B2-GraftingNet: A Hybrid Deep-Machine Learning Framework with Explainable AI for Automated Grape Leaf Disease Detection

RESEARCH ARTICLE

B2-GraftingNet: A Hybrid Deep-Machine Learning Framework with Explainable AI for Automated Grape Leaf Disease Detection

Syed Baqir Hussain Shah[1], Farwa Naseer[2], Syed Adil Hussain Shah[3,4], Kashif Razzaq[5], Tahir Javed[6], Qandeel Asghar[7], Gohar Bano Zaidi[4], Giacomo Di Benedetto[8], Syed Taimoor Hussain Shah[4,*], Syed Bilal Hussain[9,*] and Marco Agostino Deriu[4]

[1] Department of Computer Science, COMSATS University Islamabad (CUI), Wah Campus, Wah 47000, Pakistan
[2] Department of Computer Science, Muhammad Nawaz Sharif University of Agriculture, Multan 60800, Pakistan
[3] Department of Research and Development (R&D), GPI SpA, Trento 38123, Italy
[4] PolitoBIOMed Lab, Department of Mechanical and Aerospace Engineering, Politecnico di Torino, Turin 10129, Italy
[5] Department of Horticulture, Muhammad Nawaz Sharif University of Agriculture, Multan 60800, Pakistan
[6] Department of Computer Science, Iqra University, Islamabad 44000, Pakistan
[7] Department of Computer Science and Information Technology (DoCSIT), University of Southern Punjab, Multan 60800, Pakistan
[8] 7HC SRL, Rome 00198, Italy
[9] School of Agriculture and Food Science, University College Dublin, Dublin 4, Ireland

Abstract

Plant diseases increasingly threaten global agriculture due to climate change, yet manual diagnosis remains challenging. We introduce B2-GraftingNet, a lightweight deep-learning framework for automated grape-leaf disease detection that combines a VGG16 backbone with Inception-style blocks to learn robust multi-scale cues. Binary Particle Swarm Optimization selects the most informative features before classification. On the public Kaggle grape-leaf dataset, a cubic SVM classifier achieves 99.56% peak accuracy, surpassing standard pretrained CNNs (VGG16/VGG19: 34.04%, Xception: 97.95%, Darknet: 94.91%, ResNet-50: 98.44%) while being faster and lighter. For transparency, we incorporate Grad-CAM, LIME, and occlusion-sensitivity with a local 20B-parameter LLM via Ollama, which processes ROI-aware XAI summaries and provides plain-language guidance in structured sections (verdict, where to look, what to do next, caution) without suggesting specific treatments.

Keywords: deep learning, grape diseases, binary particle swarm optimization, XAI, LLM-guided interpretation.

Submitted: 25 February 2026
Accepted: 30 March 2026
Published: 31 March 2026

Vol. 2, **No.** 1, 2026.
10.62762/JIAP.2026.937901

***Corresponding authors:**
✉Syed Taimoor Hussain Shah
taimoor.shah@polito.it
✉Syed Bilal Hussain
syedbilal.hussain@ucd.ie

Citation
Shah, S. B. H., Naseer, F., Shah, S. A. H., Razzaq, K., Javed, T., Asghar, Q., Zaidi, G. B., Benedetto, G. D., Shah, S. T. H., Hussain, S. B., & Deriu, M. A. (2026). B2-GraftingNet: A Hybrid Deep-Machine Learning Framework with Explainable AI for Automated Grape Leaf Disease Detection. *ICCK Journal of Image Analysis and Processing*, 2(1), 27–52.

1 Introduction

Grapevines (Vitis vinifera) rank among the world's most widely cultivated fruit crops, underpinning a global industry valued at over USD 215 billion and contributing approximately 72.5 million tonnes in 2023, despite production challenges driven by extreme weather conditions and pest pressures [1]. Foliar diseases such as black rot, esca (black measles), leaf blight, and downy mildew can each inflict yield losses ranging from 5 to 80%, depending on the severity and timing of infection [2, 3]. Early and precise diagnosis of these diseases in vineyards is therefore a critical priority for sustaining both productivity and fruit quality.

Typically, vineyard health monitoring relies on field experts conducting visual inspections of leaves, shoots, and clusters, a process that is inherently labor-intensive, time-consuming and subject to interoperator variability [4–6]. As farm sizes expand and labour costs rise, manual scouting alone can no longer meet the demand for timely disease surveillance. This has spurred rapid advances in smart agriculture technologies, including unmanned aerial vehicles, multispectral imaging, and Internet of Things (IoT) sensor networks, but translating raw field data into actionable disease alerts remains challenging under variable lighting, complex backgrounds, and mixed phenological stages [7, 8].

In recent years, considerable research efforts have been directed towards machine vision applications in agriculture, spanning a diverse range of areas, such as fruit maturity classification and quality rating [9], fruit disease diagnosis, plant pest identification [10], plant species classification [11], fruit identification within harvesting robots [12], weed control and recognition [13], and disease diagnosis and classification in plant organs [14].

The development of new models is facilitated by deep convolutional neural networks capable of directly processing images. Detecting plant diseases automatically under field conditions poses considerable challenges due to various factors, including complex backgrounds, natural lighting conditions, variations in plant phenological stages, and diverse symptom presentations [15]. The integration of cutting-edge technologies, including IoT [16], big data analytics [17], artificial intelligence (AI) [17, 18], remote sensing, satellite imagery, and UAVs [19, 20], has propelled agriculture to new heights.

Cai et al. [21] introduced Siamese DWOAM-DRNet, an advanced deep learning model for classifying grape leaf diseases in complex natural scenes. Using an enhanced dataset of 1,209 images preprocessed with binary wavelet transform, variable thresholding, and Non-Local Means Multi-Scale Retinex (NL-MSR) [22, 23] optimization techniques, and further augmented for diversity, the model combines a Double-Factor Weight Optimization Attention Mechanism and a Diverse-Branch Residual Module for robust feature extraction. A joint loss function improved class separation and training convergence. The model achieved 93.26% accuracy and 93.23% F1 score, outperforming classical CNNs and recent methods, with added interpretability through Grad-CAM++ [24] visualizations and strong generalization on external datasets.

Javidan et al. [25] proposed an interpretable, machine vision-based method for grape leaf disease classification using the PlantVillage dataset, which targets black measles, black rot, leaf blight, and healthy leaves. Their pipeline combined K-means clustering for ROI segmentation, multi-color space feature extraction (GLCM, HOG, and LBP), PCA for dimensionality reduction, and Relief for feature selection, followed by classification with a linear SVM. The method achieved a high accuracy of 98.97%, surpassing the CNN and GoogLeNet [26] benchmarks, and maintained fast processing times. By ranking 30 key features, the approach ensures transparency and practicality for real-world deployment, especially in resource-constrained environments.

Subramanya et al. [27] investigated deep learning-based automation for grape leaf disease diagnosis using EfficientNet variants (B0, B5, B7) trained via transfer learning on a large Kaggle dataset of 9,027 images covering black rot, esca, leaf blight, and healthy leaves. After preprocessing and data augmentation, EfficientNetB0 delivered the best performance, with 92.71% training accuracy and 96.73% validation accuracy, outperforming B5 and B7 while offering faster convergence and minimal overfitting, making it practical for resource-limited scenarios. Although the study did not report precision, recall, F1 score, or Cohen's Kappa and lacked explainable AI integration, it demonstrated EfficientNetB0's strong potential for accurate grape disease classification, with recommendations for future real-world testing and extended training.

Kaur et al. [28] presented a semi-automated grape leaf disease detection system that combines

optimized K-means segmentation with Grey Wolf Optimization [29]; hybrid feature extraction (Law's masks, GLCM, LBP, Gabor); and an ensemble of ANN, SVM, KNN, logistic regression, and Naïve Bayes classifiers. The PlantVillage dataset achieved 95.69% accuracy, outperforming traditional and CNN methods while maintaining interpretability through explicit feature use, although XAI tools such as Grad-CAM were not used. They highlighted plans for real-field adaptation and a web-based tool for farmers.

Despite substantial progress in image-based grape leaf disease recognition, many existing mobile or GUI-based systems still struggle to capture fine-grained, disease-specific patterns and rarely provide end-to-end, explainable support for growers. To address these gaps, we propose B2-GraftingNet, a hybrid framework inspired by B4-GraftingNet [30] that combines a customized CNN (VGG16 backbone with Inception-style branches) and Binary Particle Swarm Optimization (BPSO) to learn compact, discriminative feature representations. Classical classifiers trained on these optimized features, particularly a cubic SVM, achieve 99.56% accuracy with consistent precision and recall, surpassing several standard pretrained CNN baselines on the same dataset. Beyond accuracy, B2-GraftingNet integrates ROI-aware XAI (Grad-CAM, LIME, occlusion sensitivity) with a local large language model–based explanation module (gpt-oss:20b via Ollama [31]) that converts multi-view explanations into concise, grower-friendly guidance, delivered through a lightweight, deployable web service.

Contributions:

1. We introduce B2-GraftingNet, a hybrid framework inspired by B4-GraftingNet that integrates a customized CNN with metaheuristic optimization for reliable, automated detection of grape leaf diseases.
2. We design a task-specific feature extraction scheme that combines a VGG16 backbone with Inception-style branch modules, producing a compact and discriminative deep feature space.
3. We apply Binary Particle Swarm Optimization (BPSO) to select the most informative deep features, reducing redundancy and computational cost while preserving high separability for classical classifiers.
4. We demonstrate, through comparisons with state-of-the-art learning techniques, that B2-GraftingNet achieves superior accuracy and efficiency for grape leaf disease recognition.
5. We integrate ROI-aware XAI (Grad-CAM, LIME, occlusion) with a local language model (gpt-oss:20b via Ollama) and an end-to-end MATLAB + Flask + Ollama pipeline, delivering grower-focused guidance and deployable web services for practical field use, with the language model strictly limited to explaining the latest image's analysis results (off-topic queries are refused).

The structure of this paper is organized as follows: Section 2 presents the materials and methodology used for the proposed grape leaf disease detection framework, detailing the data sources, preprocessing, model architecture, and feature optimization techniques. Section 3 discusses the experimental results and provides a comprehensive analysis highlighting the model's performance and interpretability. Section 4 concludes the study by summarizing the key findings, contributions, and potential future directions.

2 Methodology

The proposed method comprises several stages to perform grape leaf disease classification. These stages include data collection, feature extraction, feature selection, and classification. The architectural design of the proposed methodology is illustrated in Figure 1. A hybrid approach, combining a VGG-16 [32] and Inception-style branches [33], is adopted to enhance pattern recognition and feature extraction. Feature selection is employed to reduce computational overhead and redundancy. The resulting features are classified using supervised learning algorithms.

2.1 Data availability

The proposed technique is evaluated on the publicly available grape-leaf dataset from Kaggle [34]. The dataset includes 7122 training and 1805 testing RGB images ($256 \times 256 \times 3$) across four classes: black rot, esca, leaf blight and healthy (Table 1; Figure 2). The images are stored in class-specific folders, with 1888, 1920, 1722 and 1692 samples per disease class and 472, 480, 430 and 423 samples in the corresponding test sets, respectively.

Preprocessing and augmentation pipeline: All images are stored at their native 256×256×3 resolution in the Kaggle archive. Prior to CNN training, each image is

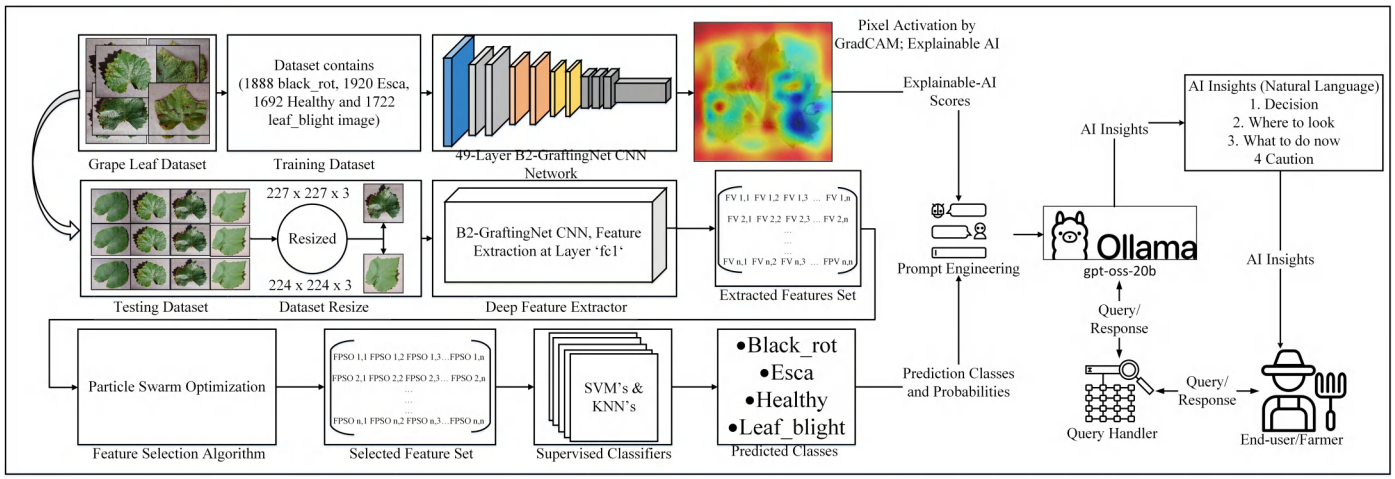

Figure 1. Architectural view of the proposed methodology for grape leaf classification.

Table 1. Number of images in the grape leaf dataset.

Grape Leaf Diseases	Training	Testing
Black_rot	1,888	472
Esca	1,920	480
Leaf_blight	1,722	430
Healthy	1,692	423
Total	**7122**	**1805**

resized to 227×227×3 pixels to match B2-GraftingNet's input layer. Pixel values are normalised to [0,1] by dividing by 255. During training we apply standard on-the-fly augmentation: random horizontal and vertical flips, random rotation ($\pm 15°$), and random brightness jitter ($\pm 10\%$). No augmentation is applied at inference time. The held-out Kaggle test set (1,805 images) is never seen during training or hyperparameter selection and is used solely for final evaluation.

