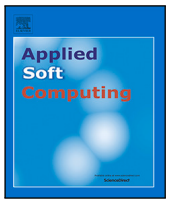
Applied Soft Computing Journal 177 (2025) 113152

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc



A generative framework for detection and classification of plant leaf disease using diffusion network

Aryan Das [a], Rajul Mahto [a], Wentao Wang [b], Ali Jamali [c], Pravendra Singh [d], Swalpa Kumar Roy [e,*]

[a] School of Computer Science Engineering, Vellore Institute of Technology, Bhopal, Madhya Pradesh 466114, India
[b] Department of Computer Science, University of Alabama at Birmingham, 10th Ave S, 35294, AL, USA
[c] Department of Geography, Simon Fraser University, 8888 University Dr, Burnaby, BC V5A 1S6, Canada
[d] Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Uttarakhand 247667, India
[e] Department of Computer Science and Engineering, Alipurduar Government Engineering and Management College, West Bengal 736206, India

ARTICLE INFO

Keywords:
Plant disease mapping
Diffusion models
Deep learning
CNN
U-net

ABSTRACT

Agriculture is a crucial driver of global economic growth, significantly contributing to employment, Gross Domestic Product (GDP), and the provision of essential food and raw materials. However, crop diseases, which often go unnoticed and untreated, cause substantial yearly reductions in farm productivity. While effective on small-scale farms, traditional manual methods for disease detection are impractical for larger operations due to their inefficiency, subjectivity, and vulnerability to human error. In contrast, deep learning technologies offer a promising solution by automating the accurate recognition and classification of crop diseases, thus enhancing efficiency and accuracy without requiring highly specialized expertise. This work presents a novel approach by introducing a diffusion model for classifying plant leaf diseases. We propose a novel diffusion-based deep learning technique (LeafDisDiff), which employs a U-Net-style architecture with position embeddings, group normalization, attention blocks, and integrated residual blocks, centring around a denoising approach for each time step. The proposed LeafDisDiff algorithm has shown superior classification accuracy across three crop disease data benchmarks: Plant Village, Bangladeshi Crop Disease, and Apple Disease. For example, LeafDisDiff outperforms several deep learning classifiers such as ViT, Cross ViT, VGG-19, CNN, ResNet-50, and EfficientNet-B3 and improves classification accuracy by approximately 9.2, 6.2, 5.7, 5.5, 2.5, and 0.7 percentage points, respectively, achieving an average accuracy of 99.49% over the Plant Village dataset. This work is motivated by recent advancements in diffusion probabilistic algorithms for generative image modelling and demonstrates the potential of these techniques in agricultural applications. Moreover, the implementation of this model can facilitate early and accurate detection of plant diseases, enabling prompt interventions like targeted pesticide application and selective breeding of disease-resistant crops. This proactive approach reduces yield losses, lowers chemical input reliance, and promotes sustainable farming. Furthermore, its automation lowers labour costs and enhances scalability, benefiting farms globally. The code is publicly available at the Link

1. Introduction

According to data from the Food and Agriculture Organization (FAO) of the United Nations, plant diseases result in an annual global economic loss of $220 billion. Detecting and categorizing plant leaf diseases are critical challenges in agriculture since they directly impact crop growth and yield. Once disease symptoms are identified on leaves, it is essential to classify them to determine appropriate actions, such as applying specific chemicals or adjusting agricultural practices [1]. Traditionally, manual observation and expert experience have been relied upon for the detection and classification of plant leaf diseases [2–5]. Farmers and agricultural professionals use visual observations, searching for symptoms like unusual colouration, spots, patches, depressions, or bumps. This approach demands substantial expertise and experience, as different conditions may exhibit similar symptoms, requiring accurate identification and classification by professionals [6,7]. While traditional methods may be effective in specific situations, they often consume more time and labour and

* Corresponding author.
E-mail addresses: aryan.das2021@vitbhopal.ac.in (A. Das), rajul.mahto2021@vitbhopal.ac.in (R. Mahto), ww9@uab.edu (W. Wang), alij@sfu.ca (A. Jamali), pravendra.singh@cs.iitr.ac.in (P. Singh), swalpa@agemc.ac.in (S.K. Roy).

https://doi.org/10.1016/j.asoc.2025.113152
Received 21 February 2024; Received in revised form 1 August 2024; Accepted 2 April 2025
Available online 12 May 2025

are susceptible to subjectivity and human error. Furthermore, these methods lack the desired accuracy, posing challenges for farmers to timely detect and manage plant leaf diseases, leading to significant production losses. In contrast, deep learning techniques can automate the detection and classification of leaf diseases, enhancing efficiency and accuracy without the need for highly specialized expertise [8,9]. As a result, deep learning-based methods are gaining prominence in the agricultural sector. Therefore, there is a pressing need to develop an effective and reliable automated method capable of efficiently and accurately distinguishing various infections on plant leaves, facilitating timely actions to mitigate crop losses and ensure the stability and sustainability of agricultural production.

Data scarcity poses a significant challenge in deep learning applications, including the classification of plant diseases, where substantial labelled data is crucial for model training. Alzubaidi et al. [10] comprehensively examines strategies such as Transfer Learning, Generative Adversarial Networks (GANs), and Self-Supervised Learning to mitigate these challenges. It explores how these advanced learning approaches can effectively enhance model performance with limited data, offering valuable insights for improving accuracy and robustness in various fields grappling with data scarcity. With the continuous progress of technology, especially in the domains of deep learning and image processing, more efficient techniques for detecting and classifying plant leaf diseases have emerged [11]. These technologies leverage extensive datasets to train classification models, enabling automatic categorization of leaf diseases. Specifically, by inputting digital images into computer systems and employing various algorithms and models to assess features such as leaf colour, texture, and shape, we can determine the presence of disease indicators. For instance, deep learning-based approaches like Convolutional Neural Networks (CNNs) have proven successful in plant disease detection. These advanced technologies not only enhance the efficiency of detection and classification but also empower farmers to promptly implement necessary measures to control disease spread, thereby minimizing production losses.

In recent times, Denoising Diffusion Probabilistic Models (DDPM) [12] have showcased outstanding performance in the domain of image generation and synthesis. The methodology involves progressively improving the quality of an initial image. More specifically, DDPM is a generative model rooted in a Markov chain framework, capturing data distribution by simulating a diffusion process that steers the input data towards a desired distribution. Although several works have applied the diffusion model to tasks such as image segmentation and object detection [13–16], the full potential of DDPM in other tasks such as plant leaf disease detection and classification remains largely untapped.

Our objective is to develop an approach capable of accurately detecting and classifying plant leaf diseases. Timely diagnosis is imperative for implementing effective treatment measures. Drawing inspiration from the success of diffusion probabilistic models in generative image modelling, we propose a novel diffusion-based approach named LeafDisDiff for the identification and classification of plant leaf diseases. The main contributions of our work are as follows: (1) We propose a novel diffusion-based method, termed LeafDisDiff, for stable and precise detection and classification of plant leaf diseases, which can effectively identify various disease types at early stages; (2) Our method can effectively eliminate unwanted noise in images by leveraging diffusion-based probabilistic techniques, targeting the stochastic diffusion process at each sampling step. This feature enables our method to perform exceptionally well in handling complex and challenging plant leaf databases; (3) We accomplish a vast assessment of LeafDisDiff on three complex and publicly accessible datasets, namely Apple Disease, Bangladeshi Crops Disease, and PlantVillage Database. The comparative analysis demonstrates the superior capability of LeafDisDiff in plant disease detection and classification relative to existing state-of-the-art models; (4) We demonstrate the versatility and robustness of LeafDisDiff in handling various plant leaf databases, providing a unified and reliable solution for automated plant disease detection.

2. Literature review

This section provides an overview of the background and related works connected to the proposed work. Various methods employed for plant disease classification will be examined. Furthermore, the diffusion model will be introduced, along with an overview of how it works.

2.1. Manual methods of plant leaf diseases classification

Plant leaf diseases are one of the significant factors affecting crop growth and yield, exerting a severe impact on agricultural production [17]. These diseases can be caused by pathogens such as fungi, bacteria, viruses, parasites, etc., which infect plant leaves, leading to discolouration, wilting, rotting, or even complete withering of the leaves [18]. The severity of plant leaf diseases depends on factors such as the type of pathogen, environmental conditions, and the host plant's resistance to the disease. Timely and accurately identifying and classifying plant leaf diseases is crucial for agricultural production [19]. Manual identification methods typically rely on professional plant pathologists who need to observe and diagnose the leaves. Specifically, manual methods typically involve the following steps: (1) Sample Collection: Plant pathologists collect samples of affected plant leaves in the field. These samples may come from different fields, seasons, and plant varieties. (2) Observation of Leaf Characteristics: Pathologists conduct detailed observations of the leaves, including their colour, shape, texture, etc. They also note the presence of spots, moulds, browning, and other signs of disease on the leaves. (3) Use of Reference Materials: Pathologists may use reference materials in plant pathology, such as atlases, literature, etc., to assist in diagnosis. These reference materials typically contain descriptions and picture comparisons of various plant disease symptoms. (4) Diagnosis: Based on the observed leaf characteristics and reference materials, pathologists make a preliminary diagnosis of the disease. This may involve distinguishing between diseases and determining possible cross-infections or mixed infections. (5) Possible Further Testing: In some cases, pathologists may take further testing measures, such as sampling for laboratory analysis or using a microscope to observe the morphological characteristics of the pathogens. While these traditional methods can identify and classify plant leaf diseases to some extent, they typically require experienced professionals and involve subjective judgment. Therefore, this approach is time-consuming, labour-intensive, and not sufficiently efficient.

2.2. Automatic recognition methods

Apart from methods based on manual identification, there are also some vision-based methods used for plant leaf disease classification. These methods typically employ image processing and pattern recognition techniques. Some traditional vision-based methods include: (1) Colour and texture feature extraction: This method describes the visual characteristics of leaves by extracting colour histograms and texture features (such as grey-level co-occurrence matrices [20], local binary patterns [21], etc.) from leaf images. Machine learning algorithms (such as support vector machines, decision trees, etc.) are then used to classify these features. (2) Morphological feature extraction: This method utilizes morphological processing techniques to extract morphological features of leaf images [22], such as leaf shape, size, edges, etc. These features are then used for classification. (3) Texture analysis: By analysing texture features of leaf images [23], such as texture direction, frequency, etc., the texture information of leaves is described. These texture features are then used for classification. (4) Feature combination and selection: Multiple visual features are combined or selected to construct more complex feature representations. These features can include colour, texture, morphology, and other aspects of information [24]. Combining or selecting these features can improve the accuracy of classification. These machine vision-based methods typically have lower computational complexity and resource requirements,

making them valuable in resource-constrained environments. However, these traditional methods often exhibit poor classification performance in complex image scenes and require manual design and selection of features [25]. Therefore, in some cases, they may not achieve the desired accuracy and generalization ability.

Automated methods have several significant advantages over manual approaches. They reduce the need for experienced professionals and minimize subjective judgment, resulting in more consistent and accurate results. Automated techniques can quickly process large datasets, providing timely diagnoses crucial for effective disease management. Additionally, these systems can be deployed in resource-constrained environments where manual expertise is limited. Their scalability and efficiency make automated methods superior, especially for high-throughput analysis and real-time monitoring, which are vital for modern agriculture.

2.3. Deep learning-based classification

In recent years, with the rapid development of deep learning technology, successes have been achieved in multiple fields such as autonomous driving [26,27], medical analysis [28,29], remote sensing [30,31]. The automated identification and classification of plant leaf diseases using computer vision and deep learning techniques have also become a hot research topic. Compared to traditional machine learning, deep learning eliminates the need for extensive feature engineering [32]. Deep learning models can learn rich feature representations from images through training on large-scale datasets, thus possessing higher accuracy and generalization capabilities. Therefore, the application of deep learning technology in the classification of plant leaf diseases holds promise for achieving rapid, accurate, and automated diagnosis and monitoring, providing crucial support for crops' healthy growth and production.

Several studies focusing on the automated diagnosis and identification of plant diseases have leveraged deep learning techniques. Convolutional Neural Network (CNN) has emerged as a potent method for vision-based tasks, showcasing its ability to discern crucial features from images efficiently and without expert intervention [33–37], and adapting seamlessly to a range of leaf disease recognition tasks [38–40]. Architectures based on Convolutional Neural Networks (CNNs) have notably advanced the understanding of intricate patterns. A novel plant leaf identification method utilizing deep CNNs was introduced by [41], capturing botanist behaviour through three integrated deep learning models. In a similar vein, [42] presented a Bi-CNN for plant disease identification and classification, employing leaf images as input and integrating fine-tuned VGG and pruned ResNet as feature extractors connected to fully connected dense networks. Additionally, [43] proposed an approach for classifying rice plant diseases using a deep convolutional neural network. The model incorporates Otsu's global thresholding technique for image binarization, effectively eliminating background noise from the images.

2.4. Transformer based classification

Recent investigations have delved into the utilization of attention mechanisms [44] and transformer architecture [45] for the classification of plant leaf diseases. The Transformer model, renowned for its self-attention mechanism [46], excels in capturing extensive dependencies within input data, leading to a holistic understanding of global information: a capability that traditional CNN architectures find challenging. This self-attention mechanism empowers the transformer to process data more efficiently by evaluating information from every position in the image. Unlike CNNs, where the receptive field is confined by the fixed size of the convolution kernel, the transformer's self-attention mechanism dynamically adjusts its focus across various locations, providing increased flexibility and effectiveness. Zeng et al. [47] introduced an image classification model named the Squeeze-and-Excitation Vision Transformer (SEViT) for large-scale and fine-grained disease identification. SEViT incorporates a ResNet with a channel attention module as its preprocessing network and employs the Vision Transformer (ViT) for feature classification. Thakur et al. [48] developed a model named "PlantXViT", which is a Convolutional Neural Network enabled with Vision Transformer. This model combines the strengths of traditional CNNs with Vision Transformers, effectively identifying a diverse range of plant diseases across various crops. Yu et al. [49] proposed an innovative transformer block utilizing transformer architecture to model long-range features. This block also employs soft split token embedding to capture local information from surrounding pixels and patches.

2.5. Diffusion model

The diffusion model simulates the spread of data points in a latent space to determine the dataset's underlying structure in a latent space [12,50]. The diffusion model involves two main processes: forward diffusion and backward diffusion. Its operation principle is to add noise (Gaussian noise) to available training data (also known as the forward diffusion process). The input data undergo gradual perturbation by introducing Gaussian noise across multiple time steps. The process is then reversed (denoising or backward diffusion process) to recover the data. The backward process attempts to restore the original input data by minimizing the difference between the predicted noise and the added noise over multiple backward time steps. The model gradually learns to eliminate noise. This denoising process learns to generate new high-quality images from a random seed (random noise image). The diffusion model has recently attracted attention due to its outstanding flexibility and performance, effectively addressing complex visual challenges such as image denoising and generation [51,52].

