applied sciences

MDPI

Article

Guava Disease Detection Using Deep Convolutional Neural Networks: A Case Study of Guava Plants

Almetwally M. Mostafa [1,*], Swarn Avinash Kumar [2], Talha Meraj [3], Hafiz Tayyab Rauf [4], Abeer Ali Alnuaim [5] and Maram Abdullah Alkhayyal [1]

1. Department of Information Systems, College of Computer and Information Sciences, King Saud University, P.O. Box 51178, Riyadh 11543, Saudi Arabia; 442204299@student.ksu.edu.sa
2. Indian Institute of Information Technology, Allahabad 211015, Uttar Pradesh, India; swarnavinashkumar@ieee.org
3. Department of Computer Science, COMSATS University Islamabad—Wah Campus, Wah Cantt 47040, Pakistan; talha_cui@ciitwah.edu.pk
4. Department of Computer Science, Faculty of Engineering & Informatics, University of Bradford, Bradford BD7 1DP, UK; h.rauf4@bradford.ac.uk or hafiztayyabrauf093@gmail.com
5. Department of Computer Science and Engineering, College of Applied Studies and Community Services, King Saud University, P.O. Box 22459, Riyadh 11495, Saudi Arabia; abalnuaim@ksu.edu.sa

* Correspondence: almetwaly@ksu.edu.sa

Abstract: Food production is a growing challenge with the increasing global population. To increase the yield of food production, we need to adopt new biotechnology-based fertilization techniques. Furthermore, we need to improve early prevention steps against plant disease. Guava is an essential fruit in Asian countries such as Pakistan, which is fourth in its production. Several pathological and fungal diseases attack guava plants. Furthermore, postharvest infections might result in significant output losses. A professional opinion is essential for disease analysis due to minor variances in various guava disease symptoms. Farmers' poor usage of pesticides may result in financial losses due to incorrect diagnosis. Computer-vision-based monitoring is required with developing field guava plants. This research uses a deep convolutional neural network (DCNN)-based data enhancement using color-histogram equalization and the unsharp masking technique to identify different guava plant species. Nine angles from 360° were applied to increase the number of transformed plant images. These augmented data were then fed as input into state-of-the-art classification networks. The proposed method was first normalized and preprocessed. A locally collected guava disease dataset from Pakistan was used for the experimental evaluation. The proposed study uses five neural network structures, AlexNet, SqueezeNet, GoogLeNet, ResNet-50, and ResNet-101, to identify different guava plant species. The experimental results proved that ResNet-101 obtained the highest classification results, with 97.74% accuracy.

Keywords: data augmentation; deep learning; guava disease; plant disease detection

check for updates

Citation: Mostafa, A.M.; Kumar, S.A.; Meraj, T.; Rauf, H.T.; Alnuaim, A.A.; Alkhayyal, M.A. Guava Disease Detection Using Deep Convolutional Neural Networks: A Case Study of Guava Plants. *Appl. Sci.* **2022**, *12*, 239. https://doi.org/10.3390/app12010239

Academic Editors: Anselme Muzirafuti and Dimitrios S. Paraforos

Received: 28 September 2021
Accepted: 17 December 2021
Published: 27 December 2021

1. Introduction

Food production is currently one of the greatest challenges with the growing global population. It is estimated that food consumption will double by 2050. Therefore, food production needs a more high-yielding and sustainable environment to increase the plant yield [1,2]. Guava is an important plant that belongs to the Myrtaceae plant family. It was initially allocated in the American tropics; guava was discovered in Portugal in the early 17th Century [3]. It is popular in tropical and nontropical countries such as Bangladesh, India, Pakistan, Brazil, and Cuba [4]. Guava contains phosphorus, calcium, nicotinic acid, and many other essential food components [5]. Furthermore, it normalizes blood pressure, has benefits for diabetes, provides immunity against dysentery, and eliminates diarrhea [6]. Regarding guava's growing environment, it can grow in a variety of soils with a wide range of pH (4.4 to 4.9), where it can also sustain intensive and extensive climate change [7].

The delightful aroma of the generally spherical guava fruit makes it attractive [8]. The growing age of fruits and vegetables may change and pass from various stages, making it challenging to recognize various factors that make them behave differently during different stages. Therefore, image acquisition of vegetables and fruits is the first important step to effectively analyze quality attributes such as color and texture. Illumination also affects receiving these features from the sensor in fruit image collection [9]. Computer-vision-based fruit and vegetable disease detection can lead to large-scale automatic vegetable and fruit monitoring [10]. This helps in taking earlier steps to take care of specific hazards that disturb actual yields, such as the need for fertilizers to be applied to increase the growth rate [11]. Various diseases affect the production of guava fruits, such as anthracnose [5], canker, dot, mummification, and rust. Farmers are very knowledgeable about these diseases, but they mostly do not know of early prevention methods to protect them against further loss. This ultimately leads to significant loss in guava production [12]. These different diseases are caused by different factors of guava plants; for example, canker is caused by algae and was first discovered by Ruehle [13].

Similarly, Dastur is another guava disease that is caused by dry rot [14]. These types of diseases affect guava production, which leads to economic and environmental loss [15]. Environmental loss is any kind of loss, including energy, water, clean air, and land loss, where as far as the economic loss is concerned, this results in financial loss in production. Pakistan is a country in the Asian Pacific whose economy is mostly based on agricultural production. The agricultural significance of Pakistan can be analyzed from its gross domestic product (GDP), with agriculture being 25% of its annual GDP [16]. Many agricultural countries produce guava as a domestic product, and Pakistan is globally fourth in guava production, as it annually produces 1,784,300 t [17]. To diagnose guava diseases in a timely manner, accurate detection is necessary, as false detection may lead to the poor production of guava species. Manual observation may be time consuming and lead to the wrong interpretations.

This led us to produce an automatic system for guava disease detection [18], as the production of guava fruits creates severe issues in developed and underdeveloped countries [19,20]. The automation of disease detection is currently the fastest, least expensive, and most accurate solution [21]. It could cost more, but it can lead to a colossal time reduction by automating the disease detection process [22]. For prediction models, the RGB color channel images are primarily used, which are visually distinguishable by color. The color features could be strong descriptors to distinguish different diseases. However, obtaining deep feature-based models could be more robust, as this covers many other aspects such as geometry, pattern, texture, and other local features. For this, a local guava disease-based RGB image dataset was collected by a high-display-quality camera here. It contains four types of disease, namely canker, dost, rust, and mummification, with the fifth category as the healthy class. Further details of the dataset are discussed in Section 3. The proposed study was inspired by deep learning (DL), and a comparative analysis of various pretrained models is proposed. The main contributions are as follows:

1. Augmented data cover different aspects of view to provide more real-time data visualization and big data usage for DL;
2. The first local Pakistani guava disease detection dataset using DL;
3. State-of-the-art DL models used to validate Pakistani guava disease detection.

The rest of the manuscript is divided into three sections. Section 2 presents the related work. Section 3 is the methodology. Section 4 outlines the results and discussion.

2. Related Work

Plant disease detection is becoming increasingly automated. However, both machine learning and deep learning methods are used [23] in order to provide intelligent automated solutions, with a few recent studies on both categories are discussed below.

2.1. Machine-Learning-Based Plant Disease Detection

Some experts confirmed the labeling to identify unhealthy from healthy guava fruits. Handcrafted features named local binary patterns (LBPs) are extracted and further reduced using principal component analysis (PCA). The multiple types of machine-learning (ML) classifiers are used, where a cubic support vector machine performed the best among the various methods in [24]. Edge- and threshold-based segmentation is performed for plant disease detection using images of leaves. Multiple features of color, texture, and shape are extracted, which are fed into a neural network classifier that classifies different plant diseases [25].

Shadows are removed from the background by enhancing, resizing, and isolating the region of interest (ROI), with clustering performed by using K-means for pomegranate fruit disease detection in [26]. Guava disease detection was performed using basic color transformation functions in image processing to detect the actual diseased parts of plant leaves. Classification was performed using support vector machine (SVM) and K-nearest neighbor (KNN) in [27]. Apple disease detection was performed using the spot segmentation method, where feature extraction and fusion were also performed. The decorrelation method was used for the fusion of extracted features in [28]. A soil-based analysis to recognize the soil indicator that plays an important role in plant yield was also used in [29]. Similarly, the weather forecasting history could play an important role in plant monitoring systems to avoid any natural hazards, as in [30]. The hue–saturation–intensity (HSI) color space was initially used, where unhealthy areas were detected using textures. Multiple features were extracted after the color conversion of the data. Features such as homogeneity, energy, and other cluster-based features were extracted. Lastly, SVM was used for classification in [31].

