core ideas such as region proposal, feature extraction, and classification were presented, along with modern deep learning-based solutions. Finally, the chapter addressed real-world challenges, such as image variability, environmental noise, and limited annotated data, all of which must be considered when developing robust AI-based agricultural systems. In the following chapter, we will explore hybrid approaches, such as ensemble learning and its integration with CNN models, to further enhance accuracy, generalization, and robustness in agricultural applications.

# Chapter 3

# Review of Image-Based Approaches in Wheat Disease Identification

## 3.1 Introduction

In recent years, image-based approaches leveraging advances in deep learning have emerged as powerful tools for the early detection and classification of wheat diseases. These methods rely on the visual symptoms present on wheat leaves, stems, or spikes, captured through images and analyzed using automated algorithms.

This chapter provides a comprehensive review of existing image-based approaches for wheat disease identification. It begins with an exploration of data collection and preprocessing techniques, including data augmentation strategies to enhance model robustness. The chapter then delves into different classification models and frameworks used for disease recognition. Finally, it highlights the current limitations and identifies research gaps that still need to be addressed to improve the accuracy, scalability, and real-world applicability of these systems.

## 3.2 Approaches in Data Collection and Preprocessing

This section discusses various methods employed in the literature for gathering data, handling challenges such as low-quality or irrelevant images, and preprocessing techniques that prepare datasets for effective utilization in classification and detection tasks.

### 3.2.1 Data Collection

The initial and foundational step in developing deep learning models for wheat disease classification is the acquisition of high-quality and diverse image data. This data can be sourced through various means, including manual and automated methods. Common sources include drone-mounted or mobile device cameras, which allow for the real-time capture of wheat leaf images under natural field conditions. Public datasets also play a crucial role; for instance, the "Wheat Leaf Dataset" available on Kaggle, utilized by [33], and the "Wheat Disease Detection" dataset, employed by [34], offer extensive repositories of labeled images. In addition to these, hybrid data collection strategies are often employed. [16], for example, combined images obtained from field visits, Kaggle datasets, and internet sources to enrich their training data. This diversity in data sources helps improve the robustness and generalization capability of the resulting models.

### 3.2.2 Data Preprocessing

Data preprocessing plays a crucial role in improving the quality and relevance of input images for deep learning models. This process typically begins with the elimination of low-quality images where disease symptoms are poorly visible [41] and the cropping of extraneous image regions to concentrate on areas of diagnostic interest. To further enhance the visual quality, advanced contrast enhancement techniques such as Contrast Limited Adaptive Histogram Equalization (CLAHE), contrast stretching, and the hypercolumn technique (Figure 3.1) are commonly employed to improve image contrast and brightness, as discussed by [34] and [10]. These methods significantly improve image contrast and brightness, making subtle disease features more discernible and thereby facilitating more accurate classification. Additionally, to isolate the region of interest and reduce background noise, methods such as background removal, color thresholding, and edge detection are utilized [2], ensuring that only the most relevant image features are presented to the model.

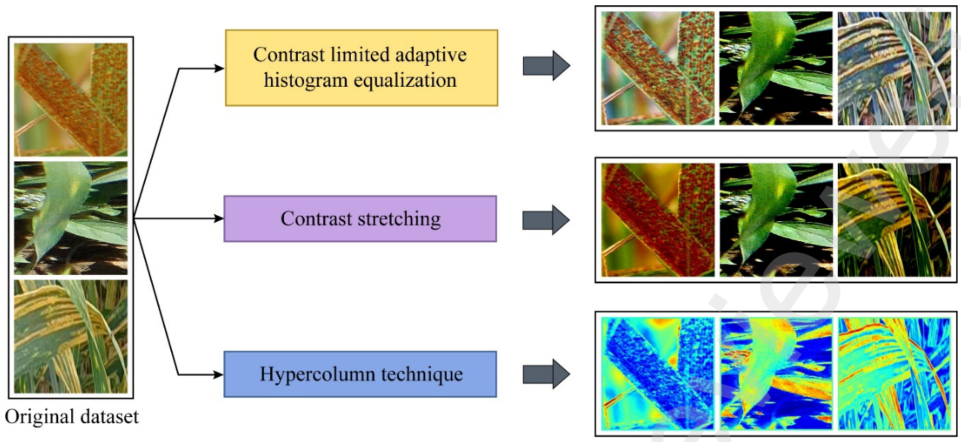



Figure 3.1: Sub-data samples of the wheat disease original dataset obtained by the CLAHE, Contrast stretching, and hypercolumn technique [34].