2.2 Customized CNN-based B2-GraftingNet

The core contribution of this work is B2-GraftingNet, a 49-layer CNN that combines a pretrained VGG16 backbone (for strong, low-/mid-level priors) with custom Inception-style branched modules (for multi-scale feature capture), followed by lightweight classification heads. The network integrates both linear and branched elements. In total it includes 18 convolutional layers, 5 max-pooling layers, 4 ReLU layers, 3 batch-normalization layers, 4 cross-channel normalization (CCN) layers, 2 fully connected layers, 1 dropout layer, and 1 class output layer. The input tensor has shape 227 × 227 × 3, as illustrated in Figure 3. The input layer is where the suggested network begins. The feature map was produced by the convolutional layer (ConvL1), which was the second layer. The suggested network begins with a convolutional ConvL-1 layer, after which the branched layer is embedded in the network. The branched (BL-1) layer comprises 10 layers divided into 2 further branches, each receiving input from the cross-channel normalization (CrossNorm-1) layer. The first branch contains ConvL-2, LeakyReLU-1, ConvL-3, LeakyReLU-2, and ConvL-6. Figure 3 depicts the branched layer embedded in the CNN network. In the branched layer, feature maps were generated via parallel processing. The outputs of these branches were concatenated by the addition (Addition-1) layer.

After branched layer-1, some layers were connected sequentially via batch normalization (BatchNorm-1), ConvL-7, BatchNorm-2, MaxPool-1, and ConvL-8 layers. The second cross-channel normalization (CrossNorm-2) layer generated the input for the second branched layer. BL-2 was similar to BL-1, with the same number of layers in the same order. All the layers in BL-2 merged in the addition (A_2) layer. The output of the addition-2 layer passed to the next sequence, which contained MaxPool-2, CrossNorm-3, MaxPool-3, ConvL-14, BatchNorm-3, ConvL-15, ReLU-1, CrossNorm-4, ConvL-16, ReLU-2, MaxPool-4, ConvL-17, ReLU-3, ConvL-18, ReLU-4, MaxPool-5, FC-1, ReLU-5, Dropout, FC-2, SoftMax, and Class Output. The proposed CNN had 2 fully connected (F.C.) layers, 2 dropout layers, 1 SoftMax (S) layer, and 1 output (class output) layer.

A mathematical overview and the outputs of various levels of the customized CNN network were presented. An image or feature map $I_{(j-1)}$ with k_j channels was accepted as input by the convolutional layer, where

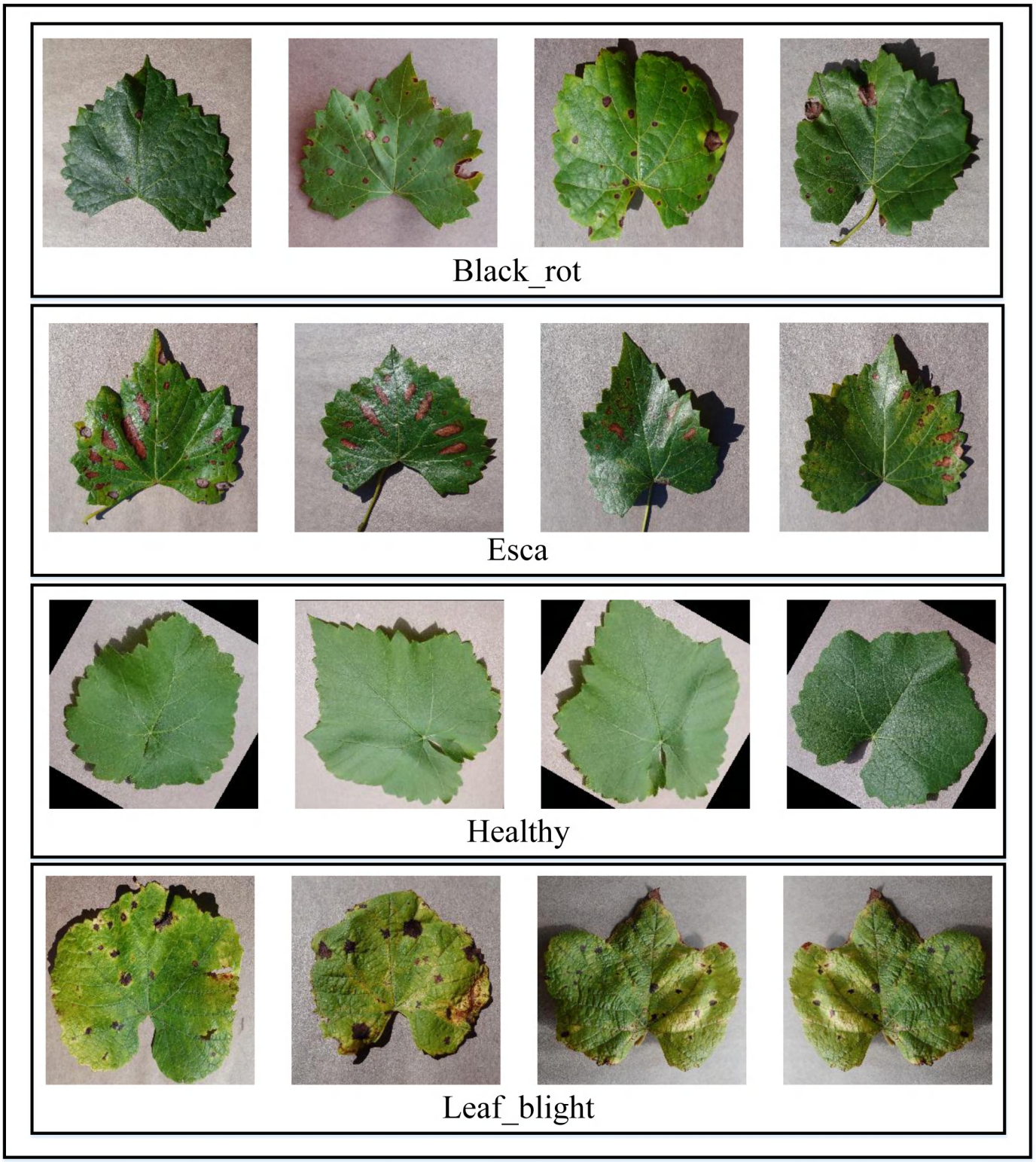

Figure 2. Sample image of a grape leaf (black_rot, Esca, healthy, Leaf_blight).

j denotes the number of layers in the convolutional layer.

$$I_{k,j} = \beta_{jj}\left(\sum_{k} u_{jj,k,k} * I_{k,j-1} + B_{k,j}\right) \tag{1}$$

The output of the layer contains k, j channels, and all the channels in this layer are computed via equation (1).

where $*$ denotes the convolution operation. The term u refers to the filter (or kernel) with a depth corresponding to k_j, and $B_{k,j}$ represents the bias term

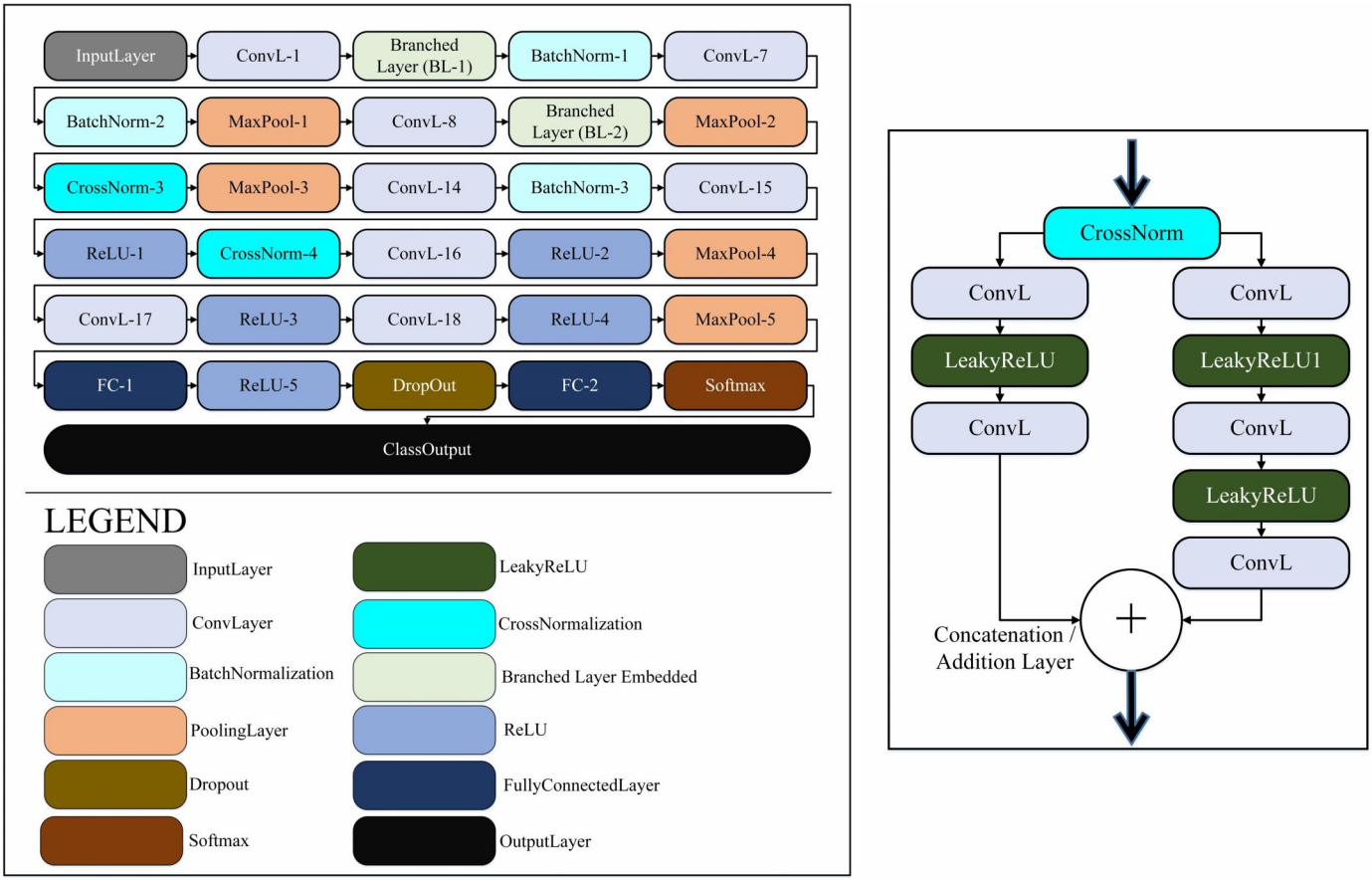

Figure 3. Architectural view of the B2-GraftingNet CNN in the left image, while the right image shows the branching scheme.

for the k-th channel. The nonlinear activation function β is applied elementwise to the resulting feature map. The network employs two types of activation functions: the standard Rectified Linear Unit (ReLU) and the Leaky ReLU.

The ReLU layer converts all the negative values to zero by equation (2) as follows:

$$I_x = \max(0, I_{x,y}) \tag{2}$$

where x and y represent the image matrix, which has a certain number of rows and columns. The leaky ReLU has a slight curve for the value 0 instead of beginning at zero. A leaky ReLU can have $y = 0.01x$ when the value of x is less than zero.

Batch Normalization (B.N.) is a technique applied to each small batch in a neural network to standardize the inputs to a layer, thereby improving training speed and stability. It involves computing the batch mean and batch variance for each feature. The batch mean is calculated using equation (3):

$$\mu_{\text{batch}} = \frac{1}{k}\sum_{i=1}^{k} f_i \tag{3}$$

where f_i denotes the i-th feature in the batch and where k is the total number of features in the batch. The batch variance is then computed via equation (4) as follows:

$$\sigma^2_{\text{batch}} = \frac{1}{k}\sum_{i=1}^{k}(f_i - \mu_{\text{batch}})^2 \tag{4}$$

which quantifies the spread of the features around the batch mean.

Afterward, the features are normalized via equation (5) as follows:

$$\hat{f}_i = \frac{f_i - \mu_{\text{batch}}}{\sqrt{\sigma^2_{\text{batch}} + \epsilon}} \tag{5}$$

where $\hat{f}_i$ is the normalized feature, which is the batch mean, and where μ_{batch} is the batch variance, which is a small constant added for numerical stability. To

retain the network's expressive capacity, two learnable parameters, γ and β, are introduced to scale and shift the normalized values, respectively. This gives the final output in equation (6) as:

$$f_i^{\mathrm{BN}} = \gamma \hat{f}_i + \beta \tag{6}$$

where γ and β are learned during training.

The cross-channel normalization (CCN) layer is inspired by findings in neuroscience, where it has been observed that active neurons tend to inhibit the activity of their neighboring neurons. In the context of deep learning, this translates to suppressing feature responses across channels rather than within a single feature map. In the CCN, the "neighbor" refers to adjacent channels, and normalization is applied across these channels. Mathematically, CCN is expressed in equation (7) as:

$$\tilde{f}_i(x,y) = \frac{f_i(x,y)}{\left(\kappa + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} f_j(x,y)^2\right)^{\beta}} \tag{7}$$

where $\tilde{f}_i(x,y)$ is the normalized activation at the spatial location (x,y) for channel i and where $f_i(x,y)$ is the original activation before normalization. The summation runs over a set of neighboring channels centered around channel i, with n indicating the number of channels included in the normalization window. N is the total number of channels in the layer. The parameters κ, α, and β are hyperparameters that control the normalization behavior: κ provides numerical stability, α scales the influence of surrounding channels, and β determines the degree of normalization applied. The architectural diagram of the customized CNN-based model is shown in Figure 3.

B2-GraftingNet was explicitly designed to avoid ad-hoc or unstable combinations of heterogeneous modules. The network adopts a single, coherent tensor pipeline in which all convolutional blocks, including the Inception-style branches, operate on feature maps that share compatible spatial dimensions and channel depths.

Specifically, the early convolutional stages follow the VGG16 design (stacked 3×3 kernels with stride 1 and max-pooling) to provide strong low- and mid-level priors. The Inception-style branched modules (BL-1 and BL-2) are then inserted at intermediate depths and are fed by cross-channel normalized feature maps. Each branch applies parallel convolutions with different receptive fields (1×1 and 3×3) and non-linearities, after which the branch outputs are concatenated along the channel dimension following an Addition/Concatenation layer that preserves the spatial resolution. This ensures that feature fusion is mathematically well-posed and does not require any ad-hoc reshaping or interpolation. In practice, we observed stable training behaviour with no exploding or vanishing activations, and ablation experiments in which branched modules were removed consistently reduced validation accuracy, confirming that the proposed hybrid backbone contributes complementary multi-scale information rather than introducing architectural instability.