3. Proposed methodology

3.1. Background

Diffusion Model: These models belong to the category of probabilistic generative models, introducing noise to data and subsequently learning to restore it for generating new samples. The three primary types of diffusion models are denoising diffusion probabilistic models (DDPMs), score-based generative models (SGMs), and stochastic differential equations (Score SDEs). Our study specifically delves into DDPMs [12], which approach diffusion models from a discrete-time perspective, treating them as latent variable models, as elaborated in the next paragraph. A denoising diffusion probabilistic model (DDPM) is a subtype of probabilistic generative model that introduces noise to data and then learns to recover it, generating new samples.

Denoising Diffusion Probabilistic Model: DDPM, which is the denoising diffusion probabilistic model, is a seminal method in image space diffusion. The concept of diffusion models involves progressively destroying all information in an image over a sequence of time steps T, where at each step a small amount of Gaussian noise is added. By the end of a large number of steps, the image becomes completely random noise, mimicking a sample from a normal distribution. This is called the forward process, wherein we apply a transition function to move from x_{T-1} to x_T. We then learn a reverse of this process via a neural network model, effectively training the model to remove noise from the image step by step. Once this model is trained, we can feed it random noise sampled from a normal distribution, and it will iteratively denoise it. By the end of numerous denoising steps, we obtain an image from the original distribution. This method is suitable for our study because its robust denoising capabilities effectively handle the noise and variability in plant disease images. By using a diffusion-based methodology with a U-Net-style architecture and integrating residual blocks, the LeafDisDiff model accurately reconstructs original image data and enhances feature

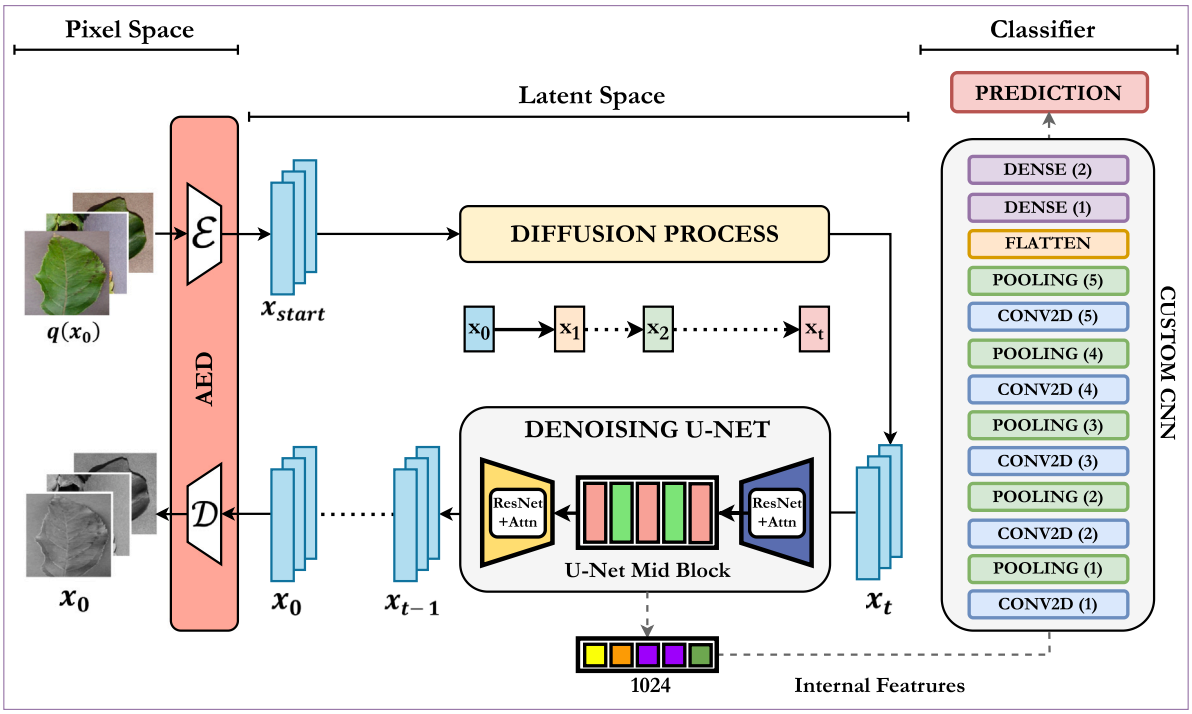


Fig. 1. Proposed LeafDisDiff Model: Diffusion-based U-Net Architecture with Residual Blocks for Leaf Disease Detection and Classification.

extraction. This results in superior classification accuracy, particularly in distinguishing subtle disease symptoms obscured by noise.

Deep Latent Variable Model: A predetermined forward diffusion process, represented as q, systematically introduces Gaussian noise to an image, progressively resulting in a fully noisy image. Conversely, a trained reverse denoising diffusion process, denoted as p_θ, engages a neural network in gradually eliminating noise from an image. This process commences from a state of pure noise, persisting until the original image is successfully restored. Both the forward and reverse processes, denoted by t, unfold over a defined duration of discrete temporal steps T (following DDPM authors [12], where $T = 1000$). Initiating at $t = 0$, a real image x_0 is sampled from the data distribution. In the forward process, noise is sampled from a Gaussian distribution at each time step t and incorporated into the image from the preceding step:

$$z_t \sim \mathcal{N}(0, \sigma^2) \tag{1}$$

$$x_t = x_{t-1} + z_t \tag{2}$$

A sufficiently large T and a well-designed noise schedule leads to the gradual formation of an isotropic Gaussian distribution at $t = T$. These steps, from initial image sampling to the convergence of the distribution, outline a fundamental process in this context.

Loss Function: We proceed with a typical implementation for the diffusion framework by introducing the forward diffusion process $q(x_t \mid x_{t-1})$, which applies Gaussian noise at each time step t, the level of input noise is defined by variance schedule's which are predefined as $\beta_1 < \beta_2 < \cdots < \beta_T < 1$. This can be represented as:

$$q(x_t \mid x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{3}$$

The Gaussian distribution is characterized by two essential parameters: a mean (μ_θ) and a variance (Σ_θ). At each time step t, a new image that is slightly noisier than the previous one is drawn. The process unfolds through successive steps from the initial image x_0, leading to a sequence of images $x_1, \ldots, x_t, \ldots, x_T$. When the variance schedule is appropriately designed, the final image x_T primarily consists of pure Gaussian noise.

To reverse the process by gradual denoising to obtain a sample from the original volumetric data x_0, we need to compute the conditional probability, which requires the distribution of all possible images. To tackle the same, we require a neural network to approximate and learn the conditional probability distribution described below:

$$p_\theta(x_{t-1} \mid x_t) \tag{4}$$

Objective Function: The central task of the neural network is to represent the mean of the conditional probability distribution while keeping the variance constant (specifically for this study, we assume a fixed variance). We use the concept of a variational autoencoder to construct an objective function for learning the mean of the backward process. This analogy allows us to employ the variational lower bound to minimize the negative log-likelihood concerning the real data x_0. For this study, the variational lower bound can be defined as the cumulative sum of losses at each time step. Now, utilizing the processes given by [53] to sample x_T at any given noise level on x_0, we can now sample out Gaussian noise, adjust its scale, and add it to x_0, thereby yielding x_T directly. This capability extends to the training phase, enabling the optimization of stochastic terms within the loss function L, alongside permitting the random selection of t during training.

$$q(x_t \mid x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I) \tag{5}$$

The final formulation of the objective function for a random time step t, given $\epsilon \sim \mathcal{N}(0, I)$, can be expressed as follows:

$$\|\epsilon - \epsilon_\theta(x_t, t)\|^2 = \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{(1-\bar{\alpha}_t)}\epsilon, t)\|^2 \tag{6}$$

This equation encapsulates the core of our approach. It involves the real image x_0, a sample of the fixed forward process's noise level, and the pure noise sample ϵ from time step t, all orchestrated by the neural network.

The proposed Leaf Disease Diffusion (LeafDisDiff) model is illustrated in 1. The model employs a diffusion-based methodology with a U-Net-style architecture consisting of an encoder and decoder integrated with residual blocks. It employs an advanced archetype wherein noise is introduced as a supplementary input to both image batches and their corresponding outputs. The figure depicts the three main steps of

the model: (1) Sampling various noise levels uniformly for a random time step, (2) Training an auto encoder–decoder (AED) based on a U-Net style neural network to predict this noise based on the corrupted images, and (3) Utilizing the internal feature information present in the bottleneck layer (U-Net Mid Block) on 1024 features to train a CNN-based custom classifier for accurate prediction of crop disease classes. The model enhances classification accuracy by utilizing internal features present in the bottleneck layers (U-Net Mid Block) as inputs for the custom classifier.

3.2. Denoising U-Net

In our work, we employ a diffusion-based methodology that leverages a U-Net-style architecture characterized by the integration of residual blocks. The core of our approach hinges on a denoising technique that predicts the noise at each time step. Unlike conventional methods that directly feed batches of images into the learning process and predict their corresponding labels, our implemented diffusion model adopts a more sophisticated paradigm. It accommodates image batches and their corresponding labels while introducing noise as an additional input. The proposed diffusion-based LeafDisDiff model is depicted in Fig. 1. In the following Section 3.2.1, we will delineate the step-by-step formulation of the diffusion process. Section 3.2.2 provides an overview of the overall architecture, including key components in U-Net architecture. Moving forward to Sections 3.2.3 and 3.2.4, we explore the working of the U-Net architecture and the image sampling process using the reverse diffusion method.

Let us consider the real data distribution $q(x_0)$, which depicts our original images from the dataset. The image has height H, width W, and N colour channels. Image samples $x_0 \sim q(x_0)$ are generated from these "real images". We proceed with a typical implementation of our proposed framework for classification, which is composed of three main steps given below:

(1) Sampling various noise levels uniformly for a random time step t as shown in the diffusion process $x_0, x_1, x_2, \ldots, x_t$ in Fig. 1 using the forward diffusion process.

(2) A U-Net-style neural network, which resembles an autoencoder–decoder (see Fig. 2), is trained to predict this noise based on the corrupted images.

(3) Utilizing the internal feature information present in the bottleneck layer (U-Net Mid Block) with 1024 features to train our CNN-based custom classifier for the accurate prediction of crop disease classes.

3.2.1. Formulation of forward diffusion process

In the forward diffusion process, a gradual introduction of noise into an image, sourced from the real distribution, unfolds across a specified number of time steps represented as T. This noise infusion adheres to a predefined variance schedule. Initially, we set T to 300-time steps and establish essential variables based on β_t, notably including the cumulative product of variances, denoted as $\bar{\alpha}_t$. Each of these variables, existing as 1-dimensional tensors, effectively stores values ranging from t to T. Of particular significance, we define an extraction function that facilitates the retrieval of the suitable t index for a batch of indices, streamlining the sampling process. In Fig. 1, the diffusion process uses this forward diffusion to obtain a completely noised image x_T, starting with an image from the volumetric data x_{start}.

3.2.2. Architecture and components of U-Net

Our U-Net network architecture resembles an autoencoder (see Fig. 2), which typically incorporates a "bottleneck" layer between the encoder and decoder. This design compels the U-Net network to retain only the most critical information in the bottleneck layer. Like an autoencoder, the U-Net network features a central bottleneck (U-Net Mid Block) ensuring the learning of essential information. It also incorporates residual connections between the encoder and decoder, significantly enhancing the U-Net network's gradient flow. The neural network receives a distorted image during a specific time step and provides a predicted noise in response. It is crucial to clarify that the anticipated noise refers to a tensor that maintains an identical size and resolution as the initial input image. This implies that the U-Net network processes tensors with matching shapes upon intake and output.

While a typical convolutional neural network is primarily designed for image classification, with the input being an image and the output a singular label, U-Net introduces a distinctive "U" shape in its architecture. This symmetric design comprises two fundamental segments: (1) the left segment, termed the contracting path, encompasses the conventional convolutional processes, and (2) the right segment, referred to as the expansive path, integrates transposed 2D convolutional layers. Our instantiation of the U-Net architecture incorporates four integral components: (1) position embeddings, (2) ResNet blocks, (3) attention mechanisms, and (4) group normalization.

1. ***Position Embeddings:*** The U-Net network deals with images that may have varying levels of noise. It is crucial to consider the temporal aspect or the specific time step (noise level) associated with each image to process these images effectively. To capture this information, we employ sinusoidal position embeddings inspired by [44]. The sinusoidal position embedding module takes a tensor of shape (batch size, 1) as input, representing the noise levels of multiple images in a batch. It transforms this input into a tensor of shape (batch size, dim), where "dim" corresponds to the dimensionality of the position embeddings. The transformed position embeddings are then added to each residual block. Position embeddings are integral to our architecture as they provide contextual information [54] about the specific segment of the input or output sequence the model is currently addressing.
2. ***ResNet Block:*** We now delineate the fundamental building block of the U-Net model, incorporating the residual architecture at its core, chosen for its exceptional performance. Our architecture follows the backbone of PixelCNN++ [55]. Our model consists of 9 blocks of ResNet layers (see Fig. 2), strategically positioned within the U-Net structure: four during downsampling, one in intermediate blocks, and four during upsampling.
3. ***Attention Block:*** Attention finds extensive utility in deep learning and natural language processing domains, serving as a pivotal component with significant implications. In our study, self-attention [44] is applied to the feature map resolution of 16×16, strategically incorporated between convolutional blocks [56].
4. ***Group Normalization:*** To address the challenges that arise during normalization, Group Normalization (GN) emerges as a viable alternative. To streamline the implementation, we replaced weight normalization [57] with group normalization [58]. Since the computation of GN remains unaffected by variations in batch size, it ensures consistent accuracy across a wide range of batch sizes. We incorporate group normalization before the attention layer within the convolutional and attention layers of the U-Net architecture.

A schematic representation of the denoising U-Net architecture used in our work presented in Fig. 2. The architecture is characterized by a series of downsampling and upsampling stages, each incorporating ResNet blocks, GroupNorm, attention mechanisms, and a residual connection. The mid-block of the U-Net network (U-Net Mid Block), crucial for the classifier, harnesses the inherent capacity of the diffusion U-Net to internally generate intricate representations of its inputs. We utilize the internal feature information present in the bottleneck layer (U-Net Mid Block) with 1024 features to train a CNN-based custom classifier for the accurate prediction of crop disease classes.

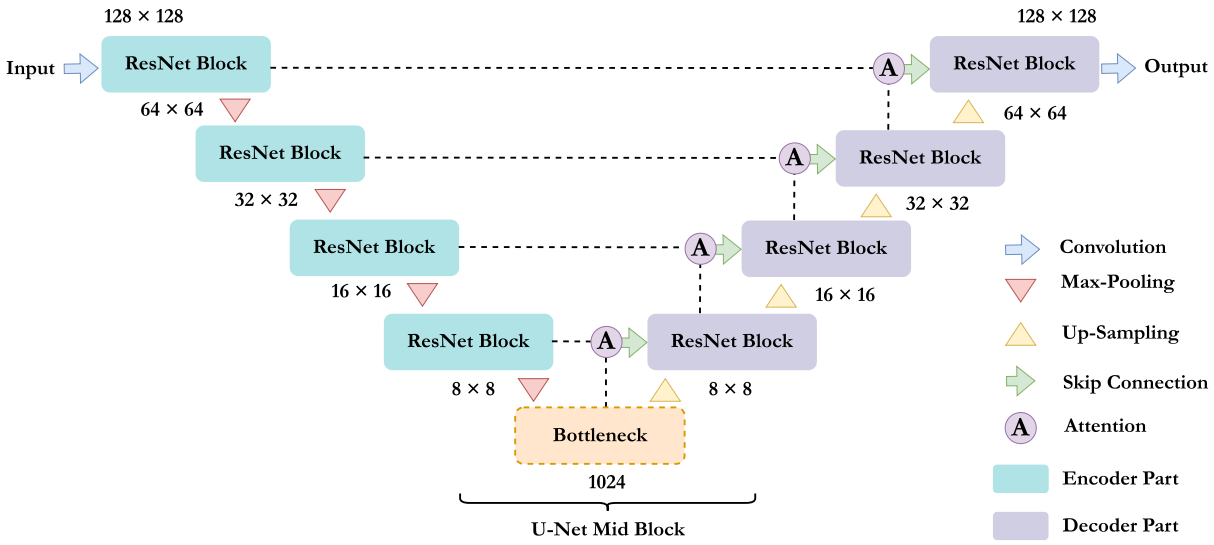


Fig. 2. Architecture Overview: Denoising U-Net for Crop Disease Detection.

