

Article

Adaptive Thresholding of CNN Features for Maize Leaf Disease Classification and Severity Estimation

Harry Dzingai Mafukidze ^{1,*} , Godliver Owomugisha ² , Daniel Otim ², Action Nechibvute ¹, Cloud Nyamhere ¹ and Felix Mazunga ¹

¹ Department of Applied Physics and Telecommunications, Midlands State University, Senga Road, Gweru P Bag 9055, Zimbabwe

² Faculty of Engineering, Busitema University, Tororo P.O. Box 236, Uganda

* Correspondence: mafukidzehd@staff.msu.ac.zw

Abstract: Convolutional neural networks (CNNs) are the gold standard in the machine learning (ML) community. As a result, most of the recent studies have relied on CNNs, which have achieved higher accuracies compared with traditional machine learning approaches. From prior research, we learned that multi-class image classification models can solve leaf disease identification problems, and multi-label image classification models can solve leaf disease quantification problems (severity analysis). Historically, maize leaf disease severity analysis or quantification has always relied on domain knowledge—that is, experts evaluate the images and train the CNN models based on their knowledge. Here, we propose a unique system that achieves the same objective while excluding input from specialists. This avoids bias and does not rely on a “human in the loop model” for disease quantification. The advantages of the proposed system are many. Notably, the conventional system of maize leaf disease quantification is labor intensive, time-consuming and prone to errors since it lacks standardized diagnosis guidelines. In this work, we present an approach to quantify maize leaf disease based on adaptive thresholding. The experimental work of our study is in three parts. First, we train a wide variety of well-known deep learning models for maize leaf disease classification, then we compare the performance of the deep learning models and finally extract the class activation heatmaps from the prediction layers of the CNN models. Second, we develop an adaptive thresholding technique that automatically extracts the regions of interest from the class activation maps without any prior knowledge. Lastly, we use these regions of interest to estimate image leaf disease severity. Experimental results show that transfer learning approaches can classify maize leaf diseases with up to 99% accuracy. With a high quantification accuracy, our proposed adaptive thresholding method for CNN class activation maps can be a valuable contribution to quantifying maize leaf diseases without relying on domain knowledge.

Keywords: CNN; transfer learning; class activation heatmap; adaptive thresholding



Citation: Mafukidze, H.D.; Owomugisha, G.; Otim, D.; Nechibvute, A.; Nyamhere, C.; Mazunga, F. Adaptive Thresholding of CNN Features for Maize Leaf Disease Classification and Severity Estimation. *Appl. Sci.* **2022**, *12*, 8412. <https://doi.org/10.3390/app12178412>

Academic Editor: Hui Yuan

Received: 24 June 2022

Accepted: 16 August 2022

Published: 23 August 2022

Publisher’s Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Crop diseases remain a global challenge to food security. According to [1], 20 to 40% of global crop production is lost to pests and diseases annually. Accordingly, the application of machine learning and deep learning models for identifying and quantifying crop diseases has emerged as one of the important themes of the 21st century, especially in precision agriculture [2]. This follows multiple efforts to automate disease monitoring in agriculture, for example [3,4]. Most deep learning models in the literature have achieved tremendous success in general image classification tasks. Therefore, it is natural to leverage the advances made in this field to meet the demands of precision agriculture [5]. Central to this is the selection of leaf image features that contain sufficient data for training the models, or the use of specialized deep learning models to extract features on the fly using a combination of various convolution and pooling layers. In most cases, deep learning

models are selected based on meeting the requirements of the project [6,7], such as training time and computational resources.