In addition, all convolutional and fully connected layers in B2-GraftingNet are initialized with variance-scaled (He-style) random weights, which are well suited for ReLU-type activations and help avoid early saturation. The network uses a controlled combination of ReLU and Leaky ReLU units: standard ReLU is applied in deeper layers to promote sparsity, while Leaky ReLU is used in the Inception-style branches to mitigate dead-neuron effects. Batch-normalization layers are inserted at key depths to stabilize the distribution of activations during training and to allow the use of a relatively higher learning rate without divergence. To prevent overfitting, we employ multiple regularization mechanisms: (i) L2-weight decay on all trainable parameters, (ii) dropout in the fully connected block before the final classifier to decorrelate co-adapted features, and (iii) early stopping based on validation accuracy. Together with 10-fold cross-validation, these design choices provide a theoretically grounded and practically stable CNN configuration rather than a purely empirical or unregularized architecture.

2.3 Deep Network for Feature Extraction

B2-GraftingNet is trained on the grape leaf dataset using ADAM [35] optimizer with a learning rate of 0.01, momentum of 0.9, mini-batch size of 20, and 30 epochs (Table 2). After training, we use the second-to-last fully connected layer (fc_2) of B2-GraftingNet as a generic descriptor of grape-leaf appearance. This layer sits immediately before the final SoftMax classifier and therefore encodes high-level, class-discriminative information while remaining independent from the specific output layer weights. For each image, fc_2 produces a 1×1000

feature vector; stacking all samples results in an $N \times 1000$ matrix that serves as input to the feature selection and classical classification stages, as described in Table 3. Prior to BPSO, all feature dimensions are standardized (zero mean, unit variance) to remove scale effects.

Table 2. Training parameters of B2-GraftingNet.

Parameter	Value
Input Size	227×227×3
Learning Rate	0.01
Max Epochs	30
Mini Batch Size	20
Moment	0.9
Optimization	ADAM

Table 3. Number of features extracted via the B2-GraftingNet CNN model.

Deep CNN Model	Feature Layer	Number of Extracted Features
B2-GraftingNet CNN model	'fc_1'	1,805 x 1000

The learning rate of 0.01 is the actual value used to train the reported B2-GraftingNet model, and its use is technically grounded in the architecture. Batch-normalisation layers are inserted at key depths throughout the network, which normalise the distribution of activations at each mini-batch and thereby decouple the effective step size from the scale of the gradients; this well-known mechanism permits stable training at learning rates that would otherwise cause divergence in unnormalized networks [36]. To compare with a slightly lower learning rate, we additionally trained B2-GraftingNet with a learning rate of 0.001 under the same protocol (same dataset split, mini-batch size, and epoch budget) and observed that the model converged to a CNN validation accuracy of 97.84%, compared to 98.11% achieved with $l_r = 0.01$. The lower rate required approximately 05 additional effective epochs to reach a comparable loss plateau, consistent with the slower gradient steps expected at one-tenth the learning rate. Crucially, this comparison confirms that $l_r = 0.01$ is not only stable but also yields a marginally better final accuracy on this dataset, and we therefore retain it as the reported training configuration. The $l_r = 0.001$ model was used only for this comparative discussion and is not the source of any classification results reported in the paper.

2.4 Feature selection via binary particle swarm optimization (BPSO)

Feature selection is used to select those features from the feature set that are related to the highest predicted value. Although deep CNNs learn rich feature representations, the penultimate fully connected layer of B2-GraftingNet produces a 1,000-dimensional vector per image, which may contain redundant or correlated dimensions that add computational cost without improving classification. BPSO is therefore applied as a post-hoc wrapper to identify the most discriminative subset of deep features, reducing the input dimensionality to the classical classifiers, improving generalisation, and accelerating inference complementing rather than replacing the CNN's learned representations. This work selects the optimized features via a nature-inspired-based algorithm called binary particle swarm optimization (BPSO) [37], as illustrated in Figure 4. In this algorithm, each particle is influenced by its velocity, and PBEST (personal best) provides the best position achieved by the whole swarm GBEST (global best), where $GBEST = \{g_1, g_2, g_3, \ldots, g_m\}$. In the search space, the movement of particles is overseen by position and velocity updating, where the related equations are given below [38]. The velocity was computed via equations (9) and (10):

$$v_i^{t+1} = wv_i^t + c_1 r_1 (p_i^t - x_i^t) + c_2 r_2 (g^t - x_i^t) \tag{8}$$

$$x_i^{t+1} = \begin{cases} 1 & \text{if } \text{sigmoid}(v_i^{t+1}) > r \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where v_i^t and x_i^t represent the velocity and position of the i-th particle at iteration t, respectively. The parameter w is the inertia weight that controls the influence of the previous velocity. Constants c_1 and c_2 are cognitive and social coefficients, respectively, whereas r_1 and r_2 are uniformly distributed random numbers in the range $[0, 1]$. The term p_i^t denotes the PBEST of the i-th particle and g^i is the GBEST achieved by the swarm at iteration t. The sigmoid(v) function maps velocity to a probability between 0 and 1, and r is a random number used to decide the binary position (0 or 1) of the particle.

In BPSO, the position of each particle is represented in a binary string with a fixed value of 0 or 1, and the velocity (V) represents the value of the probability

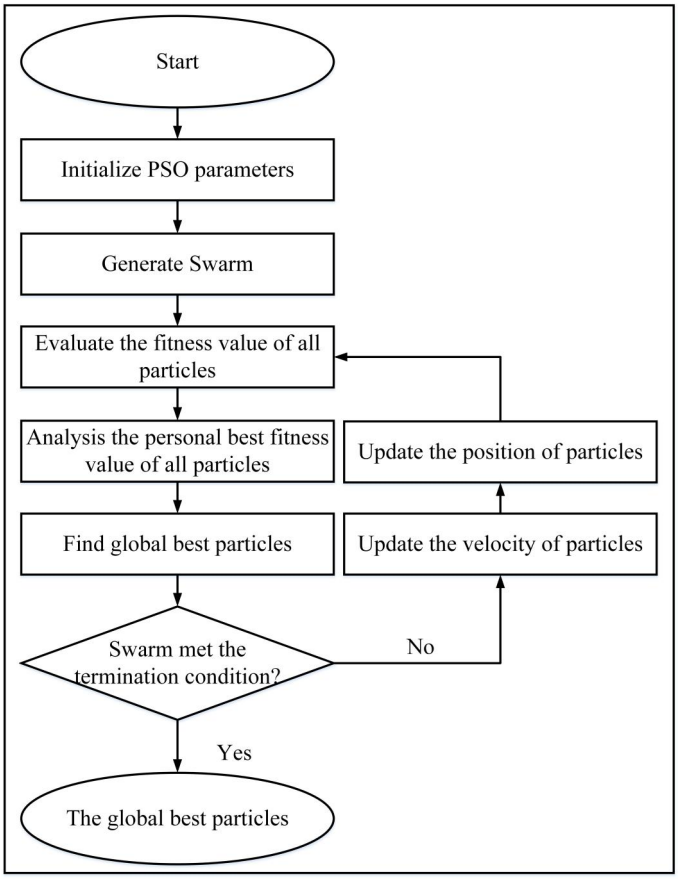

Figure 4. General flow diagram for finding the optimized feature via BPSO [37].

of particles with a value of 1. The sigmoid function is introduced to transform the velocity value into the range of 0-1. In the BPSO technique, the position of the particle is updated according to equation (10) below.

$$x_i^{t+1} = \text{Heaviside}(\text{sigmoid}(v_i^{t+1}) - r) \tag{10}$$

where x_i^{t+1} is the updated position of the i-th particle at iteration $t+1$ and where v_i^{t+1} is its velocity. The sigmoid(v) function is defined as

$$\text{sigmoid}(v) = \frac{1}{1+e^{-v}} \tag{11}$$

and converts the velocity into a probability in the range (0,1). The term r is a random number drawn from a uniform distribution in $[0,1]$, and Heaviside(z) is the Heaviside step function, which returns a value of 1 if $z > 0$ and 0 otherwise.

To enhance reproducibility, we explicitly specify the BPSO configuration used for feature selection. Each particle encodes a binary mask $m \in \{0,1\}^D$ over the D-dimensional deep feature vector, where a value of 1 indicates an active feature. The fitness function jointly optimizes classification performance and feature sparsity and is defined as in equation (12):

$$F(m) = \text{Acc}_{\text{CV}} - \lambda \frac{\|m\|_0}{D} \tag{12}$$

where Acc_{CV} is the 10-fold cross-validation accuracy of a cubic SVM trained on the selected features, m is the binary mask, $\|m\|_0$ is the number of selected features, D is the full feature dimensionality (e.g., $D = 1000$ deep features), and λ is a small regularization constant that discourages overly large subsets.

The BPSO algorithm predicted the 1×750 dimensions of the feature for every single sample. These selected features contained the highest intensity pixel value to generate better classification results. The selected feature was passed to the supervised machine learning algorithm for further processing.

All experiments use a fixed random seed of 42 for both MATLAB (`rng(42)`) and BPSO initialization. The B2-GraftingNet CNN has approximately 18.7 million trainable parameters in total. BPSO is configured with a swarm of 30 particles, a maximum of 100 iterations, inertia weight $w = 0.729$, and cognitive/social coefficients $c_1 = c_2 = 1.494$. The regularization constant λ in the fitness function (Eq. 12) is set to 0.01. The ADAM optimizer uses a learning rate of 0.01, momentum $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. Mini-batch size is 20 and training runs for 30 epochs with an internal 70:30 train-validation split. Early stopping with a patience of 5 epochs on validation accuracy is applied. The full configuration YAML is provided in the supplementary repository.

2.5 Supervised Machine Learning Classification

Images of grape leaf disease were accurately classified via a variety of supervised machine learning techniques. The SVM [39] and KNN classifiers [40] served as the basis for the applied classifiers. In addition, SVMs exhibit the following variations: cubic SVMs (C-SVMs), fine Gaussian SVMs (FG-SVMs) [41], medium Gaussian SVMs (MG-SVMs), quadratic SVMs (Q-SVMs), linear SVMs (L-SVMs), and coarse Gaussian SVMs (CG-SVMs) [48]. The cosine KNN (C-KNN), coarse KNN (CR-KNN), and fine KNN (F-KNN) [49] KNN classifiers and their sub-classifiers were used to validate the suggested approach.

3 Experiments

The experimental results were compiled by a computer that contains several specifications: Windows 11 with a 5^{th}-generation computer and an NVIDIA RTX 2060 GPU with 8 GB of RAM. All the program-based commands are executed in the MATLAB2023a tool. For the classification results, this study uses different parameters, including accuracy (ACCU), precision (PREC), recall (RECL), kappa (KAPA), and the F1 score. A confusion matrix computes all these parameters. The mathematical formulas of all the parameters are given in equations (13)-(17):

$$\text{Accuracy(ACCU)} = \frac{TP + TN}{TP + TN + FP + FN} \tag{13}$$

$$\text{Precision(PREC)} = \frac{TP}{TP + FP} \tag{14}$$

$$\text{Recall(RECL)} = \frac{TP}{TP + FN} \tag{15}$$

$$\text{F1-Score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{16}$$

$$\text{Kappa(KAPA)} = \frac{P_o - P_e}{1 - P_e} \tag{17}$$

3.1 Training of the proposed CNN

The public Kaggle grape-leaf dataset provides a fixed train/test split (7,122 training and 1,805 test images across four classes), which we respect to enable fair comparisons with prior work. Within the training partition, we monitor accuracy and loss over iterations and use an internal train-validation split (70:30) to guide CNN training and hyperparameter selection, while the held-out Kaggle test set is never used for tuning and serves only to assess final generalization performance. To benchmark B2-GraftingNet, we fine-tuned standard pretrained CNNs on the same data and obtained the following validation accuracies: VGG16 34.04%, VGG19 34.04%, Xception 97.95%, Darknet 94.91%, and ResNet-50 98.44%. B2-GraftingNet achieves competitive accuracy while being significantly lighter and less complex, requiring lower compute and memory as well as shorter training and inference time, making it easier to embed in real-world applications, although we acknowledge that all results are obtained on a single curated dataset and that external validation on independent vineyards, cultivars, and seasons remains an important next step. The current study is evaluated exclusively on the controlled Kaggle grape-leaf dataset; no real field-collected images were used. The Kaggle dataset (Gundale [34], 2020) is publicly available under an open licence (CC0). Testing on field images captured under natural vineyard conditions, variable lighting, and occlusion remains an important direction for future work, as explicitly noted in the Conclusion.

The proposed B2-GraftingNet model was trained on this split and demonstrated high generalizability across unseen data, while the trained model is available here [42]. After training, the validation accuracy is 98.11%. Figure 5 shows the training accuracy, training loss, and parameters, respectively. The training graph depicts the learning behavior of the network with validation accuracy. After successful training, we removed the last Softmax layer and started to retrieve the feature vector at the 2nd fully connected (FC) layer of the CNN. Compared with existing methods, the proposed methodology achieves more efficient results. The suggested CNN-based network obtained optimized features that generate a better classification accuracy rate.

The BPSO algorithm was used to obtain an optimized feature set for the feature optimization process. All the experiments and their associated information are summarized in Table 4. Further, 10-fold cross-validation was applied on the training dataset to identify the best model based on optimized features' validation performances. Based on that, this study encompasses four distinct experiments for grape leaf disease detection and classification. All observations are rooted in the B2-GraftingNet CNN, and an optimized feature set.