3.2.3. Working of U-Net architecture

As mentioned in the above section, the data processing module accepts an array of input images affected by noise and their corresponding noise levels arranged in a multidimensional structure. This module processes these inputs, producing an output tensor that maintains the same dimensionality as the initial set of noisy images. We employ the typical U-Net architecture, and the construction of the U-Net network proceeds through the following stages:

a. Initially, a convolutional layer is applied to the batch of noisy images, accompanied by the computation of position embeddings for the associated noise levels. Subsequently, a series of downsampling stages is implemented, each comprising ResNet blocks, GroupNorm, attention mechanisms, a residual connection, and a downsampling operation as shown in Fig. 2.
b. In the mid-block of the U-Net network, ResNet blocks are once again applied, alternating with attention mechanisms. The inherent capacity of the diffusion U-Net to internally generate intricate representations of its inputs serves as a valuable foundation that we utilize as input (1024 features) for our custom classifier to accurately predict crop disease classes (Section 3.3). These internal representations not only encapsulate a comprehensive depiction of the existing noisy input but also incorporate details regarding the noise level.
c. Finally, a sequence of upsampling stages ensues, with each stage incorporating ResNet blocks, GroupNorm, attention mechanisms, a residual connection, and an upsample operation depicted in Fig. 2 as the upsampling process (ResNet + Attn). The U-Net network culminates with the application of a ResNet block followed by a convolutional layer. This comprehensive structure aims to effectively process noisy input images, integrating various components for optimal performance.

All the layers, along with a clear depiction of their working and the crucial mid-block, are presented in Fig. 2.

3.2.4. Sampling

The sampling procedure, central to the generation of new images in a diffusion model, revolves around the reversal of the diffusion process. Commencing from $t = T$, we initiate this by sampling pure noise from a Gaussian distribution. Subsequently, we leverage the U-Net network to progressively denoise this noise, guided by the conditional probability it has acquired through learning until we arrive at time step $t = 0$. As elucidated earlier, we can derive a slightly less denoised image, designated as $x_{(t-1)}$, by incorporating the reparameterization of the mean, employing our noise predictor. It is noteworthy that the variance is predetermined and known in advance. The ultimate objective of this process is to produce an image that authentically resembles a sample from the real data distribution. The more accurate the generated image from the diffusion model is, the richer the internal feature vector generated from the bottleneck layers becomes. This ultimately aids in the classification task as shown in the experimental section.

3.3. Using diffusion U-Net as classifier

The volumetric pre-processed batches of images are fed to the encoder, compressing each image into a more miniature hidden representation known as the "bottleneck". This bottleneck step retains internal features. Subsequently, the diffusion process initiates, resulting in a completely noised image. This is followed by denoising, where our diffusion model, in collaboration with the U-Net architecture (explained in the previous section), acts as a noise predictor. The decoder then reconstructs the image using this hidden representation. By employing the features from the bottleneck layer (U-Net Mid Block) as inputs for our custom classifier, the implemented diffusion model improves classification accuracy, excelling in both accurate classification tasks and effective noise reduction. Our approach not only enhances overall classification results but also demonstrates higher accuracy, as evidenced by applying multiple classifiers to the sampled-out images from the model (see Tables 5, 6, 7, 8, 9, 10). We use the parameters specified in the hyperparameter settings given in Table 4 in conducting these experiments. As detailed in Section 4.4, a batch of images is sampled from the prepared dataset, encompassing multiple classes. Our proposed model is then employed to extract mid-block internal features from these images. These features contribute to building a classification model, considering any internal features as potential strong candidates. Subsequently, multiple classifiers are trained, and their classification accuracies are thoroughly compared. The detailed comparison of various classifiers is presented in the experiments and results section (see Tables 8, 9, 10); in the next section, we specify the architecture used in our custom classifier for the prediction of crop disease classes.

3.3.1. Custom classification network specifications

We employ a Convolutional Neural Network (CNN) architecture for image classification. The model is trained using categorical cross-entropy as the loss function, indicating a multi-class classification objective, as our dataset comprises multiple classes. For efficient parameter updates during training, we opt for the Adam optimizer with a

learning rate of 0.001. The classifier's network architecture is described as follows:

a. Input Layer (CONV2D (1)): The initial layer of our Convolutional Neural Network (CNN) is a 2D convolutional layer with 16 filters, each of size (3,3), and it utilizes the rectified linear unit (ReLU) activation function.
b. Max Pooling Layer (POOLING (1)): Following the first convolutional layer, a max pooling layer with a pool size of (2,2) is employed. This layer plays a crucial role in downsampling the spatial dimensions of the feature maps, enabling the network to focus on the most significant features.
c. Convolutional Block (CONV2D (2) and POOLING (2)): A second convolutional block, comprising a Conv2D layer with 32 filters and ReLU activation, is introduced, followed by another max pooling layer.
d. Convolutional Block (CONV2D (3) and POOLING (3)): Continuing the architectural progression, a third convolutional block is implemented. It involves a Conv2D layer with 64 filters and ReLU activation, followed by a max pooling layer.
e. Convolutional Blocks (CONV2D (4), POOLING (4), CONV2D (5), and POOLING (5)): Two additional convolutional blocks follow a similar pattern, each featuring a Conv2D layer with 64 filters and a subsequent max pooling layers.
f. Flatten Layer: After the series of convolutional and max pooling operations, a flatten layer is introduced. This layer transforms the three-dimensional output into a one-dimensional vector, preparing the data for input to the fully connected layers.
g. Dense Layer (DENSE (1)): A dense layer with 128 neurons and ReLU activation follows the flattening operation.
h. Dense Layer (DENSE (2)): The final dense layer, comprising 15 neurons, represents the output layer. It utilizes the *softmax* activation function for multi-class classification, producing probability distributions over the different classes.

4. Experimental datasets

We evaluated the performance of the proposed LeafDisDiff network on three different plant disease datasets and compared the experimental results with the various traditional and deep learning techniques to show the superiority of the developed methodology. For more details, please refer to the GitHub repository. The Apple Disease, Bangladeshi Crops Disease, and PlantVillage databases are among the datasets are utilized for experimental evaluations. Each of these datasets is described in the subsections below.

4.1. Apple disease dataset

Apple leaf disease dataset, derived from the original New Plant Diseases Dataset,[1] was meticulously crafted by isolating apple leaf images from the broader dataset. It comprises approximately 7,771 coloured images of healthy and diseased apple leaves, categorized into 4 classes, i.e., black rot, healthy, cedar apple rust, and apple scab, respectively. It is available on Kaggle.[2] The class-wise number of instances is shown in Table 1, and in addition, ten randomly selected samples are displayed in Fig. 3.

Compared to the other two datasets used in conducting the experiments this dataset consists of fewer classes and overall less number of samples as it can be noticed from Table 1.

Table 1
Classes and the number of samples per class in the Apple leaf disease dataset.

Classes	Samples
Black Rot	1987
Healthy	2008
Cedar Apple Rust	1760
Apple Scab	2016

4.2. Bangladeshi crops disease dataset

The Bangladeshi crops disease dataset is a curated compilation, encompassing 15 distinct classes that portray various crop diseases. Featuring a collection of 31,053 image files shown in Table 2, this dataset offers a comprehensive resource for exploring agricultural diseases that are common in Bangladesh. The images have been thoughtfully selected from reputable sources, including the PlantVillage dataset, Rice disease dataset, and Wheat disease dataset. Each class within the Bangladeshi Crops disease dataset represents a unique challenge these essential crops must overcome. A few instances from the 15 classes are depicted in Fig. 4. The dataset can be accessed on Kaggle.[3]

Though the dataset has a sufficient number of samples for the training and evaluation of the model along with the greater number of classes compared to the Apple Disease Dataset, it shows class imbalances as it can be derived by comparing the number of class samples from Table 2.

4.3. PlantVillage dataset

The PlantVillage dataset comprises of extensive collection of more than 50,000 in carefully selected images, encompassing healthy and infected crop plant leaves. These valuable images are conveniently accessible via the PlantVillage online platform. This dataset is essential to the advancement of mobile disease diagnostics since it makes use of crowdsourcing and machine learning to fight infectious diseases that have a major impact on crop production. PlantVillage dataset consists of 38 classes, and the number of samples in each class is shown in Table 3. Furthermore, a few examples from each class are shown in Fig. 5. This dataset can also be accessed on Kaggle.[4]

The PlantVillage dataset shows significant class imbalance, with some diseases like *Orange Haunglongbing* having 5,507 samples, while others like *Corn (maize) healthy* having only 116. This imbalance can bias models towards majority classes.

4.4. Preprocessing and preparation of datasets:

The previous section mentions a detailed description of the datasets being used in our work. For dataset preparation, we have arranged three datasets (apple disease, plant village disease, and Bangladeshi crop disease) in 3 different folders, and they contain separate subfolders inside them for multiple classes of disease containing images in *.jpg* format. The images are fetched from these during training, sampling, or inferencing.

(1) Normalization and Resizing: We preprocess the batch of images (64 images per batch) from the dataset and resize the images to 128×128, followed by centre cropping and conversion to tensor with shape $C \times H \times W$ (channel, height, width). Subsequently, a reverse transformation is defined to convert the processed tensor back into the image. This involves reversing the intensity normalization, reshaping the tensor from CHW to HWC (Height, Width, Channel), scaling values back to the range $[0, 255]$, and creating an image.

[1] https://www.kaggle.com/datasets/vipoooool/new-plant-diseases-dataset
[2] https://www.kaggle.com/datasets/ludehsar/apple-disease-dataset
[3] https://www.kaggle.com/datasets/nafishamoin/bangladeshi-crops-disease-dataset
[4] https://www.kaggle.com/datasets/abdallahalidev/plantvillage-dataset

Fig. 3. Ten samples are randomly taken from each class of Apple leaf disease datasets.

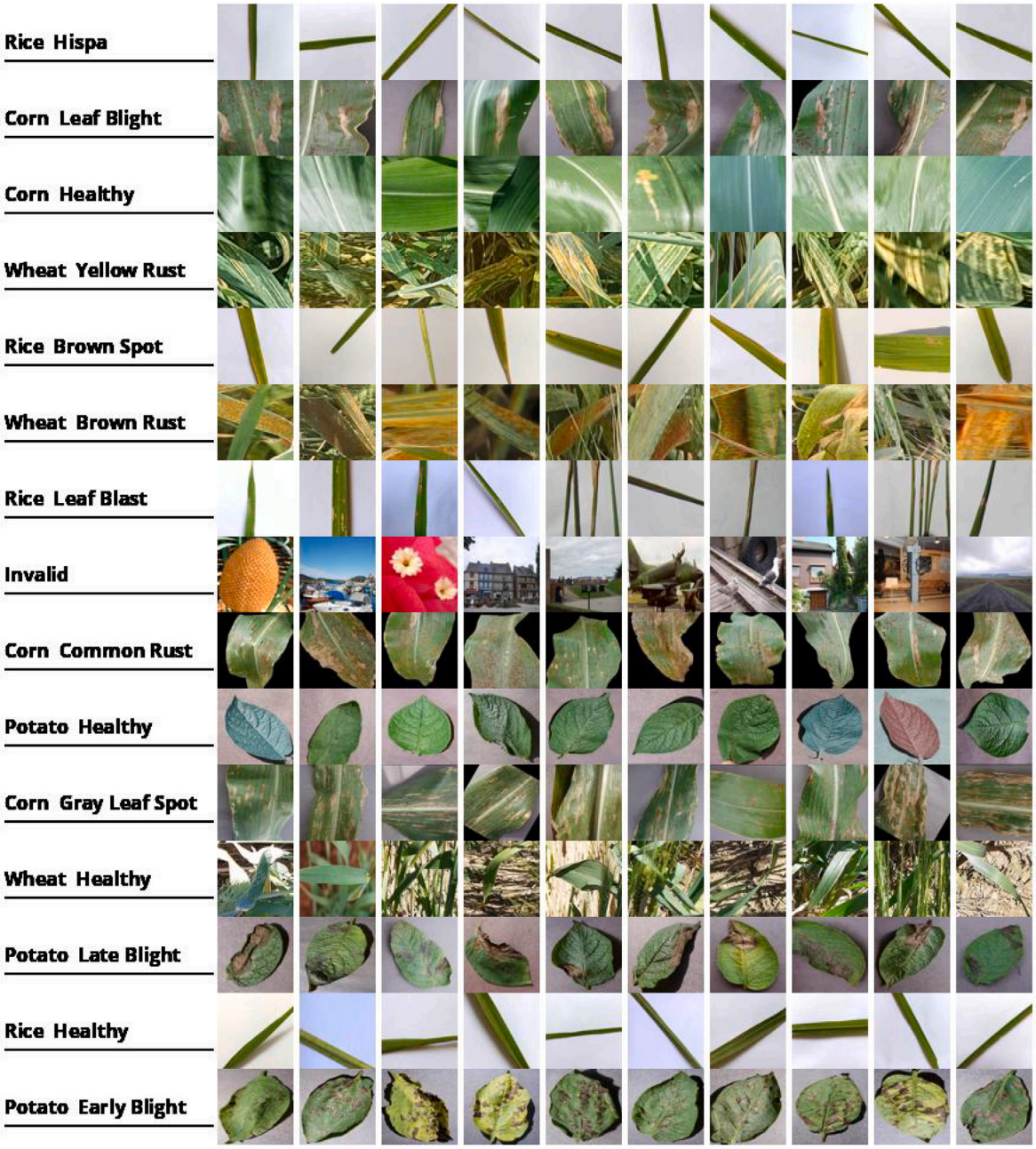


Fig. 4. Class-wise ten samples are randomly taken from Bangladeshi crop disease datasets.

Fig. 5. Class-wise ten samples are randomly taken from Plantvillage datasets.

Table 2
Classes and the number of samples per class in the Bangladeshi Crop disease dataset.