### 3.2.3 Data augmentation

Data augmentation is extensively employed to mitigate class imbalance and enhance dataset diversity, thereby improving the generalization capabilities of deep learning models. Common augmentation techniques include random rotations, cropping, zooming, flipping, and contrast adjustments (Figure 3.2), which artificially expand the training dataset by introducing varied visual representations of the original images [30, 16, 31].


Figure 3.2: Data augmentation techniques [30].

[33] use four augmentation techniques enhance dataset diversity and address class imbalance: CycleGAN, which generates realistic synthetic images through unpaired image-to-image translation; ADASYN, which adaptively oversamples difficult minority class instances; SMOTE, which creates synthetic samples by interpolating between minority class neighbors; and SMOTE-Tomek, a hybrid method that combines SMOTE with Tomek links to both oversample and remove noisy boundary samples.

## 3.3 Approaches in Wheat Disease Classification

Various classification models have been employed in the literature to address wheat disease classification effectively. These approaches, preprocessing techniques, datasets, classification categories, and results are summarized in Table **??**, which provides a comparative overview of the most recent and relevant studies in this domain.

**Transfer Learning Approaches:** [19] compares three CNN training strategies for wheat disease identification (Figure 3.3): training from scratch, fixed feature extraction, and transfer learning with fine-tuning.

- **Training from scratch:** The entire CNN is trained from the beginning with randomly initialized weights using only the target dataset (FWDI). This approach allows full customization but requires a large amount of data and significant computational power.
- **Fixed feature extraction:** In this strategy, a pre-trained CNN is employed as a fixed feature extractor, where all convolutional layers are frozen to preserve the learned representations, and only the final classification layers—such as the fully connected, batch normalization, and softmax layers are retrained on the target dataset. A similar approach was employed by [31], who fine-tuned a pre-trained EfficientNet model on their custom WheatRust21 dataset for wheat rust classification, achieving an impressive accuracy of 99.35%.
- **Transfer learning with fine-tuning:** This approach involves partially or fully retraining a pre-trained CNN on the target dataset, enabling the model to adapt learned features to the specific characteristics of the new domain. By leveraging representations learned from a large, diverse source dataset, fine-tuning enhances the model's ability to capture domain-specific patterns in the target dataset. [19] applied this strategy, achieving 92.5% accuracy on the FWDI and PlantVillage datasets using InceptionV3. Similarly, [33] fine-tuned pre-trained CNNs such as DenseNet121, ResNet50V2, MobileNetV2, and

Xception for wheat leaf disease classification, using both augmented and non-augmented datasets, and achieved 100% accuracy in predicting three classes (healthy, stripe rust, or septoria), further validating the effectiveness of fine-tuning in this domain.