Maize leaf disease quantification, or maize leaf severity analysis, is a very interesting research problem that seeks to determine the extent to which a leaf has been affected by disease [8–10]. This process relies on trained experts to visually estimate the affected area and assign a severity label. In the literature, multiple works have successfully applied multi-label automated visual classification frameworks to identify diseases and evaluate damage based on a human-defined scale [11]. The intuition is that an image can be associated with multiple labels. For example, a diseased leaf image can be classified as blight and then categorized as “low severity”, “medium severity” or “high severity”. Such methods rely on nominal scales for quantification, and have been used by Owomugisha et al. [11], among others, to indicate various levels of disease severity. This method of quantifying leaf diseases has been widely accepted to solve the problem of leaf disease quantification; however, with thousands of leaves to annotate, it becomes tedious and prone to errors. Although this solution is very popular in the literature, it cannot be applied to solve all disease quantification problems since leaf diseases have patterns that are statistical in nature, and hence more challenging for a human eye to assign a severity label.

Machine learning methods that have been applied to solve maize leaf disease classification problems can be grouped into two sets: hand-crafted feature-based models [12] and automatic feature extraction models [13]. The parameters of hand-crafted feature-based models are estimated mainly based on the domain knowledge of the problem and the statistical characteristics of the leaf image. However, one of the biggest challenges, in general, of relying on statistical characteristics such as mean, standard deviation and variance, among others, is that: (i) variations may be indistinguishable for leaf diseases with low intraclass and high interclass variability [14]; (ii) different diseases may have similar effects on leaves; (iii) a single leaf may have multiple diseases, and (iv) disease symptoms may vary. On the other hand, automatic feature extraction models, particularly CNNs, require large amounts of data to learn the discriminative features of the image classes [14], which is a major limiting factor considering the difficulties of gathering the data.

This work is motivated by the ability to quantify maize leaf diseases based on the information contained in the class activation maps (CAM) of the CNN. Unlike qualitative leaf disease severity estimation methods that rely mainly on domain knowledge, this approach uses the same deep learning model for leaf disease classification and leaf disease quantification, as described in Section 3. The critical component of our system is based on identifying which regions of the diseased leaf image have contributed more to the final decision of the CNN, and we express those regions as a percentage of the total leaf area to estimate disease severity. We compare the performance of the proposed method with state-of-the-art image segmentation techniques for leaf disease estimation. The proposed system can assign a severity estimate as a ratio without prior information from experts, and identify the following maize leaf conditions with high accuracy: Northern Corn Leaf Blight, Gray Leaf Spot, common corn rust and healthy leaves.

The main goals of the suggested method are: (1) Use transfer learning to reuse CNN models to assign the correct maize leaf disease class to the given image. (2) Use CNN class activation heatmaps to derive a percentage severity estimate based on the suggested adaptive thresholding technique. Our hypothesis is that the extracted ROI from the class activation heatmap correlates with the area covered by the disease in the original leaf image. Therefore, the area of the ROI as a percentage of the total leaf area gives us an estimate of the disease severity. We summarize the main contributions of this work as follows:

1. We propose a novel method for simultaneously classifying and quantifying maize leaf diseases without any prior knowledge of the lesions.
2. To improve the classification accuracy of maize leaf disease identification, this work investigates several models: custom CNN, Inception V3 [15], DenseNet 121 [16], DenseNet 201 [16], MobileNetV2 [17], VGG16 [18], EfficientNetB0 [19], ResNet50 [20] and LeNet [21]. The performance of the custom CNN is further examined by compar-

ing the effect of regularization techniques such as batch normalization and dropout operations. Finally, we compare the experimental results of the custom CNN with those of the transfer learning techniques to determine the effect of CNN depth and number parameters on the performance of the model.

3. We propose a unique and computationally efficient adaptive thresholding method for quantifying the diseased regions of leaf images in an unsupervised manner. The advantage of this method is that it uses spatial relationships between the pixels to predict regions that are affected by the disease.
4. Experimental results on the dataset demonstrate two things. First, the transfer learning techniques achieve higher classification accuracies of 99%. Second, our proposed adaptive thresholding approach performs well, giving good results, and therefore is a good approach for maize leaf disease quantification.
5. Comparison evaluation with the most common maize leaf disease severity analyses such as Bernser's, Niblack's and Sauvola's thresholding reveal that our approach is robust and gives good results.
6. We created an augmented dataset consisting of 46,898 samples from the original PlantVillage [22] dataset, available at: https://github.com/harryD1/maize_leaf_diseases_classification_and_quantification (accessed on 1 June 2022).