Classes	Samples	Classes	Samples
Wheat_Yellow_Rust	1156	Potato_Late_Blight	3131
Potato_Healthy	2006	Rice_Healthy	523
Corn_Healthy	3718	Invalid	1563
Wheat_Brown_Rust	1128	Corn_Common_Rust	3814
Corn_Gray_Leaf_Spot	3284	Corn_Leaf_Blight	3816
Rice_Brown_Spot	563	Potato_Early_Blight	3149
Wheat_Healthy	1497	Rice_Hispa	523
Rice_Leaf_Blast	1182		

(2) Noise Function: Now, we have a forward function that generates a noisy image by combining the original image with noise. This function obtains a noisy image at a specific diffusion time step. The dataset is then transformed, including resizing, random horizontal flipping, and normalization. These processed images are subsequently used to train our LeafDisDiff model, the architecture of which is illustrated in Fig. 1.

5. Results and discussion

5.1. Experimental setting

Hyperparameters and experimental settings play a key role in the convergence of the model during training. Table 4 shows the experimental parameters that are used in the developed deep learning models. For instance, the number of hidden layers in CNN, EfficientNet-B3, VGG-19, and ResNet-50 models was set to 8, 19, 19, and 50, respectively. The input image patch size for all the developed classifiers was 128, while the learning rate, dropout size, batch size, optimizer, activation function, epochs, and momentum were set to 128, 0.001, 0.5, 64, Adam, ReLU, 30, and 0.9, respectively.

5.2. Experiments

5.2.1. Performance analysis metrics

To assess the effectiveness of the classification, we consider four common evaluation criteria: overall accuracy (OA), class-wise accuracy (CA), average accuracy (AA), and the Kappa coefficient (Kappa). Let us examine a dataset with C classes, where N represents the total number of samples in the dataset. Additionally, we use $X_{i,i}$ to represent correctly classified samples; in other words, if the ith sample belongs to the ith class, it is correctly classified; if not, it is misclassified into the jth class and is represented by $X_{i,j}$. The overall accuracy (OA), determined by the number of samples correctly classified out of the total test samples, is defined as:

$$OA = \frac{\sum_{i=1}^{C} X_{i,i}}{N} \times 100\% \tag{7}$$

The percentage of properly classified samples in each category is indicated by the class accuracy (CA) and calculated as:

$$CA = \frac{X_{i,i}}{\sum_{i=j}^{C} X_{i,j}} \times 100\% \tag{8}$$

The average of class-wise classification accuracies determines AA which can be formulated as:

$$AA = \frac{\sum_{i=1}^{C} y_i}{C} \times 100\% \tag{9}$$

where $y_i = CA_i$. Kappa is a statistical measurement metric that mutually indicates a high degree of agreement between the generated classification map and the ground truth map, which can be defined as follows:

$$Kappa = \frac{N \sum_{i=1}^{C} X_{i,i} - \sum_{i=1}^{C} \sum_{j=1}^{C} X_i X_j}{N \times N - \sum_{i=1}^{C} \sum_{j=1}^{C} X_i X_j} \tag{10}$$

where $X_{i,i}$, X_i, and X_j represent the diagonal elements, the sum of the ith row, and the sum of the jth column of the corresponding confusion matrix, respectively.

5.2.2. Performance over apple disease dataset

Plant leaf disease classification results over the Apple disease dataset utilizes the developed CNN- and ViT-based classifiers, including CNN, ResNet-50, EfficientNet-B3, VGG-19, ViT, Cross-ViT, and the LeafDisDiff, respectively are reported in Table 5. The plant leaf disease classification results on the Apple leaf disease benchmark data illustrated the superior performance of the developed diffusion-based algorithm of the LeafDisDiff, specifically compared to the multi-head self-attention-based classifiers, i.e., ViT and Cross-ViT. The obtained results revealed that the best plant leaf disease classification performance in terms of statistical indices i.e., overall accuracy (100% ± 0.00%), average

Table 3
Classes and the number of samples per class in the Plant Village dataset.

Classes	Samples	Classes	Samples
Tomato_Late_blight	1909	Peach_Bacterial_spot	2297
Tomato_healthy	1591	Apple_Cedar_apple_rust	275
Grape_healthy	423	Tomato_Target_Spot	1404
Orange_Haunglongbing_(Citrus_greening)	5507	Pepper_bell_healthy	1478
Soybean_healthy	5090	Grape_Leaf_blight_(Isariopsis_Leaf_Spot)	1076
Squash_Powdery_mildew	1835	Potato_Late_blight	1000
Potato_healthy	152	Tomato_Tomato_mosaic_virus	373
Corn_(maize)_Northern_Leaf_Blight	985	Strawberry_healthy	456
Tomato_Early_blight	1000	Apple_healthy	1645
Tomato_Septoria_leaf_spot	1771	Grape_Black_rot	1180
Corn_(maize)_Cercospora_leaf_spot_Gray_leaf	513	Potato_Early_blight	1000
Strawberry_Leaf_scorch	1109	Cherry_(including_sour)_healthy	854
Peach_healthy	360	Corn_(maize)_Common_rust_	1192
Apple_Apple_scab	630	Grape_Esca_(Black_Measles)	1383
Tomato_Tomato_Yellow_Leaf_Curl_Virus	5357	Raspberry_healthy	371
Tomato_Bacterial_spot	2127	Tomato_Leaf_Mold	952
Apple_Black_rot	621	Pepper_bell_Bacterial_spot	997
Blueberry_healthy	1502	Tomato_Spider_mites_Two-spotted_spider_mite	1676
Cherry_(including_sour)_Powdery_mildew	1052	Corn_(maize)_healthy	116

Table 4
Hyperparameters and experimental values for various deep learning models.

Hyper-parameters	Experimental settings				
	CNN	ResNet-50	EfficientNet-B3	VGG19	LeafDisDiff
Hidden Layers	8	50	19	19	920
Input Image Size	128	128	128	128	128
Nodes per Trained Layer	55–4096	64–2048	64–2048	64–4096	64-2048
Learning Rate	0.001	0.001	0.001	0.001	0.001
Dropout	0.5	0.5	0.5	0.5	None
Batch Size	64	64	64	64	64
Optimizer	Adam	Adam	Adam	Adam	Adam
Activation Function	ReLU	ReLU	Swish	ReLU	ReLU
Epochs	30	30	30	30	20
Momentum	0.9	0.9	0.99	0.9	0.9

accuracy (99.99% ± 0.00%), and Kappa index (0.9999 ± 0.00) were obtained by the proposed LeafDisDiff network, followed by the 2D CNN with an overall accuracy (99.91% ± 0.02%), average accuracy (98.33% ± 4.30%), and kappa index (0.9942 ± 0.087), respectively. For instance, in terms of average accuracy, the developed diffusion algorithm i.e., the proposed LeafDisDiff model considerably enhances the plant leaf disease classification as compared to the other classifiers, including the ViT, Cross ViT, ResNet-50, CNN, VGG-19, and EfficientNet-B3 by about 7, 6, 2, 2, 1.5, and 1 percentage points, respectively, as reported in Table 5. Overall, the transformer-based classifiers, i.e., ViT and Cross ViT demonstrated the lowest classification performance achieved over the CNN-based algorithms, such as VGG-19 and EfficientNet-50. This can be attributed to the better generalization capability of CNNs over the vision transformer with limited available ground truth data, as shown in Table 1, highlighting the proven drawback of ViTs in plant leaf image classification applications with a few training samples.

5.2.3. Performance over Bangladeshi crops disease dataset

The classification results of the compared CNN- and ViT-based deep learning classifiers, including CNN, ResNet-50, EfficientNet-B3, VGG-19, ViT, Cross-ViT, and the proposed LeafDisDiff obtained based on the Bangladeshi crops disease dataset are illustrated in Table 6. The best classification performance in terms of overall accuracy (100% ± 0.00), average accuracy (99.82% ± 0.11), and Kappa index (0.9995% ± 0.0005), respectively were achieved by the developed LeafDisDiff algorithm. For instance, in terms of average accuracy, the LeafDisDiff classifier (99.82) outperformed the other classification algorithms, i.e., ViT, VGG-19, Cross ViT, CNN, ResNet-50, EfficientNet-B3 by about 9, 5, 4, 4, 1, and 1, respectively. Besides the developed LeafDisDiff algorithm, the CNN-based models, i.e., ResNet-50 and EfficientNet-B3, illustrated superior classification performance over the other algorithms. The ViT model obtained the worst classification results with an overall accuracy (91.88% ± 2.09), average accuracy (91.02% ± 3.01), and Kappa index (0.9184 ± 0.0104), as shown in Table 6.

5.2.4. Performance over PlantVillage dataset

Classification results of the PlantVillage dataset using the compared CNN- and ViT-based classification algorithms, including CNN, ResNet-50, EfficientNet-B3, VGG-19, ViT, Cross-ViT, and the proposed LeafDisDiff are shown in Table 7. The highest scores in terms of statistical indices of overall accuracy (100% ± 0.00), average accuracy (99.49% ± 0.50), and kappa index (0.9996 ± 0.0042) were achieved by the developed LeafDisDiff classification model. In terms of overall accuracy, the LeafDisDiff classification model (100%) outperformed the other compared CNN- and ViT-Based classifiers of the ViT, Cross ViT, VGG-19, CNN, ResNet-50, and EfficientNet-B3 by approximately 6, 3.5, 3, 3, 1, and 0.5 percentage points, respectively. Moreover, in terms of average accuracy, the LeafDfisDiff algorithm considerably improved the plant leaf disease classification performance of the ViT, Cross ViT, VGG-19, CNN, ResNet-50, and EfficientNet-B3 models by about 9, 6, 6, 5.5, 2.5, and 1, percentage points, respectively, with an average accuracy of 99.49%. In addition, in terms of kappa index, the LeafDisDiff classifier with a kappa index of 0.9996 enhanced the plant disease classification results of the ViT, Cross ViT, VGG-19, CNN, ResNet-50, and EfficientNet-B3 classifiers by approximately 7, 5, 4.5, 4, 2, and 1, percentage points, respectively, as seen in Table 7. Besides the developed LeafDisDiff classification algorithm and compared to the other implemented classifiers, the CNN model of the EfficientNet-50 illustrated the second-best plant disease classification performance with an overall accuracy (99.67% ± 2.15), average accuracy (98.80% ± 0.52), and kappa index of (0.9925 ± 0.0057). The worst performance for plant disease classification was obtained by the vision transformer of the ViT by an overall accuracy (94.19% ± 2.58), average accuracy (90.31% ± 2.99), and Kappa index of (0.9319 ± 0.0296). The most

Table 5
Results obtained using various deep learning models on the Apple leaf disease dataset.

Class	Deep models				Transformers		LeafDisDiff
	CNN	ResNet-50	Efficient-B3	VGG19	ViT	CrossViT	
1	97.76 ± 4.47	97.51 ± 3.63	98.48 ± 4.68	99.16 ± 4.37	94.48 ± 2.28	91.02 ± 3.73	99.99 ± 0.01
2	98.29 ± 3.76	96.48 ± 4.26	99.31 ± 3.41	98.60 ± 3.88	95.51 ± 3.27	99.4 ± 4.6	100.00 ± 0.00
3	97.89 ± 4.12	99.49 ± 4.92	98.65 ± 4.59	98.69 ± 4.74	93.26 ± 4.33	92.01 ± 2.44	99.99 ± 0.01
4	99.37 ± 4.85	99.33 ± 4.03	98.48 ± 4.03	97.67 ± 3.99	87.7 ± 2.14	93.14 ± 2.94	100.00 ± 0.01
OA	99.91 ± 0.02	99.82 ± 0.14	99.2 ± 0.67	99.06 ± 0.13	94.39 ± 3.77	95.26 ± 2.73	100 ± 0.00
AA	98.33 ± 4.30	98.2 ± 4.09	98.73 ± 4.17	98.53 ± 4.24	92.74 ± 3.01	93.89 ± 3.43	99.99 ± 0.00
kappa	0.9942 ± 0.087	0.991 ± 0.001	0.9893 ± 0.078	0.99 ± 0.000	0.941 ± 0.025	0.9489 ± 0.034	0.9999 ± 0.000

Table 6
Results obtained using various deep learning models on the Bangladeshi disease dataset.

Class	Deep models				Transformers		LeafDisDiff
	CNN	ResNet-50	Efficient-B3	VGG19	ViT	CrossViT	
1	95.66 ± 1.68	99.93 ± 0.07	99.94 ± 0.06	97.77 ± 2.23	89.98 ± 4.7	98.22 ± 1.78	99.94 ± 0.06
2	98.74 ± 1.26	99.94 ± 0.06	99.95 ± 0.05	91.09 ± 3.24	95.55 ± 2.05	99.9 ± 0.05	99.95 ± 0.05
3	90.5 ± 1.23	99.97 ± 0.03	99.98 ± 0.02	89.83 ± 3.18	97.11 ± 2.79	89.15 ± 2.99	99.98 ± 0.02
4	89.88 ± 2.18	98.9 ± 0.25	98.9 ± 0.53	98.77 ± 1.23	93.15 ± 2.58	96.88 ± 3.12	99.91 ± 0.09
5	96.66 ± 2.74	99.95 ± 0.05	99.97 ± 0.03	97.73 ± 2.27	92.36 ± 3.31	91.23 ± 2.38	99.97 ± 0.03
6	98.43 ± 0.74	83.86 ± 0.69	98.96 ± 0.45	99.23 ± 0.77	91.35 ± 2.65	89.94 ± 3.45	99.67 ± 0.29
7	96.77 ± 1.82	99.92 ± 0.08	99.92 ± 0.08	95.87 ± 2.94	88.7 ± 2.13	98.64 ± 1.36	99.82 ± 0.18
8	95.94 ± 2.99	98.87 ± 0.03	98.98 ± 0.89	93.41 ± 2.88	91.88 ± 2.26	91.52 ± 4.53	99.86 ± 0.14
9	99.16 ± 0.31	99.96 ± 0.04	99.96 ± 0.01	91.1 ± 2.82	88.74 ± 2.54	99.13 ± 0.87	99.93 ± 0.07
10	95.21 ± 2.26	98.17 ± 0.37	98.97 ± 0.13	95.68 ± 3.3	90.84 ± 3.94	95.66 ± 3.99	99.88 ± 0.12
11	95.76 ± 2.89	99.91 ± 0.09	99.91 ± 0.09	95.23 ± 3.36	87.8 ± 2.63	95.86 ± 3.67	99.71 ± 0.21
12	99.92 ± 0.03	98.81 ± 0.71	85.11 ± 0.57	92.01 ± 3.42	88.82 ± 3.4	94.27 ± 2.57	99.65 ± 0.09
13	95.96 ± 2.73	99.98 ± 0.02	99.97 ± 0.03	93.35 ± 3.48	89.21 ± 3.65	97.77 ± 2.21	100 ± 0
14	90.13 ± 1.08	98.89 ± 0.26	98.85 ± 0.91	88.23 ± 3.54	90.9 ± 3.11	96.31 ± 3.55	99.9 ± 0.05
15	96.18 ± 2.01	99.99 ± 0.01	99.93 ± 0.07	98.04 ± 1.96	88.92 ± 3.37	99.98 ± 0.01	99.13 ± 0.19
OA	96.36 ± 1.806	99.26 ± 0.74	98.92 ± 1.08	95.09 ± 2.015	91.88 ± 2.09	96.54 ± 2.72	100 ± 0
AA	95.66 ± 1.73	98.47 ± 0.18	98.62 ± 0.26	94.49 ± 2.71	91.02 ± 3.01	95.63 ± 2.44	99.82 ± 0.11
kappa	0.9592 ± 0.0044	0.9922 ± 0.0006	0.9883 ± 0.0003	0.9504 ± 0.0011	0.9184 ± 0.0104	0.9654 ± 0.0002	0.9995 ± 0.0005

interesting pattern for the plant disease classification based on the Plant Village dataset is the better classification performance of the CNN-based algorithms (e.g., EfficientNet-B3) than the vision transformers (e.g., ViT). The low amount of training data can explain the reason for some crop classes, such as healthy grape and healthy potato, where the vision transformer models showed a lower capability to recognize these classes than CNNs. This is because of the intrinsic issue of transformers that require a significant number of training data to reach their full potential of classification power. Moreover, several samples from the crop disease data benchmarks (i.e., Plant Village, Apple, and Bangladeshi datasets) generated by the developed LeafDisDiff algorithm are provided. These generated diseased crop samples include *potato late blight*, *pepper bell bacterial spot*, *tomato bacterial spot*, and *black rot* as seen in Fig. 6. The obtained classification results in all crop disease benchmarks illustrated the significance and contribution of the produced noisy crop disease samples in considerably improving the classification accuracy of the current cutting-edge deep learning algorithms.