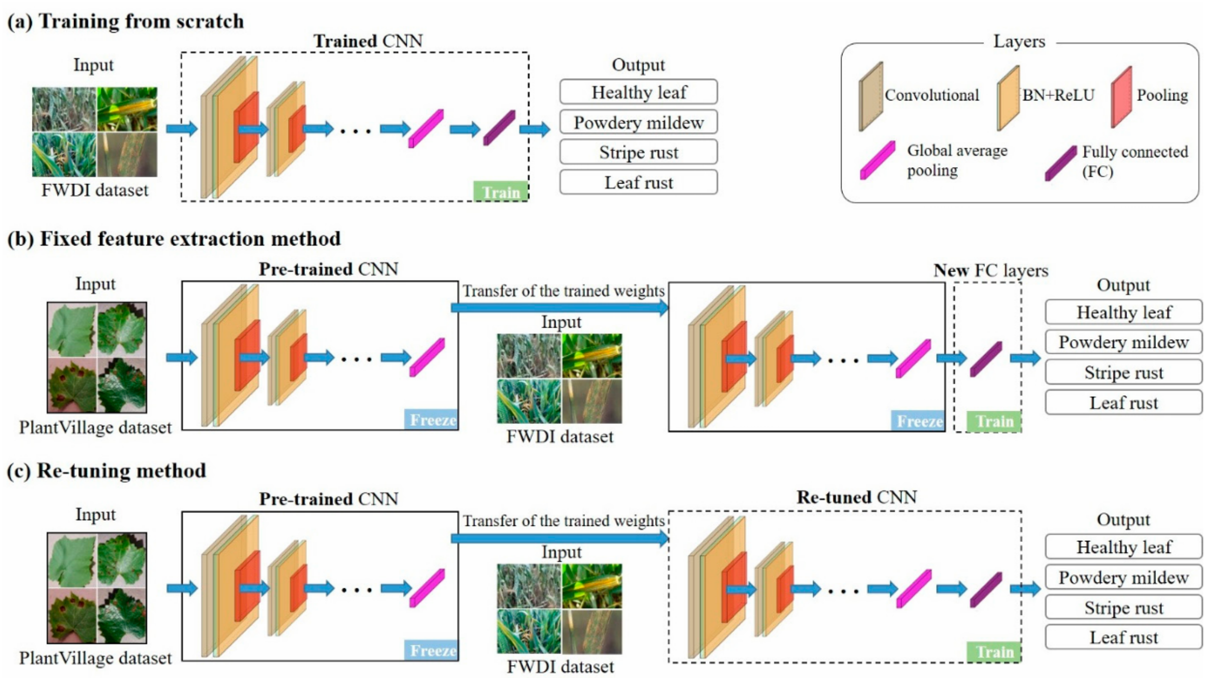



Figure 3.3: Training strategies of a convolutional neural network (CNN) [19].

**Hybrid Models:** [30] Introduced a hybrid model (Figure 3.4) that combines deep learning and traditional machine learning for image classification. Features are extracted from images using two pre-trained CNN models, MobileNet and DenseNet, and then concatenated. Particle Swarm Optimization (PSO) is applied for dimensionality reduction. Finally, the reduced features are fed into traditional classifiers (RF, SVM, DT) for training and prediction, achieving an accuracy of 98.89% across four classes.

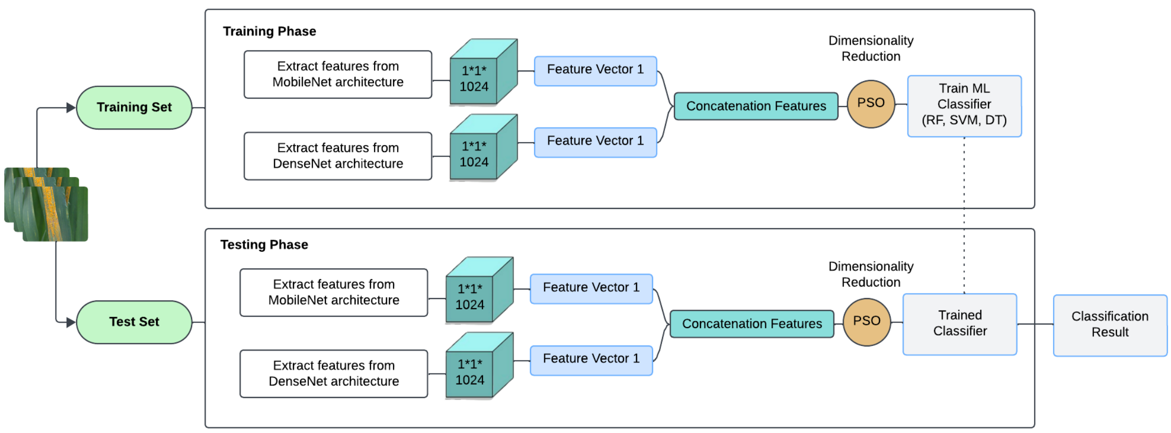



Figure 3.4: The architecture of the hybrid framework [30].

**Custom Deep Convolutional Networks:** [14] proposed a custom deep convolutional neural network (Figure 3.5) specifically designed for wheat disease classification. The architecture comprises 21 convolutional layers, 7 max-pooling layers, and 3 fully connected layers. Activation functions used include ReLU and Leaky ReLU in the convolutional layers, while the fully connected layers employ ReLU, followed by a SoftMax classifier for multi-class output. The model was trained on a dataset of over 12,000 labeled RGB images covering 10 distinct wheat disease classes and achieved an accuracy of 97.88%.