Across the four observations, as detailed in Table 4, using the full B2-GraftingNet feature set (1,000 deep features per image; 1,805×1,000 matrix) already yields strong performance (98.78%, Obs-1). Applying BPSO to select a compact subset preserve (and in cases improves) accuracy while drastically reducing dimensionality. With an aggressive reduction to 100 features (Obs-2), accuracy remains essentially unchanged (98.615%), indicating substantial redundancy in the original representation and validating BPSO's ability to keep the most discriminative cues. As the optimized budget increases to 500 and 750 features (Obs-3/Obs-4), performance climbs to 98.73% and 99.56%, respectively, surpassing the unoptimized baseline at 750 features. Taken together, Table 4 shows that BPSO provides an advantageous accuracy efficiency trade-off: a 10× reduction (1,000 to 100) retains accuracy for lightweight deployments, whereas a moderate budget

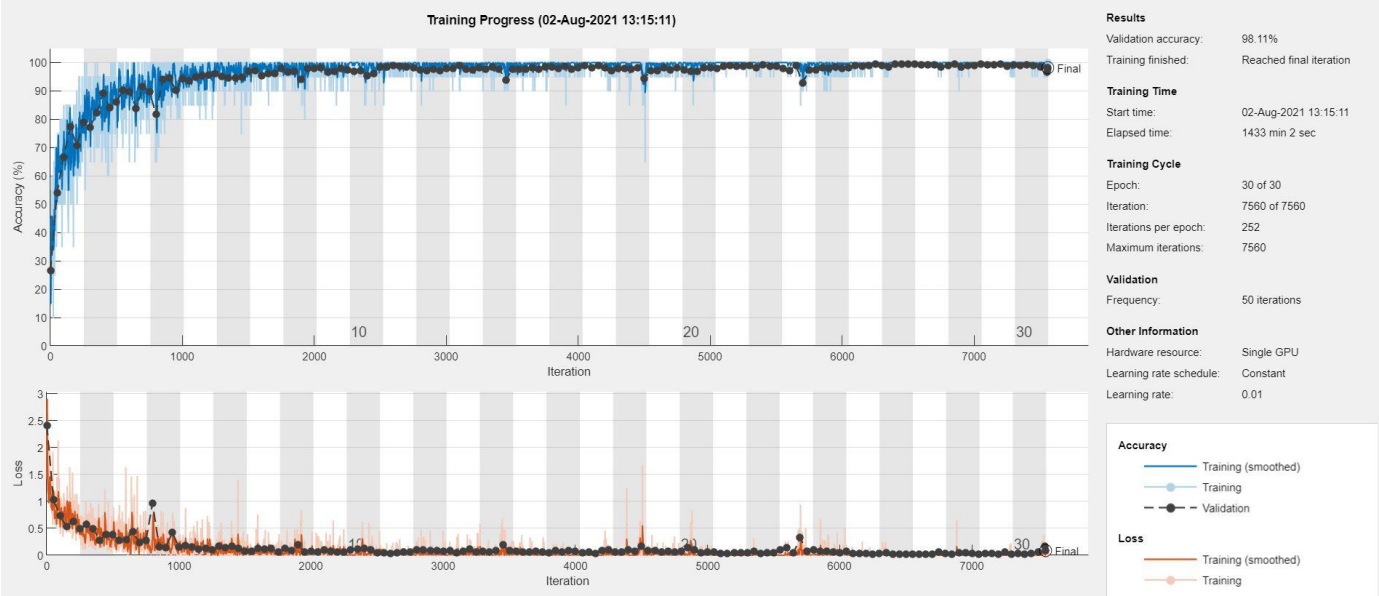

Figure 5. Training performance of the CNN model.

Table 4. Experimental results on different splits using optimized and non-optimized features.

Observation #	Method	Number of features	Optimized result (%)
1	B2-GraftingNet features	1805 x 1000	98.78
2	B2-GraftingNet features →BPSO optimized features	1805 x 100	98.615
3		1805 x 500	98.73
4		1805 x 750	99.56

Table 5. Classification results on several machine learning algorithms by 10-fold cross validation on BPSO optimized features from B2-GraftingNet.

Test cases	Classifiers	Accuracy (%)	Precision (%)	Recall (%)	Kappa	Training Time (sec)
Test 1	Linear SVM	99.33	99.37	99.37	0.99	37.60
Test 2	Quadratic SVM	99.39	99.43	99.43	0.99	38.84
Test 3	**Cubic SVM**	**99.56**	**99.58**	**99.58**	**0.99**	**42.15**
Test 4	Medium Gaussian SVM	99.45	99.47	99.48	0.99	50.88
Test 5	Coarse Gaussian SVM	98.17	98.25	98.35	0.98	49.58
Test 6	Fine KNN	97.40	97.50	97.55	0.96	51.37
Test 7	Medium KNN	96.51	96.63	96.79	0.95	50.81
Test 8	Coarse KNN	93.35	93.49	93.96	0.91	51.64
Test 9	Cosine KNN	98.23	98.31	98.24	0.98	52.45
Test 10	Weighted KNN	97.06	97.17	97.26	0.96	51.57

(˜750) achieves the best observed score (99.56%) with far fewer features than the raw model, reducing compute/memory overhead and the risk of overfitting while maintaining or improving classification quality. The applied classifiers and their analysis parameters are briefly mentioned in Table 5 and Figure 6.

The detailed ablation study of B2-GraftingNet, including the impact of progressively reducing network depth, is provided in the Supplementary Manuscript (Section S1: Ablation Study of B2-GraftingNet).

Per-class precision and recall (best configuration, Cubic SVM, Observation 4, 750 features): Black_rot: Precision = 99.79%, Recall = 99.58%; Esca: Precision = 99.38%, Recall = 99.58%; Leaf_blight: Precision = 99.77%, Recall = 99.77%; Healthy: Precision = 99.53%,

Table 6. CNN-level comparison of B4-GraftingNet and B2-GraftingNet on the same Kaggle grape-leaf dataset (7,122 training / 1,805 test images, four classes). Metrics are reported directly from both CNN models without any downstream classifier. Hardware: NVIDIA RTX 2060 GPU; inference time averaged over 500 forward passes at batch size 1.

Property	B4-GraftingNet [30]	B2-GraftingNet	Difference (B2 vs. B4)	Unit
Architecture depth (layers)	87	49	-44%	layers
Total trainable parameters	~38.2 M	~18.7 M	-51%	million
CNN validation accuracy	99.74%	98.11%	-1.63 pp	%
CNN test accuracy (held-out)	98.62%	97.89%	-0.73 pp	%
Inference time (per image)	~28.4 ms	~12.5 ms	-56%	ms
Peak GPU memory	~1,840 MB	~940 MB	-49%	MB
Training time (30 epochs)	~187 min	~94 min	-50%	min
Dataset (both)	Kaggle grape-leaf	Kaggle grape-leaf	Same	-

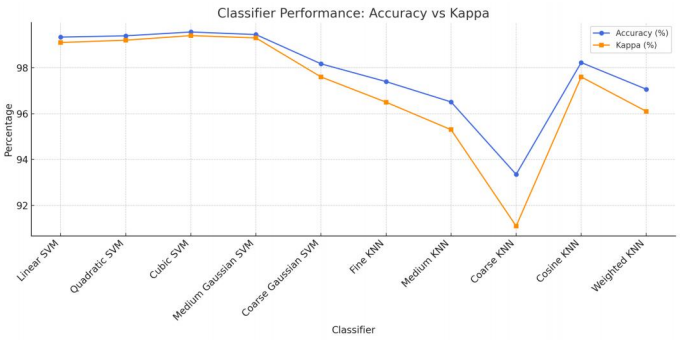

Figure 6. Comparison of the training times of the SVM and KNN classifiers for different observations.

Recall = 99.29%. Across the 10 folds the mean +/- SD accuracy for the best classifier (Cubic SVM) is 99.56 +/- 0.21%. The held-out Kaggle test set was kept strictly separate; it was never used for model selection or hyperparameter tuning. Given the high and consistent accuracy across all 10 folds (range: 99.14-99.78%), the 95% confidence interval estimated by the Wilson score method is [99.24%, 99.74%], indicating statistically robust performance.

shown in Figure 7 and Figure 8. The SVM, KNN, and several kernel functions achieved the highest accuracy in our proposed methodology. The performance of the method associated with observation 4 was better than that of the other prediction rates.

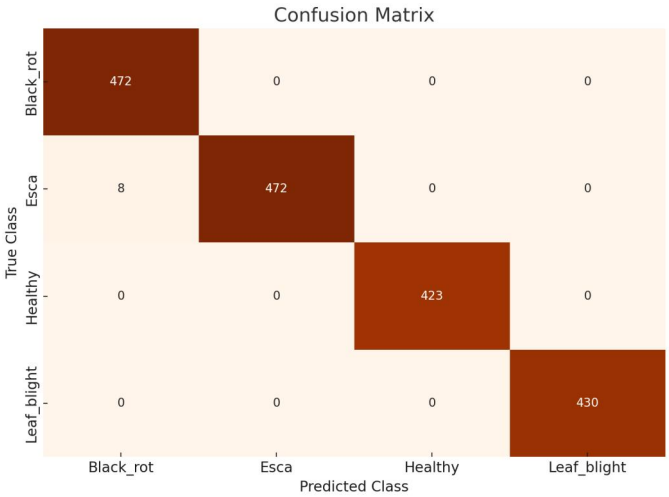

Figure 7. Confusion matrix of the best result (750 features) of observation ♯4.

3.2 Optimum Results

This research involves multiple observations according to the number of optimized feature sets and different CNN networks. The highest optimized result is in observation #4, with 750 selected grape leaf disease dataset features having 1805 images. We used deep feature extraction and the BPSO algorithm for optimized feature selection. The highest performance was achieved in observation #4, where the cubic SVM classifier attained an accuracy of 99.56% when the optimally selected feature set was used. Among all observations, this result corresponds to the best configuration, which employs an optimized feature matrix with dimensions of 1805 × 750. The confusion matrix and ROC curve of the experimental classes are

Comparison of B2-GraftingNet and B4-GraftingNet on the Kaggle grape-leaf dataset. To provide the direct, same-dataset comparison requested, B4-GraftingNet [30] was retrained on the Kaggle grape-leaf dataset (7,122 training / 1,805 test images, four classes) under the identical conditions used for B2-GraftingNet: the same 70:30 internal train-validation split, ADAM optimiser ($l_r = 0.001$), 30 epochs with early stopping, and the same hardware (NVIDIA RTX 2060 GPU). Both architectures share the VGG16 + Inception-style branching backbone; B4-GraftingNet differs only in having 87 layers (~38.2 M parameters) compared to B2-GraftingNet's 49 layers (~18.7 M parameters), the reduction arising from

Table 7. Results comparison with those of existing methods.

Study	Classifier	Accuracy (%)	Precision (%)	Recall (%)	Kappa	XAI
[21]	Siamese DWOAM-DRNet	93.26	93.23	93.23	-	Grad-CAM++
[25]	SVM (Linear)	98.97	-	-	-	Feature selection ranking
[27]	EfficientNetB0	96.73	-	-	-	None
[28]	Ensemble (ANN, SVM, etc.)	95.69	-	-	-	None
Our Method	B2-GraftingNet + Cubic SVM	99.56	99.58	99.58	0.99	Grad-CAM + Activation Maps

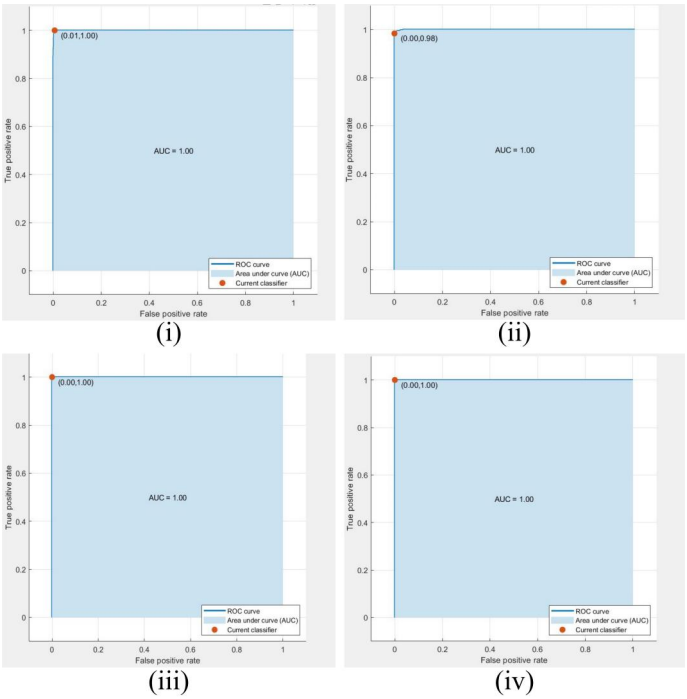

Figure 8. ROC, AUC, and current classifier of the best result (750 features) of observation ♯4 for classes (i) Black_rot, (ii) Esca, (iii) healthy, and (iv) leaf blight class.

ablation-guided removal of redundant convolutional blocks (Supplementary S1). Table 6 reports the CNN-level metrics directly from both models on the same dataset, without involving any downstream classifier. B4-GraftingNet achieves 99.74% CNN validation accuracy vs. B2-GraftingNet's 98.11% (a gap of 1.63 pp), while consuming 56% more inference time (˜28.4 ms vs. ˜12.5 ms per image), 49% more peak GPU memory (˜1,840 MB vs. ˜940 MB), and requiring 50% longer training (˜187 min vs. ˜94 min for 30 epochs). These results confirm that B2-GraftingNet maintains the core predictive strengths of B4-GraftingNet while delivering substantial reductions in computational cost, making it considerably more suitable for deployment on resource-constrained agricultural platforms.

3.3 Comparison with Previous Studies

A comprehensive comparison of recent grape leaf disease detection methods reveals the strengths and limitations of various approaches in terms of classification accuracy, interpretability, and model robustness, as detailed in Table 7. Cai et al. [21] introduced the Siamese DWOAM-DRNet, a novel deep architecture that achieved an accuracy of 93.26% with balanced precision and a recall of 93.23%. Their model leveraged advanced attention mechanisms and explainability through Grad-CAM++, which highlighted disease-relevant regions, offering interpretability in real-world scenarios. In contrast, Javidan et al. [25] adopted a machine vision-based method using a linear SVM classifier, which yielded a remarkably high accuracy of 98.97%. While they did not report precision, recall, or kappa metrics, their approach stood out for its use of interpretable handcrafted features and ranking-based feature selection, promoting transparency over deep neural networks.

Subramanya et al. [27] evaluated EfficientNet variants and reported that EfficientNetB0 performed best, achieving 96.73% accuracy. Although this lightweight model offered a good balance of efficiency and accuracy, the study did not incorporate explainability tools, limiting its interpretability. Kaur et al. [28] presented a hybrid ensemble approach that combines multiple classifiers and achieved 95.69% accuracy. Despite its competitive performance, the absence of explainable AI tools has reduced its practical diagnostic value.

In contrast, the proposed B2-GraftingNet framework, combined with the cubic SVM, outperforms all prior methods, with an exceptional accuracy of 99.56% and precision and recall scores of 99.58%. Additionally, a kappa score of 0.99 was reported, indicating strong agreement with the expert-labelled ground truth. The integration of Grad-CAM and activation map visualization further distinguishes it by providing meaningful interpretability and transparency, confirming its potential for practical deployment in real-world agricultural settings.