5.3. Comparison of various machine learning classifiers using LeafDisDiff features

To use and illustrate the feature extraction capability of the LeafDisDiff model and compare the classification performance of various traditional machine learning classifiers, including K-NN, Naïve Bayes (NB), Decision Tree (DT), Random Forest (RF), ELM, and XGBoost (XGB) based on the extracted features, as seen in Table 8, we used the developed diffusion model for feature extraction and the traditional machine learning classifiers as plant disease classifiers. As seen in Table 8, the plant disease classification results based on the Apple Disease dataset showed the better classification performance of the tree-based algorithm of XGB over the other classifiers with an average accuracy (96.50% ± 3.34), overall accuracy (97.61% ± 2.99), and kappa index of (0.9908 ± 0.0295), using the extracted features resulting from the developed LeafDisDiff model. For example, regarding average accuracy, the XGB classifier considerably outperformed the other models, including ELM, DT, RF, NB, and K-NN, with about 3, 3.5, 4, 5, and 6 percentage points, respectively. The Apple disease classification results illustrated that the worst-performed algorithm using the dataset of the Apple disease was the K-NN algorithm with an average accuracy (90.42% ± 4.68), overall accuracy (93.27 ± 2.79), and kappa index of (0.9428 ± 0.0309) compared to other classification models.

The plant disease classification results based on the Bangladeshi Disease dataset showed the classification superiority of the XGB classifier compared to the other classification models with an average accuracy (96.43 ± 2.53), kappa index (0.976 ± 0.0387), and overall accuracy of (99.02 ± 3.94) using the extracted features resulting from the developed LeafDisDiff algorithm, as seen in Table 9. For instance, in terms of average accuracy, the XGB algorithm considerably enhanced the plant disease classification results of the models, including ELM, RF, K-NN, NB, and DT, with approximately 3, 3.5, 4, 4, and 5 percentage points, respectively. The least performed classification model based on the Bangladeshi Disease dataset was the algorithm of the DT with an average accuracy (91.38 ± 3.6), overall accuracy (95.04 ± 4.03), and kappa index of (0.940 ± 0.0415) over the other classifiers. Overall, based on the classification results of all three plant disease datasets (i.e., Plant Village, Apple, and Bangladeshi), the XGB model showed the best classification performance over the other traditional ml-based classification models, including the K-NN, NB, DT, RF, and ELM models. Moreover, high classification results of the XGB model proved the excellent usefulness of the extracted features from the developed LeafDisDiff model.

The results based on the Plant Village dataset illustrated the better classification performance of the tree-based XGB model with an average accuracy (95.13% ± 3.45), Kappa index (0.97 ± 0.0343), and overall accuracy of (97.99% ± 3.60) compared to other classifiers. For instance, using the extracted features resulting from the LeafDisDiff

Table 7
Results obtained with various deep learning models on the Plant Village dataset.

Class	Deep models				Transformers		LeafDisDiff
	CNN	ResNet-50	Efficient-B3	VGG19	ViT	CrossViT	
1	92.15 ± 0.76	99.84 ± 0.48	97.34 ± 0.53	94.69 ± 0.98	88.56 ± 4.38	95.25 ± 3.64	99.45 ± 0.12
2	94.86 ± 1.34	97.79 ± 0.36	99.76 ± 0.32	97.26 ± 1.20	93.34 ± 4.39	98.3 ± 3.24	99.52 ± 0.34
3	99.43 ± 0.65	94.70 ± 0.46	96.57 ± 0.51	98.37 ± 2.87	90.31 ± 2.99	93.31 ± 2.43	99.48 ± 0.56
4	95.20 ± 0.98	99.90 ± 0.81	98.22 ± 0.89	97.95 ± 1.56	90.24 ± 3.11	95.81 ± 4.65	99.50 ± 0.78
5	97.50 ± 1.87	99.86 ± 0.27	99.10 ± 0.24	90.19 ± 0.67	91.75 ± 4.06	97.99 ± 3.88	99.51 ± 0.90
6	97.49 ± 2.12	97.83 ± 0.82	99.90 ± 0.9	98.61 ± 2.34	95.05 ± 3.39	88.12 ± 3.47	99.47 ± 0.23
7	91.99 ± 0.45	94.66 ± 0.27	98.20 ± 0.24	88.49 ± 2.09	90.67 ± 3.28	99.5 ± 4.68	99.46 ± 0.45
8	94.56 ± 1.29	99.88 ± 0.48	99.32 ± 0.53	98.65 ± 2.76	97.85 ± 2.53	96.03 ± 3.29	99.49 ± 0.67
9	95.41 ± 2.54	89.91 ± 0.53	97.01 ± 0.58	87.10 ± 1.51	91.44 ± 3.57	94.86 ± 4.14	99.49 ± 0.89
10	94.44 ± 1.02	94.69 ± 0.30	98.88 ± 0.01	95.87 ± 2.49	90.39 ± 4.48	96 ± 2.55	99.48 ± 0.01
11	94.71 ± 1.93	99.92 ± 0.19	97.89 ± 0.17	93.76 ± 1.24	88.11 ± 2.44	87.08 ± 4.12	99.48 ± 0.13
12	88.83 ± 2.27	97.77 ± 0.66	99.43 ± 0.72	90.62 ± 2.95	91.38 ± 3.09	97.29 ± 2.27	99.50 ± 0.35
13	89.51 ± 0.51	94.64 ± 0.87	96.24 ± 0.96	92.48 ± 2.15	96.77 ± 3.19	99.85 ± 3.72	99.47 ± 0.57
14	96.04 ± 1.41	99.98 ± 0.71	98.11 ± 0.64	95.97 ± 2.65	94.31 ± 4.13	98.42 ± 4.47	99.49 ± 0.79
15	91.59 ± 2.73	97.75 ± 0.30	99.54 ± 0.33	87.28 ± 0.73	96.36 ± 3.1	91.19 ± 3.37	99.51 ± 0.91
16	94.88 ± 1.08	94.68 ± 0.42	99.02 ± 0.46	88.44 ± 2.23	93.57 ± 2.61	99.2 ± 4.59	99.48 ± 0.24
17	96.77 ± 2.01	99.94 ± 0.54	99.45 ± 0.59	91.4 ± 1.63	92.61 ± 4.62	97.66 ± 4.08	99.48 ± 0.46
18	87.27 ± 2.43	97.73 ± 0.34	99.35 ± 0.37	94.51 ± 0.81	89.86 ± 4.28	96.01 ± 4.58	99.50 ± 0.68
19	95.96 ± 0.59	94.71 ± 0.94	98.55 ± 1.03	97.29 ± 3.08	89.64 ± 3.36	96.19 ± 2.73	99.47 ± 0.8
20	97.60 ± 1.55	99.96 ± 0.14	99.21 ± 0.12	91.70 ± 1.77	95.41 ± 3.08	99.06 ± 3.87	99.48 ± 0.02
21	97.52 ± 2.86	97.71 ± 0.40	99.56 ± 0.44	98.95 ± 2.80	91.55 ± 4.54	99.21 ± 3.46	99.51 ± 0.14
22	87.50 ± 1.15	94.67 ± 0.01	99.13 ± 0.01	88.59 ± 0.93	98.42 ± 3.84	96.35 ± 3.81	99.49 ± 0.36
23	90.30 ± 2.09	99.90 ± 0.25	98.33 ± 0.28	91.45 ± 2.31	93 ± 4.26	95.86 ± 3.13	99.49 ± 0.58
24	96.71 ± 2.58	97.69 ± 0.43	99.65 ± 0.47	99.24 ± 1.37	88.17 ± 2.73	95.79 ± 2.73	99.47 ± 0.70
25	90.48 ± 0.67	94.65 ± 0.04	99.01 ± 0.01	95.53 ± 3.16	89.23 ± 3.42	98.82 ± 3.95	99.49 ± 0.92
26	94.20 ± 1.68	99.87 ± 0.81	97.78 ± 0.89	98.97 ± 0.96	95.17 ± 3.51	96.45 ± 2.72	99.50 ± 0.25
27	95.35 ± 2.94	97.97 ± 0.06	99.46 ± 0.01	96.96 ± 2.40	89.21 ± 2.89	98.56 ± 2.65	99.48 ± 0.47
28	96.30 ± 1.23	87.63 ± 0.02	98.44 ± 0.01	92.81 ± 1.90	94.94 ± 3.36	92.68 ± 4.32	99.49 ± 0.69
29	87.61 ± 2.18	99.85 ± 0.97	99.88 ± 1.07	88.16 ± 2.96	97.66 ± 4.55	96.18 ± 2.08	99.49 ± 0.81
30	96.82 ± 2.74	97.65 ± 0.76	99.67 ± 0.83	93.65 ± 1.00	92.25 ± 3.87	95.54 ± 4.56	99.47 ± 0.03
31	95.62 ± 0.74	89.62 ± 0.75	99.79 ± 0.82	89.08 ± 3.21	95.25 ± 2.69	90.62 ± 2.45	99.46 ± 0.15
32	93.33 ± 1.82	98.83 ± 0.78	98.66 ± 0.86	94.00 ± 1.45	95.99 ± 3.14	95.56 ± 4.35	99.50 ± 0.37
33	92.27 ± 2.99	97.63 ± 0.8	99.97 ± 0.88	98.27 ± 2.48	98.32 ± 3.12	90.84 ± 2.29	99.49 ± 0.59
34	99.13 ± 0.31	99.91 ± 0.63	97.12 ± 0.69	99.76 ± 1.53	97.64 ± 3.77	91.04 ± 3.01	99.41 ± 0.71
35	97.95 ± 2.26	98.61 ± 0.3	99.68 ± 0.33	91.08 ± 3.11	89.06 ± 4.04	88.28 ± 4.68	99.48 ± 0.93
36	87.27 ± 2.89	99.89 ± 0.91	99.77 ± 1.00	93.20 ± 1.17	90.16 ± 3.84	98.7 ± 2.53	99.49 ± 0.26
37	90.65 ± 0.82	97.59 ± 0.25	99.00 ± 0.28	97.18 ± 1.22	94.16 ± 4.13	93.82 ± 2.77	99.49 ± 0.48
38	96.81 ± 1.95	90.20 ± 0.62	98.23 ± 0.68	87.61 ± 0.90	87.46 ± 3.49	99.87 ± 4.45	99.48 ± 0.60
OA	97.1 ± 2.38	99.43 ± 3.05	99.67 ± 2.15	96.96 ± 2.21	94.19 ± 2.58	96.56 ± 2.88	100 ± 0.00
AA	94.00 ± 1.67	97.00 ± 0.49	98.80 ± 0.52	93.82 ± 1.91	90.31 ± 2.99	93.31 ± 2.43	99.49 ± 0.50
kappa	0.9582 ± 0.0211	0.9796 ± 0.0308	0.9925 ± 0.0057	0.9569 ± 0.0219	0.9319 ± 0.0296	0.9533 ± 0.0321	0.9996 ± 0.0042

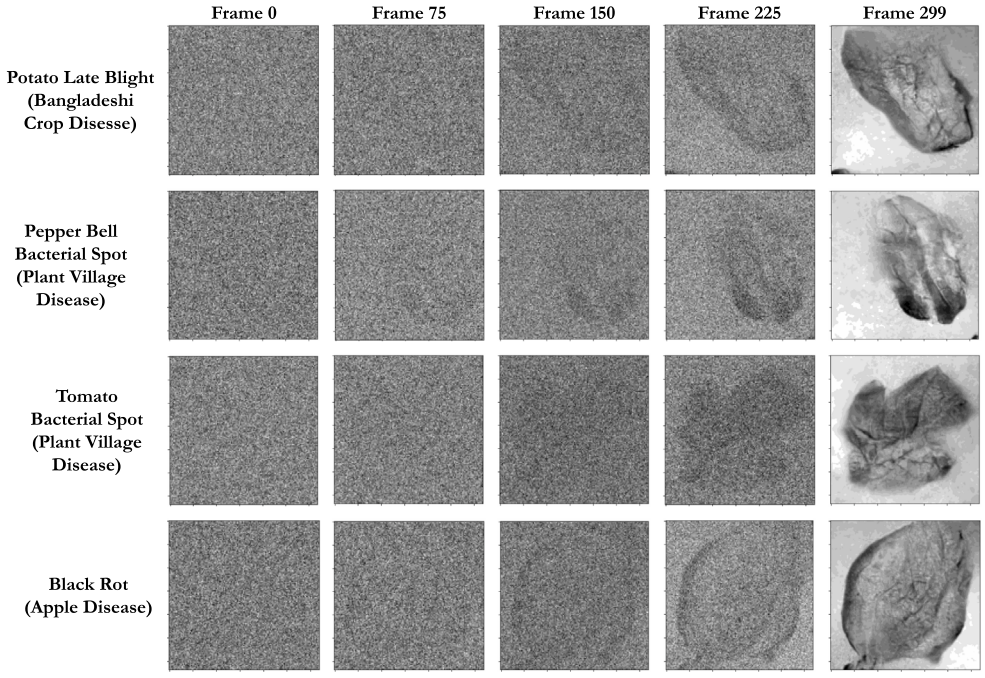


Fig. 6. Samples generated by the proposed LeafDisDiff model from randomly noisy images as examples.

Table 8
Classification results obtained with various machine learning classifiers on the Apple leaf disease dataset using LeafDisDiff features.

