mathematics

MDPI

Article

Recognition of Sunflower Diseases Using Hybrid Deep Learning and Its Explainability with AI

Promila Ghosh [1], Amit Kumar Mondal [1], Sajib Chatterjee [1], Mehedi Masud [2], Hossam Meshref [2] and Anupam Kumar Bairagi [1,*]

[1] Computer Science and Engineering Discipline, Khulna University, Khulna 9208, Bangladesh
[2] Department of Computer Science, College of Computers and Information Technology, Taif University, P.O. Box 11099, Taif 21944, Saudi Arabia
* Correspondence: anupam@cse.ku.ac.bd

Abstract: Sunflower is a crop that has many economic values and ornamental usages. However, its production can be hampered due to various diseases such as downy mildew, gray mold, and leaf scars, and it is challenging for farmers to identify disease-prone conditions with traditional approaches. Thus, a computerized model composed of vision, artificial intelligence, and machine learning is the demand of the age to detect diseases in plants efficiently. In this paper, we develop a hybrid model with transfer learning (TL) and a simple CNN using a small dataset for detecting sunflower diseases. Out of the eight models tested on the dataset of four different classes (downy mildew, gray mold, leaf scars, and fresh leaf), the VGG19 + CNN hybrid model achieves the best results in terms of precision, recall, F1-score, accuracy, Hamming loss, Matthews coefficient, Jaccard score, and Cohen's kappa metrics. The experimental outcomes show that the proposed model provides better precision, recall, and accuracy than other approaches on the benchmark dataset.

Keywords: sunflower diseases; transfer learning; deep learning; hybrid model; explainable AI; LIME

MSC: 68T07

Citation: Ghosh, P.; Mondal, A.K.; Chatterjee, S.; Masud, M.; Meshref, H.; Bairagi, A.K. Recognition of Sunflower Diseases Using Hybrid Deep Learning and Its Explainability with AI. *Mathematics* **2023**, *11*, 2241. https://doi.org/10.3390/math11102241

Academic Editor: Catalin Stoean

Received: 18 April 2023
Revised: 3 May 2023
Accepted: 8 May 2023
Published: 10 May 2023

1. Introduction

Sunflower's scientific name, Helianthus annuus, is a flora of Asteraceae, which originated in Mexico in 2100 BCE, and it has many economic and ornamental usages [1]. Many countries cultivate sunflowers to meet consumer demand. However, countries such as Ukraine and Russia are the top growers. Sunflower is invaluable from both an economic and ornamental perspective. The seeds possess oil and are also utilized as food. Moreover, we obtain hay from the leaves and a yellow dye from the flowers. However, diseases such as downy mildew, Phom blight, Verticillium wilt, leaf scars, leaf rust, and Septoria leaf spot can negatively affect sunflower production [1,2]. Moreover, most of our farmers, especially in developing countries, are illiterate and technologically unsound, which leads to more damage from such diseases. Primarily, the farmers classify or detect diseases manually through visual observation, which mainly depends on the experience of the farmers and is probably highly error-prone. Again, farmers cannot notice the signs of diseases periodically with only visual observation, which leads to substantial financial losses [1]. Moreover, spectrometers can classify plant leaves as healthy or infected, better than traditional techniques [3]. Moreover, molecular methods can also be used to identify plant diseases, namely, the chain reaction of polymerase [4] and the chain reaction of real-time polymerase [5]. Such approaches are time-consuming, expensive, and challenging and require skilled professionals to operate. Hence, an image-based self-regulating internet of things (IoT) framework will be relatively more convenient than any conventional system.

With the expansion of computer vision (CV) and machine learning (ML), it is now within our hands to create computerized models that enable the accurate and convenient

detection of plant diseases [6]. Various ML techniques [7] are utilized for the automated detection of plant diseases. ML significantly improves the speed and accuracy of plant disease detection, which can potentially enhance global food security [8]. Feature extraction is a method that is used in ML. However, most of the imperative features need to be specified by domain specialists to lower the complexity of the data and create patterns more perceptible to the algorithms in traditional ML techniques.

Recently, deep learning (DL) approaches have enhanced MLs dramatically in terms of learning and extracting crucial features from the data that are required for more accurate pattern detection [9]. The aspiration of DL algorithms comes from the operational structure of the human brain [10] and is capable of processing high dimensional data such as images, video, etc. It reduces our burden for performing complex tasks such as pattern recognition, classification, disease detection, and language analysis [6,7,11,12]. Recently, it has also been rigorously used in the Agri-domain, especially for the detection of plant disease; crop prediction; plant categorization; pest range; and pesticide impact assessment [1,6,13,14].

Within DL, the convolutional neural network [10], defined as ConvNet or CNN, is a category of DL algorithm that utilizes data with a matrix, such as an image by extracting different features. Feature extraction transforms plain data into numeric values that can be processed for further analysis [2,15]. Moreover, the CNN model is widely used for plant disease detection with reasonable accuracy [12]. However, DL models need a large dataset to learn and decide. In this scenario, transfer learning (TL) plays an important role via the transfer of proficiency from a related concept to a new problem [1]. It raises fast progress or boosted execution when modeling the next task. In TL, we reuse a pre-trained method as the beginning point for a model on a new assignment. In this research, we introduce a hybrid model of TL and CNN. A hybrid algorithm is a technique that converges two or more other techniques together that solve the exact issues [2]. Here, we want to use the TL model to extract features and then apply CNN techniques for classification from sunflower disease on a small dataset [16].

To make the DL models trustworthy, we need something to analyze and explain the outcomes. Explainable artificial intelligence (XAI) can serve our purpose in this regard; it refers to a group of methods and plans that enable people to comprehend and have confidence in the outcomes of machine learning methods that are opaque. We also employ an XAI approach that seeks clarity on how the model operates by defining model precision, impartiality, clarity, and consequences in decision-making driven by AI [17]. To examine the model without affecting its efficiency, the method employs post hoc and intrinsic methods after training. A widely used explainable artificial intelligence method called local interpretable model-agnostic explanations (LIME) is adopted to explain the functioning of ML and deep learning (DL) models by providing localized, model-agnostic interpretations [18,19].

A few studies in the literature [1,2,20] deal with the recognition of sunflower diseases using DL/ML algorithms, and current studies have not explained the outcomes of the models with intelligent tools. Moreover, there is still scope to improve the disease recognition rate to promote sunflower production. Hence, there is an urgent need to do more research in the domain. This study uses a hybrid DL technique to effectively detect sunflower diseases from a small dataset. It investigates the effectiveness of eight different hybrid algorithms in combination with a CNN, uses TL algorithms for feature extraction and CNN for classification, and evaluates the performance of the proposed approach in different standard parameters and compares it with other state-of-the-art approaches. Moreover, this research introduces the use of LIME to identify significant features of correctly classified samples and misclassified images. It provides insights into the inner workings of the model, contributing to the interpretability of DL models. The overall contributions of this paper are as follows:

- We proposed a hybrid model with a TL and a DL framework using a small dataset for detecting sunflower diseases.

- This research evaluates the proposed model using different diversified metrics such as precision, recall, confusion matrix (CM), F1-score, accuracy, Hamming loss, Matthews coefficient, Jaccard score, and Cohen's kappa.
- This research justifies the proposed model by differentiating it from other state-of-the-art models using precision, recall, and accuracy metrics.
- This research analyzes the model using explainable AI frameworks such as LIME.

This paper has the following sections. Section 2 presents related studies based on plant disease detection, while Section 3 explains the process and methods used in this study. Section 4 presents the results of the proposed approach, a comparison, and a discussion. We conclude the paper in Section 5.

2. Related Work

In [1], the authors proposed a hybrid model by ensembled VGG16 and MobileNet. The authors collected images of four sunflower diseases from Google. The model VGG-16 has 81% accuracy, and MobileNet has 86% accuracy. Inception_V3, DenseNet-121, AlexNet, CNN, ResNet-50, ResNet-101, ResNet50V2, and VGG-16 have also been adopted. The models provided the best result when in combination. The work presented in [2] is the primary research of [1], where they used MobileNet, AlexNet, InceptionV3, and DenseNet121. In [20], the authors proposed a method for classifying four types of sunflower diseases. In [1,2], the authors applied stacking ensemble learning with the VGG-16 and MobileNet architectures on a dataset of size 329. Similarly, in [2], the authors compared six different models in this dataset . In contrast, ref. [20] used a larger dataset of size 650, which they preprocessed by resizing, converting color, and enhancing contrast. They found that random forest provided the best results, although they did not provide an explanation of the significant features used by the model. Leaf scars, leaf rust, gray mold, downy mildew, and disease-free samples were present in a dataset they collected [16].

A study [21] proposed a hybrid deep learning-based framework for detecting multiple diseases in guava leaves in real time. They used modified MobileNetV2, U-Net, and YOLOv5 models and were trained and validated on two self-collected datasets. The results showed that the framework could detect five distinct disease classes with an average accuracy of 73.3%. In [14], the authors investigated the effectiveness of pre-trained classification models and transfer learning to improve results in the urban planter dataset, consisting of fifteen urban plant species.

This study [17] utilized a machine-vision-based algorithm and hyperspectral techniques to detect apple pesticide residues. They obtained the region of interest and hyperspectral images of the apples and added noise to expand the dataset. Finally, they used a convolutional neural network to detect pesticide residues. The technique is considered fast, effective, and low-cost, providing a helpful tool for pesticide residue detection in post-harvest apples. A study was conducted to identify apple leaf diseases of six kinds in [22]. The authors used DenseNet-121, regression, and multi-label classification, focusing on the loss function. Their proposed methods achieved significant accuracy, with values above 93%.

The authors of [23] focused on developing an agro-medical system for recognizing diseases of jackfruit in Bangladesh. The system uses digital images taken with an edge device and K-means clustering segmentation to mark disease-affected regions, extract features, and classify diseases using classification methods. The study addresses the challenges of disease detection and classification and evaluates the performance of the classifiers using various performance metrics.

The study [24] developed a papaya disease identification system to assist distant agronomists with cultivation. It addresses the challenges of disease detection and classification using an expert system based on machine vision. The system achieved over 90% classification accuracy, demonstrating its potential as a tool for papaya disease recognition.

The authors submitted a deep CNN model for plant disease identification with a dataset of 39 different classes of plant leaves images in [25]. They gained 96.46% identifica-

tion accuracy, higher than traditional machine learning approaches and better than popular transfer learning methods.

In [26], the authors proposed a system for detecting sunflower diseases by segmenting and classifying sunflower leaf images. The system uses particle swarm optimization for leaf image segmentation and different classification techniques for disease recognition. The experiments showed satisfactory results, achieving an average accuracy of 98%, outperforming the state-of-the-art approaches. However, the preliminary limitation of the system is the requirement for speedy image segmentation.

The authors of [27] developed a CNN-based model to identify apple diseases. The proposed ML method was compared to traditional ML algorithms and pre-trained models (InceptionV3 and VGG16) and was found to be more accurate, faster, and require less space. The proposed model achieved an accuracy of 99%.

Using a single plant leaf image, a deep neural network structure for end-to-end plant disease diagnosis is proposed in [28]. They used U-Net to separate the leaves from the background and a two-head network that extracted features from pre-trained models to recognize plant diseases. They achieved an accuracy of 87.45% for disease recognition.