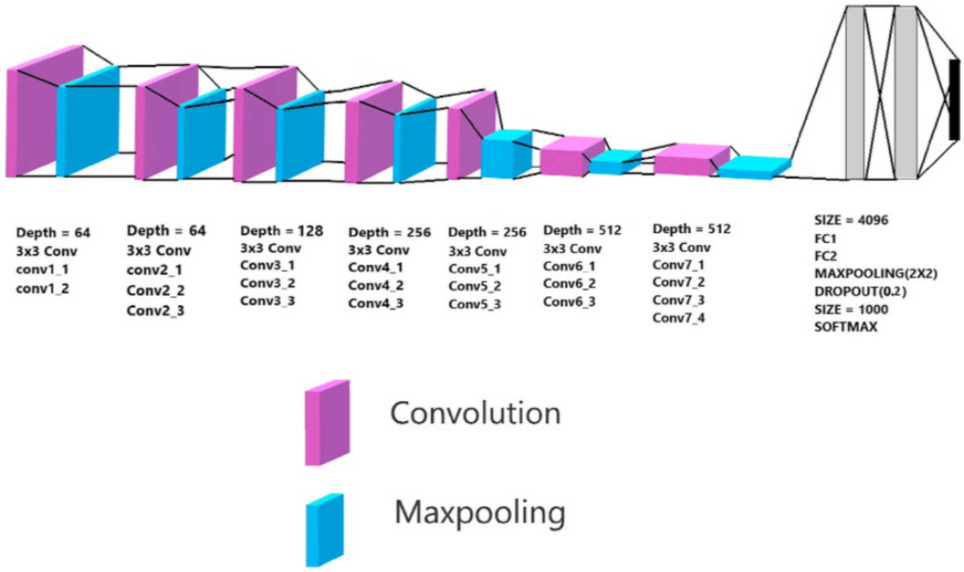



Figure 3.5: The proposed deep convolutional neural network [14].

**Few-Shot Learning:** [2] implemented a few-shot learning approach (Figure 3.6) using a Siamese network with EfficientNetB0 as a shared feature extractor to process both support and query images, generating feature embeddings for each. The support set contains a small number of labeled examples per class, while the query set includes unlabeled images to be classified. The Siamese structure ensures that the same feature extraction process is applied to both sets, enabling meaningful comparison. This model achieved an accuracy of 93.19% in classifying 18 types of wheat diseases.

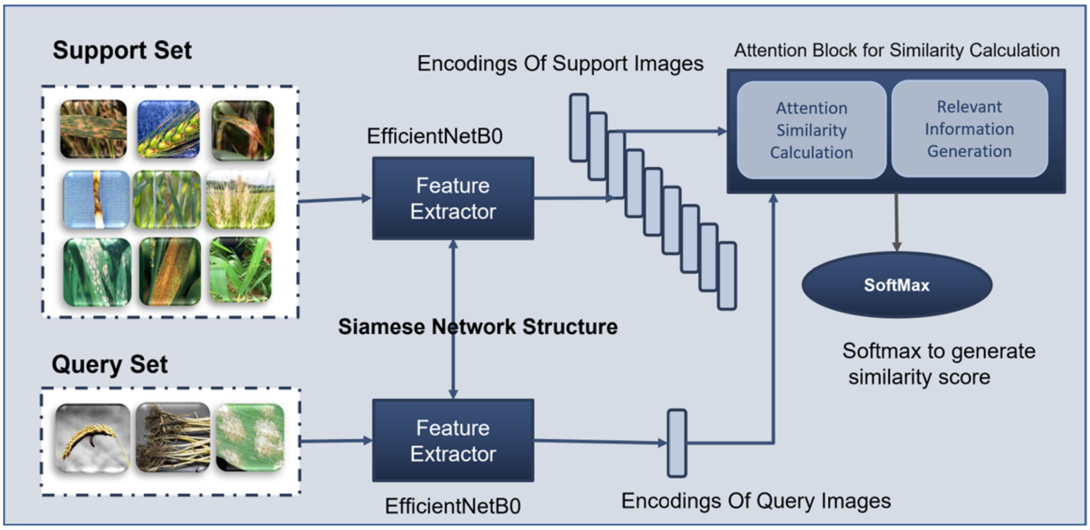



Figure 3.6: Architecture diagram of a Few-shot Learning-based network [2].