Classes	K-NN	Naïve Bayes	Decision Tree	Random Forest	ELM	XGBoost
1	88.84 ± 4.76	93.53 ± 3.97	95.37 ± 4.29	93.02 ± 4.51	91.38 ± 4.44	95.31 ± 2.48
2	95.36 ± 4.72	89.52 ± 3.93	90.56 ± 4.25	87.05 ± 4.47	94.12 ± 3.07	96.43 ± 4.57
3	87.75 ± 4.64	92.31 ± 3.85	92.78 ± 5.17	96.09 ± 4.39	95.51 ± 3.43	97.03 ± 3.03
4	89.74 ± 4.60	91.6 ± 3.81	93.61 ± 6.13	92.61 ± 4.35	91.79 ± 2.06	97.23 ± 3.28
OA	93.27 ± 2.79	94.21 ± 3.1	95.17 ± 3.08	94.53 ± 3.12	95.25 ± 3.17	97.61 ± 2.99
AA	90.42 ± 4.68	91.74 ± 3.89	93.08 ± 4.96	92.19 ± 4.43	93.2 ± 3.25	96.5 ± 3.34
Kappa	0.9428 ± 0.0309	0.9532 ± 0.029	0.9638 ± 0.0287	0.9567 ± 0.0276	0.9647 ± 0.0295	0.9908 ± 0.0295

algorithm, regarding average accuracy, the XGB model outperformed other classification algorithms, including ELM, RF, K-NN, NB, and DT, with approximately 2, 2.5, 3, 3, and 4 percentage points, respectively. The obtained plant disease classification results demonstrated that the worst-performed classifier was the tree-based DT model with an average accuracy (90.88 ± 3.53), kappa index (0.94 ± 0.0381), and overall accuracy of (94.64 ± 3.95) compared to other models, as seen in Table 10.

The classification performance of the deep learning models, namely CNN, ResNet-50, Efficient-B3, VGG19, ViT, and CrossViT, as presented in Tables 5, 6, and 7, surpasses that of the machine learning models listed in Tables 8, 9, and 10. The machine learning models in Tables 8, 9, and 10 include K-NN, Naïve Bayes, Decision Tree, Random Forest, ELM, and XGBoost. The results from these tables serve as a comparative analysis of classification performance between deep learning and machine learning-based algorithms.

5.4. Impact on classification performance of various ML and DL algorithms with different percentages of training samples

The effect of different percentages of training samples on the plant disease classification performance of the developed deep learning models, including CNN, ViT, Cross ViT, EfficientNet-B3, ResNet-50, VGG-19, and LeafDisDiff in terms of average accuracy, overall accuracy, and kappa index in all three data benchmarks (i.e., Plant Village, Apple, and Bangladeshi datasets) is shown in Fig. 7. Results illustrated the excellent capability of the developed LeafDisDiff algorithm in plant disease classification using even 5% of the reference data as training. For instance, in the Apple plant disease dataset, the classification performance of the LeafDisDiff increased by about 4%, 5%, and 5% in terms of average accuracy, overall accuracy, and kappa index, respectively, from 5% to 25% training ratios. In the Apple plant disease dataset, The CNN algorithm showed the most significant improvement in classification performance in terms of overall accuracy and kappa index by approximately 13% and 13% from 5% to 25% training ratios, as seen in Fig. 7. For the Bangladeshi plant disease data benchmark, the classification performance of the LeafDisDiff increased by about 3%, 3%, and 3% in terms of overall accuracy, average accuracy, and kappa index, respectively, from 5% to 25% training ratios, presenting its excellent efficiency and capability in plant classification task even with few training data. The obtained results revealed that the classification accuracy of the LeafDisDiff model using only a 5% training ratio was higher than that of ViT, Cross ViT, CNN, and VGG-19 algorithms utilizing a 25% training ratio. Moreover, in the Plant Village data benchmark, the classification performance of the LeafDisDiff enhanced by about 3%, 3%, and 6% in terms of overall accuracy, average accuracy, and kappa index, respectively, from 5% to 25% training ratios. Similar to the Bangladeshi plant disease dataset, the classification accuracy of the LeafDisDiff algorithm with only a 5% training ratio was superior over the ViT, Cross ViT, CNN, and VGG-19 algorithms using a 25% training ratio. Overall, the LeafDisDiff classification model showed a high level of plant disease classification performance using only a few training data (5% of the reference data), and the plant classification results were consistently high from 5% to 25% training ratios, as seen in Fig. 7.

The effect of different percentages of training samples on the plant disease classification performance of the traditional machine learning classifiers, including K-NN, NB, DT, RF, ELM, and XGB, in terms of average accuracy, overall accuracy, and kappa index in all three data benchmarks (i.e., Plant Village, Apple, and Bangladeshi datasets) is presented in Fig. 8. As discussed, these machine learning models are trained on the extracted features resulting from the LeafDisDiff model. The obtained plant disease classification results showed the superiority of the XGB algorithm compared to other classifiers in all three plant disease benchmarks. In the Apple plant disease dataset, the classification results of the XGB algorithm improved by approximately 5%, 7%, and 7%, respectively, in terms of average accuracy, overall accuracy, and kappa index from 5% to 25% training ratios. The XGB model illustrated much higher classification accuracy in the Bangladeshi data benchmark than the other developed traditional algorithms. For instance, the classification accuracy of the XGB using only a 5% training ratio was better than that of K-NN, NB, DT, RF, and ELM algorithms utilizing a 25% training ratio. Moreover, in the Plant Village dataset, the classification performance of the XGB classifier enhanced by approximately 5%, 7%, and 8%, respectively, in terms of average accuracy, kappa index, and overall accuracy, from 5% to 25% training ratios, as seen in Fig. 8.

5.5. GradCam visualization

GradCAM [59] visualization of several randomly selected leaf samples over the three benchmark datasets (i.e., Apple, Bangladeshi, and PlantVillage datasets) are presented in Fig. 9. Grad-CAM creates a coarse map of localization that highlights the most critical areas of the image for the plant disease prediction/classification by using the gradients of any given target (diseased leaf) flowing into the last convolutional layer. Grad-CAM allows us to visually confirm the locations of our network's eyes, ensuring that it focuses on the right relationships in the image and activating surrounding them. In other words, GradCam Visualization helps us understand where the diffusion-based LeafDisDiff algorithm is looking to recognize various plant diseases. For instance, Fig. 9(a) illustrates the inputs image (i.e., raw plant disease image) for *apple scab* disease where Fig. 9(b) presents the region highlighted by the GradCAM visualization function that the developed LeafDisDiff algorithm is focused to correctly detect that specific apple disease (i.e., $applescab$).

5.6. Probability outcomes

Probability Outcomes of several randomly selected leaf samples in Apple, Plant Village, and Bangladeshi data benchmarks are shown in Fig. 10. For example, Fig. 10(a) and (b) present the input images (i.e., raw diseased plant images) and the probability outputs of the plant diseases resulting from the diffusion-based algorithm of the LeafDisDiff for the Apple disease data benchmark. As seen in Fig. 10(a), the highest probability (higher than 80% probability) for the input image resulting from the LeafDisDiff model is for class 1 (i.e., apple scab disease), while as shown in Fig. 10(b), the highest probability (higher than 90% probability) for the input image resulting from the LeafDisDiff algorithm is for class 2 (i.e., apple black rot disease), illustrating correct apple disease classification outcome by the developed model.

Table 9
Classification results obtained with various machine learning classifiers on the Bangladeshi Crops disease dataset using LeafDisDiff features.

Classes	K-NN	Naïve Bayes	Decision Tree	Random Forest	ELM	XGBoost
1	88.99 ± 4.91	89.21 ± 3.92	91.00 ± 3.68	97.18 ± 2.82	98.04 ± 1.96	97.93 ± 2.07
2	95.15 ± 4.47	94.08 ± 4.56	93.24 ± 4.21	94.06 ± 5.94	92.38 ± 3.56	97.57 ± 2.43
3	87.76 ± 6.19	99.95 ± 0.05	87.01 ± 3.86	89.92 ± 6.2	93.8 ± 4.67	96.35 ± 3.65
4	90.51 ± 7.76	88.62 ± 6.92	88.43 ± 4.03	98.38 ± 1.62	87.82 ± 2.8	98.85 ± 1.15
5	87.33 ± 6.11	96.49 ± 3.51	88.73 ± 3.96	96.74 ± 2.86	88.03 ± 2.18	97.52 ± 2.02
6	90.75 ± 2.4	90.8 ± 4.94	92.11 ± 3.14	93.94 ± 5.74	97.89 ± 2.11	97.92 ± 2.08
7	96.01 ± 3.99	91.42 ± 4.78	92.64 ± 4.02	95.34 ± 3.93	93.81 ± 3.18	97.93 ± 2.07
8	88.03 ± 5.06	94.74 ± 3.96	88.23 ± 4.08	87.76 ± 5.75	93.15 ± 3.05	98.53 ± 1.47
9	94.02 ± 5.69	98.75 ± 1.25	93.80 ± 3.12	92.29 ± 4.68	97.56 ± 2.44	94.44 ± 4.51
10	97.82 ± 2.18	98.15 ± 1.85	92.73 ± 3.3	97.3 ± 2.7	90.17 ± 3.4	97.61 ± 2.39
11	98.57 ± 1.43	93.2 ± 2.51	88.76 ± 3.02	87.07 ± 3.69	91.43 ± 3.75	97.97 ± 2.03
12	88.11 ± 7.28	87.51 ± 5.11	92.36 ± 4.09	98 ± 2	95.51 ± 3.17	97.98 ± 2.02
13	90.26 ± 6.36	89.34 ± 6.22	90.65 ± 3.29	90.88 ± 4.21	91.12 ± 4	95.16 ± 3.98
14	96.1 ± 3.9	91.77 ± 6.48	96.83 ± 3.17	89.05 ± 6.79	90.25 ± 4.19	87.14 ± 3.58
15	92.38 ± 5.37	88.14 ± 3.2	94.17 ± 3.08	87.82 ± 7.54	96.89 ± 2.46	93.54 ± 2.51
OA	95.62 ± 4.17	96.16 ± 4.34	95.04 ± 4.03	96.35 ± 3.96	96.46 ± 4.78	99.02 ± 3.94
AA	92.12 ± 4.87	92.81 ± 3.95	91.38 ± 3.6	93.05 ± 4.43	93.19 ± 3.13	96.43 ± 2.53
Kappa	0.945 ± 0.0392	0.950 ± 0.0443	0.940 ± 0.0415	0.951 ± 0.0412	0.952 ± 0.0413	0.976 ± 0.0387

Table 10
Classification results obtained with various machine learning classifiers on the Plant Village dataset using LeafDisDiff features.

Classes	K-NN	Naïve Bayes	Decision Tree	Random Forest	ELM	XGBoost
1	92.87 ± 4.30	91.14 ± 3.39	92.05 ± 3.57	99.54 ± 3.90	87.52 ± 3.15	95.58 ± 3.63
2	87.45 ± 4.26	97.38 ± 3.35	88.49 ± 3.53	89.08 ± 3.86	92.22 ± 3.08	94.85 ± 4.12
3	87.09 ± 4.18	89.93 ± 3.27	87.87 ± 3.45	87.78 ± 3.78	93.52 ± 4.39	96.74 ± 2.72
4	88.98 ± 4.14	88.77 ± 3.23	88.58 ± 3.41	91.74 ± 3.74	97.18 ± 3.63	97.78 ± 3.28
5	99.05 ± 4.34	99.57 ± 3.43	92.67 ± 3.61	96.44 ± 3.94	90.84 ± 2.71	91.84 ± 2.11
6	88.05 ± 4.38	98.21 ± 3.47	94.79 ± 3.65	91.67 ± 3.98	94.45 ± 3.63	97.48 ± 4.68
7	87.94 ± 4.10	97.02 ± 3.19	93.15 ± 3.37	94.34 ± 3.70	94.49 ± 4.41	96.97 ± 2.05
8	98.32 ± 4.06	88.80 ± 3.15	98.62 ± 3.33	87.23 ± 3.66	90.95 ± 3.37	95.71 ± 4.44
9	87.75 ± 4.42	93.86 ± 3.51	91.57 ± 3.69	97.91 ± 4.02	90.17 ± 3.93	94.79 ± 2.91
10	94.25 ± 4.46	99.31 ± 3.55	92.19 ± 3.73	88.24 ± 4.06	98.08 ± 2.09	97.1 ± 4.7
11	91.33 ± 4.02	93.95 ± 3.11	89.03 ± 3.29	95.78 ± 3.62	98.72 ± 3.16	96.85 ± 4.28
12	90.78 ± 3.98	90.01 ± 3.07	88.89 ± 3.25	97.63 ± 3.58	96.46 ± 4.19	97.94 ± 4.44
13	89.30 ± 4.50	89.74 ± 3.59	90.64 ± 3.77	91.91 ± 4.10	96.16 ± 3.22	96 ± 3.27
14	94.04 ± 4.54	87.55 ± 3.63	89.72 ± 3.81	88.25 ± 4.14	98.03 ± 3.52	97.35 ± 2.87
15	94.22 ± 3.94	87.94 ± 3.03	94.28 ± 3.21	88.87 ± 3.54	93.3 ± 2.56	97.45 ± 3.97
16	89.03 ± 3.90	90.40 ± 2.99	90.89 ± 3.17	89.15 ± 3.50	87.96 ± 2.27	93.57 ± 2.98
17	87.24 ± 4.58	87.24 ± 3.67	87.70 ± 3.85	88.05 ± 4.18	98.56 ± 3.82	97.43 ± 2.06
18	92.60 ± 4.62	91.60 ± 3.71	91.58 ± 3.89	99.22 ± 4.22	87.58 ± 3.03	92.29 ± 4.05
19	89.66 ± 3.86	88.37 ± 2.95	88.28 ± 3.13	94.13 ± 3.46	94.62 ± 3.05	96.41 ± 4.26
20	88.01 ± 3.82	94.02 ± 2.91	87.79 ± 3.09	89.23 ± 3.42	88.85 ± 3.46	90.21 ± 3.08
21	98.35 ± 4.66	92.15 ± 3.75	87.30 ± 3.93	92.76 ± 4.26	87.93 ± 2.11	90.43 ± 3.55
22	98.45 ± 4.70	96.78 ± 3.79	95.04 ± 3.97	90.31 ± 4.30	92.62 ± 2.9	90.45 ± 3.83
23	87.11 ± 3.78	89.06 ± 2.87	91.12 ± 3.05	91.32 ± 3.38	88.41 ± 4.11	90.29 ± 3.52
24	99.92 ± 3.74	92.46 ± 2.83	88.30 ± 3.01	99.62 ± 3.34	93.62 ± 3.54	89.3 ± 3.8
25	97.47 ± 4.74	91.38 ± 3.83	88.05 ± 4.01	93.32 ± 4.34	88.17 ± 4.27	93.54 ± 2.5
26	91.53 ± 4.78	93.67 ± 3.87	90.34 ± 4.05	87.86 ± 4.38	96.16 ± 4.54	98.4 ± 3.01
27	97.89 ± 3.70	95.71 ± 2.79	94.63 ± 2.97	94.77 ± 3.30	94.95 ± 3.32	87.2 ± 3.37
28	98.12 ± 3.66	90.16 ± 2.75	90.20 ± 2.93	89.53 ± 3.26	92.53 ± 2.24	96.62 ± 2.91
29	95.89 ± 4.82	88.64 ± 3.91	92.53 ± 4.09	87.54 ± 4.42	87.46 ± 2.96	95.53 ± 4.58
30	91.05 ± 4.86	96.42 ± 3.95	93.74 ± 4.13	94.11 ± 4.46	90.27 ± 3.9	97.53 ± 3.19
31	90.89 ± 3.62	99.76 ± 2.71	87.24 ± 2.89	99.26 ± 3.22	97.45 ± 3.55	97.68 ± 4.38
32	88.34 ± 3.58	87.35 ± 2.67	92.39 ± 2.85	91.57 ± 3.18	97.1 ± 2.85	92.6 ± 3.29
33	90.85 ± 4.90	88.28 ± 3.99	97.91 ± 4.17	98.33 ± 4.50	97.97 ± 4	95.13 ± 3.26
34	90.47 ± 4.94	89.30 ± 4.03	95.38 ± 4.21	91.45 ± 4.54	96.34 ± 4.07	98.55 ± 3.24
35	87.63 ± 3.54	92.33 ± 2.63	87.88 ± 2.81	90.49 ± 3.14	90.24 ± 4.28	92.73 ± 3.03
36	93.42 ± 3.50	88.47 ± 2.59	87.45 ± 2.77	94.65 ± 3.10	95.04 ± 4.15	97.55 ± 2.85
37	89.29 ± 4.98	91.66 ± 4.07	87.27 ± 4.25	89.44 ± 4.58	93.61 ± 4.23	98.57 ± 4.75
38	97.81 ± 5.02	87.23 ± 4.11	87.92 ± 4.29	98.24 ± 4.62	94.31 ± 3.55	98.49 ± 2.08
OA	95.66 ± 3.57	95.52 ± 4.00	94.64 ± 3.95	96.04 ± 3.91	96.52 ± 4.08	97.99 ± 3.60
AA	92.17 ± 4.26	91.99 ± 3.35	90.88 ± 3.53	92.65 ± 3.86	93.26 ± 3.45	95.13 ± 3.45
kappa	0.95 ± 0.0367	0.94 ± 0.0403	0.94 ± 0.0381	0.95 ± 0.0350	0.95 ± 0.0377	0.97 ± 0.0343