The authors developed an efficient disease detection system for tomato plants using computer vision and deep learning in [29]. Their work focused on three diseases: target spot, leaf miner, and Phoma Rot of tomatoes. The system used two models: an F-RCNN-trained anomaly identification model with a confidence value of 80% and a transfer learning disease recognition model with 95.75% accuracy. The automated image-capturing system was tested in natural conditions and achieved an accuracy of 91.67% in recognizing diseases of tomato leaves. The article [30] highlights the importance of agriculture in India and how plant diseases can lead to a significant loss in agricultural production. Manually identifying plant diseases is challenging due to time limitations and disease diversity, and there is a need for automatic disease detection. The article presents a study that uses transfer learning to train an EfficientNet B7 deep architecture on grape plant images to detect four types of diseases: leaf blight, black rot, stable, and black measles. The study uses a logistic regression technique to down-sample the collected features and identify the most discriminant traits. The proposed technique achieves an accuracy of 98.7% using state-of-the-art classifiers after 92 epochs. The article concludes by suggesting the proposed technique's effectiveness for plant disease identification in grapevines and provides a fair comparison to existing procedures. The research [31] discusses the negative impact of weeds on crop quality and yields and how traditional weed removal methods are time-consuming and challenging. The study proposes a deep convolutional neural network-based Inception V4 architecture approach for identifying weed density in soya bean crop fields using RGB weed and crop images. The study uses data cleaning to eliminate background and foreground vegetation and vegetation segmentation to identify the weed-density area, which is a significant challenge. The approach is validated using the crop weed field image dataset (CFWID) and achieves an accuracy of 98.2% using 4384 weed images. The proposed approach has been generalized to different weed species in the soya bean crop, with precision, recall, and F1 scores of 97%, 99%, and 98%, respectively. The study [32] discusses the importance of agriculture in a country's economy and the need for better crop yields. Deep learning (DL) techniques have been successful in image recognition, but they require large datasets to prevent overfitting. Image augmentation techniques such as rotation, zoom, and shift can generate new images from the original set, but they do not necessarily improve classification accuracy. The article proposes two new algorithms, IPTA and IMHSA, to address the issue of limited dataset and overfitting in convolutional neural network models. The proposed approach involves transforming original images into augmented ones using an adaptive supervised learning approach and unsupervised RGB image segmentation. The article presents experimental results that demonstrate that the proposed approach improves the performance of the convolutional neural network model and solves the overfitting problem, resulting in better classification accuracy of RGB images. Anand Muni Mishra et al. discuss [33] the growing popularity of automating agricultural food production to detect and

identify weeds in crops. Weeds compete for nutrients, space, and resources with target crops, and their growth rate is assessed in different types of soil in the rabi crop field. The study collected weed image data from ten different rabi crops and ten different weeds in three different locations in Madhya Pradesh, India using Intel Real Sense LiDAR and Canon digital cameras. The collected images were used to train and validate two deep learning models, EfficientNet-B7 and Inception V4, with 97% and 94% accuracy, respectively. The study found that EfficientNet-B7 outperformed Inception V4 in terms of weed classification accuracy. Previous studies have primarily focused on utilizing large datasets or data augmentation techniques in their research. Unfortunately, there is currently a lack of research on the topic of sunflower disease. To address this gap, we have leveraged the advancements in deep learning to develop a sunflower disease detection system that can support the agriculture industry. The summary of the related works is illustrated in Table 1.

Table 1. Summary of the related works.

Paper	Year	Crop and Number of Data (Images)	Methods and Results
[1]	2022	Sunflower, 329	VGG-16 + MobileNet, accuracy 89.2%
[2]	2021	Sunflower, 329	VGG-16 + MobileNet, accuracy 89.2%
[20]	2021	Sunflower, 650	Random forest, accuracy 90.68%
[21]	2023	Guava Leaf, 1196	GIP-MU-Net model, 92.41% accuracy
[14]	2022	urban planter, 1500	DenseNet201, 96% accuracy
[17]	2019	Apple, 72	CNNS, Recognition rate 99.09%
[22]	2022	Apple leaf, 2462	DenseNet-121, accuracy 93.71%
[23]	2022	Jackfruit, 480	Random forest, accuracy 90%
[24]	2022	Papaya, 243	SVM, accuracy 95.2%
[25]	2019	Plant leaves, 61,486	Deep CNNs, accuracy 96.46%
[26]	2019	Sunflower leaf	particle swarm optimization, 98%
[27]	2019	Apple's leaf, 2486	Faster Convolution Neural Network, 99%
[28]	2019	Plant Leaves, 40,000	U-Net, 98%
[29]	2018	Tomato Plant Leaf, 4923	Alexnet, accuracy 95.75%
[30]	2022	Grape leaves, 16,579	EfficientNet B7, accuracy 98.7%
[31]	2022	Soya bean, 4384	Inception V4, accuracy of 98.2%
[33]	2022	Weed, 3670	IEfficient Net-B7, accuracy of 97%

Our approach is designed to minimize training time complexity, which we achieve by utilizing transfer learning techniques and a small dataset specific to sunflower disease. Transfer learning has been shown to reduce the training time required for deep learning models while maintaining high accuracy in disease classification tasks. In transfer learning, a pre-trained model is used as a starting point for a new task, allowing the new model to inherit some of the pre-trained model's learned features. This approach can be particularly beneficial for medical image analysis tasks, where large labeled datasets are often difficult to obtain. A study [34] applied transfer learning to classify chest radiographs for pneumonia detection. The authors used a pre-trained DenseNet-121 model and fine-tuned it on a dataset of chest radiographs. They found that transfer learning significantly reduced the training time compared to training the model from scratch, while still achieving high accuracy in pneumonia detection.

3. Materials and Methods

The working process of the submitted terminology is presented in Figure 1. The process consists of two major steps, namely, data preprocessing and application of the hybrid DL model. The details of each of these steps are described in the subsequent subsections.

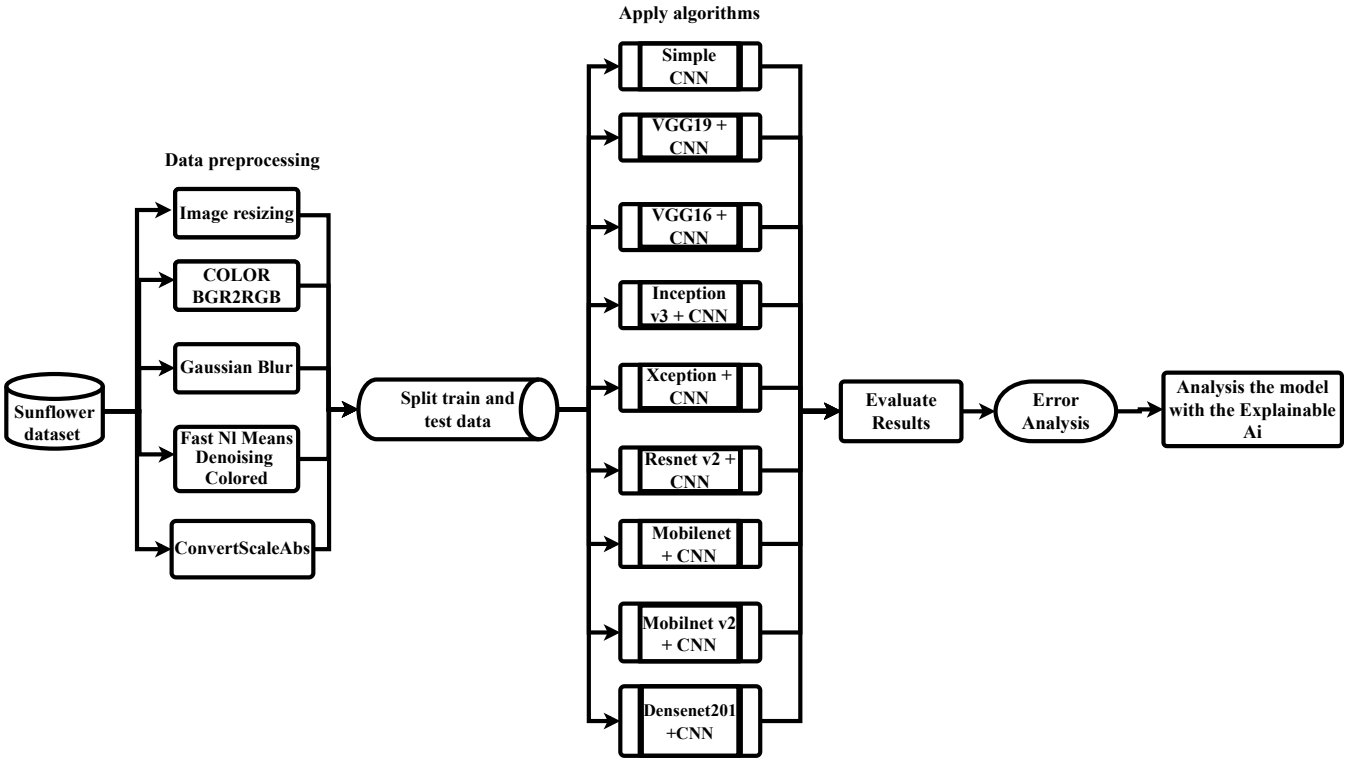

Figure 1. The entire working process of the proposed research.

3.1. Dataset

We collect the dataset from [16]. The dataset contains 467 images of fine and infected sunflower leaves and flowers with downy mildew 72, gray mold 120, downy mildew 141, and fresh leaves 134 instances. The images of size 512 × 512 were collected in November 2021; the location was Bangladesh Agricultural Research Institute—BARI at Gazipur. The samples of different categories of images from the dataset are presented in Figure 2.

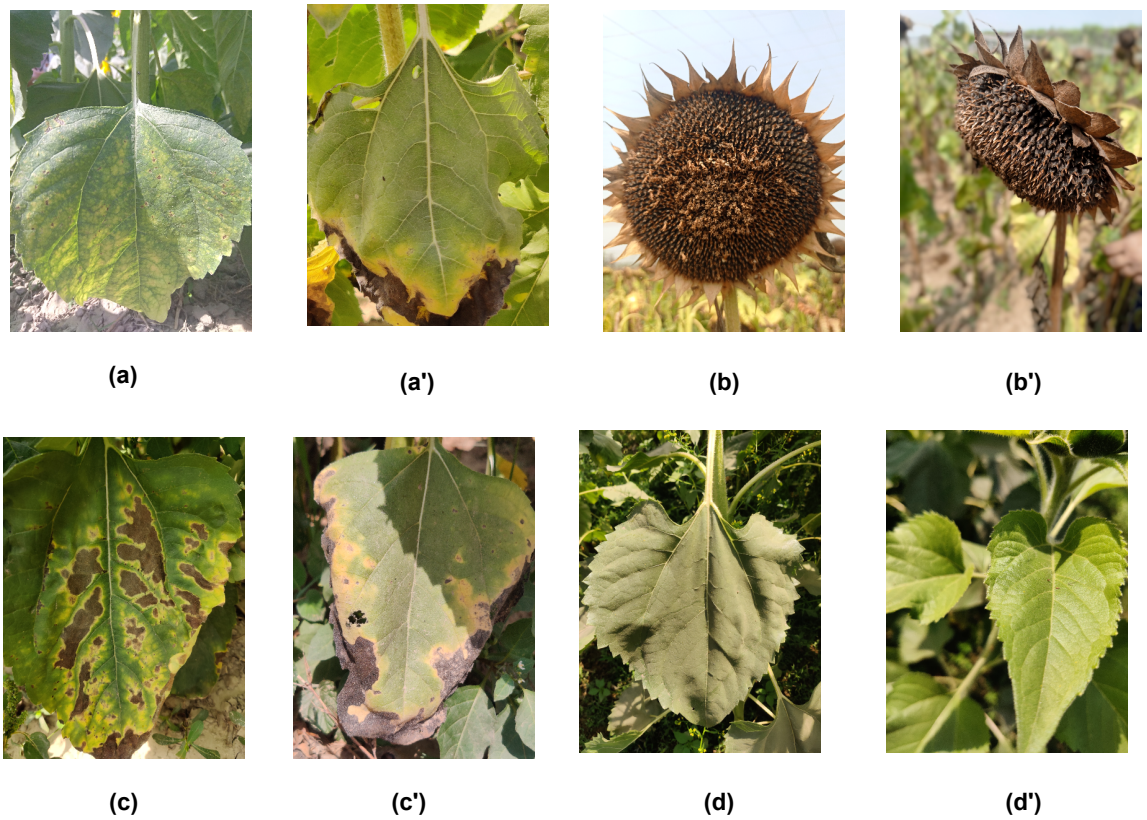

Figure 2. Sample dataset—(**a**,**a'**) downy mildew, (**b**,**b'**) gray mold, (**c**,**c'**) leaf scars, and (**d**,**d'**) fresh leaf.