We structure this paper as shown in Figure 1. We begin by giving an overview of related studies on leaf disease classification and severity estimation in Section 2. Section 3 presents the materials and methods that are used in the experiments. In Section 4, we present the results and performance of the methods based on their ability to predict the correct class among four maize disease classes. In Section 5, we present the discussion, give an outline of the work's limitations, and give future directions of our research. Section 5 concludes the work.

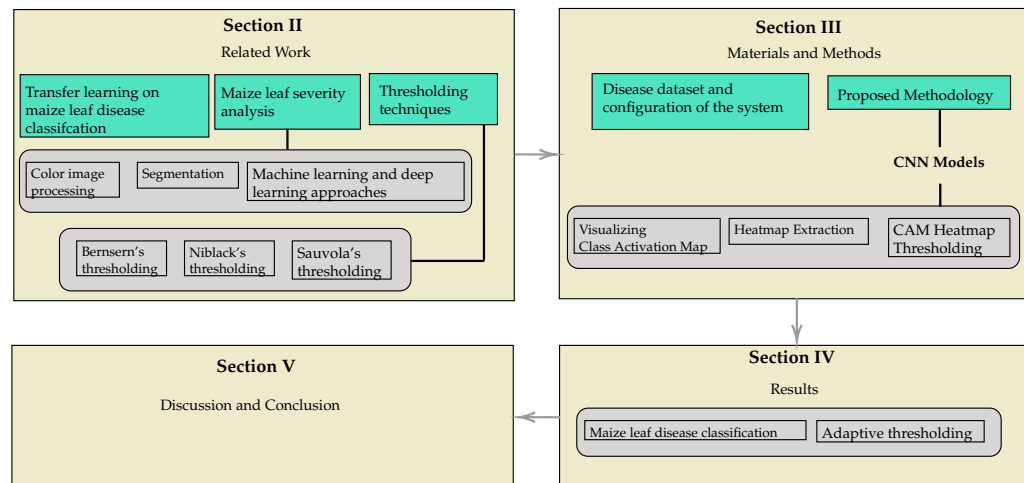


Figure 1. Paper layout.

2. Related Work

In this section, we give an overview of the research and methods that are relevant to the present study. In Section 2.1, we summarize transfer learning techniques that have been used for maize leaf disease classification. Then, in Section 2.2, we give a summary of the most recent work on maize leaf disease severity estimation. Finally, we conclude by studying the current state-of-the-art thresholding techniques in Section 2.2.3.

2.1. Transfer Learning on Maize Leaf Disease Classification

In its simplest terms, transfer learning is adapting or reusing the knowledge that the CNN has learned previously and applying that knowledge to a new set of problems [23]. For example, in Figure 2, we can have a CNN such as VGG16 that has learned to recognize objects from the ImageNet database and then use all or part of that knowledge to recognize

the maize leaf diseases in our database. In most cases, we transfer the knowledge gained from a lot of data to a problem that has relatively little training data. This is helpful because most image recognition problems from popular datasets such as ImageNet (14 million images and 1000 classes), MS-COCO (1.5 million images and 1000 classes) and CIFAR10 (60,000 images and 10 classes) rely on similar low-level features, and the CNN will have learned the basic structure of image objects, thereby increasing the learning rate on the new dataset.