2.2. Deep-Learning-Based Plant Disease Detection

Demand for deep-learning (DL)-based studies is increasing due to their promising results. Big data are used for DL model training for the prediction of automated detection. Therefore, a similar study used more than 54,000 images of 14 crop diseases with 26 different diseases types. A deep convolutional neural network (DCNN) was proposed with a 99.35% accuracy achieved on the held-out test dataset. Lastly, smartphone-assisted automatic crop disease detection was proposed and an app was suggested for development in [32]. A similar big data dataset was used for plant disease detection. The open dataset of more than 87,000 images was used with 25 different plant categories. Multiple DCNN architectures were used, and the best-performing network achieved a 99.53% accuracy. The reported results showed that this tool model can be used for real-time plant disease identification [33]. Symptom-based gaps found by researchers that cover it by proposing their own CNN with a visualization technique were also missing in previous architectures.

Modified networks for plant disease identification were applied that improved the results of [34]. In-depth features and transfer learning using famous architectures were applied on famous models' architectures. Deep-feature-based classification using SVM and other ML classifiers showed better results than those of the transfer-learning method. Moreover, the fully connected layer of state-of-the-art architectures such as VGG-16, VGG-19, and AlexNet showed better accuracy than that of other fully connected layers [35]. The images of specific conditions and various symptoms were acquired in real time, and these were missed in public datasets. To tackle this limitation, data augmentation was performed, which took a single input leaf image from multiple views that covered certain conditions on the same leaf input image. It also covered multiple diseases affecting leaves. Augmentation-based predictions increased the accuracy by 12%. Furthermore, the data limitation suggested using data augmentation in [36]: the Plant–Village dataset contained differently annotated apple black rot images. Fine-tuned DL models were trained, and the best accuracy was achieved by VGG-16, at 90.4% [37]. Different ML- and DL-based methods are shown in Table 1.

Table 1. Summary of recent studies on the detection of guava diseases.

References	Year	Species	Domain	Method	Results
[38]	2021	Plant–Village	Deep learning	C-GAN and DenseNet121-based transfer learning	Accuracy (5 classes) = 99.51% Accuracy (7 classes) = 98.65% Accuracy (10 classes) = 97.11%
[4]	2021	Guava plant disease	Machine learning	HSV, RGB, LBP, and classical machine learning methods	99% accuracy
[39]	2020	Cotton plant disease	Deep learning	Proposed CNN	∼
[40]	2019	Plant disease dataset	Deep learning	Traditional augmentation and GAN for data generation and neural network	Accuracy = 93.67%
[41]	2019	Tomato and brinjal	Deep learning	GLCM, adaptive neuro-fuzzy system	Tomato accuracy = 90.7%, brinjal accuracy = 98.0%
[42]	2019	Potato tuber	Deep learning	CNN	90–10 split training–testing accuracy = 96%
[33]	2018	Open plant dataset	Deep learning	AlexNet, VGG, and other CNN	Best accuracy = 99.53%
[43]	2018	Papaya leaves	Machine learning	HOG features, random forest classifier	Accuracy = 70.14%

Although DL has shown excellent results in plant disease detection, it still faces some challenges. The big data challenge compromises previous studies because they used limited data. The data limitation can be reduced by using various strategies that also produce a more confident model by covering a different aspect of a specific input sample of plant disease [44].

3. Methodology

The automation of plant disease monitoring is taking the place of manual monitoring. Many researchers have used real-time experimentation of plant disease monitoring and achieved satisfying results. A local Pakistani dataset was collected for guava plant and fruit disease detection in the proposed study. Data augmentation was used to meet the big data usage challenge, where it was also used to cover model overfitting problems. Data augmentation was performed using the affine transformation method; to enhance the region of interest (ROI), unsharp masking and the histogram equalization method were performed, better sharpening the ROI and removing any existing noise in the augmented data. The final augmented and enhanced data were fed into various fine-tuned state-of-the-art classification methods. All the steps are shown in Figure 1.

In the framework, augmented and enhanced images were given to 5 different predefined architectures by replacing their last layer according to the given data classes. AlexNet was the first model in the ImageNet competition that changed the image classification and object detection using deep-learning models. SqueezeNet, GoogLeNet, and ResNet followed with many others. Famous ones with different kinds of architectures were used to check the effectiveness of a given local guava dataset. The proposed method showed an initial step on a newly collected local Pakistani dataset, where more methods can be adopted using a different ML and DL technology. The details of the dataset before and after augmentation and the details of other used CNN architectures are shown in Section 3.1 by discussing their fine-tuned parameters, with the weights in Sections 3.2–3.4. The achieved results on the validation data are shown in Section 4.

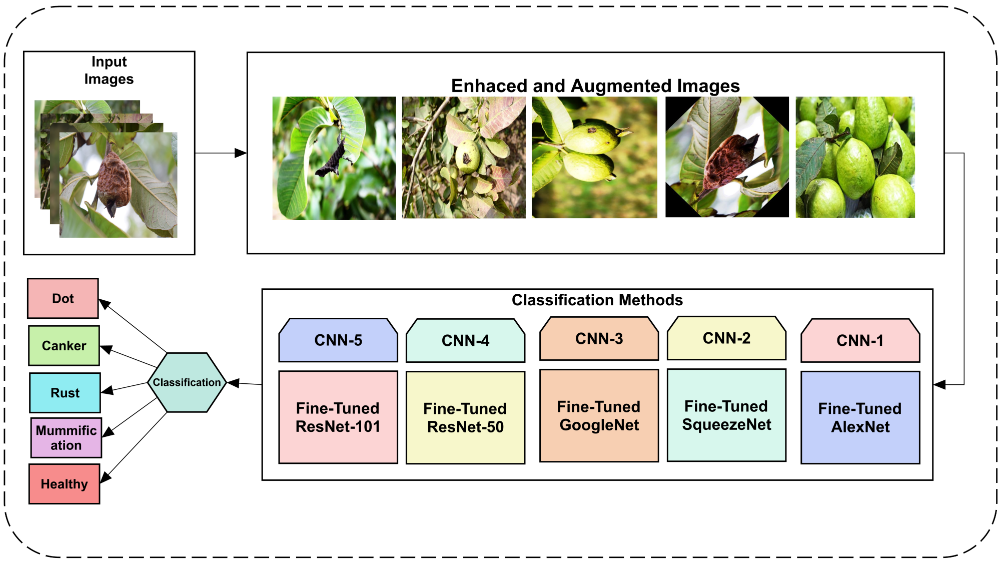

Figure 1. Proposed framework for guava plant disease detection.

3.1. Dataset Normalization

The dataset was initially collected using a high-definition camera with different resolutions. Images were of different angles and orientations, which may lead to the misguidance of the prediction model due to the illusion factor, spatial resolution changes, camera settings, background changes, and many other real-time factors. Therefore, the data were first resized to be equal in size using the bicubic interpolation method. This uses 4 by 4 neighborhood pixels to interpolate the 16 nearest pixels, primarily used in many image-editing tools. This improved the results as compared to those of the bilinear and nearest-neighbor methods. Interpolation was used to resize the image. The resized image was again augmented and enhanced; its histogram-based representation is shown in Figure 2.