3.2. Data Preprocessing

We use different preprocessing techniques in this work, and they are discussed briefly as follows. After applying the preprocessing techniques on all the images, we obtained the preprocessed images, and the samples are shown in Figure 3.

- **Image Resizing:** The images have been resized from 512 × 512 pixels to 150 × 150 pixels.
- **Color Conversion:** By default, *OpenCV* reads images in BGR format using *imread()*. The *cvtColor()* method can be used to convert BGR to RGB, and vice-versa [35].
- **Image Blurring:** *GaussianBlur* is a function in *OpenCV* that uses a Gaussian filter to smooth an image by replacing each pixel value with the average of surrounding pixels. It reduces image noise and detail, making it a common pre-processing step for computer vision tasks such as edge detection [35].
- **Image Denoising:** *fastNlMeansDenoisingColored* is a function in *OpenCV* for denoising colored images. It is based on the non-local means denoising algorithm, which can effectively remove noise from images while preserving the details. It is fast and can be used for real-time applications [35].
- **Scaling and Converting** *convertScaleAbs* is a function in *OpenCV* that scales and calculates the absolute values of a matrix/array. It converts the result of mathematical operations into a visually displayable representation and is often used to adjust image brightness, contrast, and other properties in image processing and computer vision tasks [35]. We have used $\alpha = 1.00$ and $\beta = 0$ for brightness control and contrast control, respectively.

After preprocessing, the dataset has been split into two sections—training and testing data sets, with 80% of the data used for training and the remaining 20% used for testing. The sample number in different categories for training and test data is illustrated in Figure 4.

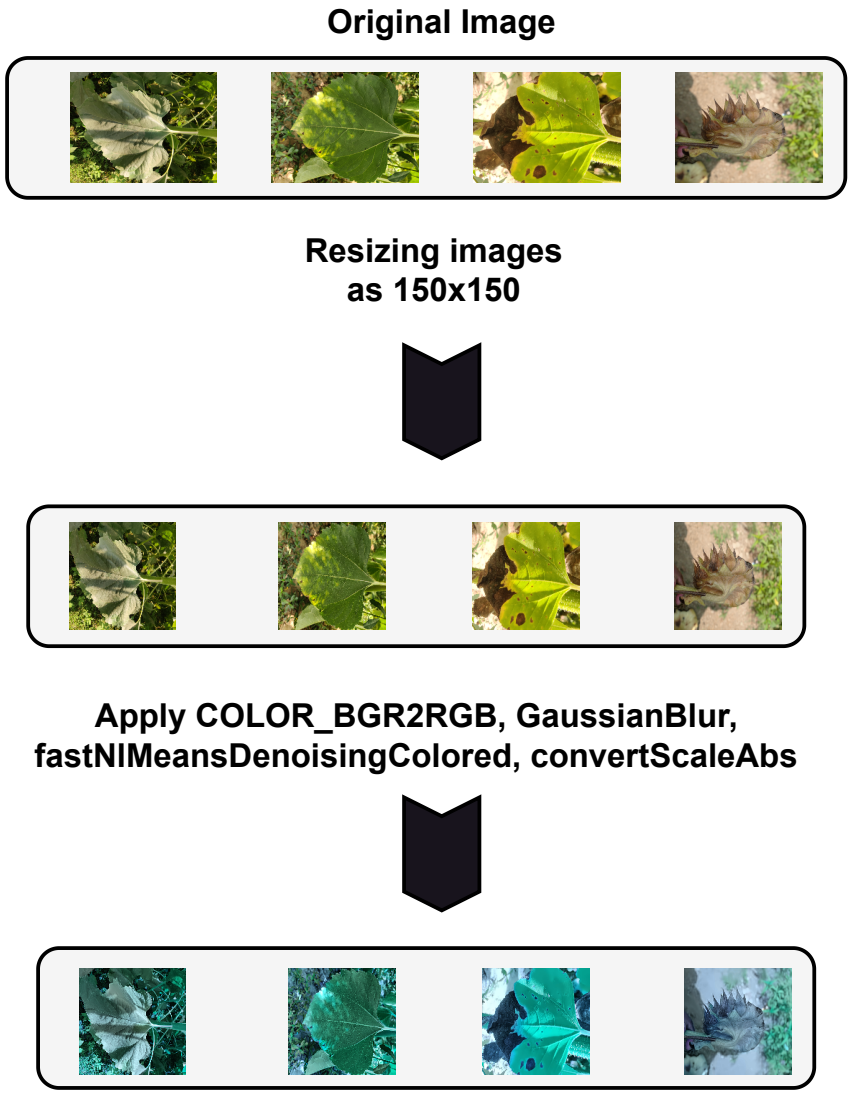

Figure 3. The preprocessed images from the original.

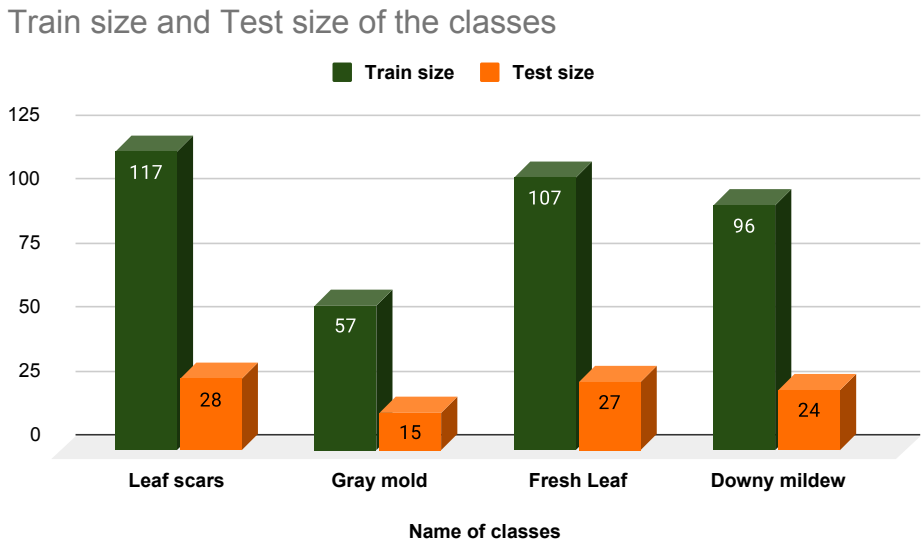

Figure 4. Number of samples for different classes.

3.3. The Architectures of Convolutional Neural Network (CNN)

DL became an influential tool for its capacity to regulate an enormous number of data. Pattern recognition is one of the sustainable applications of DL. CNN is one of the widespread methods of DL, which is most commonly applied in visual metaphors. The basic CNN structure is presented in Figure 5.

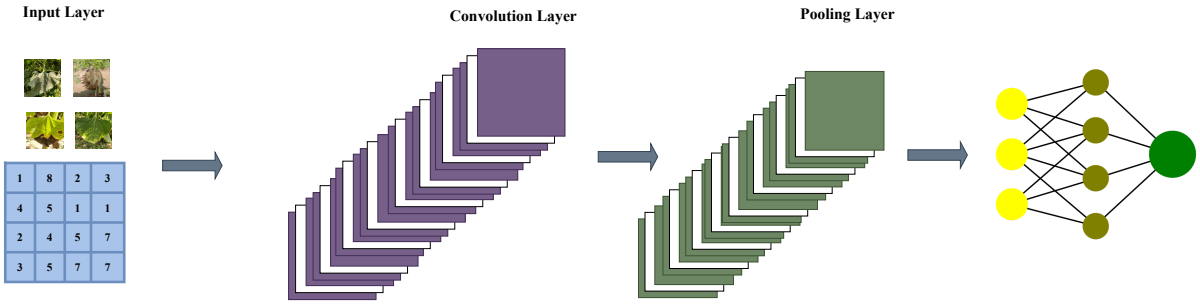

Figure 5. The structure of CNN.

In CNNs, the kernel is applied to convolutional layers on the original image or other feature maps. These layers are defined by different parameters, the most important of which are the number and size of kernels. Convolution is a mathematical process that creates a third function by combining two operations. The process is demonstrated partially in Figure 6, where a 2×2 kernel is put to an input sample image to produce the convolved feature, which is then passed on to the next layer. CNNs are composed of many layers of artificial neurons, which are mathematical methods that determine the weighted sum of inputs and produce an activation value. The first layer in a ConvNet extracts basic features, such as horizontal edges, which are then passed on to subsequent layers that detect more complex features. The number of features in each dimension in a CNN is determined by the kernel, padding, and stride size. Padding is the number of pixels added to the image. Stride is the adjustment of the kernel's action on the image.

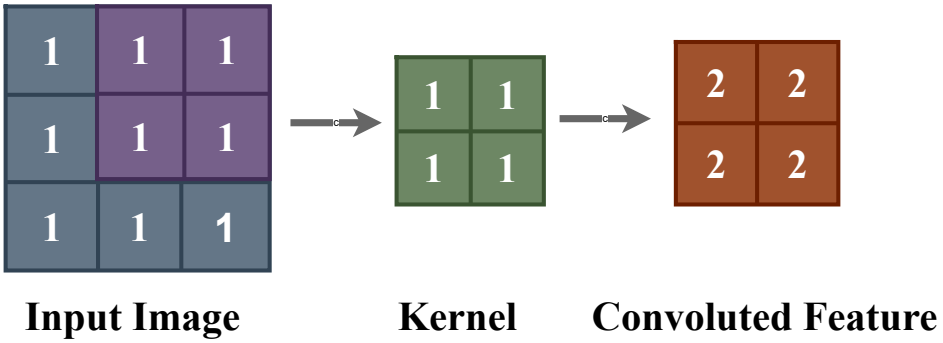

Figure 6. Convoluted feature calculation for a particular portion.