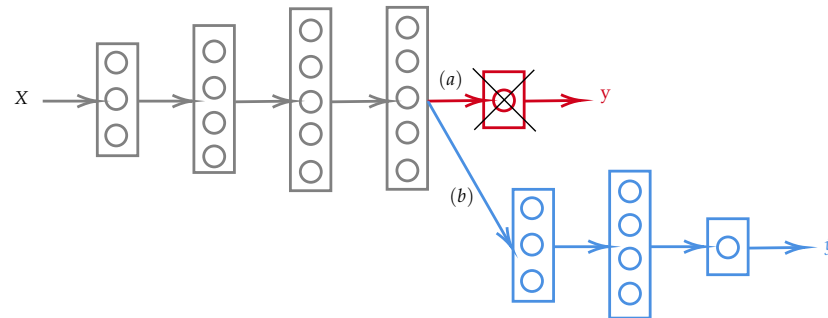


Figure 2. Illustration of the transfer learning process. Transfer learning involves adapting a pre-trained CNN model to suit the requirements of a new problem. In this case, consider the original model (a) that has been trained to recognize objects from a dataset such as ImageNet. If X is the input image and y is the predicted class of that model, then by removing some layers, including the last layer from the original model, we can obtain an architecturally different model (b) through adding more layers for learning new information from a different dataset to make new predictions $\hat{y}$.

Transfer learning has been used in several studies to address some of the limitations of deep learning models in generic image recognition problems. From the literature, it is clear that transfer learning leverages less training data, decreases training time and greatly improves the performance of deep learning models, as we shall see in this paper.

Most early studies in this field focus on the use of handcrafted features for leaf disease detection and classification. Notably, the work by Panigrahi K.P et al. proposes an early maize leaf disease detection and classification system using machine learning [24]. In their work, they extract distinctive maize leaf features such as shape, color and texture and apply machine learning techniques such as Naive Bayes, Decision Tree, K-Nearest Neighbor, Support Vector Machine and Random Forest to classify the leaf images [24]. Their results demonstrated that the Random Forest classifier performs well, giving accuracies of over 79%, while the Support Vector Machine achieves a lower accuracy of just over 77%.

In a similar study, Alehegn et al. developed maize leaf disease recognition and classification using only a Support Vector Machine [25]. They extracted similar features (texture, color and morphology), but in this instance, the authors managed to obtain good accuracies of 95% from a dataset of only 800 images. This can be attributed to a small dataset.

These results and other works [26–29] demonstrate the feasibility of using handcrafted features for crop leaf disease classification; however, the limitations of relying on such techniques naturally include: (1) the need for expert knowledge, (2) lack of robustness and (3) extensive time needed to identify relevant features from leaf images [30].

In order to improve performance, some authors have presented a unique approach to identifying maize leaf diseases. They consider a learning paradigm that automatically learns the features of maize leaf diseases without the need for expert knowledge. For instance, DeChant et al. developed an automated system to identify Northern Leaf Blight in maize plants using deep learning [31]. They trained several CNNs to identify the presence of the disease on maize leaf images. Although their task was a binary classification problem, the authors managed to achieve a classification accuracy of 96% on the test set.

A common technique in the literature is to combine a deep learning feature extraction module with a supervised classifier such as SVM [32], Decision Tree [33] or K-Nearest Neighbor [33,34]. This kind of architecture leverages the convolutional operation of the

feature extraction layers and the power of the supervised classifier to improve classification performance. This combination has already offered good results in generic image classification tasks that require modelling spatial dependencies [35]. However, significant results were demonstrated by authors who introduced the concept of transfer learning for plant leaf disease classification [36].

This is a promising approach to address the requirements of maize leaf disease identification: First, performance is greatly improved compared with developing models from scratch [36]. Second, transfer learning from a pre-trained network does not require a lot of training data compared with training a model from the ground up, and the training time is significantly lower. To customize the base convolutional network, we first delete the classifier from the model, and then we add a new classifier that will be trained from the ground up so that we can reuse the historical features learned previously from the initial problem, as shown in Figure 2.