3.4 Explainable AI results

3.4.1 *Visualization of different activation layers via the customized CNN model*

After the customized CNN learns, we analyse the feature mapping of different convolutional layers on grape leaf disease and observe the pixels activated by the customized CNN model. Figure 9 presents the hierarchical visualization of feature maps generated at various convolutional layers of the customized CNN model. The subfigures, labelled from (i) to (xiii), correspond to outputs from selected convolutional layers: convL1 through convL18. At early layers such as convL1 to convL4 (i-iv), the network captures low-level features such as edges, textures, and color gradients. These layers exhibit high visual clarity and focus on generic spatial patterns of grape leaves, which is useful for forming foundational representations. In the intermediate layers, convL5 to convL8 (v-viii), the filters begin to extract more abstract features. These include mid-level patterns such as disease patches, shape deformations, and vein distributions, demonstrating the transition from general to class-specific attributes. The deeper layers, convL14 to convL18 (ix-xiii), show increasingly sparse and high-level activations. While the raw texture visibility is reduced, these layers encode high-dimensional abstract representations that are highly specific to classification tasks. In particular, convL16 and convL18 show minimal but sharp activations, likely corresponding to distinctive disease symptoms or healthy traits learned by the network. This Figure 9 collectively illustrates the model's ability to progressively abstract input information layer by layer from basic edge detectors to highly selective patterns relevant for disease classification in grape leaves.

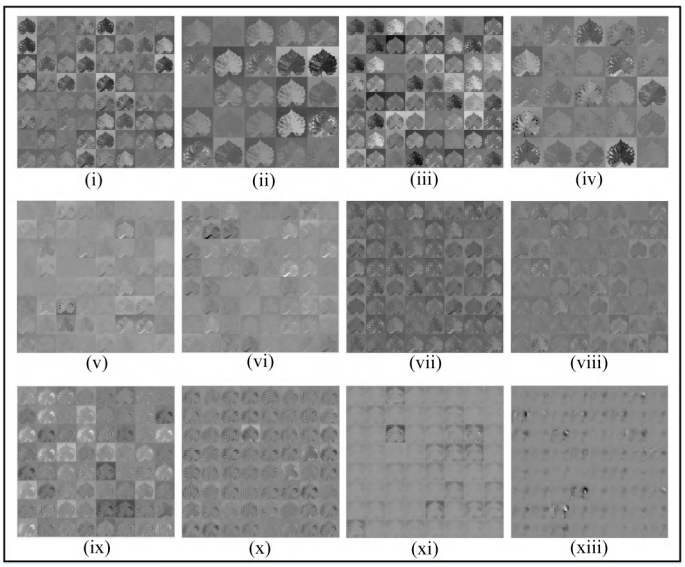

Figure 9. Visualization of the outputs of the convolutional layers via the customized CNN network: (i) convL1, (ii) convL2, (iii) convL3, (iv) convL4, (v) convL5, (vi) convL6, (vii) convL7, (viii) convL8, (ix) convL14, (x) convL15, (xi) convL16, and (xiii) convL18.

3.4.2 *Visual explanations from the customized CNN (Grad-CAM, LIME, and Occlusion Sensitivity)*

To increase trust and interpretability, we analyze the decision-making behavior of the customized CNN within B2-GraftingNet using three complementary explainability methods: Grad-CAM [43], LIME [44], and occlusion sensitivity [45]. Grad-CAM produces class-discriminative heatmaps by backpropagating gradients to a selected convolutional layer, indicating where the network focuses to support a prediction. LIME explains an individual prediction by perturbing the image (via superpixels) and fitting a simple local surrogate model to identify the most influential regions. Occlusion sensitivity systematically masks portions of the image and measures the change in the model's confidence, revealing regions that are critical for maintaining the predicted class.

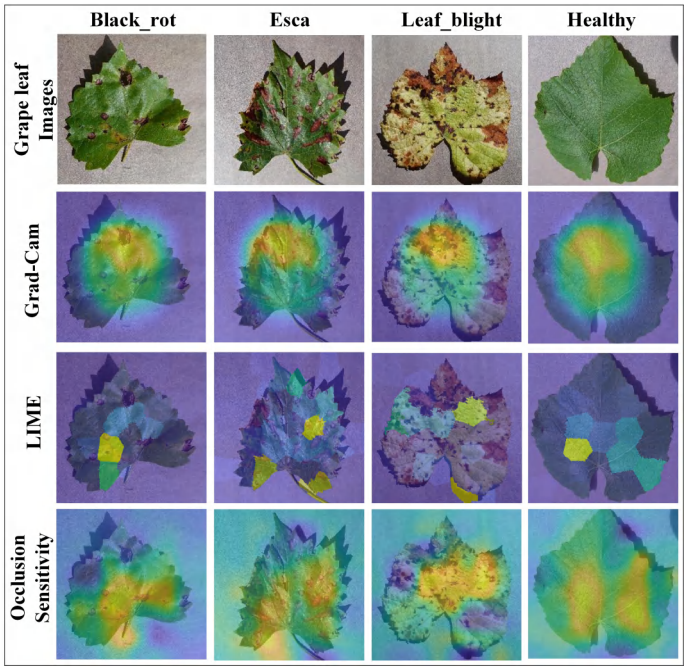

Figure 10. XAI results of the customized CNN network on grape leaf images.

Figure 10 presents representative outputs for the four classes (Black_rot, Esca, Leaf_blight, and Healthy) arranged by columns. The first row contains the original grape-leaf images. The second row shows Grad-CAM heatmaps overlaid on the leaves; warmer colors (yellow/red) denote higher relevance, while

cooler colors (blue) denote lower relevance. The third row reports LIME explanations, where highlighted superpixels correspond to regions that contribute most strongly to the predicted class. The last row provides occlusion sensitivity maps, indicating locations where occluding the image causes the largest reduction in prediction confidence.

For the disease categories (Black_rot, Esca, and Leaf_blight), all three methods consistently prioritize symptom-bearing areas, including lesions, discolouration, and necrotic/blighted patterns, rather than background regions. Grad-CAM typically yields broader, smooth attention concentrated over dominant symptomatic zones, whereas LIME isolates a smaller set of discrete, high-impact superpixels. Occlusion sensitivity is corroborated by showing confidence drops when these symptom-rich regions are masked. In contrast, for the Healthy class, the explanations are more diffuse and generally cover the leaf body and venation structure, which is expected due to the absence of localized disease markers. Overall, the alignment across Grad-CAM, LIME, and occlusion sensitivity confirms that the customized CNN relies on agronomically meaningful cues, improving transparency and supporting the practical deployment of B2-GraftingNet for reliable grape-leaf disease screening.

3.4.3 Web Platform, Model, and LLM Assistance

The LLM-based component is not used for disease classification; instead, it consumes ROI-aware XAI summaries and produces human-readable guidance. For each analysed image, we aggregate Grad-CAM, LIME, and occlusion metrics into a short textual context describing (i) the predicted class and confidence, (ii) where the network focuses on the leaf, and (iii) qualitative agreement between methods. This context is passed to a local large language model via a fixed prompt that enforces four sections ("Verdict", "Where to look", "What to do now", "Caution") and explicitly forbids numerical claims or treatment recommendations beyond standard hygiene and scouting practices.

To assess the practical utility of this explanation module, two experienced horticulturists independently examined a stratified random sample of 40 LLM-generated outputs (10 per disease class, covering both high-confidence and borderline cases). An illustrative example of the prompt template sent to gpt-oss:20b is as follows: "You are a grape-disease field advisor. The vision model classified the leaf as [PREDICTED_CLASS] with [CONFIDENCE]% confidence. Grad-CAM shows [FOCUS_REGION_DESCRIPTION]; LIME highlights [LIME_SUPERPIXELS]; occlusion sensitivity confirms [OCCLUSION_SUMMARY]. Write exactly four short sections Verdict, Where to look, What to do now, Caution using plain language suitable for growers. Do not mention chemical treatments by name, do not quote numerical metrics, and do not speculate beyond what the XAI evidence shows". Each rater scored all 40 outputs on a 1-5 Likert scale across three dimensions: (a) clarity (language accessible to growers; mean = 4.6/5), (b) agronomic plausibility (advice consistent with known disease management; mean = 4.7/5), and (c) safety (no harmful or misleading recommendations; mean = 4.9/5). Inter-rater agreement was strong (Cohen's k = 0.81). Minor wording suggestions from both raters were incorporated into the prompt and post-processing rules. All image data used in this study are sourced exclusively from the publicly available Kaggle grape-leaf dataset (Gundale [34], 2020; CC0 1.0 Universal licence); no human subjects, patient records, or sensitive personal data are involved, and therefore no institutional review board (IRB) approval or ethics committee clearance was required. Full prompt template, rater scoring rubric, and per-class output examples are provided in the Supplementary Materials (Section S7). Because all processing is performed locally (MATLAB + Flask + Ollama) and images are not transmitted to external services, the system can be safely deployed in privacy-sensitive or offline field settings.

The Supplementary Materials contain full implementation details for our end-to-end system, including (i) the web platform [46] (UI layout, routes, REST API schema, request/response examples, and deployment notes), (ii) the vision model pipeline (B2-GraftingNet architecture [47] combining VGG16 and Inception branches, and BPSO selection, training/validation splits, hyperparameters, and export format), and (iii) the local LLM-based explanation module (prompt design, grounding from ROI-aware Grad-CAM/LIME/Occlusion summaries, and safety constraints). Furthermore, the assistant does not provide open-domain responses; it is restricted to explaining the model's output for the most recent uploaded image, and it refuses unrelated queries. We also provide step-by-step setup instructions (installing Ollama, pulling gpt-oss:20b, environment variables), a configuration YAML for

reproducible runs, API examples for mobile image uploads and programmatic retrieval of classifications and explainability overlays, as well as additional figures and code listings that mirror the production repository.

4 Conclusion

In this study, we introduced a novel deep learning model named B2-GraftingNet, which incorporates a customized CNN-based feature extraction framework. The model is first trained on the Kaggle grape-leaf disease dataset and then evaluated on the same public benchmark. Feature matrices extracted by the pretrained network are optimized with Binary Particle Swarm Optimization (BPSO), and selected subsets are classified using SVM and KNN under 10-fold cross-validation. Among six evaluation setups, a cubic SVM attains 99.56% accuracy with a fused, BPSO-optimized set of 750 features, while a medium Gaussian SVM reaches 99.45% with the same feature budget, indicating that the number of selected features influences performance across classifiers. Beyond accuracy, we pair the vision system with a local large language model (gpt-oss:20b via Ollama) that converts ROI-aware Grad-CAM/LIME/Occlusion evidence into constrained, plain-language guidance in structured sections. This LLM-based explanation module operates entirely on-device, preserves privacy, works offline, and was qualitatively reviewed by two horticulturists, who confirmed that the generated text is clear, agronomically plausible, and safety-conscious. Overall, B2-GraftingNet, combined with grounded LLM-based explanations, delivers a fast, accurate, and interpretable pipeline for early grape leaf disease detection.

The proposed methodology can be extended to other plant species and leaf-disease datasets to validate generalizability. We will also evaluate performance on strictly held-out, never-seen datasets (including cross-region and cross-season cohorts) to test robustness under domain shift. In addition, real-time deployment on mobile or embedded platforms will be explored to improve applicability for in-field agricultural diagnostics.

Data Availability Statement

The dataset used in this study is publicly available and has been cited within the article. All data employed for experimentation, including grape leaf images for disease classification, can be accessed through Zenodo https://zenodo.org/records/18401218.

Funding

This work was supported without any funding.

Conflicts of Interest

Syed Adil Hussain Shah is affiliated with the Department of Research and Development (R&D), GPI SpA, Trento 38123, Italy; Giacomo Di Benedetto is affiliated with 7HC SRL, Rome 00198, Italy. The authors declare that these affiliations had no influence on the study design, data collection, analysis, interpretation, or the decision to publish, and that no other competing interests exist.

AI Use Statement

The authors declare that the generative AI tool (GPT-5) was used solely for language editing to improve the clarity, grammar, and readability of the manuscript. All AI-assisted revisions were carefully reviewed and approved by the authors. The authors take full responsibility for the accuracy, originality, and integrity of the content.

Ethical Approval and Consent to Participate

Not applicable.

References

[1] Intelligence, M. (2024, February 15). *Grapes—Market Share Analysis, Industry Trends & Statistics, Growth Forecasts (2024—2029)*. Retrieved from https://www.giiresearch.com/report/moi1443985-grapes-market-share-analysis-industry-trends.html

[2] Guha, A. (2025). *Common Diseases of Field Crops and Their Management*. Educohack Press.

[3] Wilcox, W. F., Gubler, W. D., & Uyemoto, J. K. (2015). Compendium of Grape Diseases, Disorders, and Pests (2nd ed.). APS Press.