A CNN has multiple layers of artificial neurons that produce the weighted sum of sample inputs and determine an activation value. The first layer extracts basic features, such as edges, and passes them on to subsequent layers to detect more complicated features. However, CNN cannot directly understand which maps a feature looks at and how large the region is. Each feature is found at the center of its input space when the dimensions of each feature map are specified and equal to the dimensions of the input data. The convolution operation is commonly used to calculate the feature map in a convolutional layer and is defined as follows [36,37]:

$$y_{ij} = \sum_{k,l} x_{kl} \cdot w_{ij,kl} + b_{ij} \quad (1)$$

where y_{ij} is the esteem of the feature map at position (i, j), x_{kl} is the value of the input at area (k, l), $w_{ij,kl}$ is the weight of the filter at area (i, j, k, l), and b_{ij} is the bias term for the feature map at area (i, j).

The pooling layer reduces the spatial dimensions of the convolved feature, which reduces the computational power needed to process the data. We use max-pooling in this research. Max pooling picks the highest pixel value from the part of the image covered by the kernel or filter, as shown in Figure 7. Max pooling also acts as a noise filter, discarding noisy activations and reducing noise through dimension reduction. The dense layer, a part of the convolutional neural network (CNN), receives input from the preceding layer and is utilized to categorize pictures based on the results generated by convolutional layers. The dropout layer is a filter that blocks some neurons from contributing to the following layer while allowing others to operate unaltered. Flattening is the procedure of modifying data into a 1D array so that it may be passed to the next layer. The final layer flattens the convolutional layers to create one feature vector linked to the ultimate classification model, the fully connected layer.

The study implemented a simple CNN model to evaluate classification performance. This model included a convolutional layer, a pooling layer (either max or average), and an utterly connected layer. In the CNN, the activation function changes the weighted sum of sample inputs from a node into an output in a network layer. The architecture of a simple CNN is shown in Table 2.

This describes the output sizes and parameter calculations for three convolutional layers in a neural network. The first layer is a kernel size of 3 × 3 and of 32 filters, and it produces an output shape of (148, 148, 32) after max-pooling. The second layer is similar and produces an output shape of (74, 74, 32). The third layer has a smaller output shape of (17, 17, 32) and produces 9248 flattened outputs.

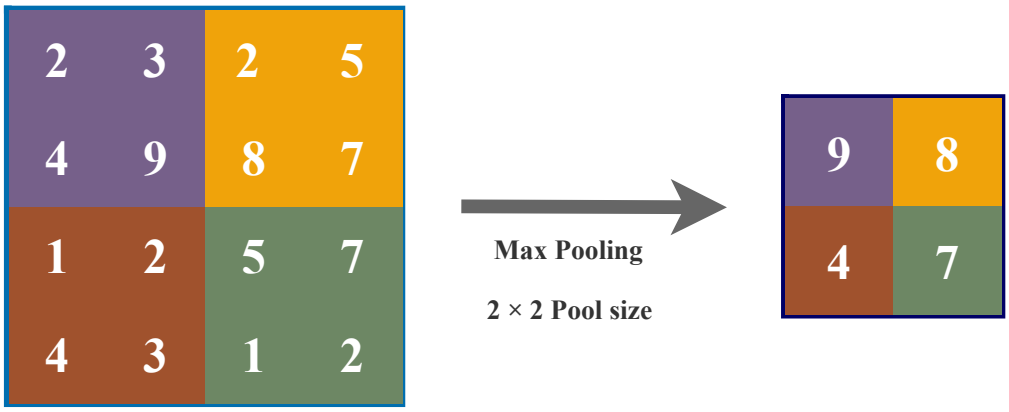

Figure 7. Max pooling computation.

Table 2. The architecture of CNN.

Variable	Value
Input Size	150, 150, 3
Kernal Size	3, 3
Pool size (Maxpooling2D)	2, 2
Dropout	0.2
Activation Function	Relu, Softmax

We have also applied a simple CNN with the extracted feature in the hybrid model. We have achieved the features through transfer learning.

3.4. Transfer Learning Model as Feature Extractor

TL is a strategy for predictive modeling on a separate but similar situation that can then be reapplied partially or entirely to accelerate the activity and improve the execution of a model on the situation of interest. It reuses the weights in one or multiple layers from a TL network model in a further model and either maintains the weights specified, fine-tunes them, or adjusts the weights completely when training the model. TL has a lower training time for a neural network, and there is a lower abstraction error.

In our work, we use the pre-trained models, namely, Xception, VGG16, VGG19, InceptionV3, MobileNet, MobileNetV2, ResNet, DenseNet121, and DenseNet201, as feature extractors and then created a hybrid model with a single flattened layered CNN model for the categorizing purpose. The working process of the hybrid model is shown in Figure 8. The following subsections briefly describe the pre-trained models used in our work.

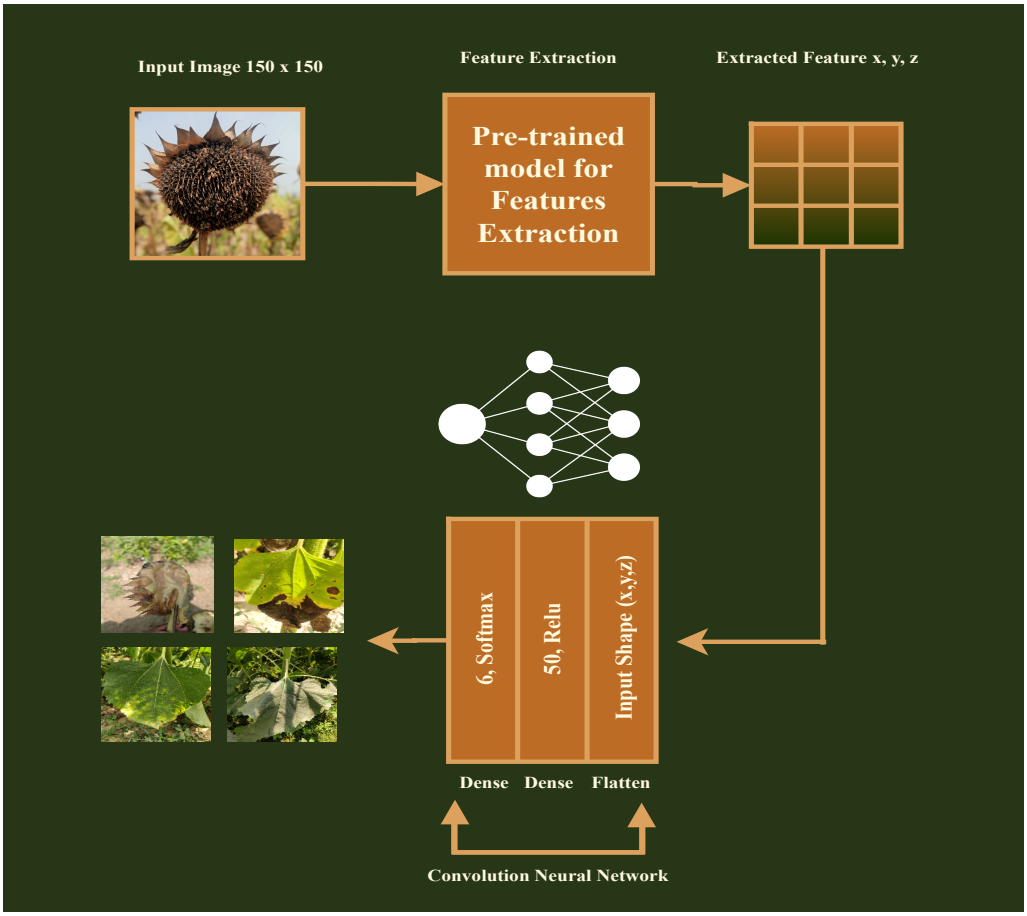

Figure 8. This is how the hybrid models work.

3.4.1. VGG19 and VGG16

The researcher used deep CNN with up to 19 weight layers to classify images in the ImageNet challenge dataset and found that increased depth improved accuracy [38]. They used the VGG19 model for feature extraction and achieved the best accuracy at epoch 11. They used mini-batch gradient descent and a batch size of 256, they regularized the activity with weight decay and dropout regularization, and they used a learning rate of 10^{-2}. The VGG19 architecture has three additional deep convolution layers compared to VGG16, with the numbers 16 and 19 indicating the number of weight layers in the CNN. The training process ended after 74 epochs and 370K iterations.

3.4.2. InceptionV3

The Inception model addresses issues in deep neural networks, such as overfitting and computational cost, using kernel transformations and smart factorization [39]. The Inception Net V3 uses the RMSProp optimizer, BatchNorm, and Label Smoothing. The combination of InceptionV3 + CNN and VGG16 + CNN achieved the best training time accuracy at epoch eight and had a faster training time than other models in the study. Table 2 illustrates the other training time accuracy results.

3.4.3. Xception

The Xception is a deep learning model that uses 36 convolutional layers for feature extraction and includes linear residual connections in 14 modules. It has a linear stack of depthwise separable convolution layers with residual connections, making it easy to define and modify [39]. The model extracted 38,400 features from an input shape of (5, 5, 1536), with 1,920,356 trainable parameters. Xception + CNN achieved the best training accuracy at ten epochs. The extracted features were fed into a flatten layer in the same neural network architecture as previous models.

3.4.4. ResNetV2

The ResNet architecture is a neural network with 34 layers inspired by VGG-19. Keras provides pre-trained ResNet V1 and V2 models with 50, 101, or 152 layers. ResNet V2 with 152 layers was used in the system, achieving the best training accuracy at epoch 10. ResNet152V2 was also used in the research, and the extracted features were implemented on a CNN. ResNetV2 operates batch normalization before each weight layer, and utilizing ResNet has greatly improved neural network implementation with more layers [39]. The number of features extracted from the pre-trained model is displayed in Table 3.

Table 3. The features number.

Name of the Methods	Features
VGG19 + CNN	8192
Inception + CNN	2880
Xception + CNN	38,400
ResNetV2 + CNN	12,800
MobileNet + CNN	5120
DenseNet201 + CNN	2048
MobileNetV2 + CNN	5120
VGG16 + CNN	41,472

3.4.5. MobileNet and MobileNetV2

MobileNet is a neural network that uses depthwise separable convolutions except for the first layer, which is a full convolution. It has 28 layers, with batch normalization and ReLU activation after each convolutional layer. Downsampling is achieved through stride convolution, and a global mean pooling layer is used to reduce the spatial resolution to 1 before the fully connected layer. MobileNetV2 is similar to the original MobileNet but employs inverted residual blocks with bottleneck features and has fewer parameters [39]. MobileNet can support any sample input dimension greater than 32 $\times$ 32, and larger image sizes can result in better results. MobileNetV2 extracted 5120 features, with the input shape for the single flattened layer being (4, 4, 320), and the number of trainable parameters was 256,356. Table 4 shows the number of trainable parameters of all the models.

Table 4. The number of trainable parameters.