2.2. Maize Leaf Severity Analysis

Bock et al. [37] define severity analysis as determining the total area of the leaf that is affected by the disease, and such calculations can be estimated by using the leaf color or shape features [38]. Typically, the procedure follows a three-step process to estimate the severity: image pre-processing, image segmentation and image analysis. Most disease severity estimation work in the literature can be grouped into one of two sets: (1) color image processing techniques or (2) machine learning and deep learning methods. In color image processing, researchers use digital image processing to segment the diseased parts of a leaf and determine the total affected area, while in (2), researchers extract the relevant features of the leaf diseases [38] and pass the information to either machine learning or deep learning algorithms. Some authors have reported significant progress using third-party image processing software [39].

2.2.1. Color Image Processing

The motivation for using color image processing techniques for leaf disease severity analysis stems from the fact that it is possible to segment the leaf image based on the color of the pixels [39]. Basically, we subdivide the image into its constituent regions or objects until all the regions of interest have been isolated [39]. In what follows, we review works that adopt color changes to segment leaf images.

In 2006, Barbora et al. assessed the impact of the two-spotted spider mite on host plants using third-party image processing software (Mathematica). In their work, they converted the leaf images to greyscale and then measured the total damaged area based on a pre-determined threshold. Then, they expressed the result as a ratio of the total number of pixels of the leaf image to obtain a severity estimate [40]. Their results show that this method is reliable and can identify significant differences in pixel damage due to different concentrations of spider mites.

2.2.2. Segmentation

Several studies have used a threshold [40–47] for leaf disease severity analysis. However, such techniques generally suffer from low classification rates due to poor sensitivity. Namita addresses this problem by using an adaptive threshold for automatic segmentation. In their work, they first convert the RGB color images to Lab color space, and then the adaptive threshold value is obtained from the maximum intensity values of the blue color channel. Finally, disease quantification is performed by taking the ratio of the diseased parts to the ratio of the entire leaf image. Their method is computationally efficient, and, above all, delivers superior results even on poor-quality images. In particular, they were able to achieve a 99% classification accuracy on cherry crop images.

2.2.3. Machine Learning and Deep Learning

Diseases mainly alter the color, shape and intensity of leaves [48]. It is, therefore, reasonable to extract meaningful information from such deformations and use it as features to represent the respective diseases. The most common features that authors use from the color moments are the mean, standard deviation and skewness, as defined in Equations (1)–(3) [48]:

$$\mu_i = \frac{1}{N} \sum_{j=1}^N f_{ij} \quad (1)$$

$$\sigma_i = \left[\frac{1}{N} \sum_{j=1}^N (f_{ij} - \mu_i)^2 \right]^{\frac{1}{2}} \quad (2)$$

$$\gamma_i = \left[\frac{1}{N} \sum_{j=1}^N (f_{ij} - \mu_i)^3 \right]^{\frac{1}{3}} \quad (3)$$

where μ_i , σ_i and γ_i ($i = 1, 2, 3 \dots$) represent the mean, standard deviation and skewness of each image, respectively [48].

Several authors have also reported success with features extracted from the histograms of diseased leaf images. Without a doubt, the histogram of an image provides important statistics that are relevant in image processing tasks [49]. They are simple to calculate, making them a tool of choice in near real-time digital image processing applications [49]. For example, we can use Equations (4) and (5) to extract some statistical variables that can help us segment the digital image. Developing from the previous intuition of Section 2.2.3, if we have a grayscale image I with a total of N pixels and K possible intensity values $0 \leq g < K$, we can compute the mean and variance from the histogram of the image as follows [49]:

$$\mu_I = \frac{1}{N} \cdot \sum_{u,v} I(u,v) = \frac{1}{N} \cdot \sum_{g=0}^{K-1} g \cdot h(g), \quad (4)$$

$$\sigma_I^2 = \frac{1}{N} \cdot \sum_{u,v} [I(u,v) - \mu_I]^2 = \frac{1}{N} \cdot \sum_{g=0}^{K-1} (g - \mu_I)^2 \cdot h(g). \quad (5)$$

where μ_I and σ_I^2 represent the mean and the variance of the image, respectively.