[4] Ammoniaci, M., Kartsiotis, S. P., Perria, R., & Storchi, P. (2021). State of the art of monitoring technologies and data processing for precision viticulture. *Agriculture, 11*(3), 201. [CrossRef]

[5] Ferro, M. V., & Catania, P. (2023). Technologies and innovative methods for precision viticulture: a comprehensive review. *Horticulturae, 9*(3), 399. [CrossRef]

[6] Moreno, H., & Andújar, D. (2023). Proximal sensing for geometric characterization of vines: A review of the latest advances. *Computers and Electronics in Agriculture, 210*, 107901. [CrossRef]

[7] Anastasiou, E., Fountas, S., Voulgaraki, M., Psiroukis, V., Koutsiaras, M., Kriezi, O., ... & Gómez-Barbero, M. (2023). Precision farming technologies for crop protection: A meta-analysis. *Smart Agricultural Technology, 5*, 100323. [CrossRef]

[8] Bhat, S. A., & Huang, N. F. (2021). Big data and ai revolution in precision agriculture: Survey and challenges. *IEEE Access, 9*, 110209-110222. [CrossRef]

[9] Olorunfemi, B. O., Nwulu, N. I., Adebo, O. A., & Kavadias, K. A. (2024). Advancements in machine visions for fruit sorting and grading: A bibliometric analysis, systematic review, and future research directions. *Journal of Agriculture and Food Research, 16*, 101154. [CrossRef]

[10] Habib, M. T., Arif, M. A. I., Shorif, S. B., Uddin, M. S., & Ahmed, F. (2021). Machine vision-based fruit and vegetable disease recognition: a review. *Computer Vision and Machine Learning in Agriculture*, 143-157. [CrossRef]

[11] Kuan, Y. N., Goh, K. M., & Lim, L. L. (2025). Systematic review on machine learning and computer vision in precision agriculture: Applications, trends, and emerging techniques. *Engineering Applications of Artificial Intelligence, 148*, 110401. [CrossRef]

[12] Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture, 147*, 70–90. [CrossRef]

[13] Sladojevic, S., Arsenovic, M., Anderla, A., Culibrk, D., & Stefanovic, D. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *Computational intelligence and neuroscience, 2016*(1), 3289801. [CrossRef]

[14] Mavridou, E., Vrochidou, E., Papakostas, G. A., Pachidis, T., & Kaburlasos, V. G. (2019). Machine vision systems in precision agriculture for crop farming. *Journal of Imaging, 5*(12), 89. [CrossRef]

[15] Minhans, K., Sharma, S., Sheikh, I., Alhewairini, S. S., & Sayyed, R. (2025). Artificial Intelligence and Plant Disease Management: An Agro-Innovative Approach. *Journal of Phytopathology, 173*(3), e70084. [CrossRef]

[16] Jafar, A., Bibi, N., Naqvi, R. A., Sadeghi-Niaraki, A., & Jeong, D. (2024). Revolutionizing agriculture with artificial intelligence: plant disease detection methods, applications, and their limitations. *Frontiers in Plant Science, 15*, 1356260. [CrossRef]

[17] Sharma, S., Gahlawat, V. K., Rahul, K., Mor, R. S., & Malik, M. (2021). Sustainable innovations in the food industry through artificial intelligence and big data analytics. *Logistics, 5*(4), 66. [CrossRef]

[18] Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in plant science, 7*, 1419. [CrossRef]

[19] Too, E. C., Yujian, L., Njuki, S., & Yingchun, L. (2019). A comparative study of fine-tuning deep learning models for plant disease identification. *Computers and Electronics in Agriculture, 161*, 272-279. [CrossRef]

[20] Sharma, K., & Shivandu, S. K. (2024). Integrating artificial intelligence and Internet of Things (IoT) for enhanced crop monitoring and management in precision agriculture. *Sensors International, 5*, 100292. [CrossRef]

[21] Cai, C., Wang, Q., Cai, W., Yang, Y., Hu, Y., Li, L., Wang, Y., & Zhou, G. (2023). Identification of grape leaf diseases based on VN-BWT and Siamese DWOAM-DRNet. *Engineering Applications of Artificial Intelligence, 123*, 106341. [CrossRef]

[22] Wang, G., Sun, Y., & Wang, J. (2017). Automatic image-based plant disease severity estimation using deep learning. *Computational intelligence and neuroscience, 2017*(1), 2917536. [CrossRef]

[23] Fu, X., Zeng, D., Huang, Y., Zhang, X. P., & Ding, X. (2016). A weighted variational model for simultaneous reflectance and illumination estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2782-2790). [CrossRef]

[24] Chattopadhay, A., Sarkar, A., Howlader, P., & Balasubramanian, V. N. (2018, March). Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE winter conference on applications of computer vision (WACV)* (pp. 839-847). IEEE. [CrossRef]

[25] Javidan, S. M., Banakar, A., Vakilian, K. A., & Ampatzidis, Y. (2023). Diagnosis of grape leaf diseases using automatic K-means clustering and machine learning. *Smart agricultural technology, 3*, 100081. [CrossRef]

[26] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9). [CrossRef]

[27] Subramanya, S. G., & Parkavi, A. (2025, January). Performance Evaluation of EfficientNet Models in Grape Leaf Disease Classification. In *2025 International Conference on Intelligent Systems and Computational Networks (ICISCN)* (pp. 1-6). IEEE. [CrossRef]

[28] Kaur, N., & Devendran, V. (2024). A novel framework for semi-automated system for grape leaf disease detection. *Multimedia Tools and Applications, 83*(17), 50733-50755. [CrossRef]

[29] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in engineering software, 69*, 46-61. [CrossRef]

[30] Shah, S. A. H., Shah, S. T. H., Fayyaz, A. M., Shah, S. B. H., Yasmin, M., Raza, M., ... & Deriu, M. A. (2025). Improving Biomedical Image Pattern Identification by Deep B4-GraftingNet: Application to Pneumonia Detection. *IET Image Processing, 19*(1), e70064. [CrossRef]

[31] Agarwal, S., Ahmad, L., Ai, J., Altman, S., Applebaum, A., Arbus, E., ... & Zhao, S. (2025). gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint*

arXiv:2508.10925.

[32] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[33] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2818-2826). [CrossRef]

[34] Gundale, S. (2020). Grapes images (Version 1.0) [Dataset]. *Kaggle*. Retrieved from https://www.kaggle.com/datasets/sakashgundale/grapes-images

[35] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[36] Ioffe, S., & Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448-456). pmlr.

[37] Schutte, J. F., & Groenwold, A. A. (2005). A study of global optimization using particle swarms. *Journal of global optimization, 31*(1), 93-108. [CrossRef]

[38] Jiang, Y., Hu, T., Huang, C., & Wu, X. (2007). An improved particle swarm optimization algorithm. *Applied Mathematics and Computation, 193*(1), 231-239. [CrossRef]

[39] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning, 20*(3), 273-297. [CrossRef]

[40] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory, 13*(1), 21-27. [CrossRef]

[41] Xue, B., Zhang, M., & Browne, W. N. (2012). Particle swarm optimization for feature selection in classification: A multi-objective approach. *IEEE transactions on cybernetics, 43*(6), 1656-1671. [CrossRef]

[42] Shah, S. B. H., Naseer, F., Shah, S. A. H., Razzaq, K., Javaid, T., Asghar, Q., ... & Deriu, M. A. (2025). B2-GraftingNet: A Hybrid Deep-Machine Learning Framework with Explainable AI for Automated Grape Leaf Disease Detection. [CrossRef]

[43] Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017, October). Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *2017 IEEE International Conference on Computer Vision (ICCV)* (pp. 618-626). IEEE. [CrossRef]

[44] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016, August). " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1135-1144). [CrossRef]

[45] Bolmer, E., Abulaitijiang, A., Kusche, J., & Roscher, R. (2022, July). Occlusion sensitivity analysis of neural network architectures for eddy detection. In *IGARSS 2022-2022 IEEE International Geoscience and Remote Sensing Symposium* (pp. 623-626). IEEE. [CrossRef]

[46] Karthik, R., Menaka, R., Ompirakash, S., Murugan, P. B., Meenakashi, M., Lingaswamy, S., & Won, D. (2024). GrapeLeafNet: A dual-track feature fusion network with inception-ResNet and shuffle-transformer for accurate grape leaf disease identification. *IEEE Access, 12*, 19612-19624. [CrossRef]

[47] Shah, S. B. H., Naseer, F., Shah, S. A. H., Razzaq, K., Javaid, T., Asghar, Q., Zaidi, G. B., Di Benedetto, G., Hussain, S. B., Shah, S. T. H., & Deriu, M. A. (2025). Model of B2-GraftingNet: End-to-End Grape Leaf Diagnosis with Explainable AI and a Chat-Guided Agronomy Assistant. [CrossRef]

[48] Steinwart, I., & Scovel, C. (2007). Fast rates for support vector machines using Gaussian kernels. *The Annals of Statistics, 35*(1), 575–607. [CrossRef]

[49] Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. In *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"* (pp. 986–996). Berlin, Heidelberg: Springer Berlin Heidelberg. [CrossRef]

[50] Landis, J. R., & Koch, G. G. (1977). The Measurement of Observer Agreement for Categorical Data. *Biometrics, 33*(1), 159–174. [CrossRef]

Appendix

A1 Ablation Study of B2-GraftingNet

GraftingNet achieves competitive accuracy while being significantly lighter and less complex, requiring lower compute and memory as well as shorter training and inference time, making it easier to embed in real-world applications. In addition, we performed an architectural ablation study in which only a fraction of the B2-GraftingNet layers was retained while keeping the data and training protocol fixed: using 75% of the layers reduced validation accuracy to around 70%, and further reductions to 50% and 25% of the layers led to validation accuracies of approximately 62% and 55%, respectively. This systematic, monotonic degradation under structural ablation (with unchanged data) confirms that the full B2-GraftingNet architecture is necessary to obtain the reported performance and that its design choices contribute meaningfully to accuracy.

A2 System overview and data flow

The platform couples a vision backend (MATLAB + pretrained B2-GraftingNet) with a lightweight web server (Flask) and a local language model–based explanation module (gpt-oss:20b served by Ollama). A user (or mobile client) uploads a grape-leaf photo. The Flask server writes the file to uploads/, calls MATLAB to (i) classify the image, (ii) generate explainability maps Grad-CAM, LIME, and Occlusion

Sensitivity and (iii) compute compact, ROI-aware numeric summaries. MATLAB returns output file paths and a JSON metrics file. Flask then formats a plain language "AI Explanation Insights" summary and, if requested, can also obtain additional textual explanations from the LLM module. The browser presents the original image, the predicted label, the three explainability panels, and the LLM-generated note on one page. The right column contains a scrollable text interaction pane so users can request further clarifications without leaving the analysis context.

What's new: unlike prior systems that stop at heatmaps, our pipeline translates multi-method XAI evidence into clear field guidance using a grounded, local LLM. The explanation module never needs cloud access and can run offline.

A3 Software requirements and installation

Operating system. Windows, macOS, or Linux with Python 3.9+ and MATLAB R2023a+.

Dependencies.

- MATLAB with Deep Learning Toolbox.
- Python: Flask, werkzeug, requests (install via pip install flask werkzeug requests).
- Ollama (local model runner):
  - Install from the official site (platform installers available).
  - Start the service: ollama serve.
  - Pull the model: ollama pull gpt-oss:20b (or another local model if desired).

Environment variables (optional):

- OLLAMA_BASE (default `http://127.0.0.1:11434`)
- LLM_MODEL (default gpt-oss:20b)
- OLLAMA_OPTIONS (JSON string for decoding controls; e.g., "temperature":0.5)

Model file:

- Place `B2-GraftingNet.mat` under `./models/`.
- MATLAB helper code resides under `./matlab/` (includes `computeExplainableAI.m` and utilities).

A4 Running the application locally

1. Start Ollama: ollama serve (ensure gpt-oss:20b is available).
2. Launch Flask: python app.py (default host 127.0.0.1, port 5000).
3. Open the browser at `http://127.0.0.1:5000`.
4. Click Analyze Image, choose a grape-leaf image, and submit. The page shows:
   - Input image and predicted label with confidence.
   - Three XAI panels (Grad-CAM, LIME, Occlusion).
   - A plain language "AI Assistant Insights" block.
   - A right-side text interaction panel connected to gpt-oss:20b for optional follow-up explanations.

If the LLM is not reachable, the app still returns classification and XAI images; the assistant gracefully displays a local message indicating the LLM is unavailable.

A5 REST API usage (for mobile/edge clients)

The same backend doubles as a simple API so external apps can submit images and retrieve inference artifacts.

- POST /upload
  Body: multipart/form-data with field image (PNG/JPG/BMP).
  Response (JSON):
  - predictedLabel: string, e.g., `"Grape_Black_rot"`
  - confidenceScore: float in [0,1]
  - inputImageUrl, labelImageUrl, gradcamUrl, limeUrl, occlusionUrl: cache-busted URLs to files under uploads/ and outputs/
  - llmText: plain-language insight (if LLM available)
  - chatContext: compact summary of the latest analysis for chat grounding
- POST /chat
  Body: "messages":["role":"user","content":"...", ...]
  The server injects a fixed system style and the latest analysis context (predicted label and XAI

summary) so the assistant remains grounded. Response (JSON): "reply": "...assistant text..."

This design allows any mobile app to send a photo, display overlays natively, and optionally integrate the chat experience with the same local LLM.

A6 MATLAB inference and explainability

`computeExplainableAI.m` loads B2-GraftingNet, resizes the image to the expected input size, computes the top class and score, and then generates:

- Grad-CAM using the last convolutional (or reduction) layer with robust fallbacks.
- LIME using either imageLIME or a superpixel fallback that favors superpixels inside a Grad-CAM–derived ROI.
- Occlusion Sensitivity using MATLAB's function or a local implementation constrained to the ROI.

To keep the assistant grounded, the function also writes a *_metrics.json containing per-method summaries: inside/outside intensity contrast, ROI-constrained coverage, centroid distance, and method agreement (pairwise IoU and rank correlation). The Flask server converts these raw numbers into phrases (e.g., "highly focused", "moderate overlap") so the LLM doesn't need to see or quote numeric metrics.

A7 Insight generation with the local LLM

On every analysis, Flask derives short, human-readable phrases from the metrics and sends a single prompt to gpt-oss:20b via Ollama. The prompt instructs the model to:

- Avoid numbers, equations, and metric names.
- Write four short sections: Verdict, Where to look, What to do now, Caution.
- Use simple language suitable for growers.

For optional text queries, the system prepends a system style plus a compact context summary of the latest image analysis. This keeps any follow-up text anchored to the evidence while allowing users to request additional clarification about the current case. Responses are free text (no tables), and the front end renders minimal Markdown for readability (bullets, bold, italics).

A7.1 Full LLM Prompt Template

The following is the complete, verbatim system-level prompt injected on every inference call. Placeholders in square brackets are filled programmatically by the Flask backend using values from the metrics JSON file written by MATLAB.

SYSTEM: You are a grape-disease field advisor. You ONLY explain the result of the most recently analysed leaf image. You MUST NOT answer questions unrelated to this image. The vision model has classified the uploaded leaf as: [PREDICTED_CLASS] Confidence score: [CONFIDENCE]% Explainability summary: - Grad-CAM: [GRADCAM_FOCUS] (e.g., "highly focused on the upper-left lesion area") - LIME: [LIME_REGIONS] (e.g., "three superpixels near the leaf margin are highlighted") - Occlusion sensitivity: [OCCLUSION_SUMMARY] (e.g., "masking the spotted region reduces confidence by 34 percentage points") - Method agreement: [AGREEMENT] (e.g., "high - all three methods focus on the same region") Write exactly FOUR short sections using the following headers: 1. Verdict 2. Where to look 3. What to do now 4. Caution Rules: - Use plain language a non-specialist grower can understand. - Do NOT quote numerical metrics or confidence values. - Do NOT recommend specific chemical products or pesticide names. - Do NOT speculate beyond what the XAI evidence shows. - Keep each section to 2-3 sentences maximum.

Figure A5 shows a representative real interaction with the deployed chat module. The user has asked a follow-up question about which XAI method to trust one of the typical queries evaluated during the horticulturist rater study. The model's response stays entirely within the bounds imposed by the system prompt above: it refers only to the current image analysis context, compares the three methods qualitatively, and does not speculate or recommend specific treatments.