6. Conclusion

This work introduces a novel approach for detecting and classifying plant leaf diseases. Our methodology employs a diffusion-based deep learning technique named LeafDisDiff, featuring a U-Net-style architecture with position embeddings, group normalization, attention blocks, and integrated residual blocks. Unlike conventional methods that directly input batches of images into the learning process for label prediction, our approach emphasizes a denoising technique that predicts noise at each time step. Our diffusion model takes a sophisticated approach by considering image batches and their corresponding labels, introducing noise as an additional input. This methodology excels in accurate classification tasks and noise reduction, thereby enhancing the overall quality of classification results. Quantitative and qualitative evaluations on the Apple disease dataset, Bangladeshi Crops disease dataset, and PlantVillage dataset demonstrate that our proposed

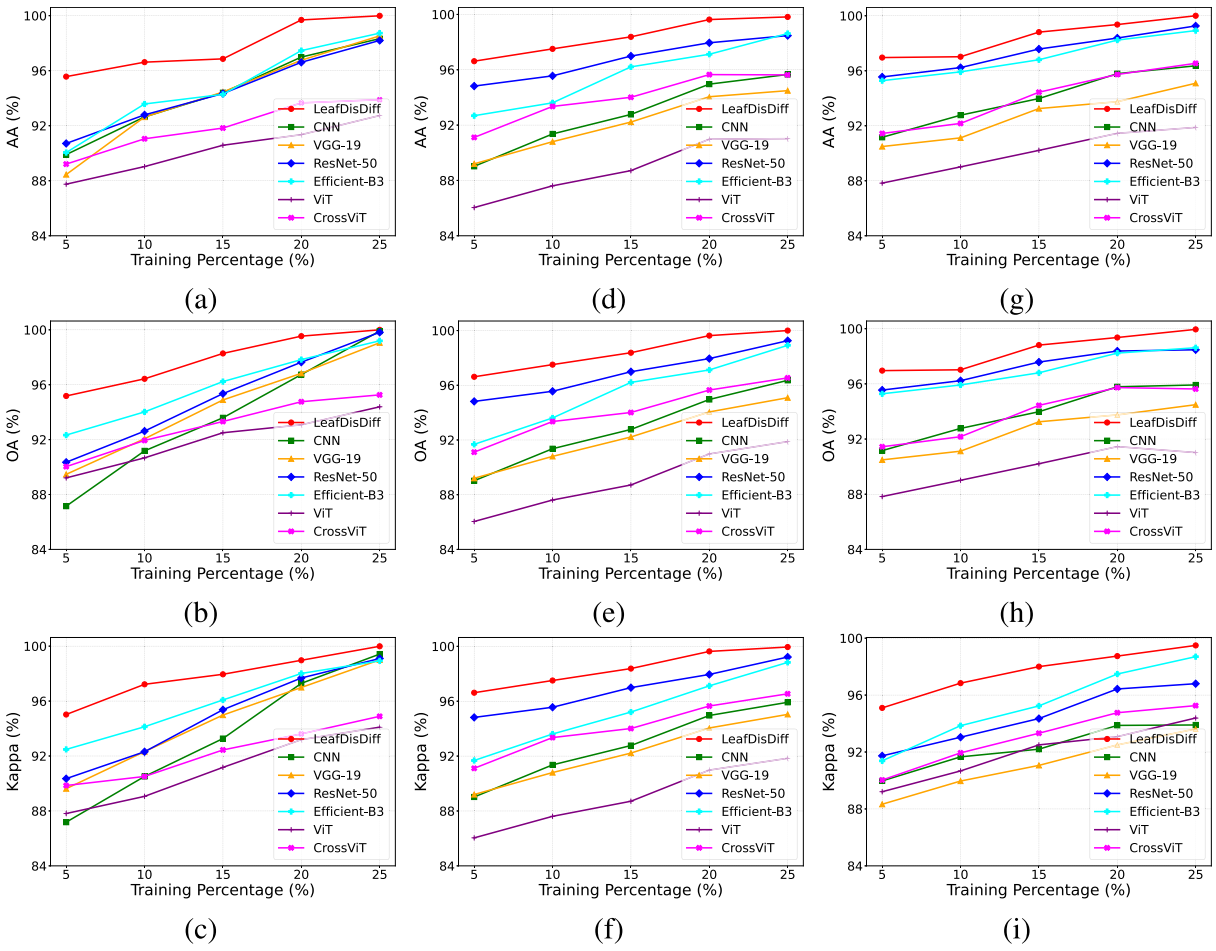


Fig. 7. Classification accuracies achieved by various deep learning techniques, including the proposed LeafDisDiff, in terms of AA, OA, and kappa (κ) using different percentages of training samples randomly selected from: (a)–(c) Apple leaf disease, (d)–(f) Bangladeshi crop disease, and (g)–(i) Plant Villages datasets, respectively.

approach achieves significant improvements, leading to state-of-the-art performance.

6.1. Future prospects and limitations

Employing the diffusion-based deep learning technique, LeafDisDiff, to plant disease classification, showcasing significant improvements in classification accuracy. Using diverse data benchmarks, including Plant Village, Bangladeshi Crop Disease, and Apple Disease, ensures robust generalization across varying real-world agricultural conditions. Despite progress, challenges like background elements obscuring symptoms, inconsistent capture conditions, and symptoms lacking distinct boundaries persist. Our approach effectively addresses these issues. The U-Net style architecture with attention and residual blocks isolates key features, aiding in complex situations and accurately differentiating overlapping disease patterns. The diverse images used ensure generalization across different situations and lighting conditions. While LeafDisDiff shows substantial improvements, further enhancements can elevate performance. Integrating sophisticated decoders, like conditional autoregressive models, can better capture complex dependencies. Exploring alternative scaling methods and a multi-resolution approach will preserve more information during normalization, leading to more accurate classification across various datasets. Future work involves incorporating multimodal data to enhance the model's diagnostic capabilities. Enhancing localization within the model will support precise disease detection in diverse environments and conditions, aiding agricultural platforms in effectively managing plant health.

CRediT authorship contribution statement

Aryan Das: Visualization, Software, Methodology, Data curation, Conceptualization. **Rajul Mahto:** Writing – original draft, Visualization, Software, Resources, Methodology, Investigation, Data curation, Conceptualization. **Wentao Wang:** Data curation, Investigation, Resources, Visualization, Writing – original draft. **Ali Jamali:** Writing – original draft, Validation. **Pravendra Singh:** Writing – review & editing, Validation. **Swalpa Kumar Roy:** Writing – review & editing, Methodology, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research is partly supported by SERB, Govt. of India under Project Grant No. SRG/2022/001390.

Data availability

Data will be made available on request.

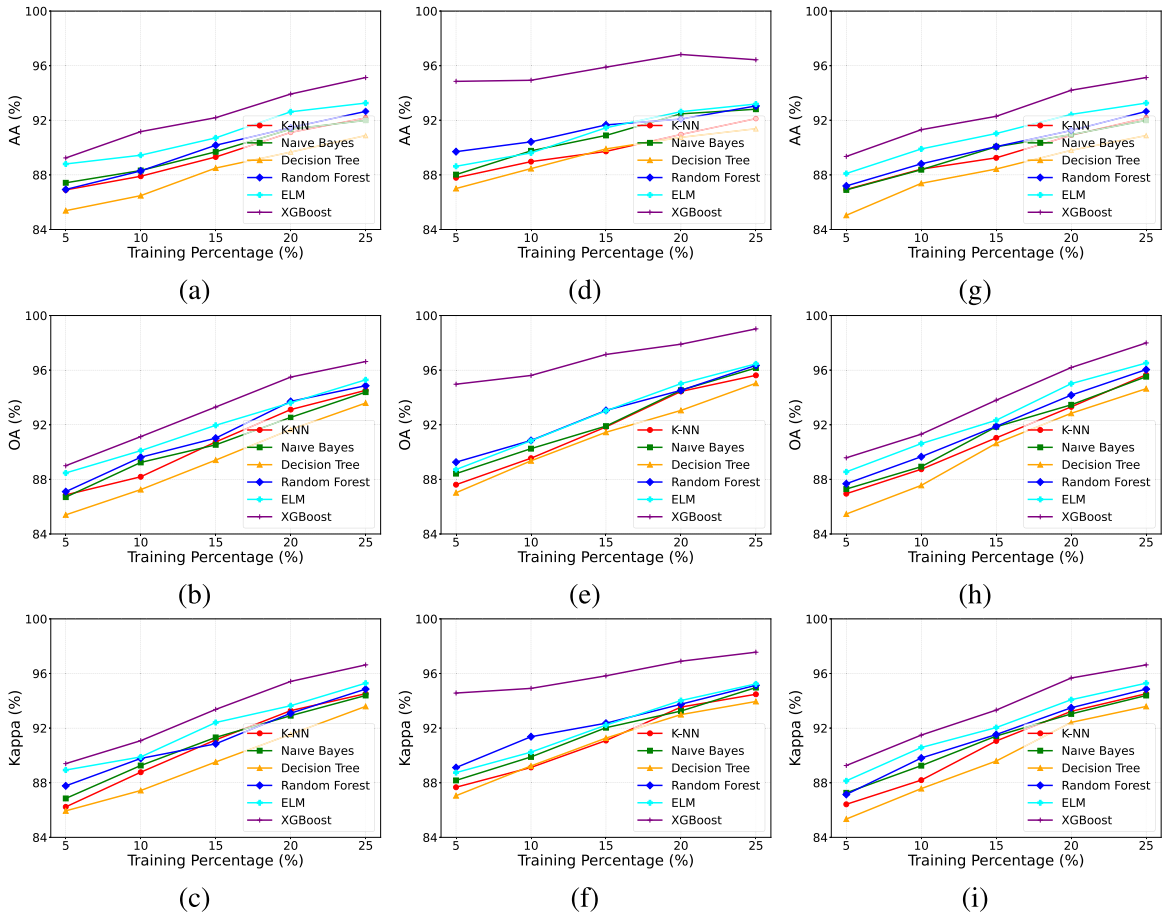


Fig. 8. Classification accuracies achieved using various machine learning classifiers in terms of AA, OA, and kappa (κ) with different percentages of training samples randomly selected from (a)–(c) Apple leaf disease, (d)–(f) Bangladeshi crop disease, and (g)–(i) Plant Villages datasets, respectively.

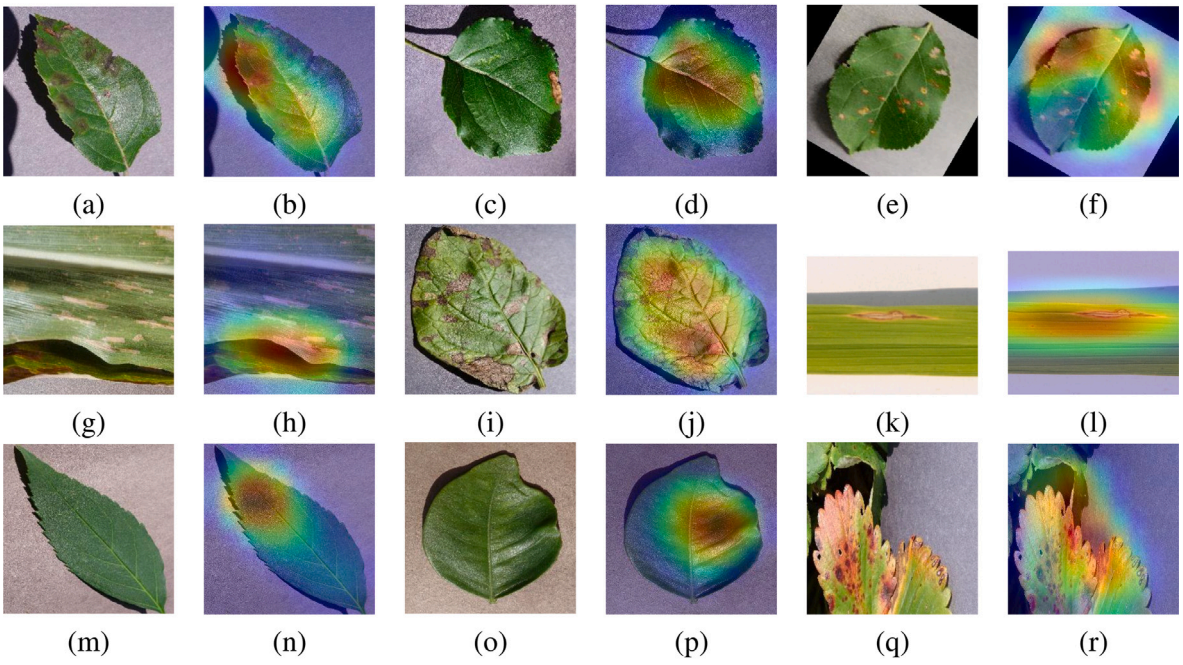


Fig. 9. GradCAM Visualization of LeafDisDiff Model for Sample Leaf Images.