Name of the Methods	Trainable Params
Simple CNN	1,204,038
VGG19 + CNN	409,956
InceptionV3 + CNN	144,356
Xception + CNN	1,920,356
ResNetV2 + CNN	640,356
MobileNet + CNN	819,556
DenseNet201 + CNN	102,756
MobileNetV2 + CNN	256,356
VGG16 + CNN	2,073,956

3.4.6. DenseNet201

DenseNet is a deep learning architecture designed to address the issue of decreased accuracy in deep neural networks [39]. It uses efficient connections between layers to make networks deeper and easier to train. DenseNet201 was used for feature extraction in a CNN with an input layer shape (4, 4, 128), achieving high accuracy at epoch 26. DenseNet-121 is a model in the DenseNet family specifically designed for image classification. DenseNet uses dense blocks to connect all layers directly with each other and has been shown to have better feature representation and computation efficiency than ResNet with fewer parameters. However, DenseNet requires high-quality GPU memory for concatenation operations, which can be reduced through memory-efficient implementation.

3.5. Local Interpretable Model-Agnostic Explanations (LIME)

XAI is an area of AI that focuses on creating AI systems that can provide human-understandable explanations for their predictions or decisions [40]. One common approach for making AI models more explainable is LIME. This is a process used to explain the predictions made by machine learning models, including image classification models [41]. LIME achieves this by temporarily replacing the complex model with a simpler, more interpretable model for a particular instance. This approximation is accomplished by assigning varying weights to the instance's features based on their significance to the prediction. The weights are determined through an optimization process that maximizes the correspondence between the complex and approximating models.

The equation of LIME is given by [41]:

$$L(x, f, \pi) = \sum_{i=1}^{d} \pi_i(x) f_i(x) + C|\pi|_1 \quad (2)$$

where x is the instance being explained, f is the approximating model, π is the feature importance weights, d is the number of features, and C is a regularization parameter. The first term in the equation is the weighted sum of the features, and the second term is the regularization term that encourages sparsity in the feature weights. The optimization problem is to find the weights π that minimize the difference between the predictions of the black-box model and the approximating model for the instance x.

LIME is a technique that generates a simple and interpretable model for a specific image to explain the predictions of a complex image classification model. This is achieved by assigning different weights to the pixels based on their relevance to the prediction and creating an approximation through an optimization process. The most significant features can then be identified and visualized through a heatmap, with the most important pixels highlighted as the explanation for the image classification.

4. Results and Discussion

In DL image classification, performance evaluation is carried out by monitoring training loss and accuracy. Training loss measures the error made by the model while predicting training data and is minimized by adjusting the model's parameters. Training accuracy represents the proportion of exact predictions the model makes on the training data. It helps detect overfitting, where the model memorizes the training data but cannot derive new data.

These deep learning models were evaluated for achieving 100% training accuracy on a given dataset as Table 5. The results revealed that the number of epochs required to reach this level of accuracy varied across models. Specifically, the simple CNN model took 20 epochs, while VGG19 + CNN, InceptionV3 + CNN, Xception + CNN, and ResNetV2 + CNN required significantly fewer epochs, taking only 11, 8, 10, and 10 epochs, respectively. The DenseNet201 + CNN model took 24 epochs, while the MobileNet + CNN and MobilnetV2 + CNN models required 39 epochs to reach 100% accuracy. The VGG16 + CNN model required 8 epochs to achieve perfect training accuracy. Figure 9 presents the training accuracy and training loss against the number of epochs for all of the considered models. The figure shows that the models achieve the best fit at lower epochs, which suggests a better balance of model complexity and fitting and might be less prone to overfitting than in the other models. However, further testing and evaluation of validation and test data are necessary to confirm this conclusion.

The confusion matrix is a commonly used metric in image classification that allows us to evaluate the performance of a model by distinguishing between correctly classified and misclassified images belonging to different classes [36]. This metric is useful for calculating overall accuracy, identifying class imbalances and obtaining a detailed breakdown of a model's performance for individual classes. In Figure 10, this research presents the confusion matrices of the considered models for all the classes. The results indicate that the VGG19 + CNN model outperforms other models in classifying images. Specifically, the VGG19 + CNN model correctly classifies all samples of the fresh leaf and gray mold classes (27 and 15 samples), while some misclassifications may occur due to the similarity in patterns among other classes.

Precision, recall, *F*1-score, and accuracy are commonly used performance metrics in evaluating image classification models. These metrics provide a quantitative assessment of the models for accurately classifying images. The standard mathematical is used to evaluate these metrics as follows: true positive (*TP*), false positive (*FP*), true negative (*TN*), and false negative (*FN*).

$$precision = \frac{TP}{TP + FP} \tag{3}$$

$$recall = \frac{TP}{TP + FN} \tag{4}$$

$$F1\text{-}Score = 2 \times \frac{precision \times recall}{precision + recall} \tag{5}$$

$$accuracy = \frac{TN + TP}{TP + FP + FN + TN} \tag{6}$$

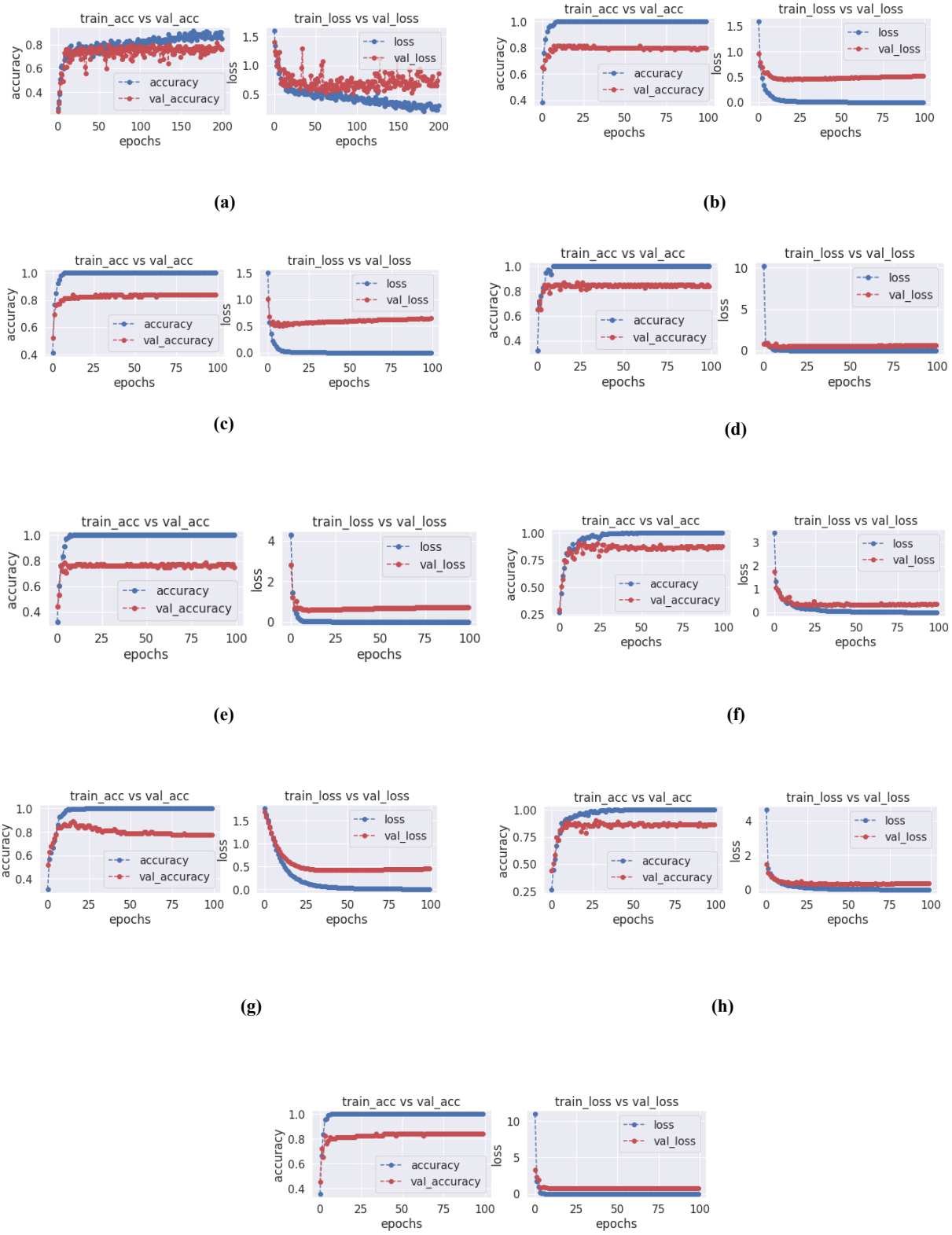

Figure 9. Training accuracy and training loss: (**a**) Simple CNN, (**b**) VGG19 + CNN, (**c**) InceptionV3 + CNN, (**d**) Xception + CNN, (**e**) ResNetV2 + CNN, (**f**) MobileNet + CNN, (**g**) DenseNet201 + CNN, (**h**) MobileNetV2 + CNN, and (**i**) VGG16 + CNN.

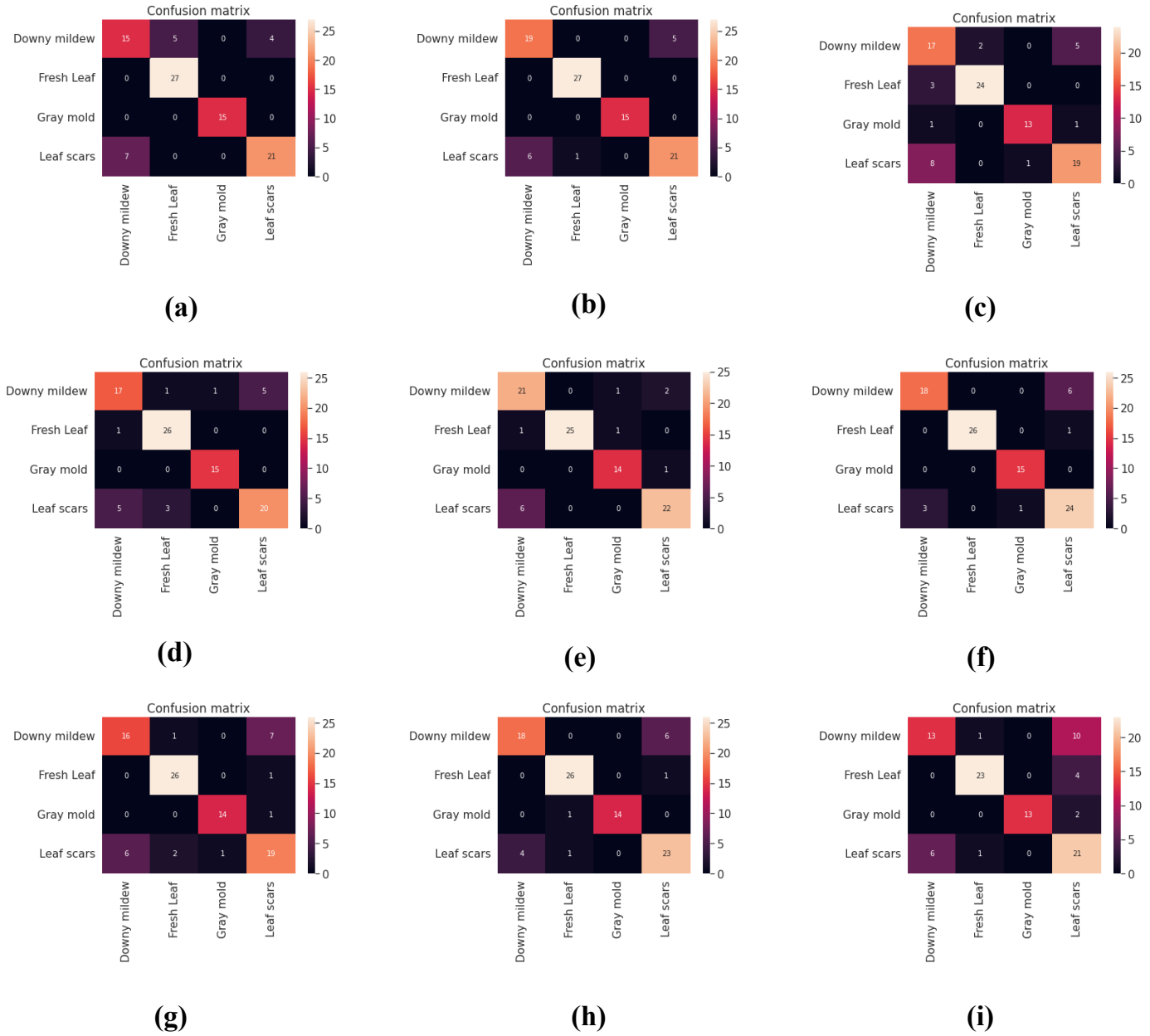

Figure 10. Confusion matrix: (**a**) Simple CNN, (**b**) VGG19 + CNN, (**c**) InceptionV3 + CNN, (**d**) Xception + CNN, (**e**) ResNet v2 + CNN, (**f**) MobileNet + CNN, (**g**) DenseNet201 + CNN, (**h**) MobileNetV2 + CNN, and (**i**) VGG16 + CNN.