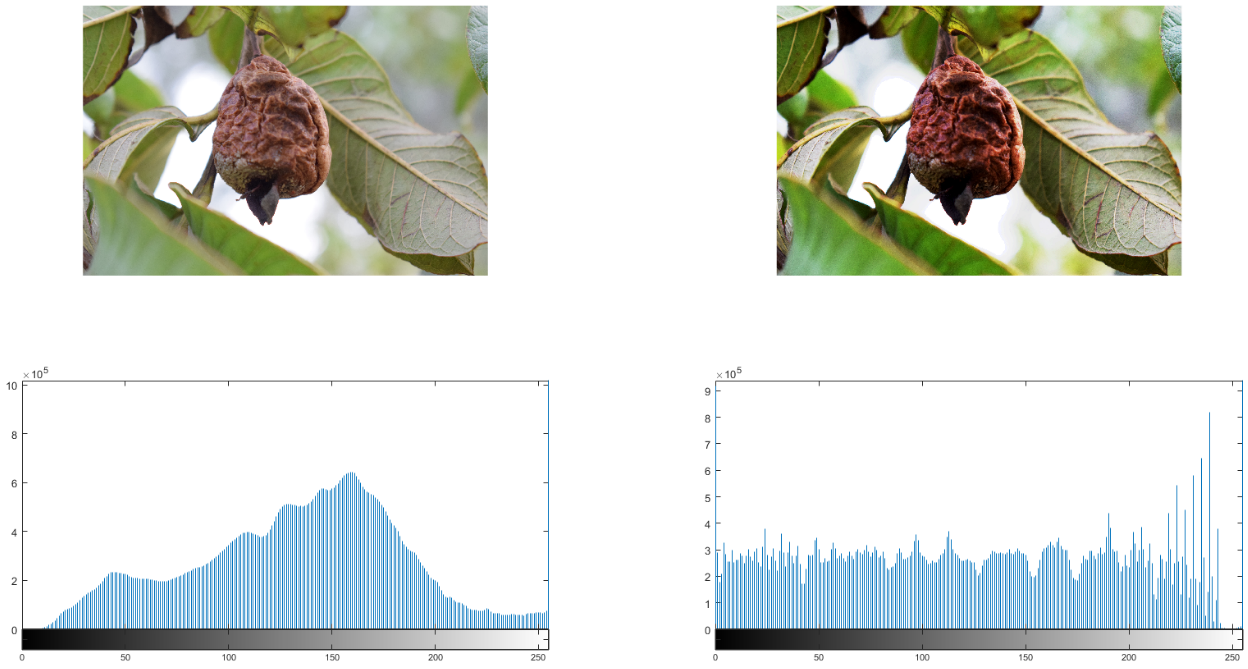

Figure 2. (**top**, **left**) Original and (**top**, **right**) enhanced. (**bottom**, **left**) Original and (**bottom**, **right**) enhanced image histogram.

The histogram shows that the data intensity levels were equalized after the preprocessing of data resizing and enhancement.

3.2. Data Augmentation and Enhancement

Resized images were rotated or transformed using the affine transformation method. Different angles with a 360 rotation were used with an angle difference of 45°. There were 9 angles in total in each sample instance applied for all classes, namely 0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°, and 360°. Affine transformation was calculated as given in Equation (1), and the applied augmentation sample for each category is shown in Figure 3.

$$Rotation = \begin{bmatrix} cos(a) & sin(a) & 0 \\ -sin(a) & cos(a) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

The angle of rotation according to an affine rotation is shown in Equation (1). A is the angle value that was changed nine times for an image to obtain the new rotated image.

Data enhancement was applied with a combination of unsharp masking and color histogram equalization, and both of these methods were applied and calculated using Equations (2) and (3).

$$f\ (I) = \alpha f - \beta f_l \tag{2}$$

The output enhanced image was calculated in *f(I)*, where α and β are constant, to be the input image that is multiplied where the original image is processed and subtracted via low-pass filter process mask f_l. Histogram equalization was used in the image processing to enhance a given RGB image, and the three channels were individually evaluated using Equation (3).

$$T_k = (L-1)cdf(P) \tag{3}$$

The cumulative distribution of the given intensity was calculated over the probability of occurrences, as calculated in Equation (4)

$$Cdf\ (P) = \sum_{k=-\infty}^{P} Prob(k) \tag{4}$$

This calculated accumulative distributive value was then multiplied with maximal value intensity, and the newly calculated transformed intensity was calculated and mapped to the corresponding pixels throughout the given image.

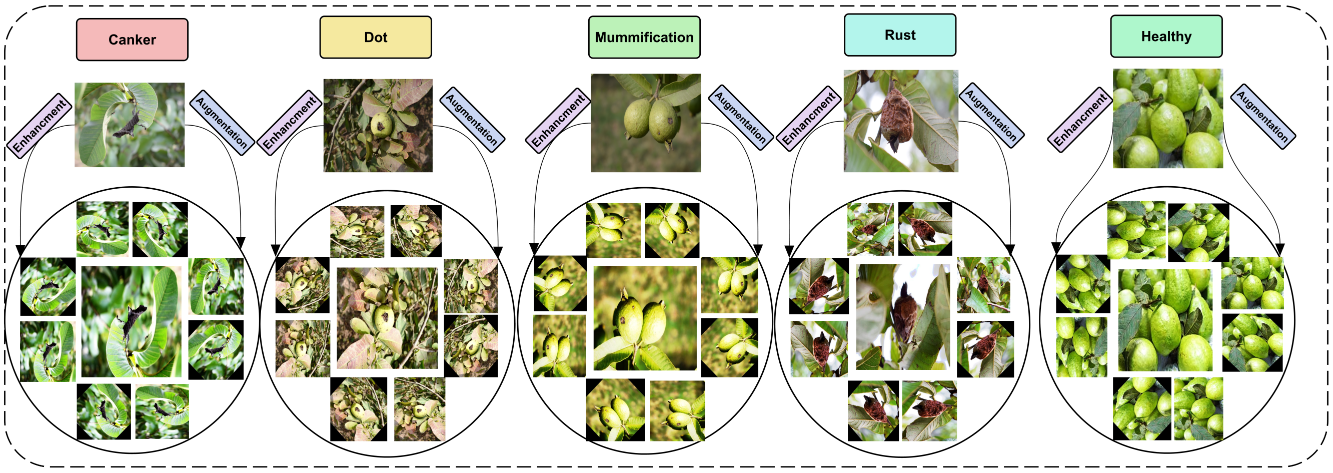

Figure 3. Five types of guava species image samples with their enhanced and rotated 9 angle images in circular view in 4 circles.

Rotated images covered a different aspect of the actual time occurrence, which could be any of the orientations for the user. The training and predictions of rotated augmented data offered promising results.

3.3. Basics of Convolutional Neural Networks

There are many proposed CNN architectures used in various aspects of intelligent classification and object-detection systems. These architectures have slight differences in their networks, and the analyzed primary layers and components are discussed here. We discuss these basics before explaining state-of-the-art models of classification that were also used.

3.3.1. Convolutional Layer

The convolutional layer is called as such when at least one convolutional operation is used in the input layers of an architecture. The convolutional operation uses various parameters such as kernels, where the kernel size is specified for parameter initialization. Similarly, padding and stride size are also initialized and used in convolutional operations. The convolutional operation is summarized in Equation (5).

$$Conv_i^l = Bias_i^l + \sum_{j=1}^{a_i^{(l-1)}} w_{i,j}^{(l-1)} * C_i^l \tag{5}$$

In Equation (5), $Conv_i^l$ is the output of a convolved operation in which $Bias_i^l$ is the bias matrix, with the ith iterative region of operation on which convolved window w is evolving, and i, j represents the window size of the rows and columns. Iterated convolved window C_i^l is multiplied with the corresponding pixels of the given image, where the selected area is defined by window size $w_{i,j}$.

3.3.2. Batch Normalization

Batch normalization is a normalization operation, such as the min–max data normalization performed in data cleaning. Batch normalization is a normalization in which a batch of input data is normalized, and it can be written as in Equation (6).

$$x_i' = \frac{x_i - \mu_B}{\sigma_B^2} \tag{6}$$

It normalizes data, where the data transformation has taken place, such as a mean output close to 0, and the standard deviation output remains close to 1. In Equation (6), input x of a particular instance is subtracted from the mean (μ) of batch b, where after subtraction, a ratio is calculated over the square of the standard deviation (σ) of that particular batch (B) where instance x belongs, and a normalized value of x_i' is returned as output.

3.3.3. Pooling Layer

Pooling pools over a specific item from some scenarios, where pooling in CNN is used to calculate a max, min, and average pool to take a single output value from a defined kernel window. The stride is also used as a parameter to define the ongoing or iterating step for a pooling value. The pooling value is calculated as in Equation (7).

$$D_{output} = x_h * x_w * x_d \tag{7}$$

The output dimension after performing pooling is represented as D_{output}, where x is the input instance and instance height, the width represented as h, w, and the color channel dimension is represented as d, for instance x.

3.3.4. Rectified Linear Unit

ReLU is an activation unit where other activation units such as tanh and sigmoid are also used, and it is used in various studies. ReLU is also called the piecewise linear function, and it simply outputs an identical input variable to the input if it is >0; otherwise, it is 0. Lastly, it maximally excludes the misguiding value in calculating an output class by a prediction model of artificial intelligence. We can simply write the ReLU as in Equation (8).

$$ReLU = \max(0, x) \tag{8}$$

The linear behavior of this activation function makes it a commonly used activation function. In Equation (8), the output is shown as ReLU, where input x is to be taken as the max, which is calculated very straightforwardly if the input value is positive >0; then, it outputs the simple input as it is where <0, or negative values are only taken as 0.