**Lightweight CNNs for Efficient Classification:** [10] introduced a lightweight multiscale CNN model optimized for mobile and edge computing applications. By integrating Inception modules with residual blocks and attention mechanisms like Convolutional Block Attention Model (CBAM) and Efficient Channel Attention (ECA), the model enhanced feature extraction while reducing computational costs, achieving 98.78% accuracy on wheat disease classification tasks. Likewise, [20] proposed a deep learning approach using MobileNetV2 and EfficientNet variants to build a lightweight yet accurate CNN-based model for wheat leaf disease detection in smart agriculture, achieving 94% test accuracy.

The studies are complementary as they address different challenges in wheat disease classification. Figure 3.7 summarizes the main approaches used in this field, providing a taxonomy that includes transfer learning, hybrid models, custom CNN architectures, few-shot learning, lightweight CNNs, and attention mechanisms. Transfer learning helps overcome limited data by leveraging pre-trained models, while hybrid models combine deep learning and traditional machine learning to enhance performance. Custom and task-specific CNNs are tailored to the unique features of wheat diseases, improving accuracy. Few-shot learning enables effective classification with minimal labeled data, and lightweight CNNs optimize real-time detection for mobile and edge devices, balancing performance with computational efficiency. Each approach targets a specific problem, enhancing the overall classification process.

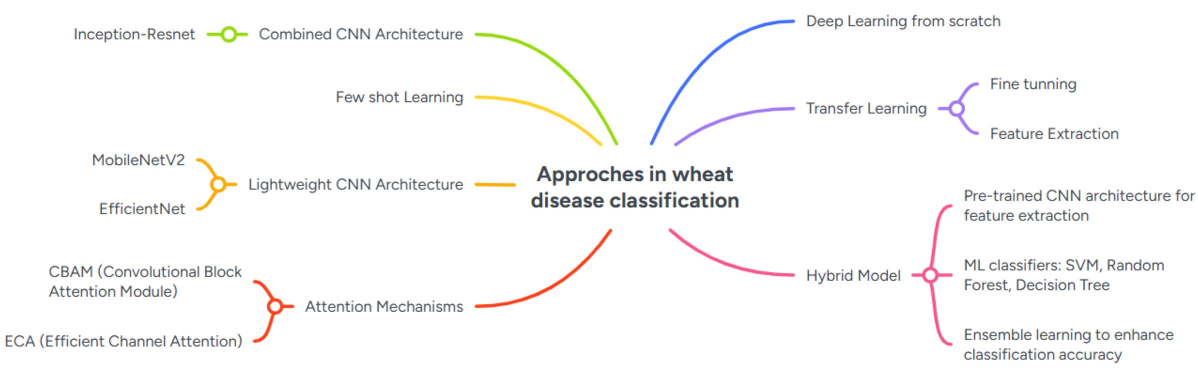



Figure 3.7: Taxonomy of Approaches for Classifying Wheat Diseases. [15]

Table 3.1: Related Work on Wheat Disease Classification

| **Reference** | **Approach** | **Image Preprocessing** | **Size (images)** | **Categories** | **Dataset** | **Results** |
|---|---|---|---|---|---|---|
| [33] | Pre-trained CNNs (DenseNet121, ResNet50V2, MobileNetV2, Xception) | Data Augmentation (CycleGAN, ADASYN, SMOTE, SMOTETomek) | 407 | 3 (healthy, stripe rust, septoria) | Wheat Leaf Dataset | CycleGAN-augmented + MobileNetV2: 100% accuracy |
| [30] | Hybrid (MobileNet + DenseNet + PSO + SVM/DT/RF) | Resize 224x224, normalization, augmentation (cropping, flipping, rotation, noise, zooming) | 887 | 4 (yellow rust, brown rust, loose smut, healthy) | Custom Dataset | 98.89% |
| [34] | Deep models + Ensemble learning | CLAHE, hypercolumn | 2400 | 3 (healthy, yellow rust, brown rust) | Wheat Disease Detection Dataset | 99.72% |
| [14] | Improved deep CNN architecture | / | 12,000 | 10 (e.g., karnal bunt, powdery mildew, healthy...) | LWDCD2020 | 97.88% |
| [2] | Few-shot learning with EfficientNet and attention | / | 1530 | 18 classes | PlantVillage, CGIAR, manual, Google Images | 93.19% |
| [21] | CropNet (EfficientNetB0 + shallow CNN layers) | Resize 256x256, StandardScaler normalization | / | 5 (healthy, septoria, mildew, brown/yellow rust) | / | 99.80% |
| [10] | Inception-ResNet-CE with CBAM/ECA attention | Contrast enhancement, data augmentation | >12,000 | 7 (healthy, rusts, powdery, smut...) | LWDCD2020, PlantVillage, CGIAR, Wheat Leaf | 98.76% |
| [31] | EfficientNet transfer learning | Contrast enhancement, resize, augmentation | 6556 | 4 (stripe rust, stem rust, leaf rust, healthy) | WheatRust21 | 99.35% |
| [19] | 7 CNNs evaluated (e.g., VGG-16, ResNet-50) | Resize 224x224, normalization, geometric augmentation | 2643 | 4 (healthy, mildew, rusts) | FWDI, PlantVillage | 92.5% |