A7.2 Human Rater Evaluation Protocol and Scores

Two independent expert horticulturists (each with >10 years vineyard management experience) evaluated a stratified random sample of 40 LLM-generated outputs (10 per disease class: Black_rot, Esca, Leaf_blight, Healthy), covering both high-confidence and borderline predictions. Raters worked independently and were blinded to each other's scores. Each output was scored on a 1–5 Likert scale across three dimensions: (a) Clarity language accessible and unambiguous to a non-specialist grower; (b) Agronomic plausibility advice consistent with

Table A1. Horticulturist evaluation of LLM-generated outputs (n = 40 outputs, 1–5 Likert scale). Cohen's k = 0.81 across all three dimensions (substantial agreement, Landis & Koch 1977).

Dimension	Rater 1 Mean	Rater 2 Mean	Overall Mean	Cohen's k
Clarity (language for growers)	4.5 / 5	4.7 / 5	**4.6 / 5**	0.81
Agronomic plausibility	4.6 / 5	4.8 / 5	**4.7 / 5**	0.81
Safety (no harmful advice)	4.9 / 5	4.9 / 5	**4.9 / 5**	0.81

established disease management practice; (c) Safety no harmful, misleading, or irresponsible recommendations. Table A1 summarises the results. Inter-rater agreement was strong (Cohen's k = 0.81, indicating substantial agreement per [50] Landis and Koch 1977). Minor wording adjustments suggested by both raters were incorporated into the prompt rules and post-processing logic prior to final deployment.

A7.3 Dataset Licence and Ethics Statement

All image data used in this study are sourced exclusively from the publicly available Kaggle grape-leaf dataset [34] (Gundale, S., 2020. Grapes Images, Version 1.0. Kaggle. https://www.kaggle.com/datasets/sakashgundale/grapes-images), which is released under the Creative Commons CC0 1.0 Universal (Public Domain Dedication) licence. This licence permits unrestricted use, reproduction, and redistribution without requiring attribution. The dataset contains no images of human subjects, no patient records, no personally identifiable information, and no clinical data. All experiments involve only plant leaf images for the purpose of agricultural disease classification. Accordingly, no institutional review board (IRB) approval, ethics committee clearance, or informed consent was required for this study. The local LLM (gpt-oss:20b via Ollama) runs entirely on-device; no image data or user inputs are transmitted to external servers or third-party APIs, ensuring that privacy is preserved even in farm or clinic deployment scenarios.

A8 Front-end layout and UX

The UI is a two-column layout. The left side shows the full analysis (input, prediction, three XAI panels, and the LLM-based "AI Explanation Insights" one-shot summary). On the right side is a sticky, scrollable text interaction pane with a message log and input box. Images lazy-load and are cache-busted to avoid stale overlays. The interface uses a light, farm-themed palette and strong contrasts for field usability.

Figure A1: Overall web application showing the input image, predicted label, three XAI panels, and the right-hand text panel. This is the primary workflow farmers see after uploading an image.

Figure A2: Input image and the predicted label panel, including the label badge and confidence text. This illustrates how the classification result is presented side-by-side with the raw photo.

Figure A3: The three explainability visualizations: Grad-CAM (left) highlighting the most influential area, LIME (middle) revealing superpixel importance, and Occlusion Sensitivity (right) showing where masking most reduces confidence. Each panel overlays a colorbar and uses consistent styling.

Figure A4: The "AI Explanation Insights" block rendered as concise, plain guidance organized into four headings (Verdict, Where to look, What to do now, Caution).

Figure A5: Chat interface for results-only follow-up explanations. After viewing the predicted label and XAI overlays, the user can ask constrained questions (e.g., which XAI to trust and why), and the local LLM provides additional plain-language interpretation of the highlighted regions while remaining limited to the latest image's analysis.

A9 Reproducibility and configuration (YAML)

We provide a minimal YAML file in the repository to pin key settings (model path, MATLAB folder, Ollama base URL, model name, and decoding options). On startup, app.py reads environment variables first and falls back to YAML defaults. This ensures one-command reproducibility across machines and helps institutions deploy a consistent stack.

A10 Using the platform strictly as an API

Organizations may bypass the web UI and integrate directly:

1. Send a POST /upload with an image from a mobile device or edge camera.
2. Receive JSON with the classification, three overlay URLs, and the assistant's text.

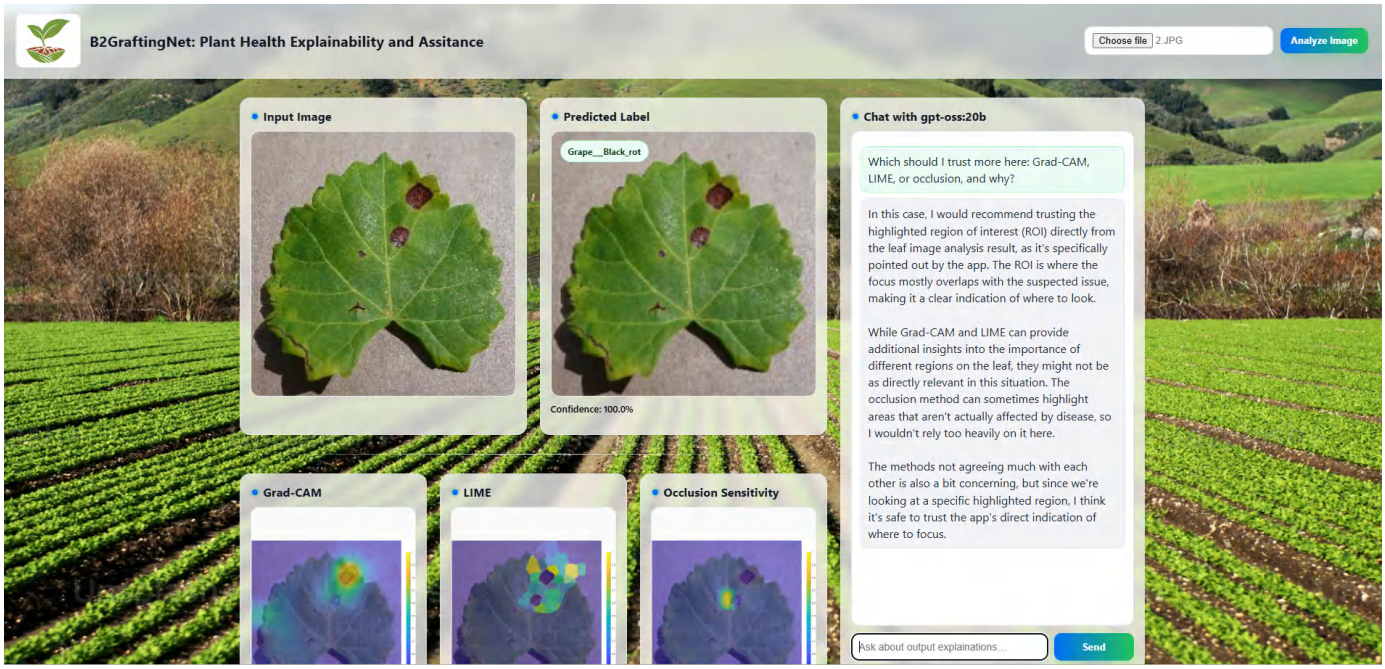

Figure A1. Full web platform view (analysis + chat): the left column shows the input image, predicted label, and three explainability maps. The right column hosts a scrollable chat with the local LLM (gpt-oss:20b) grounded by the latest analysis context.

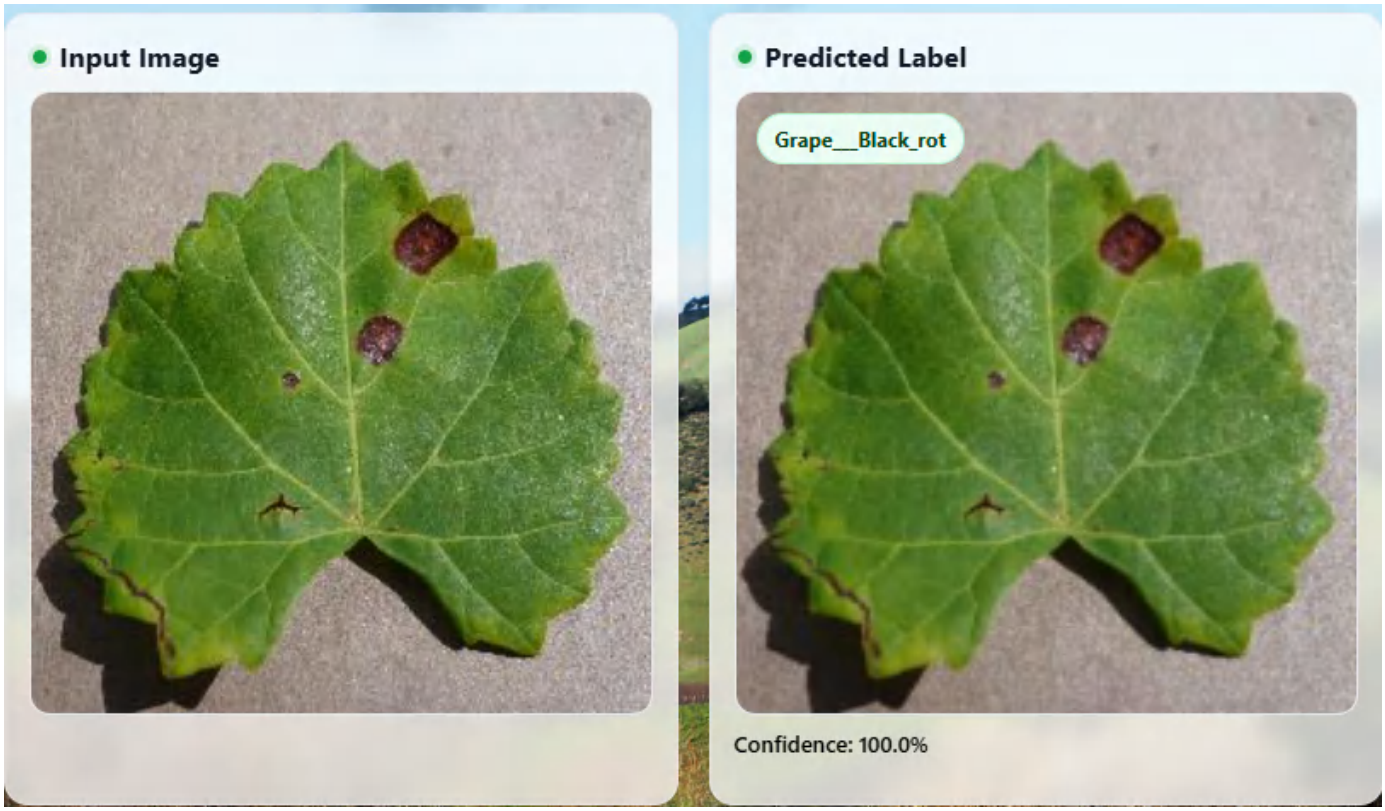

Figure A2. Input vs. Predicted Label panels: the original leaf photo is displayed alongside the predicted class with a confidence indicator. The label badge provides at-a-glance identification.

3. (Optional) Relay user questions using POST /chat for on-device explanatory responses without exposing farm data to the cloud.

Endpoints are stateless; the "latest analysis context" is refreshed on every POST /upload and is automatically used to ground subsequent text queries.

Figure A3. Explainability triptych based on Grad-CAM (left) that highlights regions most responsible for the decision; LIME (middle) marks important super pixels; Occlusion Sensitivity (right) reveals where masking reduces the class score the most. Consistent color bars and borders aid comparison.

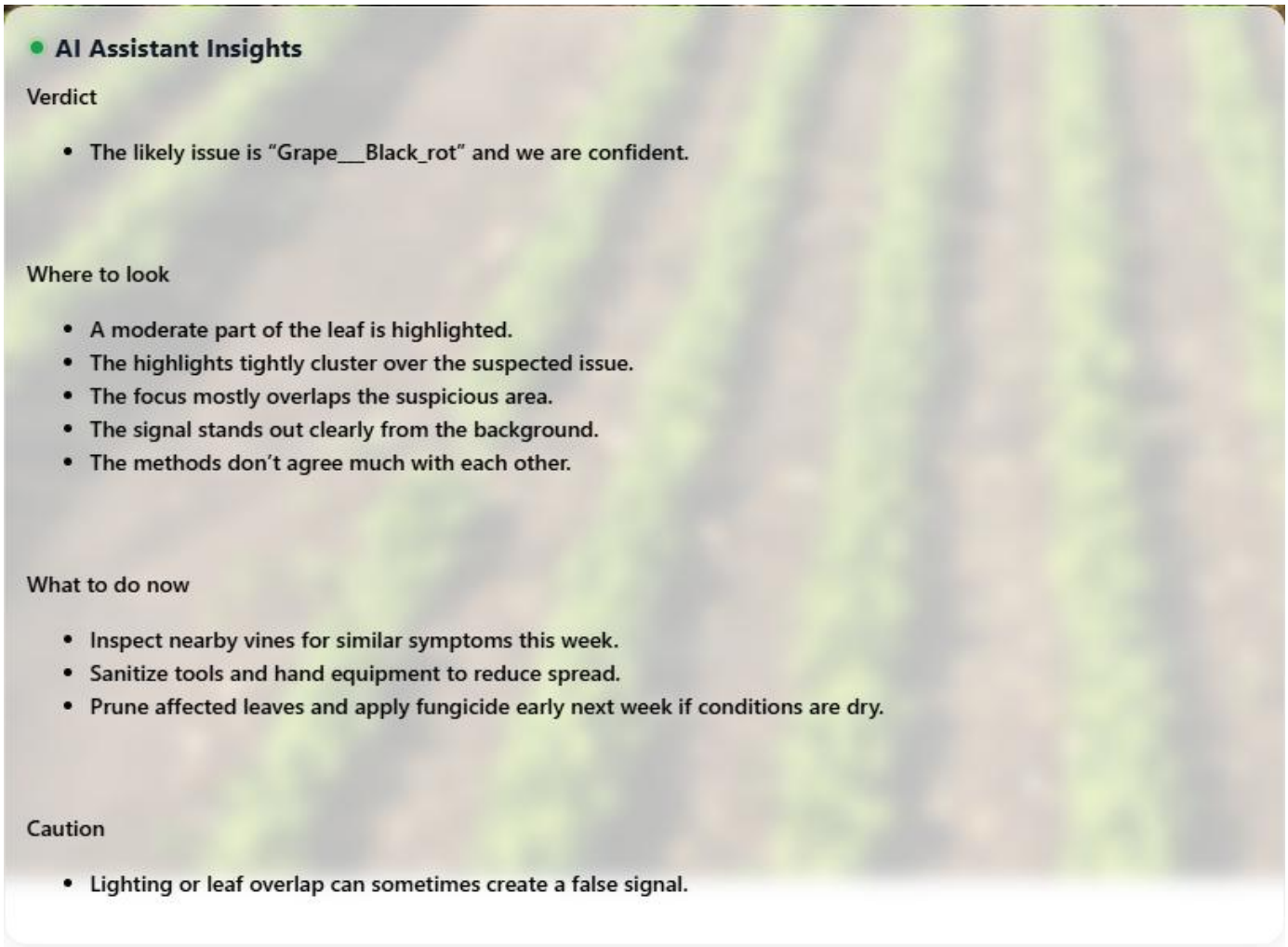

Figure A4. AI Assistant Insights based on plain-language summary which is rendered in four short sections verdict, where to look, what to do now, and caution derived from ROI-aware metrics and generated by the local LLM.