3.3.5. Softmax

The softmax function returns probability values in the range of 0–1, where maximum likelihood returns a higher probability value. It is somehow matched to multilinear regression, where multiple classes are predicted using internal values. The softmax function can be calculated as Equation (9).

$$\sigma(\overrightarrow{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \tag{9}$$

In Equation (5), the softmax operation is calculated as input vector $\overrightarrow{z}$ where z_i are all the input values of vector z. Exponential function e is applied over each value that gives a positive value greater than 0. The denominator value confirms all values sum up to give one value. The final K is the output class number that changes from application to application.

3.4. Classification Using the AlexNet Architecture

AlexNet was the first model in deep learning to change the trend of image identification and classification tasks. It was initially proposed for detecting and classifying objects using a benchmark dataset from ImageNet. Using the AlexNet architecture, the image size for the input layer is taken to be 227 × 227 × 3. In this architecture, there are only 5 convolutional layers and 3 fully connected layers, giving 25 layers in total. The last layers were altered in the proposed framework, and then, we used fine-tuned network parameters. The modifications are shown in Table 2.

In Table 2, all layers above remain the same as in AlexNet, where Fc-8 is first altered with four layers, and the two layers of the softmax activation class output correspondingly give the output for the five categories of guava species.

Table 2. AlexNet for guava disease detection.

Layers	Categories	Activations	Weights
Data Layer	Image Input	227 × 227 × 3	-
Convolve-1	Convolution	55 × 55 × 96	11 × 11 × 3 × 96
ReLU-1	ReLU	55 × 55 × 96	-
Normalization-1	Cross-Channel Normalization	55 × 55 × 96	-
Pool-1	Max-Pooling	27 × 27 × 96	-
Convolve-2	Grouped Convolution	27 × 27 × 256	5 × 5 × 48 × 128
ReLU-2	ReLU	27 × 27 × 256	-
Normalization-2	Cross-Channel Normalization	27 × 27 × 256	-
Pool-2	Max-Pooling	13 × 13 × 256	-
Convolve-3	Convolution	13 × 13 × 384	3 × 3 × 256 × 384
ReLU-3	ReLU-3	13 × 13 × 384	-

Table 2. *Cont.*

Layers	Categories	Activations	Weights
Convolve-4	Grouped Convolution	13 × 13 × 384	3 × 3 × 192 × 192
ReLU-4	ReLU	13 × 13 × 384	-
Convolve-5	Grouped Convolution	13 × 13 × 256	3 × 3 × 192 × 128
ReLU-5	ReLU	13 × 13 × 256	-
Pool-5	Max-Pooling	6 × 6 × 256	-
FC-6	Fully Connected	1 × 1 × 4096	4096 × 9216
ReLU-6	ReLU	1 × 1 × 4096	-
Drop-6	Dropout	1 × 1 × 4096	-
Fc-7	Fully Connected	1 × 14096	4096 × 4096
ReLU-7	ReLU	1 × 1 × 4096	-
Drop-7	Dropout	1 × 1 × 4096	-
FC-8	Fully Connected	1 × 1 × 5	5 × 4096
Softmax	Softmax	1 × 1 × 5	-
Class Output	Classification	-	-

3.5. Classification Using GoogLeNet Architecture

Google developers focused on the proposed AlexNet model and then introduced the inception module and changed it sequentially by stacking up layers. This introduced different and smaller kernel size windows with more layers in them. It became the winner of the 2014 ILSVRC competition. The inception modules that were the fundamental contribution by GoogLeNet are shown for the trained architecture on the guava dataset, and the layer-by-layer parameters are shown in Table 3.

Table 3. GoogLeNet network used for guava disease detection.

Layers	Categories	Activations	Weights
Inception-3a-1×1	Convolution	28 × 28 × 64	1 × 1 × 192 × 64
Inception-3a-3×3	Convolution	28 × 28 × 128	3 × 3 × 196 × 128
Inception-3a-5×5	Convolution	28 × 28 × 32	5 × 5 × 16 × 132
Inception-4a-1×1	Convolution	14 × 14 × 192	1 × 1 × 480 × 192
Inception-4a-3×3	Convolution	14 × 14 × 208	3 × 3 × 96 × 208
Inception-4a-5×5	Convolution	14 × 14 × 48	5 × 5 × 16 × 48
Inception-5a-1×1	Convolution	7 × 7 × 256	1 × 1 × 832 × 256
Inception-5a-3×3	Convolution	7 × 7 × 320	3 × 3 × 160 × 320
Inception-5a-5×5	Convolution	7 × 7 × 128	5 × 5 × 32 × 128
FC	Fully Connected	1 × 1 × 5	5 × 1024
Softmax	Softmax	1 × 1 × 5	-
Classification	Classification Output	-	-

The overall architecture remains similar, where the last three layers are altered, and the upper-layer connections remains connected.

3.6. Classification Using the SqueezeNet Architecture

SqueezeNet was introduced with five modules. It is claimed that the 3 × 3 kernel size should be reduced to 1 × 1, reducing the size of the overall parameter. Downsampling is also reduced into layers. More feature maps are thus learned by the layers. The introduced fire module contains the squeeze layer, and it has 1 × 1 filters. They are fed an expanding layer that is a mixture of 1 × 1 and 3 × 3 kernels. The SqueezeNet layer architecture and parameters with altered layers are shown in Table 4.

Table 4. SqueezeNet network used for guava disease detection.

Layers	Categories	Activations	Weights
Fire3-Squeeze-1×1	Convolution	56 × 56 × 16	1 × 1 × 128 × 16
Fire4-Squeeze-1×1	Convolution	28 × 28 × 32	1 × 1 × 128 × 32
Fire5-Squeeze-1×1	Convolution	28 × 28 × 32	1 × 1 × 256 × 32
Fire6-Squeeze-1×1	Convolution	14 × 14 × 48	1 × 1 × 256 × 48
Fire7-Squeeze-1×1	Convolution	14 × 14 × 48	1 × 1 × 384 × 48
Fire8-Squeeze-1×1	Convolution	14 × 14 × 64	1 × 1 × 384 × 64
Fire9-Squeeze-1×1	Convolution	14 × 14 × 64	1 × 1 × 512 × 64
Last-convolve	Convolution	14 × 14 × 5	1 × 1 × 512 × 5
ReLU-Convolve	ReLU	14 × 14 × 5	1 × 1 × 512 × 5
Pool-4	Global Average Pooling	1 × 1 × 5	-
Softmax	Softmax	1 × 1 × 5	-
Classification	Classification Output	-	-

The fire modules in the hyperparameter continuation produce three tunable parameters, namely s1 × 1, e1 × 1, and e3 × 3. AlexNet's level of accuracy was achieved by the actual SqueezeNet with 50× fewer parameters, and the model size was reduced to just 0.5 MB because of the decrease in the kernel sizes and the fire modules used in this architecture.

3.7. Classification Using the ResNet-50 Architecture

ResNet was introduced with the residual block concept mainly to answer the overfitting issue created in DL models. It uses a considerable number of layers, such as 50, 101, and 152. As ti is suggested by GoogLeNet to use a small kernel size, it uses small convolutional kernels where denser or more layers are used to meet or improve the validity of the data. The introduced residual block uses a 1 × 1 layer that reduces the dimension, a 3 × 3 layer, and a 1 × 1 layer used to restore the dimensions of the given input. The layer-based ResNet was used, so we used a 50- and 101-layer architecture; the 50-layer architecture of ResNet is shown in Table 5.

Table 5. ResNet-50 network used for guava disease detection.

Layers	Categories	Activations	Weights
Res-2a	Convolution	56 × 56 × 256	1 × 1 × 64 × 256
Res-3a	Convolution	28 × 28 × 512	1 × 1 × 256 × 512
Res-4a	Convolution	14 × 14 × 1024	1 × 1 × 512 × 1024
Res-5a	Convolution	7 × 7 × 2048	1 × 1 × 1024 × 2048
FC	Fully Connected	1 × 1 × 5	5 × 2048
Softmax	Softmax	1 × 1 × 5	-
Class-Output	Classification	-	-

The basic residual branches of ResNet-50 with their learned parameters in guava disease detection are shown in Table 5. The residual block re-concatenates spatial information from the previous block to preserve information in each calculated feature map of the residual block. For the proposed framework, the last layers are altered with five categories to classify them on the basis of previous learning on the augmented guava data of the proposed study.

3.8. Classification Using ResNet-101 Architecture

The ResNet-based study produced many variants of its introduced residual blocks, such as 18, 19, 34, 50, and 101, and the densest of 152. Making it increasingly denser did not improve the accuracy after a certain point. This may be due to many factors, such as learning saturation, loopholes in the proposed architecture, and hyperparameter optimization. Therefore, the mainly used networks were ResNet-50 and 101. The learned-weight-based architecture of the residual blocks using the ResNet-101 architecture for guava disease detection are described in Table 5; the difference between 50 and 101 is in their architecture. There are 347 layers in total in ResNet-101 and 177 layers in the ResNet-50 model. Learning mainly changed after Res-branch 4a, as hundreds of layers are added after it learns in different ways, as described in the ResNet-101 architecture.