## 3.4 Core Limitations and Research Gaps in Wheat Disease and Pest Detection

Deep learning has shown great promise in automating wheat disease detection, offering timely and precise solutions for sustainable agriculture. However, key challenges still limit its widespread adoption.

Based on the previously discussed studies, various approaches have been explored for wheat disease classification, ranging from transfer learning using pre-trained convolutional neural networks (CNNs) to the development of custom architectures tailored specifically for disease classification tasks. The proposed models for wheat disease classification that detect five or fewer disease classes generally achieve accuracies of 99

Additionally, the number of disease classes addressed in most proposed models for wheat detection remains limited compared to the actual variety of diseases observed in the field, with a maximum of five classes, primarily due to the difficulty of manual annotation and the high computational resources required to train models on larger, more diverse datasets. Although several public datasets for wheat disease classification are available, such as those on Kaggle, these sources often contain low-quality images, inconsistent labeling, or irrelevant data, which can hinder model training and performance. This necessitates manual cleaning and preprocessing of the datasets before they can be effectively utilized, increasing the time and effort needed for research.

Moreover, the performance of these models is often affected by inconsistencies in image data. Therefore, diversifying image data is crucial; photos should be taken under varying lighting conditions, such as on sunny and cloudy days, and from different distances, including both close-up and wider shots. Creating a custom dataset by selecting diverse images from different public datasets is important to ensure this variability and improve model robustness.

Beyond dataset issues, several structural and architectural limitations remain. Most current models use a single-stage classification pipeline that attempts to identify all classes in one pass. This approach can be inefficient, especially when simple distinctions (e.g., healthy vs. diseased) could be resolved earlier using lightweight inference mechanisms. The lack of hierarchical or multi-stage classification pipelines presents an opportunity for improvement in computational efficiency and deployment readiness.

Furthermore, few models explicitly address deployment constraints such as low computational power, memory limitations, or real-time processing requirements, which are crucial for applications in edge computing or mobile agricultural tools used directly in the field. The absence of design considerations for lightweight architectures and early-exit strategies limits the practical applicability of many existing solutions.

Another significant limitation is the lack of evaluation on real-world field images that exhibit high variability in terms of background, illumination, occlusion, and resolution. While many models achieve excellent performance on curated datasets, their generalization to uncontrolled field conditions remains underexplored. Robustness under such conditions is essential for real-life adoption by farmers and agronomists.

Finally, although attention mechanisms and hybrid models have been proposed in recent literature, few studies explicitly quantify the trade-offs between accuracy and computational cost in multi-class settings, especially when scaling up to cover a wider range of diseases.

## 3.5 Conclusion

While deep learning has shown strong potential in wheat disease detection, a major limitation across current studies is the restricted number of disease classes addressed—often no more than five. This narrow scope limits the practical applicability of these models in real agricultural environments, where a wider range of diseases must be identified accurately. The difficulty of collecting and annotating diverse, high-quality images, along with the computational cost of training on large-scale multi-class datasets, remains a significant barrier. Additionally, many existing models struggle to maintain high accuracy when scaled to include more disease categories.

These challenges highlight the need for systems capable of handling a greater number of classes without compromising accuracy. In the next chapter, we present the design and implementation of our proposed solution, which is specifically developed to support accurate multi-class classification and improved generalization to real-world conditions.

# Part II

# Contribution

# General conclusion

The general conclusion goes here.