We show the mean classification precision, recall, F1-Score, and accuracy of all of the considered models for four classes in Table 6. It reveals that the VGG19 + CNN model achieves better precision, recall, F1-Score, and accuracy than any of the other models, and the values are 93%, 93%, 93%, and 93%, respectively. Here, an F1-Score of 93% suggests a good balance between precision and recall. Moreover, the model can correctly predict classes in 93% of cases. Moreover, the InceptionV3 + CNN model exhibits poor results among the models, as well as average values of precision, recall, F1-Score, and accuracy are 79%, 78%, 78%, and 78%, respectively.

Table 5. The number of epochs.

Name of the Methods	Epochs
Simple CNN	20
VGG19 + CNN	11
InceptionV3 + CNN	8
Xception + CNN	10
ResNetV2 + CNN	10
MobileNet + CNN	39
DenseNet201 + CNN	24
MobilnetV2 + CNN	39
VGG16 + CNN	8

Table 6. Average classification precision, recall, F1-Score, and accuracy.

Name of the Model	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)
Simple CNN	81	80	79	78
VGG16 + CNN	81	79	80	78
InceptionV3 + CNN	79	78	78	78
Xception + CNN	86	86	86	80
ResNetV2 + CNN	88	87	87	84
MobileNet + CNN	87	84	85	84
MobileNetV2 + CNN	86	84	85	85
DenseNet201 + CNN	80	81	81	87
VGG19 + CNN	93	93	93	93

Hamming loss (*HL*) [42], Matthews correlation coefficient (*MC*) [43], Jaccard score (*JS*) [44], and Cohen's Kappa (*CK*) [45] are also some of the prominent evaluation metrics used for evaluating models in multiclass image classification problems. Hamming loss measures the average number of incorrect class predictions a classifier makes, whereas MC considers a classifier's number of *TP*, *FP*, *TN*, and *FN* predictions. The Jaccard score measures the similarity between the predicted class labels and the actual class labels. It is defined as the size of the intersection divided by the size of the union of the expected and actual class labels. Moreover, Cohen's kappa assesses the consistency between the predicted and actual class labels while considering the agreement that may occur by chance. These metrics can be defined as follows:

$$HL = \frac{1}{n}\sum_{i=1}^{n} \mathbb{I}[y_i \neq \hat{y}_i], \tag{7}$$

where n is the total number of samples, y_i is the true label for the ith sample, and $\hat{y}_i$ is the predicted label for the ith sample.

$$MC = \frac{(TN \times TP - FN \times FP)}{\sqrt{(FP + TP) \times (FN + TP) \times (FP + TN) \times (FN + TN)}}. \tag{8}$$

$$JS = \frac{|A \cap B|}{|A \cup B|} \tag{9}$$

where A and B are the sets of predicted and actual class labels, respectively, and $|S|$ represents the number of elements in set S.

$$CK = \left(\frac{P_o - P_e}{1 - P_e}\right), \tag{10}$$

where P_o is the agreement between the predicted and actual class labels and P_e is the agreement that is expected to happen.

We present the value of Hamming loss, Matthews correlation coefficient, Jaccard score, and Cohen's kappa for different models in Table 7. The table reveals that the VGG19 + CNN model provides the lowest HL value but the highest MCC, JS, and κ values; the values are 7%, 90%, 86%, and 90%, respectively. The HL value suggests that this model incorrectly predicts 7% of the cases, which is similar to the accuracy of the model as mentioned above. The MCC of 90% indicates strong agreement between the predicted and the actual labels. Moreover, the JS and κ values also suggest a similar conclusion between the prediction and the actual value, which indicates the consistency of the model. The HL of VGG19 + CNN is 6% lower than ResNetV2 + CNN, which is the second lowest and 15% lower than VGG16 + CNN and InceptionV3 + CNN, which are the worst. In the case of MCC, VGG19 + CNN shows 7% better than ResNetV2 + CNN, which is the second highest and 20% better than VGG16 + CNN and InceptionV3 + CNN, which are the lowest. Moreover, VGG19 + CNN gives 9% and 7% higher values of JS and κ, respectively, than that of ResNetV2 + CNN, which is the second highest in both the metrics. Moreover, these values are 23% and 20% better for VGG19 + CNN than those of both VGG16 + CNN and InceptionV3 + CNN, which exhibit the lowest values. In this study, several performance metrics were used to evaluate the performance of various deep learning models for recognizing diseases in sunflower plants. These metrics included precision, recall, F1-score, accuracy, Hamming loss, Matthews coefficient, Jaccard score, and Cohen's kappa. The results showed that the VGG19 + CNN model outperformed all of the other models in terms of these metrics. Specifically, the VGG19 + CNN model showed higher precision, recall, F1-score, accuracy, and Cohen's kappa than the other models.

Table 7. Value of Hamming loss, Matthews Correlation coefficient, Jaccard score, and Cohen's Kappa.

Name of the Methods	HL (%)	MCC (%)	JS (%)	CK (%)
Simple CNN	21	72	65	71
VGG16 + CNN	22	70	63	70
InceptionV3 + CNN	22	70	63	70
Xception + CNN	15	80	74	80
ResNetV2 + CNN	13	83	77	83
MobileNet + CNN	16	77	72	78
MobileNetV2 + CNN	16	77	72	78
DenseNet201 + CNN	20	73	66	73
VGG19 + CNN	7	90	86	90

Moreover, the VGG19 + CNN model was found to have a lower Hamming loss and a higher Matthews coefficient and Jaccard score than the other models. This indicates that the VGG19 + CNN model was better at predicting the presence or absence of multiple diseases at the same image and had higher agreement with the ground truth labels.

Overall, these results suggest that the VGG19 + CNN model is a promising approach for recognizing diseases in sunflower plants. However, it is important to note that further studies are needed to validate these findings on larger datasets and under different conditions.

4.1. Comparison with the Existing Works

We show a comparison with [20] considering the same dataset [16] with the same classes in Table 8. The table indicates that the presented VGG19 + CNN model achieves better results in terms of precision, recall, and accuracy metrics than all the models of [20]. It reveals that the proposed model achieves 22.09%, 45.06%, 14.32%, 40.35%, and 35.85% better precision than LR, NB, RF, J48, and K-star, respectively. In the case of recall and accuracy, the proposed model attains 24.82%, 48.50%, 16.07%, 42.39%, and 35.19%, as well as 5.65%, 14.77%, 2.15%, 13.17%, and 11.55%, respectively, compared to LR, NB, RF, J48, and K-star. Moreover, the proposed approach obtains better accuracy than the other works [1,2] in the domain, which also utilize DL algorithms on a different small sunflower dataset. In this scenario, the proposed approach achieves 3.80% better accuracy than both of the mentioned works.

Recent studies have shown that DL models outperform traditional ML techniques in various image recognition tasks. For example, a study conducted by researchers at Google found that a DL model outperformed traditional ML models on a large-scale image recognition task, achieving a top-5 error rate of 6.70%, compared to 26.20% for traditional ML models [46]. Another study conducted by researchers at Microsoft found that a DL model achieved state-of-the-art performance on a large-scale image recognition dataset, achieving a top-1 error rate of 3.57% [47]. DL has shown superior performance in image recognition tasks due to its ability to learn hierarchical features and handle a large number of data. These advantages have been demonstrated in various studies, and the continued advancements in DL are expected to drive further improvements in image recognition accuracy. TL has shown superior performance in terms of image recognition compared to traditional ML techniques due to its ability to efficiently use pre-trained models, utilize complex deep neural network architectures, and help with overfitting. TL enables us to fine-tune pre-trained models on smaller, task-specific datasets, reducing the time and resources required for training a new model from scratch. Additionally, pre-trained models provide a starting point for the model that has already learned to generalize well on a large and diverse dataset. A study conducted by researchers at Google found that TL using pre-trained CNNs was an effective way to improve the accuracy of DL models for image recognition tasks, particularly for datasets with limited training data [48]. The results of this research, which combines VGG19 and CNN, are better than existing methods of [20] even with a small dataset. Specifically, the proposed approach improves precision, recall, and accuracy by 14.32%, 16.07%, and 2.15%, respectively, compared to random forest and perfomrs far better than other approaches of [20]. The combination of VGG19 and CNN can create a deep neural network architecture that is well-suited for image recognition tasks. VGG19 is a DL model that is pre-trained on a large dataset and has learned to extract essential features from raw image data. Using VGG19 for feature extraction enables the model to identify important patterns and features that might not be easily detected by traditional machine-learning techniques. The CNN used in the hybrid approach is capable of learning multiple levels of abstract representations from the extracted features, which can help capture more nuanced and complex relationships in the data. This can lead to higher accuracy in image recognition tasks. TL techniques used in this hybrid approach can help reduce the amount of time and the number of resources required for training a new model from scratch, and they can also improve the generalization ability of the model.

Table 8. Comparison with the previous method.

Compared Models	Precision (%)	Recall (%)	Accuracy (%)
Logistic regression (LR) [20]	70.91	68.18	87.35
Naive Bayes (NB) [20]	47.94	44.50	78.23
Random Forest (RF) [20]	78.68	76.93	90.85
J48 [20]	52.65	50.61	79.83
K-star [20]	57.15	57.81	81.45
Proposed Model (VGG19 + CNN)	93.00	93.00	93.00

4.2. Explainability with AI

LIME works by generating a simple, interpretable explanation for a prediction by training a linear model on perturbed versions of the input data. The idea is to generate a large number of perturbed samples that are similar to the original input and then to train a simple model on these perturbed samples. The coefficients of this simple model can then be used to determine which features are most impactful for the prediction.