4. Results and Discussion

The proposed study used augmented data of actual given locally collected data in Pakistan for guava disease detection. The data had enough images to train the DL model. The dataset details for before and after augmentation are shown in Table 6.

Table 6. Dataset description with and without augmentation.

Categories	**Number of Images (without Augmentation)**	**Number of Images (with Augmentation)**
Canker	77	693
Dot	76	684
Mummification	83	747
Rust	70	630
Healthy	15	135
Total	321	2889

4.1. Evaluation Measure

There are mainly four types of prediction instances, which we can consider in the formulation of these above-mentioned evaluation measures: true positive (*TP*), false positive (*FP*), true negative (*TN*), and false negative (*FN*). These are described in detail below.

4.1.1. Accuracy

Accuracy is the most commonly used measure in the ML and DL domains for classification. It can briefly be described as truly predicted instances over total instances, including wrong and right predictions. In terms of the four types used above, the equation for accuracy can be written as follows:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{10}$$

The equation can be described as the ratio of the summation of *TP* and *TN* over the summation of *TP*, *TN*, *FP*, and *FN*.

4.1.2. Specificity

This is the measure among the right predictions over the total that were from both the positive and negative classes. Briefly, negatively labeled objects are measured over the total of true- and false-negative instances. It can be written as:

$$Specificity = \frac{TN}{(TN + FP)} \tag{11}$$

The specificity equation is defined as the ratio over *TN* and the summation of *TN* and *FP*.

4.1.3. F1 Score

The F1 score is an essential measure, as it is calculated for both the essential measures of recall and precision. Recall is also known as sensitivity and is the measure to detect positive class predictions among true positives and false negatives. For the F1 score, we need to calculate the sensitivity as follows:

$$Recall\ (Sensitivity) = \frac{TP}{(TP + FN)} \tag{12}$$

The second measure, precision, is also separately calculated and used in the F1 score measurement. Precision is calculated to obtain a truly predicted class over true and false positives. It is represented as:

$$Precision = \frac{TP}{(TP + FP)} \tag{13}$$

After obtaining the precision and recall, the F1 score can be calculated. After having *TP* over *TP* and *FN* by recall and by obtaining *TP* over *TP* and *FP*, we can obtain more precise measurements for true-positive predictions. The final F1 score can be calculated as:

$$F1 - score = 2 \times \frac{Precision * Recall}{Recall + Precision} \tag{14}$$

Therefore, the F1 score equation can be defined as two multiplied by the ratio of the product and summation of precision and recall.

4.1.4. Kappa–Cohen Index

The last measure is to have confidence about the statistical analysis of the results, as statistical analysis is broadly used in many aspects of scientific work. Therefore, a statistical measure that gives confidence over confusion matrix values was calculated for evaluation. The kappa index gives the confidence over a certain range of confusion-matrix-based calculated values. If its value range lies in the range of 0–20, it promises that 0–4% of the data are reliable to use for prediction. If its index value lies in the range of 21–39, it promises a 4–15% data reliability. If it lies between 40 and 59, it promises 15–35% data reliability. If it lies between 60 and 79, it promises a 35% to 63% reliability. If it lies between 80 and 90, then it promises strong data reliability, and if it lies at more than 90, that means 82% to 100% reliability. It can be calculated as:

$$Agreement = \frac{\left(\frac{cm_1 * rm_1}{n}\right) + \left(\frac{cm_2 x rm_2}{n}\right)}{n} \tag{15}$$

The agreement type was calculated using cm_1, cm_2, rm_1, and rm_2, where cm_1 represents Column 1 and cm_2 represents Column 2. rm_1 and rm_2 represent Rows 1 and 2 of any two-class confusion matrix. This formulation is the general form of a two-class confusion matrix; in the case of our proposed methodology, there are five columns and rows that extend the formulation to up to five rows and columns.

The total augmented data were later split into a 70/30 ratio for training and testing data, where the fine-tuned parameters for each of the five training models remained different and showed different results. The training and testing data number of instances became 2023 and 866, respectively.

There were five different kinds of architectures applied to classify guava diseases using the augmented image data. The individual class testing data prediction results are discussed with their overall results. The evaluation measures accuracy, sensitivity, specificity, precision, recall, and the kappa index were also used as statistical measures. The results are shown in Table 7.

Table 7. Results obtained on the basis of individual class testing data.

Models	Categories	Accuracy	Specificity	F1-Score	Precision	Kappa
AlexNet	Canker	0.98077	0.99088	0.97608	0.97143	0.52096
	Dot	0.98049	0.99697	0.98529	0.99015	0.5309
	Healthy	1	1	1	1	0.90762
	Mummification	0.98661	0.99533	0.98661	0.99533	0.48509
	Rust	0.98942	0.99705	0.98942	0.98942	0.5648
	Overall	0.9850	0.9960	0.9875	0.9875	0.9531
GoogLeNet	Canker	0.96635	0.99696	0.9781	0.99015	0.52859
	Dot	0.97561	0.98638	0.96618	0.95694	0.52695
	Healthy	1	1	1	1	0.90762
	Mummification	0.99429	0.99377	0.97297	0.98182	0.49202
	Rust	0.99471	0.99114	0.98172	0.96907	0.56
	Overall	0.9758	0.9937	0.9798	0.9796	0.9242
SqueezeNet	Canker	0.97596	0.99392	0.97831	0.98068	0.52404
	Dot	1	1	0.96698	1	0.51541
	Healthy	1	1	1	1	0.90762
	Mummification	0.98214	0.9891	0.97561	0.96916	0.48364
	Rust	0.91534	1	0.9558	1	0.5866
	Overall	0.9711	0.9924	0.9753	0.9772	0.9098
ResNet-50	Canker	0.99519	0.99696	0.99281	0.99043	0.51957
	Dot	1	0.99697	0.99515	0.99034	0.52957
	Healthy	1	1	1	1	0.90762
	Mummification	0.98661	1	1	1	0.48733
	Rust	1	1	1	1	0.56351
	Overall	0.9954	0.9988	0.9962	0.9962	0.9856
ResNet-101	Canker	0.99519	0.98784	0.97872	0.96279	0.51484
	Dot	0.98049	0.99849	0.98772	0.99505	0.53178
	Healthy	1	1	1	1	0.90762
	Mummification	0.98214	1	1	1	0.48887
	Rust	0.98413	0.99557	0.98413	0.98413	0.56544
	Overall	0.9861	0.9964	0.9883	0.9884	0.9567

The individual class and overall results for each model were evaluated. Several evaluation measures were used: accuracy, specificity, F1 score, precision, and kappa.

The first model, AlexNet, showed 98% accuracy, 99% specificity, a 97.60% F1 score, 97.14% precision, and a 0.5296 value for kappa as the canker class prediction results of the testing data. Accuracy is a general measure over all positive and negative instances that are either wrongly or correctly predicted. The 98% accuracy of the canker class showed accurate predictions of positive and negative classes and mainly showed excellent and satisfactory results in TN predictions over TN and FP; specificity showed that true negatives were mostly predicted right among TN and FP. Precision showed TP over TP and FP, with a 97.14% value; this means that it had less accurate predictions. For a positive class to see the combined effect of TP and TN, the F1 score measure was used. The F1 score showed a 97% value, which summarizes both of the above measures. The kappa index showed a weak level of agreement for the canker class. The dot class showed 98.04% accuracy, 99.69% specificity, 98.52% F1 score, 99% precision, and a 0.53 kappa value. The dot class showed less accuracy than that of the canker class, but it needs to be discussed with another evaluation measure to analyze the predictions of positive and negative instances.