# Bibliography


[1] T. Ahmed and N. H. N. Sabab. "Classification and understanding of cloud structures via satellite images with EfficientUNet". In: *SN Computer Science* 3.1 (2022), p. 99.

[2] A. Alharbi, M.U.G. Khan, and B. Tayyaba. "Wheat disease classification using continual learning". In: *IEEE Access* (2023).

[3] E. Alvarez-Vanhard, T. Corpetti, and T. Houet. "UAV & satellite synergies for optical remote sensing applications: A literature review". In: *Science of Remote Sensing* 3 (2021), p. 100019. DOI: `10.1016/j.srs.2021.100019`.

[4] Laith Alzubaidi et al. "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions". In: *Journal of Big Data* 8 (2021), pp. 1–74.

[5] N.R. Aquino et al. "The effect of data augmentation on the performance of convolutional neural networks". In: *Brazilian Society of Computational Intelligence* 10 (2017).

[6] Yoshua Bengio, Ian Goodfellow, and Aaron Courville. *Deep Learning*. Vol. 1. Cambridge, MA, USA: MIT Press, 2017, pp. 23–24.

[7] G. Crocioni et al. "Li-ion batteries parameter estimation with tiny neural networks embedded on intelligent IoT micro-controllers". In: *IEEE Access* 8 (2020), pp. 122135–122146. DOI: `10.1109/ACCESS.2020.3007046`.

[8] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri. "Activation functions in deep learning: A comprehensive survey and benchmark". In: *Neurocomputing* 503 (2022), pp. 92–108.

[9] Etienne Duveiller et al. *Wheat Diseases and Pests: A Guide for Field Identification*. CIMMYT, 2012.

[10] X. Fang, T. Zhen, and Z. Li. "Lightweight Multiscale CNN Model for Wheat Disease Detection". In: *Applied Sciences* 13.9 (2023), p. 5801.

[11] U. B. Farook et al. "A Review on Insect Pest Complex of Wheat (Triticum aestivum L.)" In: *Journal of Entomology and Zoology Studies* 7.1 (2019), pp. 1292–1298.

[12] Manuel Figueroa, Kim E. Hammond-Kosack, and Peter S. Solomon. "A Review of Wheat Diseases—A Field Perspective". In: *Molecular Plant Pathology* 19.6 (2018), pp. 1523–1536.

[13] S. Ghazal, A. Munir, and W. S. Qureshi. "Computer Vision in Smart Agriculture and Precision Farming: Techniques and Applications". In: *Artificial Intelligence in Agriculture* (2024).

[14] L. Goyal et al. "Leaf and Spike Wheat Disease Detection & Classification Using an Improved Deep Convolutional Architecture". In: *Informatics in Medicine Unlocked* 25 (2021), p. 100642.

[15] W. Haider et al. “A Generic Approach for Wheat Disease Classification and Verification Using Expert Opinion for Knowledge-Based Decisions”. In: *IEEE Access* 9 (2021), pp. 31104–31129. DOI: 10.1109/ACCESS.2021.3059421.

[16] A. Hassan et al. “Wheat Leaf Localization and Segmentation for Yellow Rust Disease Detection in Complex Natural Backgrounds”. In: *Alexandria Engineering Journal* 107 (2024), pp. 786–798.

[17] M.I. Hossen et al. “Transfer learning in agriculture: a review”. In: *Artificial Intelligence Review* 58.4 (2025), p. 97.

[18] G. Idoje, T. Dagiuklas, and M. Iqbal. “Survey for Smart Farming Technologies: Challenges and Issues”. In: *Computers & Electrical Engineering* 92 (2021), p. 107104.

[19] J. Jiang et al. “Evaluation of diverse convolutional neural networks and training strategies for wheat leaf disease identification with field-acquired photographs”. In: *Remote Sensing* 14.14 (2022), p. 3446.

[20] O. Jouini, K. Sethom, and R. Bouallegue. “Wheat leaf disease detection using CNN in Smart Agriculture”. In: *2023 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE. June 2023, pp. 1660–1665.

[21] O. Jouini et al. “Wheat Leaf Disease Detection: A Lightweight Approach with Shallow CNN Based Feature Refinement”. In: *AgriEngineering* 6.3 (2024), pp. 2001–2022.