The images shown in Figure 11 have been generated by LIME. The display only includes the super-pixels in the correct region section, with the contour of the superpixel highlighted and the background included as well. In the correlation section, the image displayed areas of super-pixels colored in green, indicating an increase in the probability that the image belongs to a specified class. The top five positive features have been highlighted in green in the correlation section [49]. Meanwhile, the super-pixels colored in red signify a decrease in the probability. The heatmap image produced by LIME is created by overlaying a transparent mask on the original image, with the transparency of the mask representing the significance of each feature. The areas with the highest transparency indicate the features with the greatest impact. To generate the heatmap, LIME selects a local region around the prediction and trains a simple linear model on that region [49]. The simple model approximates the complex machine learning model, and the weights of the linear model signify the relative importance of each feature. The heatmap is generated by applying the transparent mask on the image, with the transparency of the mask indicating the magnitude of the weights from the linear model. The explanation module of the LIME package has been adopted for the justification [40].

The confusion matrix of VGG19+CNN (Figure 10b) showed that the leaf scars samples had the highest number of misclassifications, with many of them being classified as downy mildew. To understand this misclassification better, we adopted LIME to generate a set of perturbations of model behavior. The results showed that the misclassified sample had some features that were similar to the correctly classified downy mildew sample (Figure 11). Still, the significant region was different from leaf scars (Figure 12). The heatmap showed that the color warmth was not the same for both samples in Figure 12. Therefore, the model may have misinterpreted the leaf scars because its significant region was not similar to that of the correctly classified downy mildew. In Figure 12, there is also a misclassified sample, downy mildew, which has been misclassified as leaf scars. When we compared Figure 11 to the correctly classified downy mildew, we found that the vital area was not the same.

Overall, VGG19 + CNN has strong performance with high precision, recall, F1-Score, and accuracy, as well as strong agreement between the predicted and actual labels, as shown by MCC, JS, and κ. The misclassified images of the model show consistent patterns or trends in the features that caused inaccurate predictions. Therefore, the model needs to be adjusted and the quality of the training data needs to be enhanced to avoid making similar errors in the future.

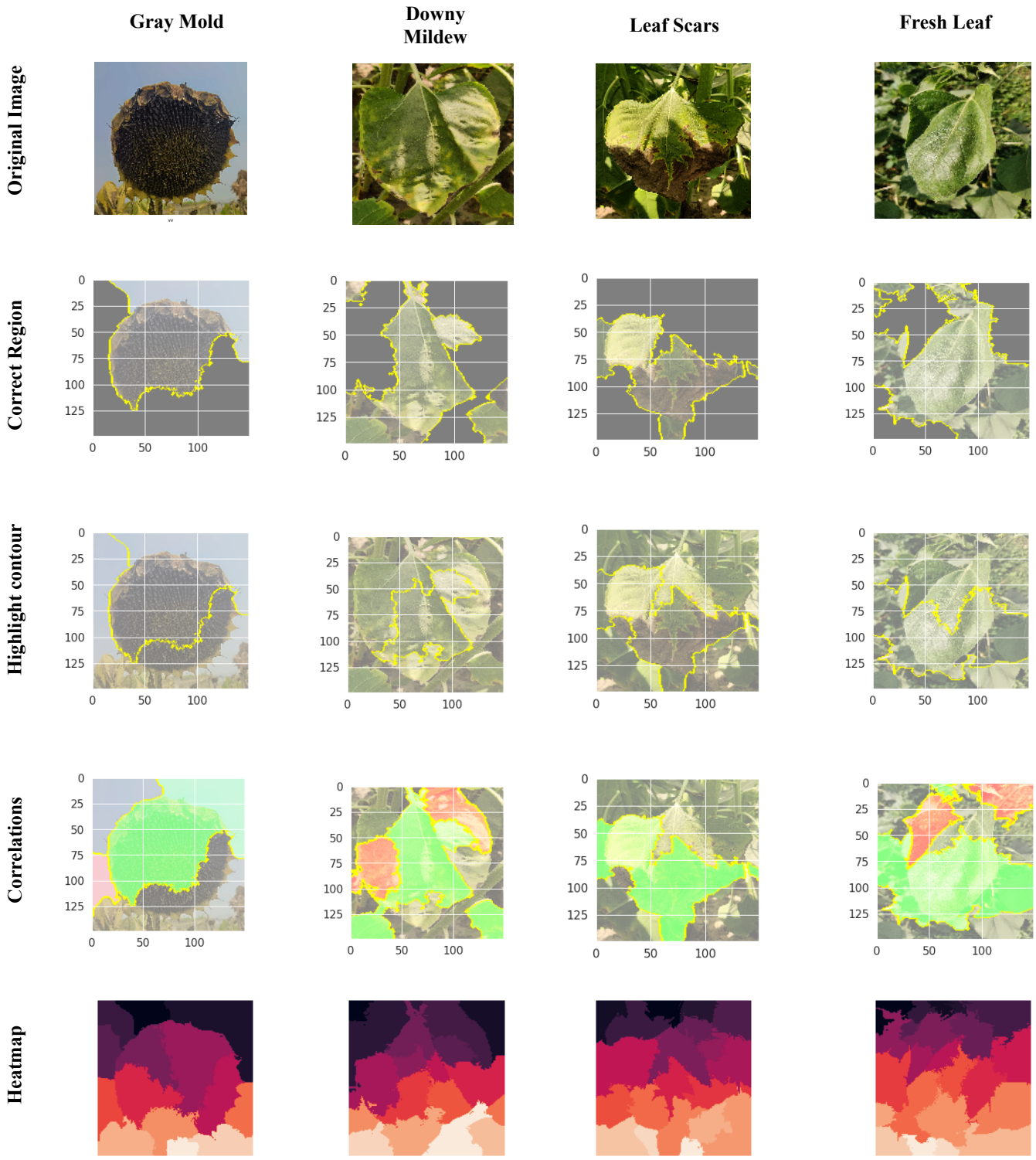

Figure 11. Lime analysis.

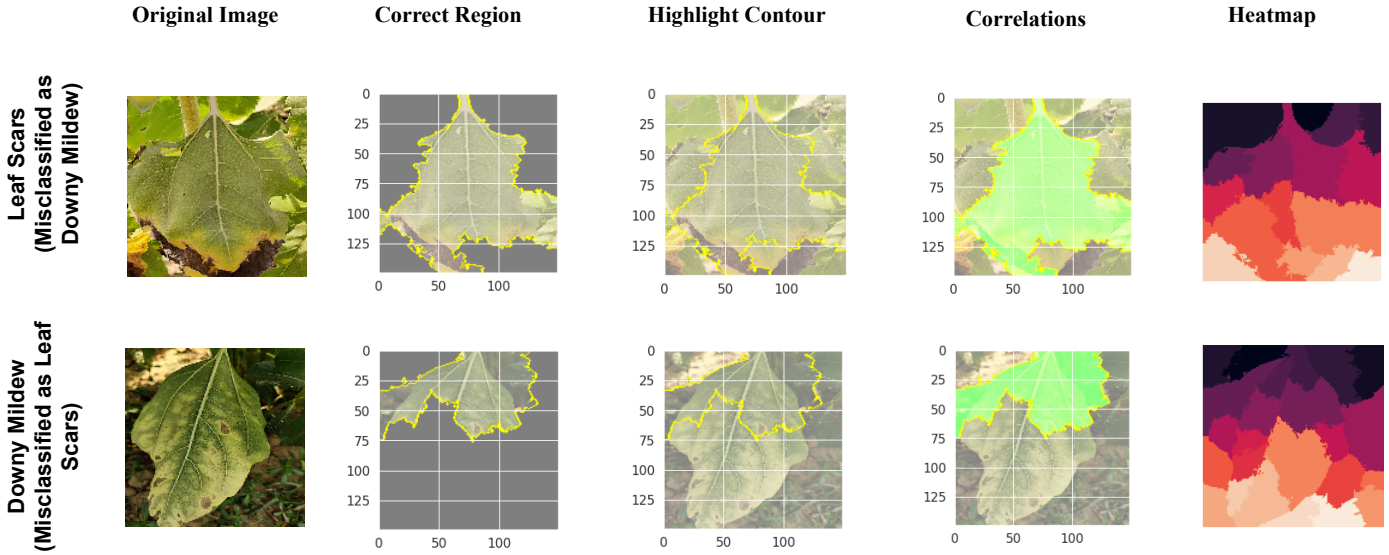

Figure 12. Misclassification analysis by LIME.

5. Conclusions

Sunflowers, with their leaves and grains, have a range of benefits and are used in various fields. Thus, it is crucial to have a plentiful supply of sunflowers through proper production. This can be ensured by detecting any disorder in sunflower plants early, and modern computing and artificial intelligence can play a role in this regard. While the VGG19 + CNN model demonstrated promising results in recognizing diseases in sunflowers along with fresh leaves, it is important to note that the model's performance may be limited by its inability to differentiate between certain diseases. For instance, the model showed no misclassifications in identifying gray mold and fresh leaves, but its accuracy may have been lower when distinguishing between more closely related diseases. Additionally, the dataset used in this research may not fully represent the diverse range of disease presentations that can occur in sunflowers, which could further limit the model's generalizability to real-world scenarios. Our proposed hybrid deep-learning model can successfully classify three sunflower diseases and fresh leaves with an accuracy of 93% on a small dataset. The results show that the proposed model provides better precision, recall, and accuracy than [4] on the same dataset. The proposed approach achieves at least 2.15% better classification accuracy than [4]. In the future, we will try to enhance the performance metrics using a more carefully designed model with appropriate data preprocessing techniques.

Author Contributions: Conceptualization, P.G. and A.K.B.; methodology, P.G., A.K.M., and A.K.B.; software, P.G. and S.C.; validation, A.K.M. and A.K.B.; formal analysis, P.G. and A.K.B.; investigation, P.G. and A.K.B.; resources, A.K.M. and S.C.; data curation, P.G.; writing—original draft preparation, P.G. and A.K.B.; writing—review and editing, P.G., M.M., H.M. and A.K.B.; visualization, P.G.; supervision, A.K.B.; project administration, M.M. and A.K.B.; and funding acquisition, M.M. and H.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Deanship of Scientific Research, Taif University.

Data Availability Statement: The data used in this paper are published in brief and available at https://doi.org/10.1016/j.dib.2022.108043 (accessed on 15 April 2023).