Specificity, which was 99.7% in the case of the dot class, represented TN among TN and FP where the precision was 99% for TP over TP and FP. The F1 score of this measure showed 98.52%, which summarizes the recall and precision, which could be considered to be more promising factors compared to sensitivity, specificity, and precision. The last statistical measure showed a weak level of agreement for this class. The third and healthy class showed accurate results in all networks where the data agreement value was also 0.97, showing an extraordinary level of agreement on the given data. The next class, mummification, showed 98.66% accuracy, 99.53% specificity, a 98.66% F1 score, a 99.53% precision value, and a 0.485 value for kappa. The accuracy for the mummification class was slightly better than that for the canker and dot classes. Other values such as specificity were lower than those for canker and dot, but with large differences. The precision value was higher than that of the canker class and lower than that of the dot class. To summarize both precision and specificity, the F1 score was used, which was nearer to that of the dot class and higher than that of the canker class. The kappa value is a decision-maker index, with a 0.48 value, which was less than that of the canker and dot classes, but also had weak agreement with the data reliability. The kappa value for the last class was 0.56 due to the one value for both precision and specificity, but 56 also lies in the weak agreement class. Lastly, the overall or mean results were evaluated, showing 98.50% accuracy, 99.60% specificity, a 98.75% F1 score, 98.75% precision, and a 0.9531 value for the kappa index. The mean or overall value was an actual representation of a model that produced good results; it was either about the accuracy, specificity specifically for TN, and precision for the TP value, and the F1 score represents the recall and precision. A kappa value of more than 90% is a strong agreement level, and the data reliability is also high when kappa is more than 90. Therefore, AlexNet overall showed satisfactory results.

GoogLeNet showed testing results on the canker class as follows: 96.635% accuracy, 99.69% specificity, 97.81% F1 score, 99.01% precision, and 0.5285 kappa value. Accuracy showed promising results where, if examining the specificity value over TN values, it showed a value of 99.69%. Similarly, the precision value over the TP values showed 99%, and the F1 score over precision and recall showed a value of 97.81%, which showed more confidence than precision and specificity did. The last important measure is the kappa index, 0.5285, which showed weak promise as it was of only one class over other classes. The mean value showed the actual effect of the kappa stat. The other dot class showed values of 97.56% accuracy, 98.638% specificity, a 96.618% F1 score, 95.69% precision, and a 0.5269 kappa value. The accuracy value as compared to that of the canker class was lower. In the case of the dot class, where the specificity value was also slightly lower, its precision value was lower than that of the canker class. The F1 score based on precision and recall showed a slightly higher score in the canker class, where the last kappa Cohen index was similar and lied on weak agreement of data reliability. The mummification showed a 99.42% value of the accuracy, a 99.37% value of the specificity, a 97.29% value for the F1 score, a 98.18% precision value, and a 0.49 kappa value. For an accuracy value higher than those of the canker and dot classes, where the specificity value was lower than that of the canker, and slightly lower than that of the dot class, this means that some TN instances had variation in these cases. The F1 score showed a lower score than that of the dot class and higher than that of the canker class. This means the combined effect of TP and FP was more promising for mummification compared to that for the dot class. The last class of rust showed 99.47% accuracy, 99.114% specificity, a 98.172% value for the F1 score, 96.907% for precision, and 0.56 for the Cohen index. The accuracy value was higher than that of the canker, dot, and mummification classes. The F1 score showed a 98.17% value that was nearest the canker, mummification, and dot classes. Therefore, the accuracy value was the measure to analyze the test prediction results where other values such as the F1 score also mattered and made the results distinguishable.

SqueezeNet testing showed results for the canker class of 97.596% accuracy, 99.392% specificity, a 97.838% F1 score, 98% precision, and a 0.52 kappa value. The accuracy value, a general assumption of model performance, had a lower value than that of specificity, precision, and the F1 score. The specificity value was much higher, which means that true negatives over the TN and FP had higher rates for the canker class. TP cases were also predicted with a 98% value over the FP and TP as the precision scores. The recall- and precision-based F1 score showed 97.83%, which is intermediate between the precision and specificity. The kappa value was 0.52 and lied on the weak agreement of the data. The second dot-class-based results showed 100% accuracy, 100% specificity, a 96.68% F1 score, 100% precision, and a 0.51 kappa value. Although the accuracy was good for this class, specificity showed a 100% result. The F1 score that covered recall and precision validity had a 96.68% score and 0.52 kappa value. The mummification class showed a predictivity of 98.214% accuracy, 98.91% specificity, a 97.56% F1-score, a 96.916% precision value, and 0.48 for the kappa index. The last class of rust showed predictivity measures of 91.53% accuracy, 100% specificity, a 95.58% F1 score, and 100% precision. The accuracy value was lower than that of the other three classes where precision and specificity were 100% in this case. The global or overall results for the all classes had a primary issue. Accuracy was 97.11%, lower than that of AlexNet and GoogLeNet, where specificity was lower than that of both AlexNet and GoogLeNet, and the F1 score, precision, and kappa were lower for this model testing the data predictions. The kappa index showed promise for these data.

ResNet-50 and -101 had much more improved results than those of AlexNet, SqueezeNet, and GoogLeNet. ResNet-50 showed canker class results of 99.51 accuracy, 99.68% specificity, a 99.28% of F1 score, a 99.03% precision value, and 0.51 for precision value of the kappa index. Compared to the previous cases of AlexNet, GoogLeNet, and SqueezeNet, the results were overall improved for all measures. Similarly, for the dot class, the accuracy value was 100%, 99.697% specificity, a 99.515% F1 score, a 99.034% precision value, and a 0.52 kappa index. The canker class results were not better than those of the dot class if we look at the accuracy measures, with only a slight difference in the F1 scores. Mummification results showed 98.661% accuracy, 100% specificity, a 100% F1 score, 98.25% precision, and a 0.48 kappa value. Although it had 100% accuracy as an individual class, the specificity and precision values were also higher than those of the canker and dot classes. The rust class showed 100% accuracy, 100% specificity, a 100% F1 score, and a 100% precision value. The overall results were improved as compared to the above models' mean results. The global mean results were 99.54% accuracy, which was better than that of SqueezeNet, GoogLeNet, and AlexNet. Specificity was also better than that in the three models. The F1 score and precision were both better than those in the previously discussed three models. Although it had a higher value than that of the previous models, the last kappa had the same class of confidence. The data reliability was higher for all models.

The last model of the proposed study also achieved good results as compared to the other models. The canker class showed predictivity values of 99.519% accuracy, 98.784% specificity, an F1 score of 97.87%, a precision of 96.27%, and a 0.51 kappa index. ResNet-50's accuracy had a dominant result compared to the previous class results of the other models, while other evaluation values also showed more improvement in this model. If the accuracy value was improved, other values were not so improved, but different cases showed overall improvement of the results. In ResNet-101, the dot class showed 98.049% accuracy, 99.84% specificity, 99.505% precision, and a 0.53 kappa value. Accuracy, specificity, precision, and the F1 score were overall improved for each class, which did not happen in the previous models' results for any class. The mummification class again showed 100% accuracy in this model testing. The mummification class showed 100% accurate results in other models where other values were not improved to such an extent: 98.41% accuracy, 99.84% specificity, 98.93% F1 score, and 99.47% precision. The last class showed consistency in the improvement of the results for each class by also showing promising results here. Lastly, the overall mean results of ResNet-50 proved it to be more accurate than the four other models. The accuracy was also better than that of the others. Similarly, other measures also showed excellent performance. The graphical illustration of all five models' mean testing results is shown in Figure 4.

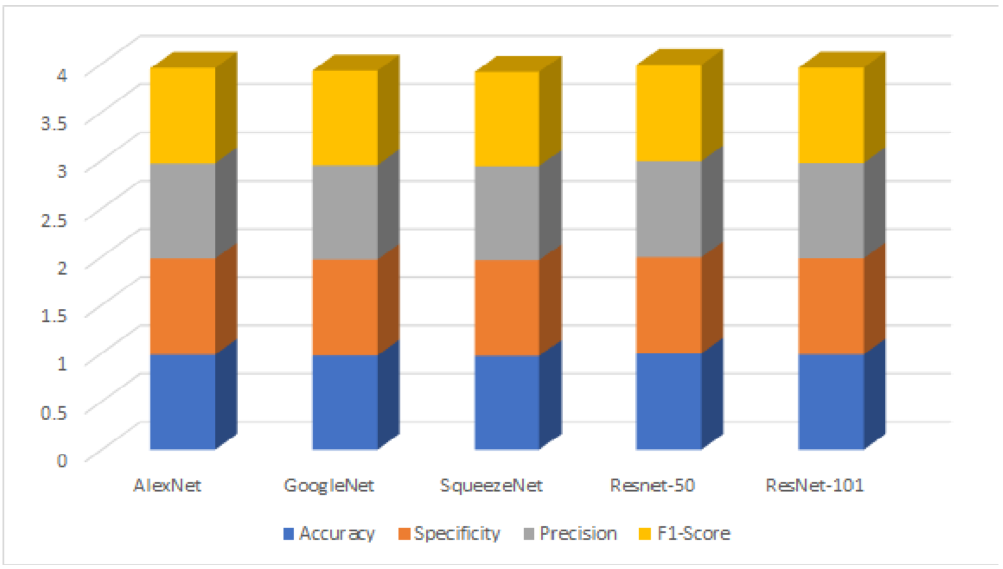

Figure 4. Guava disease classification result visualization.