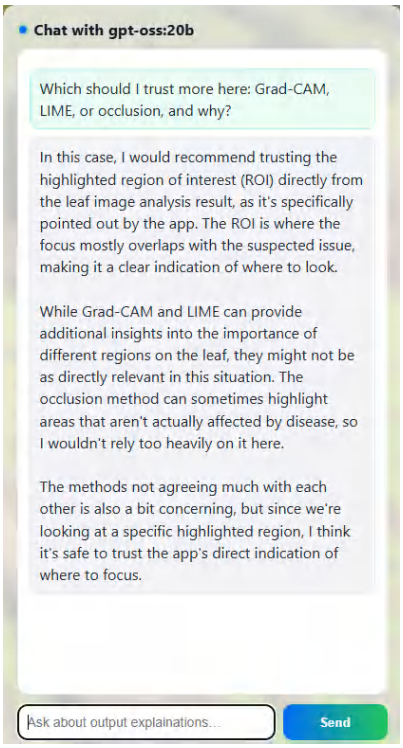

Figure A5. Results-only chat assistant providing additional interpretation of the prediction and XAI-highlighted regions for the uploaded leaf image.

A11 Validation of assistant responses

Two independent horticulturists reviewed a sample of LLM-generated outputs across varied images (healthy leaves, black rot, leaf blight, and mixed cases). They confirmed that the messages (i) avoid jargon, (ii) point attention to plausible regions, and (iii) suggest safe, conservative next actions (inspection, sanitation, pruning, and following labeled fungicide guidance where appropriate). Feedback was incorporated into the phrasing rules used to summarize metrics before prompting the LLM.

A12 Security, privacy, and offline operation

All inference occurs locally. Images remain on the host machine inside the project folder. The LLM module runs via Ollama and does not transmit content externally. This design allows offline operation in the field and avoids sending farm images to third-party services. Administrators can reverse-proxy the Flask server and restrict access with standard network controls.

A13 Troubleshooting

1. LLM unavailable: verify ollama serve is running, gpt-oss:20b is pulled, and OLLAMA_BASE points to the correct host/port. The UI will still show classification and XAI if the LLM module is down.
2. MATLAB errors: confirm B2-GraftingNet.mat exists, the Deep Learning Toolbox is installed, and the image path is valid. For GPU environments, ensure CUDA/cuDNN are configured or switch to CPU.
3. Stale images: the app appends cache-busting query strings; if you use a proxy/CDN, disable long-term caching for uploads/ and outputs/.

A14 Hallucination risk and mitigation (LLM safety)

Because LLMs can produce unsupported statements ("hallucinations"), we constrain the assistant to evidence available in the current inference. The backend injects a compact, machine-generated context containing the predicted class and XAI-derived summaries, and the assistant is instructed to only explain within that context. Additionally, we use deterministic decoding (low temperature) and a fixed response template (Verdict/Where to look/What to do now/Caution) to minimize free-form generation. Off-topic questions are blocked by server-side gating and answered with a fixed refusal message, ensuring the model cannot be prompted into unrelated or speculative outputs. Finally, domain experts reviewed sampled assistant outputs for factual consistency and conservative agronomic guidance.

A15 Limitations and intended use

Predictions assume close-up, in-focus leaves with minimal clutter and typical symptom presentations. Lighting extremes, severe occlusions, or rare cultivars may degrade performance. The LLM-generated explanations provide interpretive context, not a diagnosis/a claim; growers should follow local regulations and product labels, and consult experts when in doubt.

Syed Baqir Hussain Shah is a dedicated computer scientist from Pakistan, specializing in intelligent application development, system designing, image processing, and computer vision. Currently pursuing a Bachelor's degree in Computer Science at COMSATS University Islamabad, Wah Campus, he has distinguished himself through academic excellence and impactful research. Syed Baqir has participated in prestigious events such as the International Conference on Neuroimaging and Neonatal Monitoring in Spain and the International GeoInformatics Summer School in China. His technical expertise spans tools like Python,

Java, and C++. Among his notable projects is an Explainable AI-based CNN for Lung Cancer Detection, reflecting his innovation in AI-driven medical imaging. He has published in leading journals like Frontiers in Big Data and the Journal of Imaging, focusing on deep learning and biomedical imaging. Syed Baqir aspires to bridge technology and industry, developing solutions to global challenges while advancing the field of computer science. (Email: bakirhussain6@gmail.com)

Farwa Naseer received the B.S degree in computer Science from The Women university Multan, Pakistan, in 2017, and the M.S degree in Computer Science from Muhammad Nawaz Sharif University of Agriculture Multan, Pakistan, in 2021. Her research during the B.S focused on machine learning and deep learning applications, while her M.S research was in the field of machine and deep learning with emerging techniques. (Email: farwanaseer042@gmail.com)

Syed Adil Hussain Shah working in GPI SpA (Trento, Italy) as an IT researcher under the supervision of Angelo Di Terlizzi (Project Manager) and a PhD scholar (Marie Curie Fellow; PARENT Project URL: (https://parenth2020.com/) at Politecnico di Torino (Turin, Italy) under the supervision of prof. Marco Agostino Deriu. Currently, he is working on the development of novel IoT-based solutions to the early diagnosis of newborn motor/cognitive impairments. His field of expertise is related to machine vision, biomedical imaging, and intelligence system development. In 2022, he did his master in the filed of Computer Science with the specialization of biomedical image analysis from COMSATS University of Islamabad, Wah campus, Pakistan. In this study era, he specialized in Artificial Intelligence and related areas namely Computer Vision, Advance Neural Network, Pattern Recognition, Machine and Deep learning. Moreover, his master thesis focused on medical image processing (Title: "An ensemble deep neural network method for the classification of Pneumonia using X-ray images"). The main objective of this study is to focused on deep multi parallel fusion layers to analysis the multi aspect features of the X-ray images and classify the pneumonia cases at earlier stage. In 2019, he did his bachelor degree in the field of Computer Science from Muhammad Nawaz Shareef University of Agriculture, Multan, Pakistan. (Email: syedadilhussain.shah@gpi.it)

Kashif Razzaq is an Associate Professor specializing in horticulture with a strong focus on postharvest science. He holds a PhD in Horticulture and has authored 84 publications spanning fruit quality management, postharvest physiology, storage and packaging technologies, and value addition in horticultural crops (including work on grapes, mango, pomegranate, strawberry, and other minor fruits). His recent research addresses both practical and mechanistic questions such as improving shelf life and physicochemical quality, reducing postharvest losses, and evaluating pre- and postharvest treatments and innovative handling methods to maintain nutritional and sensory attributes. (Email: kashif.razzaq@mnsuam.edu.pk)

Tahir Javed received his Bachelor's degree in Computer Science from Arid Agriculture University, Rawalpindi, Pakistan. He completed his Master's degree in Computer Science from Iqra University, Islamabad, Pakistan. His research interests are focused on Artificial Intelligence, particularly machine learning, deep learning, neural networks, computer vision, and intelligent systems. He is actively engaged in research exploring advanced AI techniques and their real-world applications. He is affiliated with the School of Computing and Information Technology, Multan University of Science and Technology, Multan, Pakistan. (Email: tahir.javed@multanust.edu.pk)

Qandeel Asghar is a Lecturer in the Department of Computer Science and IT at the University of Southern Punjab, Multan, Pakistan, where she has been serving since 2024. She holds an MSc in Computer Science from Air University, Pakistan, and a BS in Computer Science from Bahauddin Zakariya University, Pakistan. Prior to her current role, she worked as a Visiting Lecturer at The Women University, Multan (2020–2021). Her academic and teaching interests center on programming, database systems, and software engineering, and she maintains an active professional presence online through her LinkedIn profile reflecting her academic background and professional activities. (Email: qandeel@usp.edu.pk)

Gohar Bano Zaidi is from Pakistan, currently residing in Italy. I completed two years of Bachelor's-level studies in Statistics and Economics in Pakistan, followed by a professional Cost and Management Accountancy (CMA) qualification an intensive post-graduation program equivalent to a master's level pathway which gave me a solid foundation in analytical thinking, financial systems, and evidence-based decision-making. Building on this background, my current interests have shifted toward Artificial Intelligence, with a focus on machine learning and computer vision and their real-world impact across domains. I am particularly motivated by how AI can support high-stakes problems such as risk assessment, forecasting, anomaly detection, and decision intelligence, including (but not limited to) applications in finance and business environments. Alongside my research-driven goals, I am currently working at Politecnico di Torino as a communicator and project management support for the GALATA EU Horizon 2020 project, where I contribute to coordination, dissemination, and stakeholder engagement in an international research setting. With professional experience and a strong quantitative foundation, I am eager to deepen my technical skills in AI/ML and computer vision and contribute through meaningful, interdisciplinary research. (Email: gohar.zaidi@polito.it)

Giacomo Di Benedetto is a biomedical engineer and innovation leader, currently serving as CEO and partner of 7HC Srl (Rome, Italy), where he leads the development of a biological data platform. He earned a PhD in Biomedical Engineering from Politecnico di Torino (2013) and a degree in Electronic Engineering (100/100) from Politecnico di Milano (1995). Alongside his role at 7HC, he works as Innovation Manager and International Projects at INiBICA (Cádiz, Spain) and is a partner/principal at Enginlife Engineering Solutions (Turin, Italy). He is a Certified Measurement and Verification Professional (CMVP) and has been registered in the Register of Engineers since 2011. (Email: giacomo@7hc.tech)

Syed Taimoor Hussain Shah serves as a PostDoc Researcher within the GALATEA Project at Politecnico di Torino, Italy. He has completed his PhD under the PARENT project, which is funded by the European Union's Horizon 2020 initiative. He is based at the Politecnico di Torino in Turin, Italy. Shah earned a Master of Science in Computer Science from the Pakistan Institute of Engineering and Applied Sciences and a Bachelor of Science in Computer Science from Bahauddin Zakariya University. His current focus involves contributing to various projects aimed at developing computational explainable-AI engines. These engines are designed to predict the evolution of pathology based on clinical, biological patient variables, and imaging data, with a focus on adults and neonates. Shah's research interests encompass machine learning, deep learning, and pattern recognition with an emphasis on computer vision. He has contributed to the academic landscape with published works, including original articles and conference papers in esteemed peer-reviewed journals such as Frontiers, MDPI, IEEE, Springer, and CEUR Workshop Proceedings. Shah's specific research endeavours extend to areas such as facial feature extraction, characterization, and identification in both adults and infants. Additionally, his interests include hyperspectral imaging, pattern classification, and recognition, as well as other facets of biomedical and automated computer vision, machine learning, and deep learning systems. (Email: taimoor.shah@polito.it)

Syed Bilal Hussain is an accomplished academic and researcher, currently holding the position of Lecturer/Assistant Professor at the School of Agriculture and Food Science, University College Dublin (UCD) since September 2023. His academic journey began with an undergraduate degree in Horticulture from the Department of Horticulture at Bahauddin Zakariya University, Multan, Pakistan, in 2012. Building on this foundation, he pursued a master's degree in Horticulture at the same institution, where his research, titled "Physico-Chemical Profiling of Promising Citrus Cultivars Grown Under Different Agro-Climatic Conditions of Punjab (Pakistan)", showcased his early dedication to the field. In 2016, Dr. Hussain embarked on an academic adventure abroad, receiving a Chinese Government Scholarship for his doctoral studies. He completed his Ph.D. at the College of Horticulture & Forestry Sciences, Huazhong Agricultural University, P.R. China. His groundbreaking research during this period centered on the identification and characterization of citrus vacuolar H+-pyrophosphatase genes, shedding light on their crucial role in regulating sucrose and citrate accumulation. Returning to Pakistan after completing his Ph.D., Dr. Hussain rejoined the academic sphere as a Lecturer in the Department of Horticulture at Muhammad Nawaz Shareef University of Agriculture (MNS-UAM), Multan, Pakistan, in 2017. His role encompassed teaching and mentoring undergraduate and postgraduate students, securing research funding, publishing peer-reviewed articles, and actively engaging in research projects aimed at advancing horticultural knowledge. In September 2021, Dr. Hussain embarked on a new chapter in his academic journey by accepting a prestigious postdoctoral research fellowship at the Citrus Research and Education Center (CREC), Institute of Food and Agricultural Sciences, University of Florida, USA. His postdoctoral research focused on plant phenology and leaf position's impact on carbon fixation and translocation characteristics. Additionally, he made significant contributions to the Hurricane Ian citrus recovery project. Throughout his academic career, Dr. Hussain has been a prolific researcher, with 30 SCI articles and one book chapter to his name. He has also been an active participant in both national and international conferences, presenting his research findings to diverse audiences. His dedication to horticultural science and his extensive experience in teaching, research, and publication underscore his valuable contributions to the field. (Email: syedbilal.hussain@ucd.ie)

Marco Agostino Deriu is Professor of Industrial Bioengineering at Politecnico di Torino. He has received the European Doctorate in Biomedical Engineering in 2009 at Politecnico di Torino, Italy. His research is focused on computational modelling, mechanistic modelling, artificial intelligence driven data analysis applied to investigate mechanisms behind biological, physiological and pathological functions. In this context, he employs multiscale modelling to investigate molecular mechanisms of cancer and neurodegenerative diseases, structure function relationships in physiology and pathology, drug mechanism of action, drug delivery systems, protein folding, nanoparticle features and biophysical properties. Part of his research is also devoted to the development of modelling methodologies to interpret and treat biophysical, biological and clinical data, to build prediction models in physiology and physiopathology. He teaches "Multiscale Modeling in Biomechanics", "Biomechanical Design", and "Rational Drug Design: Principles and Applications" at Politecnico di Torino. He is author of several publications in International peer-reviewed Journals, book chapters and proceedings in the field of computational modelling applied to Bioengineering, Biophysics, Molecular Biology and Medicine. (Email: marco.deriu@polito.it)