Acknowledgments: The researchers would like to thank the Deanship of Scientific Research, Taif University, for funding this project.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Malik, A.; Gayatri, V.; Vishal, J.; Sathyapriya, E.; Akash, S.; Isha, B.; Manik, R.; Evans, A. Design and Evaluation of a Hybrid Technique for Detecting Sunflower Leaf Disease Using deep learning Approach. *J. Food Qual.* **2022**, *2022*, 9211700. [CrossRef]
2. Sirohi, A.; Malik, A. A hybrid model for the classification of Sunflower Diseases using deep learning. In Proceedings of the 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM), London, UK, 28–30 April 2021. Available online: https://ieeexplore.ieee.org/document/9445342 (accessed on 15 April 2023).
3. Sasaki, Y.; Okamoto, T.; Imou, K.; Torii, T. Automatic Diagnosis of Plant Disease—Spectral Reflectance of Healthy and Diseased Leaves. *IFAC Proc. Vol.* **1998**, *31*, 145–150. [CrossRef]
4. Haber, S. Diagnosis of Flame Chlorosis by Reverse Transcription-Polymerase Chain Reaction (RT-PCR). *Plant Dis.* **1995**, *79*, 626. [CrossRef]
5. Koo, C.; Malapi-Wight, M.; Kim, H.S.; Cifci, O.S.; Vaughn-Diaz, V.L.; Ma, B.; Kim, S.; Abdel-Raziq, H.; Ong, K.; Jo, Y.K.; et al. Development of a Real-Time Microchip PCR System for Portable Plant Disease Diagnosis. *PLoS ONE* **2013**, *8*, e82704. [CrossRef] [PubMed]
6. Thorat, A.; Kumari, S.; Valakunde, N.D. An IOT Based Smart Solution for Leaf Disease Detection. In Proceedings of the 2017 International Conference on Big Data, IoT and Data Science (BID), Pune, India, 20–22 December 2017. [CrossRef]
7. Masud, M.; Sikder, N.; Nahid, A.A.; Bairagi, A.K.; AlZain, M.A. A machine learning Approach to Diagnosing Lung and Colon Cancer Using a deep learning-Based Classification Framework. *Sensors* **2021**, *21*, 748. [CrossRef]
8. Varshney, D.; Babukhanwala, B.; Khan, J.; Saxena, D.; Singh, A.K. Plant Disease Detection Using machine learning Techniques. In Proceedings of the 2022 3rd International Conference for Emerging Technology (INCET), Karnataka, India, 27–29 May 2022.
9. Patel, K.; Patel, A. Plant Disease Diagnosis Using Image Processing Techniques—A Review on Machine and deep learning Approaches. *Ecol. Environ. Conserv.* **2022**, *28*, 351–362. [CrossRef]
10. Schmidhuber, J. Deep learning in Neural Networks: An Overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef]
11. Khan, M.A.M.; Kee, S.H.; Sikder, N.; Al Mamun, M.A.; Zohora, F.T.; Hasan, M.T.; Bairagi, A.K.; Nahid, A.A. A Vision-Based Lane Detection Approach for Autonomous Vehicles Using a convolutional neural network Architecture. In Proceedings of the 2021 Joint 10th International Conference on Informatics, Electronics & Vision (ICIEV) and 2021 5th International Conference on Imaging, Vision & Pattern Recognition (IcIVPR), Kitakyushu, Japan, 16–20 August 2021.
12. Sikder, N.; Ahad, M.A.R.; Nahid, A.A. Human Action Recognition Based on a Sequential deep learning Model. In Proceedings of the 2021 Joint 10th International Conference on Informatics, Electronics & Vision (ICIEV) and 2021 5th International Conference on Imaging, Vision & Pattern Recognition (IcIVPR), Kitakyushu, Japan, 16–20 August 2021.
13. Van Klompenburg, T.; Kassahun, A.; Catal, C. Crop Yield Prediction Using machine learning: A Systematic Literature Review. *Comput. Electron. Agric.* **2020**, *177*, 105709. [CrossRef]
14. Litvak, M.; Divekar, S.; Rabaev, I. Urban Plants Classification Using Deep-Learning Methodology: A Case Study on a New Dataset. *Signals* **2022**, *3*, 524–534. [CrossRef]
15. Khatun, M.; Hossain, T.M.; Miah, M.M.; Khandoker, S.; Rashid, M.A. Profitability of Sunflower Cultivation in Some Selected Sites of Bangladesh. *Bangladesh J. Agric. Res.* **2016**, *41*, 599–623. [CrossRef]
16. Sara, U.; Rajbongshi, A.; Shakil, R.; Akter, B.; Sazzad, S.; Uddin, M.S. An Extensive Sunflower Dataset Representation for Successful Identification and Classification of Sunflower Diseases. *Data Brief* **2022**, *42*, 108043. [CrossRef] [PubMed]
17. Jiang, B.; He, J.; Yang, S.; Fu, H.; Li, T.; Song, H.; He, D. Fusion of Machine Vision Technology and Alexnet-Cnns deep learning Network for the Detection of Postharvest Apple Pesticide Residues. *Artif. Intell. Agric.* **2019**, *1*, 1–8. [CrossRef]
18. Kinger, S.; Kulkarni, V. Explainable AI for deep learning Based Disease Detection. In Proceedings of the 2021 Thirteenth International Conference on Contemporary Computing (IC3-2021), Noida, India, 5–7 August 2021.
19. Vishwarupe, V.; Joshi, P.M.; Mathias, N.; Maheshwari, S.; Mhaisalkar, S.; Pawar, V. Explainable AI and Interpretable machine learning: A Case Study in Perspective. *Procedia Comput. Sci.* **2022**, *204*, 869–876. [CrossRef]
20. Rajbongshi, A.; Biswas, A.A.; Biswas, J.; Shakil, R.; Akter, B.; Barman, M.R. Sunflower Diseases Recognition Using computer vision-Based Approach. In Proceedings of the 2021 IEEE 9th Region 10 Humanitarian Technology Conference (R10-HTC), Bangalore, India, 30 September–2 October 2021.
21. Rashid, J.; Khan, I.; Ali, G.; ur Rehman, S.; Alturise, F.; Alkhalifah, T. Real-Time Multiple Guava Leaf Disease Detection from a Single Leaf Using Hybrid deep learning Technique. *Comput. Mater. Contin.* **2023**, *74*, 1235–1257. [CrossRef]
22. Zhong, Y.; Zhao, M. Research on deep learning in Apple Leaf Disease Recognition. *Comput. Electron. Agric.* **2020**, *168*, 105146. [CrossRef]
23. Habib, M.T.; Mia, M.J.; Uddin, M.S.; Ahmed, F. An in-Depth Exploration of Automated Jackfruit Disease Recognition. *J. King Saud-Univ.—Comput. Inf. Sci.* **2022**, *34*, 1200–1209. [CrossRef]
24. Habib, M.T.; Majumder, A.; Jakaria, A.Z.M.; Akter, M.; Uddin, M.S.; Ahmed, F. Machine Vision Based Papaya Disease Recognition. *J. King Saud-Univ.—Comput. Inf. Sci.* **2020**, *32*, 300–309. [CrossRef]
25. Geetharamani, G.; Pandian, A. Identification of Plant Leaf Diseases Using a Nine-Layer Deep convolutional neural network. *Comput. Electr. Eng.* **2019**, *76*, 323–338. [CrossRef]
26. Singh, V. Sunflower Leaf Diseases Detection Using Image Segmentation Based on particle swarm optimization. *Artif. Intell. Agric.* **2019**, *3*, 62–68. [CrossRef]

27. Agarwal, M.; Kaliyar, R.K.; Singal, G.; Gupta, S.K. FCNN-LDA: A Faster Convolution Neural Network Model for Leaf Disease Identification on Apple's Leaf Dataset. In Proceedings of the 2019 12th International Conference on Information & Communication Technology and System (ICTS), Surabaya, Indonesia, 18 July 2019.
28. Huang, S.; Liu, W.; Qi, F.; Yang, K. Development and Validation of a deep learning Algorithm for the Recognition of Plant Disease. In Proceedings of the 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Zhangjiajie, China, 10–12 August 2019.
29. De Luna, R.G.; Dadios, E.P.; Bandala, A.A. Automated Image Capturing System for deep learning-Based Tomato Plant Leaf Disease Detection and Recognition. In Proceedings of the TENCON 2018—2018 IEEE Region 10 Conference, Jeju, Republic of Korea, 28–31 October 2019.
30. Kaur, P.; Harnal, S.; Tiwari, R.; Upadhyay, S.; Bhatia, S.; Mashat, A.; Alabdali, A.M. Recognition of Leaf Disease Using Hybrid convolutional neural network by Applying Feature Reduction. *Sensors* **2022**, *22*, 575. [CrossRef]
31. Mishra, A.M.; Harnal, S.; Gautam, V.; Tiwari, R.; Upadhyay, S. Weed Density Estimation in Soya Bean Crop Using Deep convolutional neural networks in Smart Agriculture. *J. Plant Dis. Prot.* **2022**, *129*, 593–604. [CrossRef]
32. Nagaraju, M.; Chawla, P.; Upadhyay, S.; Tiwari, R. Convolution Network Model Based Leaf Disease Detection Using Augmentation Techniques. *Expert Syst.* **2021**, *39*, e12885. [CrossRef]
33. Muni Mishra, A.; Harnal, S.; Mohiuddin, K.; Gautam, V.; Nasr, O.A.; Goyal, N.; Alwetaishi, M.; Singh, A. A deep learning-Based Novel Approach for Weed Growth Estimation. *Intell. Autom. Soft Comput.* **2022**, *31*, 1157–1173. [CrossRef]
34. Rajpurkar, P.; Irvin, J.; Zhu, K.; Yang, B.; Mehta, H.; Duan, T.; Ding, D.; Bagul, A.; Langlotz, C.; Shpanskaya, K.; et al. CheXNet: Radiologist-Level Pneumonia Detection on Chest X-rays with deep learning. *arXiv* **2017**, arXiv:1711.05225.
35. Bradski, G.; Kaehler, A. *Learning Opencv: Computer Vision with the Opencv Library*; O'Reilly: Beijing, China, 2012.
36. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2017.
37. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
38. Chollet, F.; Zhu, Q.S.; Rahman, F.; Gardener, T.; Lee, T.; Qian, C.; de Marmiesse, G.; Jin, H.; Zabluda, O.; Watson, M.; et al. "Keras." Github. 2015. Available online: https://github.com/fchollet/keras (accessed on 10 April 2023).
39. Ribeiro, M.T.; Singh, S.; Guestrin, C. "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, San Francisco, CA, USA, 13–17 August 2016. [CrossRef]
40. Ribeiro, M.T. Local Interpretable Model-Agnostic Explanations (LIME)—Lime 0.1 Documentation. Read the Docs. Available online: https://lime-ml.readthedocs.io/ (accessed on 13 March 2023).
41. Hastie, T.; Friedman, J.; Tisbshirani, R. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*; Springer: New York, NY, USA, 2017.
42. Tsoumakas, G.; Katakis, I. Multi-Label Classification. In *Database Technologies*; IGI Global: Hershey, PA, USA , 2009; pp. 309–319. [CrossRef]
43. Baldi, P.; Brunak, S.; Chauvin, Y.; Andersen, C.A.; Nielsen, H. Assessing the accuracy of Prediction Algorithms for Classification: An Overview. *Bioinformatics* **2000**, *16*, 412–424. [CrossRef]
44. Chung, N.C.; Miasojedow, B.; Startek, M.; Gambin, A. Jaccard/Tanimoto Similarity Test and Estimation Methods for Biological Presence-Absence Data. *BMC Bioinform.* **2019**, *20*, 644. [CrossRef]
45. Cohen, J. A Coefficient of Agreement for Nominal Scales. *Educ. Psychol. Meas.* **1960**, *20*, 37–46. [CrossRef]
46. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; IEEE: Piscataway, NJ, USA, 2015.
47. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.
48. Oquab, M.; Bottou, L.; Laptev, I. Learning and Transferring Mid-Level Image Representations Using Convolutional Neural Networks. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014.
49. Garreau, D.; Mardaoui, D. What does LIME really see in images? *arXiv* **2021**, arXiv:2102.06307.