The overall results showed that the kappa value had overall excellent data reliability for all models. The mummification, the healthy and dot classes were more distinct classes to distinguish them from the other two classes, as they showed 100 true results many times. The densest model with more residual connections showed more accurate results, which means that using a small kernel size with an increasingly denser network improved the classification results.

For individual cases or data-based testing analysis, the confusion matrices were designed and evaluated, and they are shown in Table 8.

Table 8. Confusion matrix obtained using several state-of-the-art networks.

Networks	AlexNet					GoogLeNet					SqueezeNet					ResNet-50					ResNet-101				
Classes	C	D	H	M	R	C	D	H	M	R	C	D	H	M	R	C	D	H	M	R	C	D	H	M	R
C	204	0	0	3	1	201	3	0	2	2	203	0	0	5	0	207	1	0	0	0	207	0	0	0	1
D	3	201	0	0	1	0	200	0	2	3	0	205	0	0	0	0	205	0	0	0	2	201	0	0	2
H	0	0	40	0	0	0	0	40	0	0	0	0	40	0	0	0	0	40	0	0	0	0	40	0	0
M	3	0	0	221	0	2	5	0	216	1	1	3	0	220	0	2	1	0	221	0	3	1	0	220	0
R	0	2	0	0	187	0	1	0	0	188	3	11	0	2	173	0	0	0	0	189	3	0	0	0	186

The confusion matrices of all architectures showed that the rust class was the most distinguishable among other guava diseases. If we discuss the AlexNet model results, four wrong cases were predicted as wrong in the rust and mummification classes, four wrong predictions were found for the dot class, where the wrong predictions lied on canker and rust. There were three wrong predictions for mummification, in the class of canker, and there were two wrong predictions in the rust class; these wrong predictions were two for dot. The second GoogLeNet architecture made 201 correct predictions in the canker class, and seven wrong predictions lied in the dot, mummification, and rust class, while no prediction lied in the healthy class. The five wrong predictions for the dot class were two predicted as mummification and three was rust, whereas 200 were predicted as right. This makes it one case less accurate than AlexNet, as that made four wrong predictions in the dot class. Mummification was predicted as nine wrong classes in the canker, dot, mummification, and rust categories, where two-hundred sixteen cases were rightly predicted. In rust, 188 cases were rightly predicted. One was predicted wrongly in the dot class. It was highly more efficient than AlexNet was in this category, as that predicted two wrong cases in the dot class, and GoogLeNet predicted only one wrong. The SqueezeNet architecture model made five wrong predictions, with two-hundred and three correct predictions of the canker class.

The five wrong predictions were in the mummification class. In the dot class, there was no wrong prediction, and all 205 test instances were rightly predicted. In mummification, there were four wrongly predicted cases and two-hundred and twenty rightly predicted cases. Most were predicted in the dot class with three instances, with one predicted in the canker class. Rust had 16 wrong cases, and most were in the dot class—11 out of 16. ResNet-50 was similar to ResNet-101, where the difference was mainly of several layers and its parameters. ResNet-50 predicted one wrong cases for the canker class with one wrong prediction in dot, and two-hundred and seven were correctly predicted. All dot class predictions (205) were correctly predicted in the dot class. The mummification class in this model's predictions had 221 correct predictions in dot. It has one wrong and two wrong predictions in the canker class. ResNet-101 also had higher accuracy results as compared to those of all other models. According to its confusion matrix, it made one wrong prediction for the canker class into the rust class. As compared to ResNet-50, it had one wrong case prediction, but in a different class. Regarding the dot class, ResNet-50 made no wrong predictions, and ResNet-101 made four wrong predictions. In the third case of the mummification class, there were four wrong predictions, and ResNet-50 made three wrong predictions. There were three wrong predictions in the last class, rust. There were 186 correct predictions for ResNet-101.

The above analysis of the five models shows that ResNet-50 had an overall high rate of correct predictions; for wrong predictions in the dot class data, it was highly difficult for each model, as it was predicted as the wrong class in most cases. The other classes also misled the models, where the most challenging and least robust class was dot. Therefore, the dot class may need more confident and robust approaches to classify it from other classes. The mummification class had no wrong predictions in ResNet-50 and -101, where it only had a higher rate of wrong predictions in the cases of SqueezeNet with four, where the two remaining models also did not make very many accurate predictions for this class. The healthy class overall in all models remained accurate with no wrong prediction by any model. It made the normal class easily distinguishable by any model. However, the challenge was differentiating the guava disease categories.

5. Conclusions

Guava is an important plant to monitor with the growing population; its production demand is also increasing. Pakistan is a leading global guava producer. Hence, for automatic monitoring, the study proposed a DL-based guava disease detection system. Data were preprocessed and enhanced using a color histogram and unsharp masking method. Enhanced data were then augmented over the nine angles using the affine transformation method—augmented enhanced data used by five DL networks by altering their last layers. The AlexNet, GoogLeNet, SqueezeNet, ResNet-50, and ResNet-101 architectures were used. The results of all networks showed adequate measurements, and ResNet-101 was the most accurate model. Future work should use more data augmentation methods such as generative adversarial networks. Other federated-learning-based DL architectures can be applied for classification to obtain more robust and confident results for this Pakistani guava disease dataset.

Author Contributions: Conceptualization, A.M.M., S.A.K., T.M., H.T.R., A.A.A. and M.A.A.; funding acquisition, A.M.M., A.A.A. and M.A.A.; methodology, A.M.M., S.A.K., T.M., H.T.R., A.A.A. and M.A.A.; software, A.M.M. and S.A.K.; visualization, H.T.R.; writing—original draft, A.M.M., S.A.K., T.M., H.T.R., A.A.A. and M.A.A.; supervision, H.T.R. All authors have read and agreed to the published version of the manuscript.

Funding: The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through Research Group No. RG-1441-425.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare that there is no conflict of interest.