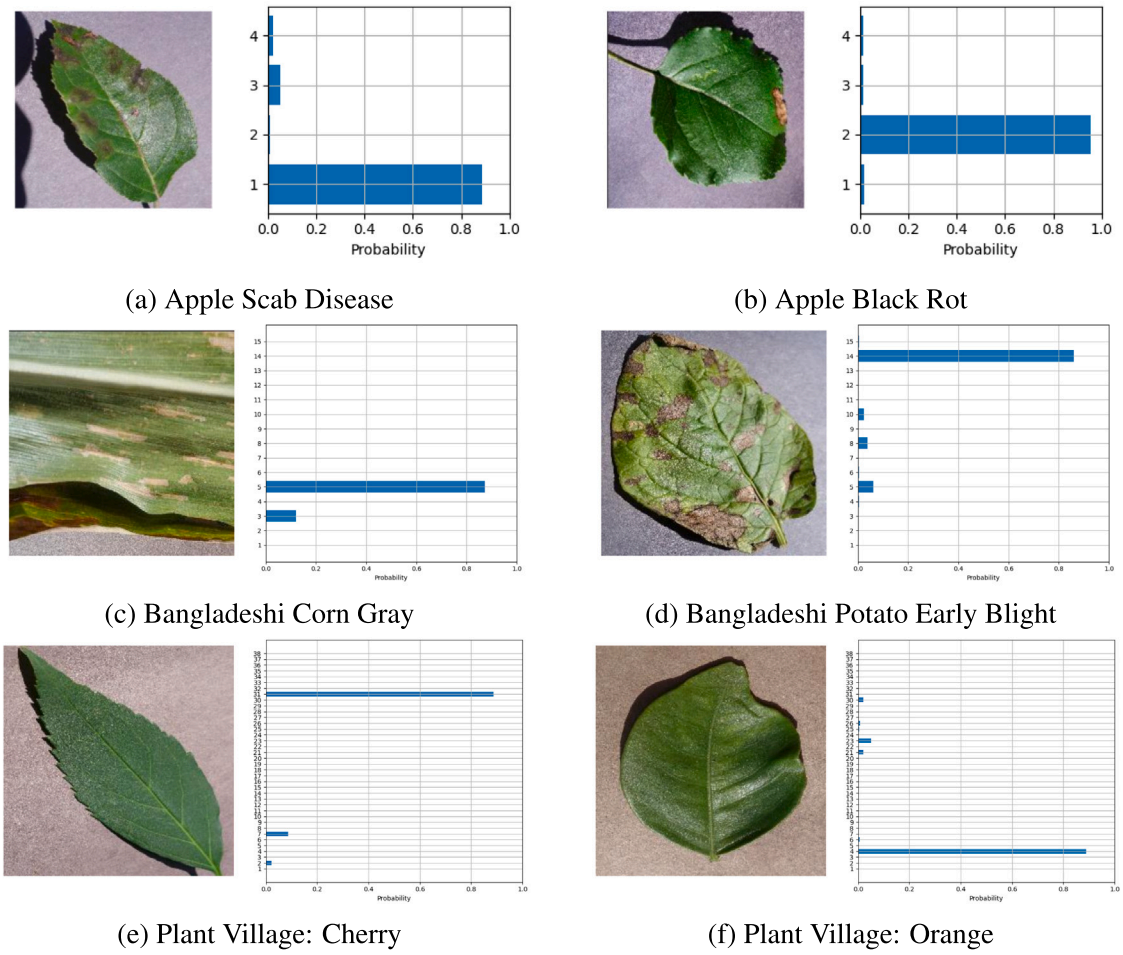


Fig. 10. Probability Outputs of Leaf Samples from Various Datasets.

References

[1] S.D. Khirade, A. Patil, Plant disease detection using image processing, in: 2015 International Conference on Computing Communication Control and Automation, 2015, pp. 768–771, http://dx.doi.org/10.1109/ICCUBEA.2015.153.

[2] L.W. Kuswidiyanto, H.-H. Noh, X. Han, Plant disease diagnosis using deep learning based on aerial hyperspectral images: A review, Remote. Sens. 14 (23) (2022) 6031.

[3] A. Lowe, N. Harrison, A.P. French, Hyperspectral image analysis techniques for the detection and classification of the early onset of plant disease and stress, Plant Methods 13 (1) (2017) 80.

[4] D.I. Patrício, R. Rieder, Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review, Comput. Electron. Agric. 153 (2018) 69–81.

[5] H. Orchi, M. Sadik, M. Khaldoun, On using artificial intelligence and the internet of things for crop disease detection: A contemporary survey, Agriculture 12 (1) (2021) 9.

[6] L.C. Ngugi, M. Abelwahab, M. Abo-Zahhad, Recent advances in image processing techniques for automated leaf pest and disease recognition–A review, Inf. Process. Agric. 8 (1) (2021) 27–51.

[7] C. Li, T. Zhen, Z. Li, Image classification of pests with residual neural network based on transfer learning, Appl. Sci. 12 (9) (2022) 4356.

[8] H. Tian, T. Wang, Y. Liu, X. Qiao, Y. Li, Computer vision technology in agricultural automation—A review, Inf. Process. Agric. 7 (1) (2020) 1–19.

[9] H. Yin, W. Yi, D. Hu, Computer vision and machine learning applied in the mushroom industry: A critical review, Comput. Electron. Agric. 198 (2022) 107015.

[10] L. Alzubaidi, J. Bai, A. Al-Sabaawi, J. Santamaría, A.S. Albahri, B.S.N. Al-dabbagh, M.A. Fadhel, M. Manoufali, J. Zhang, A.H. Al-Timemy, et al., A survey on deep learning tools dealing with data scarcity: definitions, challenges, solutions, tips, and applications, J. Big Data 10 (1) (2023) 46.

[11] C. Sarkar, D. Gupta, U. Gupta, B.B. Hazarika, Leaf disease detection using machine learning and deep learning: Review and challenges, Appl. Soft Comput. (2023) 110534.

[12] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, in: Advances in Neural Information Processing Systems, vol. 33, Curran Associates, Inc., 2020, pp. 6840–6851.

[13] H. Wang, J. Cao, R.M. Anwer, J. Xie, F.S. Khan, Y. Pang, DFormer: Diffusion-guided transformer for universal image segmentation, 2023, arXiv preprint arXiv:2306.03437.

[14] F. Tang, J. Ding, L. Wang, M. Xian, C. Ning, Multi-level global context cross consistency model for semi-supervised ultrasound image segmentation with diffusion model, 2023, arXiv preprint arXiv:2305.09447.

[15] Z. Chen, K. Sun, X. Lin, R. Ji, CamoDiffusion: Camouflaged object detection via conditional diffusion models, 2023, arXiv preprint arXiv:2305.17932.

[16] I.E. Hamamci, S. Er, E. Simsar, A. Sekuboyina, M. Gundogar, B. Stadlinger, A. Mehl, B. Menze, Diffusion-based hierarchical multi-label object detection to analyze panoramic dental X-rays, 2023, arXiv preprint arXiv:2303.06500.

[17] W.B. Demilie, Plant disease detection and classification techniques: a comparative study of the performances, J. Big Data 11 (1) (2024) 5.

[18] A. Khulbe, et al., Insect-pests and diseases in greenhouse cultivation and their biological control, in: Protected Cultivation, Apple Academic Press, 2024, pp. 217–254.

[19] P. Sajitha, et al., A review on machine learning and deep learning image-based plant disease classification for industrial farming systems, J. Ind. Inf. Integr. 38 (2024) 100572.

[20] Y. Sari, et al., Classification of chili leaf disease using the gray level co-occurrence matrix (GLCM) and the support vector machine (SVM) methods, in: 2021 Sixth International Conference on Informatics and Computing, ICIC, 2021, pp. 1–4.

[21] A. Aruraj, et al., Detection and classification of diseases of banana plant using local binary pattern and support vector machine, in: 2019 2nd International Conference on Signal Processing and Communication, ICSPC, 2019, pp. 231–235.

[22] S.B. Jagtap, et al., Agricultural plant leaf disease detection and diagnosis using image processing based on morphological feature extraction, IOSR J. VLSI Signal Process. 4 (2014) 24–30.

[23] L.S. Pinto, et al., Crop disease classification using texture analysis, in: 2016 IEEE International Conference on Recent Trends in Electronics,Information & Communication Technology, RTEICT, 2016, pp. 825–828.

[24] M. Keivani, et al., Automated analysis of leaf shape, texture, and color features for plant classification, Trait. Du Signal 37 (1) (2020) 17–28.
[25] R. Archana, et al., Deep learning models for digital image processing: a review, Artif. Intell. Rev. 57 (1) (2024) 11.
[26] X. Chi, J. Liu, M. Lu, R. Zhang, Z. Wang, Y. Guo, S. Zhang, BEV-SAN: Accurate BEV 3D object detection via slice attention networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 17461–17470.
[27] Y. Man, L.-Y. Gui, Y.-X. Wang, BEV-guided multi-modality fusion for driving perception, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 21960–21969.
[28] K. He, C. Gan, Z. Li, I. Rekik, Z. Yin, W. Ji, Y. Gao, Q. Wang, J. Zhang, D. Shen, Transformers in medical image analysis, Intell. Med. 3 (1) (2023) 59–78.
[29] H. Zhao, T. Wang, S. Liu, X. Xie, X. Zhou, Z. Hou, L. Jiao, Y. Ma, Y. Li, J. Luo, et al., An effective morphological analysis framework of intracranial artery in 3D digital subtraction angiography, in: International Conference on Neural Information Processing, Springer, 2023, pp. 50–61.
[30] M.E. Paoletti, X. Tao, L. Han, Z. Wu, S. Moreno-Álvarez, S.K. Roy, A. Plaza, J.M. Haut, Parameter-free attention network for spectral–spatial hyperspectral image classification, IEEE Trans. Geosci. Remote Sens. 61 (2023) 1–17.
[31] S.K. Roy, A. Deria, D. Hong, B. Rasti, A. Plaza, J. Chanussot, Multimodal fusion transformer for remote sensing image classification, IEEE Trans. Geosci. Remote Sens. 61 (2023) 1–20.
[32] S. Dargan, et al., A survey of deep learning and its applications: a new paradigm to machine learning, Arch. Comput. Methods Eng. 27 (2020) 1071–1092.
[33] T. Kattenborn, J. Leitloff, F. Schiefer, S. Hinz, Review on Convolutional Neural Networks (CNN) in vegetation remote sensing, ISPRS J. Photogramm. Remote Sens. 173 (2021) 24–49, http://dx.doi.org/10.1016/j.isprsjprs.2020.12.010.
[34] Z. Li, F. Liu, W. Yang, S. Peng, J. Zhou, A survey of convolutional neural networks: Analysis, applications, and prospects, IEEE Trans. Neural Netw. Learn. Syst. 33 (12) (2022) 6999–7019, http://dx.doi.org/10.1109/TNNLS.2021.3084827.
[35] Z. Lu, Y. Bian, T. Yang, Q. Ge, Y. Wang, A new siamese heterogeneous convolutional neural networks based on attention mechanism and feature pyramid, IEEE Trans. Cybern. (2023) 1–12, http://dx.doi.org/10.1109/TCYB.2022.3207431.
[36] J. Zhu, H. Li, Z. Rao, H. Ji, Identification of slightly sprouted wheat kernels using hyperspectral imaging technology and different deep convolutional neural networks, Food Control 143 (2023) 109291, http://dx.doi.org/10.1016/j.foodcont.2022.109291.
[37] D. Keerthana, V. Venugopal, M.K. Nath, M. Mishra, Hybrid convolutional neural networks with SVM classifier for classification of skin cancer, Biomed. Eng. Adv. 5 (2023) 100069, http://dx.doi.org/10.1016/j.bea.2022.100069.
[38] D.S. Joseph, P.M. Pawar, R. Pramanik, Intelligent plant disease diagnosis using convolutional neural network: A review, Multimedia Tools Appl. 82 (14) (2023) 21415–21481.
[39] T. Shi, Y. Liu, X. Zheng, K. Hu, H. Huang, H. Liu, H. Huang, Recent advances in plant disease severity assessment using convolutional neural networks, Sci. Rep. 13 (1) (2023) 2336.
[40] M.A.B. Bhuiyan, H.M. Abdullah, S.E. Arman, S. Saminur Rahman, K. Al Mahmud, BananaSqueezeNet: A very fast, lightweight convolutional neural network for the diagnosis of three prominent banana leaf diseases, Smart Agric. Technol. 4 (2023) 100214, http://dx.doi.org/10.1016/j.atech.2023.100214.
[41] A. Beikmohammadi, K. Faez, A. Motallebi, SWP-LeafNET: A novel multistage approach for plant leaf identification based on deep CNN, Expert Syst. Appl. 202 (2022) 117470.
[42] D.S. Rao, R.B. Ch, V.S. Kiran, N. Rajasekhar, K. Srinivas, P.S. Akshay, S. Mohan, B.L. Bharadwaj, Plant disease classification using deep bilinear CNN, Intell. Autom. Soft Comput. 31 (1) (2022) 161–176.
[43] S.K. Upadhyay, A. Kumar, A novel approach for rice plant diseases classification with deep convolutional neural network, Int. J. Inf. Technol. (2022) 1–15.
[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[45] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, et al., A survey on vision transformer, IEEE Trans. Pattern Anal. Mach. Intell. 45 (1) (2022) 87–110.
[46] S. Chaudhari, V. Mithal, G. Polatkan, R. Ramanath, An attentive survey of attention models, ACM Trans. Intell. Syst. Technol. (TIST) 12 (5) (2021) 1–32.
[47] Q. Zeng, L. Niu, S. Wang, W. Ni, SEViT: a large-scale and fine-grained plant disease classification model based on transformer and attention convolution, Multimedia Syst. 29 (3) (2023) 1001–1010.
[48] P.S. Thakur, P. Khanna, T. Sheorey, A. Ojha, Explainable vision transformer enabled convolutional neural network for plant disease identification: PlantXViT, 2022, arXiv preprint arXiv:2207.07919.
[49] S. Yu, L. Xie, Q. Huang, Inception convolutional vision transformers for plant disease identification, Internet Things 21 (2023) 100650.
[50] P. Dhariwal, A. Nichol, Diffusion models beat gans on image synthesis, Adv. Neural Inf. Process. Syst. 34 (2021) 8780–8794.
[51] L. Yang, et al., Diffusion models: A comprehensive survey of methods and applications, ACM Comput. Surv. 56 (4) (2023) 1–39.
[52] Y. Li, J. Guo, H. Qiu, F. Chen, J. Zhang, Denoising diffusion probabilistic models and transfer learning for citrus disease diagnosis, Front. Plant Sci. 14 (2023) 1267810.
[53] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, S. Ganguli, Deep unsupervised learning using nonequilibrium thermodynamics, in: International Conference on Machine Learning, PMLR, 2015, pp. 2256–2265.
[54] J. Gehring, M. Auli, D. Grangier, D. Yarats, Y.N. Dauphin, Convolutional sequence to sequence learning, in: International Conference on Machine Learning, PMLR, 2017, pp. 1243–1252.
[55] T. Salimans, A. Karpathy, X. Chen, D.P. Kingma, Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications, 2017, arXiv preprint arXiv:1701.05517.
[56] X. Chen, N. Mishra, M. Rohaninejad, P. Abbeel, Pixelsnail: An improved autoregressive generative model, in: International Conference on Machine Learning, PMLR, 2018, pp. 864–872.
[57] T. Salimans, D.P. Kingma, Weight normalization: A simple reparameterization to accelerate training of deep neural networks, Adv. Neural Inf. Process. Syst. 29 (2016).
[58] Y. Wu, K. He, Group normalization, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018, pp. 3–19.
[59] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-cam: Visual explanations from deep networks via gradient-based localization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 618–626.