[22] P. L. Kashyap et al. *Identification Guide for Major Diseases and Insect-Pests of Wheat*. Tech. rep. 18. Technical Bulletin, 2018.

[23] J. D. Kelleher. *Deep Learning*. MIT Press, 2019.

[24] Nikhil Ketkar and Eloi Santana. *Deep Learning with Python*. Vol. 1. Springer, 2017.

[25] Andrej Krenker, Janez Bešter, and Andrej Kos. “Introduction to the Artificial Neural Networks”. In: *Artificial Neural Networks: Methodological Advances and Biomedical Applications*. InTech, 2011, pp. 1–18.

[26] Pawan Kumar et al. “Estimation of winter wheat crop growth parameters using time series Sentinel-1A SAR data”. In: *Geocarto International* 33.9 (2018), pp. 942–956.

[27] A. Li et al. “Water surface object detection using panoramic vision based on improved single-shot multibox detector”. In: *EURASIP Journal on Advances in Signal Processing* 2021 (2021), pp. 1–15.

[28] J. Li et al. “Implicit sparse regularization: The impact of depth and early stopping”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 28298–28309.

[29] Y. R. Mehta. *Wheat Diseases and Their Management*. Springer, 2014. DOI: 10.1007/978-3-319-06477-0.

[30] J. Nagpal, L. Goel, and P.S. Shekhawat. “A hybrid deep learning neural network for early plant disease diagnosis using a real-world Wheat–Barley vision dataset: challenges and solutions”. In: *International Journal of Data Science and Analytics* (2024), pp. 1–22.

[31] S. Nigam et al. “Deep Transfer Learning Model for Disease Identification in Wheat Crop”. In: *Ecological Informatics* 75 (2023), p. 102068.

[32] P. Purwono et al. “Understanding of convolutional neural network (CNN): A review”. In: *International Journal of Robotics and Control Systems* 2.4 (2022), pp. 739–748.

[33] S. T. Y. Ramadan et al. “Improving Wheat Leaf Disease Classification: Evaluating Augmentation Strategies and CNN-Based Models With Limited Dataset”. In: *IEEE Access* (2024).

[34] H. C. Reis and V. Turk. “Integrated Deep Learning and Ensemble Learning Model for Deep Feature-Based Wheat Disease Detection”. In: *Microchemical Journal* 197 (2024), p. 109790.

[35] Shaoqing Ren et al. “Faster R-CNN: Towards real-time object detection with region proposal networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (2016), pp. 1137–1149.

[36] Z. Ren and C. Du. “A review of machine learning state-of-charge and state-of-health estimation algorithms for lithium-ion batteries”. In: *Energy Reports* 9 (2023), pp. 2993–3021.

[37] Uferah Shafi et al. “A multi-modal approach for crop health mapping using low altitude remote sensing, internet of things (IoT) and machine learning”. In: *IEEE Access* 8 (2020), pp. 112708–112724.

[38] I. Sharma et al. “Enhancing Wheat Production—A Global Perspective”. In: *The Indian Journal of Agricultural Sciences* 85.1 (2015), pp. 03–13.

[39] J. Singh et al. “Important Wheat Diseases in the US and Their Management in the 21st Century”. In: *Frontiers in Plant Science* 13 (2023), p. 1010191.

[40] C. Tan et al. “A survey on deep transfer learning”. In: *Artificial Neural Networks and Machine Learning – ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4–7, 2018, Proceedings, Part III*. Springer International Publishing, 2018, pp. 270–279.

[41] X. Yao, F. Yang, and J. Yao. “YOLO-Wheat: A Wheat Disease Detection Algorithm Improved by YOLOv8s”. In: *IEEE Access* (2024).

[42] D. Zhang et al. “Precision Agriculture: Temporal and Spatial Modeling of Wheat Canopy Spectral Characteristics”. In: *Agriculture* 15.3 (2025), p. 326. DOI: 10.3390/agriculture15030326.

[43] Hong Zhao et al. “Research on a learning rate with energy index in deep learning”. In: *Neural Networks* 110 (2019), pp. 225–231.

[44] Zhi-Qi Zhao et al. “Object detection with deep learning: A review”. In: *IEEE Transactions on Neural Networks and Learning Systems* 30.11 (2019), pp. 3212–3232.

# Appendix A

# Dependencies and libraries

```
annex a
```