References

1. Hunter, M.C.; Smith, R.G.; Schipanski, M.E.; Atwood, L.W.; Mortensen, D.A. Agriculture in 2050: Recalibrating targets for sustainable intensification. *Bioscience* **2017**, *67*, 386–391. [CrossRef]
2. Mirvakhabova, L.; Pukalchik, M.; Matveev, S.; Tregubova, P.; Oseledets, I. Field heterogeneity detection based on the modified FastICA RGB-image processing. *J. Phys. Conf. Ser.* **2018**, *1117*, 012009. [CrossRef]
3. Bose, T.; Mitra, S. Guava. In *Fruits Tropical and Subtropical*; Bose, T.K., Ed.; Naya Udyog: Calcutta, India, 1990; pp. 280–303.
4. Almadhor, A.; Rauf, H.T.; Lali, M.I.U.; Damaševičius, R.; Alouffi, B.; Alharbi, A. AI-Driven Framework for Recognition of Guava Plant Diseases through Machine Learning from DSLR Camera Sensor Based High Resolution Imagery. *Sensors* **2021**, *21*, 3830. [CrossRef] [PubMed]
5. Amusa, N.; Ashaye, O.; Oladapo, M.; Oni, M. Guava fruit anthracnose and the effects on its nutritional and market values in Ibadan, Nigeria. *World J. Agric. Sci.* **2005**, *1*, 169–172. [CrossRef]
6. Al Haque, A.F.; Hafiz, R.; Hakim, M.A.; Islam, G.R. A Computer Vision System for Guava Disease Detection and Recommend Curative Solution Using Deep Learning Approach. In Proceedings of the 2019 22nd International Conference on Computer and Information Technology (ICCIT), Dhaka, Bangladesh, 18–20 December 2019; pp. 1–6.
7. Keith, L.M.; Velasquez, M.E.; Zee, F.T. Identification and characterization of *Pestalotiopsis* spp. causing scab disease of guava, Psidium guajava, in Hawaii. *Plant Dis.* **2006**, *90*, 16–23. [CrossRef]
8. Pachanawan, A.; Phumkhachorn, P.; Rattanachaikunsopon, P. Potential of Psidium guajava supplemented fish diets in controlling Aeromonas hydrophila infection in tilapia (*Oreochromis niloticus*). *J. Biosci. Bioeng.* **2008**, *106*, 419–424. [CrossRef]
9. Sonka, M.; Hlavac, V.; Boyle, R. *Image Processing, Analysis, and Machine Vision*; Cengage Learning: Boston, MA, USA, 2014.
10. Somov, A.; Shadrin, D.; Fastovets, I.; Nikitin, A.; Matveev, S.; Hrinchuk, O. Pervasive agriculture: IoT-enabled greenhouse for plant growth control. *IEEE Pervas. Comput.* **2018**, *17*, 65–75. [CrossRef]
11. Bharathi Nirujogi, D.M.M.; Naidu, L.N.; Reddy, V.K.; Sunnetha, D.S.; Devi, P.R. Influence of carrier based and liquid biofertilizers on yield attributing characters and yield of guava cv. Taiwan White. *Pharma Innov. J.* **2021**, *10*, 1157–1160.
12. Sain, S.K. *AESA BASED IPM Package for Guava*. National Institute of Plant Health Management: Telangana, India, 2014.
13. Ruehle, G.; Brewer, C. *The FDA Method. Official Method of US Food and Drug Administration, US Department of Agriculture, and National Assn. of Insecticide and Disinfectant Manufacturers for Determination of Phenol Coefficients of Disinfectants*; MacNair-Dorland Co.: New York, NY, USA, 1941; pp. 189–201.
14. Misra, A. Guava diseases—their symptoms, causes and management. In *Diseases of Fruits and Vegetables: Volume II*; Springer: Berlin/Heidelberg, Germany, 2004; pp. 81–119.
15. Shadrin, D.; Pukalchik, M.; Uryasheva, A.; Tsykunov, E.; Yashin, G.; Rodichenko, N.; Tsetserukou, D. Hyper-spectral NIR and MIR data and optimal wavebands for detection of apple tree diseases. *arXiv* **2020**, arXiv:2004.02325.
16. Raza, S.A.; Ali, Y.; Mehboob, F. Role of agriculture in economic growth of Pakistan. *Int. Res. J. Financ. Econ.* **2012**, 180–186.
17. Pariona, A. Top Guava Producing Countries in the World. *Worldatlas*. 2017. Available online: https://www.worldatlas.com/articles/top-guava-producingcountries-in-the-world.html (accessed on 17 July 2018).
18. Pujari, J.D.; Yakkundimath, R.; Byadgi, A.S. Grading and classification of anthracnose fungal disease of fruits based on statistical texture features. *Int. J. Adv. Sci. Technol.* **2013**, *52*, 121–132.
19. Gilligan, C.A. Sustainable agriculture and plant diseases: An epidemiological perspective. *Philos. Trans. R. Soc. B Biol. Sci.* **2008**, *363*, 741–759. [CrossRef]
20. Adenugba, F.; Misra, S.; Maskeliūnas, R.; Damaševičius, R.; Kazanavičius, E. Smart irrigation system for environmental sustainability in Africa: An Internet of Everything (IoE) approach. *Math. Biosci. Eng.* **2019**, *16*, 5490–5503. [CrossRef]
21. Zhou, R.; Kaneko, S.; Tanaka, F.; Kayamori, M.; Shimizu, M. Disease detection of Cercospora Leaf Spot in sugar beet by robust template matching. *Comput. Electron. Agric.* **2014**, *108*, 58–70. [CrossRef]
22. Ali, H.; Lali, M.; Nawaz, M.Z.; Sharif, M.; Saleem, B. Symptom based automated detection of citrus diseases using color histogram and textural descriptors. *Comput. Electron. Agric.* **2017**, *138*, 92–104. [CrossRef]
23. Stasenko, N.; Chernova, E.; Shadrin, D.; Ovchinnikov, G.; Krivolapov, I.; Pukalchik, M. Deep Learning for improving the storage process: Accurate and automatic segmentation of spoiled areas on apples. In Proceedings of the 2021 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Glasgow, UK, 17–20 May 2021; pp. 1–6.
24. Almutiry, O.; Ayaz, M.; Sadad, T.; Lali, I.U.; Mahmood, A.; Hassan, N.U.; Dhahri, H. A Novel Framework for Multi-Classification of Guava Disease. *CMC—Comput. Mater. Continua* **2021**, *69*, 1915–1926. [CrossRef]
25. Gavhale, K.R.; Gawande, U. An overview of the research on plant leaves disease detection using image processing techniques. *IOSR J. Comput. Eng. (IOSR-JCE)* **2014**, *16*, 10–16. [CrossRef]
26. Deshpande, T.; Sengupta, S.; Raghuvanshi, K. Grading & identification of disease in pomegranate leaf and fruit. *Int. J. Comput. Sci. Inf. Technol.* **2014**, *5*, 4638–4645.
27. Thilagavathi, M.; Abirami, S. Application of image processing in diagnosing guava leaf diseases. *Int. J. Sci. Res. Manag. (IJSRM)* **2017**, *5*, 5927–5933.

28. Khan, M.A.; Lali, M.I.U.; Sharif, M.; Javed, K.; Aurangzeb, K.; Haider, S.I.; Altamrah, A.S.; Akram, T. An optimized method for segmentation and classification of apple diseases based on strong correlation and genetic algorithm based feature selection. *IEEE Access* **2019**, *7*, 46261–46277. [CrossRef]
29. Gasanov, M.; Petrovskaia, A.; Nikitin, A.; Matveev, S.; Tregubova, P.; Pukalchik, M.; Oseledets, I. Sensitivity analysis of soil parameters in crop model supported with high-throughput computing. In *International Conference on Computational Science*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 731–741.
30. Gasanov, M.; Merkulov, D.; Nikitin, A.; Matveev, S.; Stasenko, N.; Petrovskaia, A.; Pukalchik, M.; Oseledets, I. A New Multi-objective Approach to Optimize Irrigation Using a Crop Simulation Model and Weather History. In *International Conference on Computational Science*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 75–88.
31. Arivazhagan, S.; Shebiah, R.N.; Ananthi, S.; Varthini, S.V. Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features. *Agric. Eng. Int. CIGR J.* **2013**, *15*, 211–217.
32. Mohanty, S.P.; Hughes, D.P.; Salathé, M. Using deep learning for image-based plant disease detection. *Front. Plant Sci.* **2016**, *7*, 1419. [CrossRef]
33. Ferentinos, K.P. Deep learning models for plant disease detection and diagnosis. *Comput. Electron. Agric.* **2018**, *145*, 311–318. [CrossRef]
34. Saleem, M.H.; Potgieter, J.; Arif, K.M. Plant disease detection and classification by deep learning. *Plants* **2019**, *8*, 468. [CrossRef]
35. Türkoğlu, M.; Hanbay, D. Plant disease and pest detection using deep learning-based features. *Turk. J. Electr. Eng. Comput. Sci.* **2019**, *27*, 1636–1651. [CrossRef]
36. Barbedo, J.G.A. Plant disease identification from individual lesions and spots using deep learning. *Biosyst. Eng.* **2019**, *180*, 96–107. [CrossRef]
37. Wang, G.; Sun, Y.; Wang, J. Automatic image-based plant disease severity estimation using deep learning. *Comput. Intell. Neurosci.* **2017**, *2017*, 2917536.
38. Abbas, A.; Jain, S.; Gour, M.; Vankudothu, S. Tomato plant disease detection using transfer learning with C-GAN synthetic images. *Comput. Electron. Agric.* **2021**, *187*, 106279. [CrossRef]
39. Patil, B.V.; Patil, P.S. Computational method for Cotton Plant disease detection of crop management using deep learning and internet of things platforms. In *Evolutionary Computing and Mobile Sustainable Networks*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 875–885.
40. Arsenovic, M.; Karanovic, M.; Sladojevic, S.; Anderla, A.; Stefanovic, D. Solving current limitations of deep learning based approaches for plant disease detection. *Symmetry* **2019**, *11*, 939. [CrossRef]
41. Sabrol, H.; Kumar, S. Plant leaf disease detection using adaptive neuro-fuzzy classification. In *Science and Information Conference*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 434–443.
42. Oppenheim, D.; Shani, G.; Erlich, O.; Tsror, L. Using deep learning for image-based potato tuber disease detection. *Phytopathology* **2019**, *109*, 1083–1087. [CrossRef]
43. Ramesh, S.; Hebbar, R.; Niveditha, M.; Pooja, R.; Shashank, N.; Vinod, P. Plant disease detection using machine learning. In Proceedings of the 2018 International Conference on Design Innovations for 3Cs Compute Communicate Control (ICDI3C), Bangalore, India, 25–28 April 2018; pp. 41–45.
44. Nagaraju, M.; Chawla, P. Systematic review of deep learning techniques in plant disease detection. *Int. J. Syst. Assur. Eng. Manag.* **2020**, *11*, 547–560. [CrossRef]