In general, extracting leaf features such as color, shape or intensity is the basis of any leaf disease quantification problem. In fact, a majority of digital image processing techniques and third-party software packages for severity analysis rely on using one or more such features.

3. Materials and Methods

This section describes the materials and methods we used for our work. First, we describe our dataset, then the CNN models, and lastly, the proposed algorithm for maize leaf disease severity estimation.

3.1. Disease Dataset and Configuration of the System

In this study, we approach the problem of maize leaf disease quantification by building and testing a wide range of deep learning models on the publicly available PlantVillage maize leaf dataset. The original PlantVillage dataset for maize leaf diseases contains 4188 images spread across 4 different maize leaf diseases. In total, 1162 of the images are healthy, 574 are Gray Leaf Spot, 1306 are Common Rust, and 1146 are Blight. Images in the dataset are raw RGB color images of different sizes. Since deep learning algorithms require large amounts of training data, we build an augmented dataset from the existing PlantVillage dataset by applying six different augmentation methods, and we obtain a total of 46,896 images. We apply the following transforms:

- Resizing— (224×224) ;
- Shear—level 0.2;
- Rotation—90 degrees;
- Flipping—up_down flip;
- Saturation;
- Mean filtering;
- Flipping meaned image;
- Cropping—central_fraction = 0.8;
- Cropping—central_fraction = 0.5.

The resulting augmented PlantVillage dataset contains 9379 training images and 2345 test images for each class. Figure 3 shows the histogram of the train–test distribution of the final images, and Figure 4 shows the four different maize leaf disease classes randomly selected from the training set.

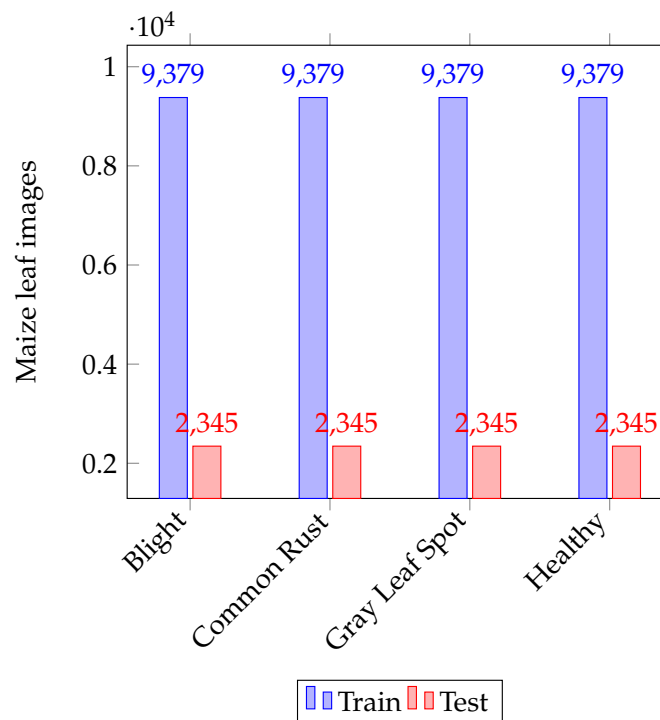


Figure 3. Training and test image distribution. The final dataset for maize leaf disease classification and quantification contains 9379 training images of each class and 2345 test images of each class. This represents an 80–20% train–test split. In total, there are 46,898 images in our augmented PlantVillage maize leaf disease dataset.

We then use this augmented dataset to train several well-known deep learning models, along with the proposed shallow CNN that contains only three convnet layers. The purpose of the CNN model is to classify the input image into the four maize leaf disease classes. The CNN identifies the specific class of the image by using class activation maps. The next process is leaf disease severity estimation using class activation heatmaps. The method accepts the class activation heatmap as a 2D matrix and then identifies the number of pixels that have contributed to the class decision. These pixels are expressed as a percentage of the total leaf area. Finally, the method returns the percentage area of the affected region.

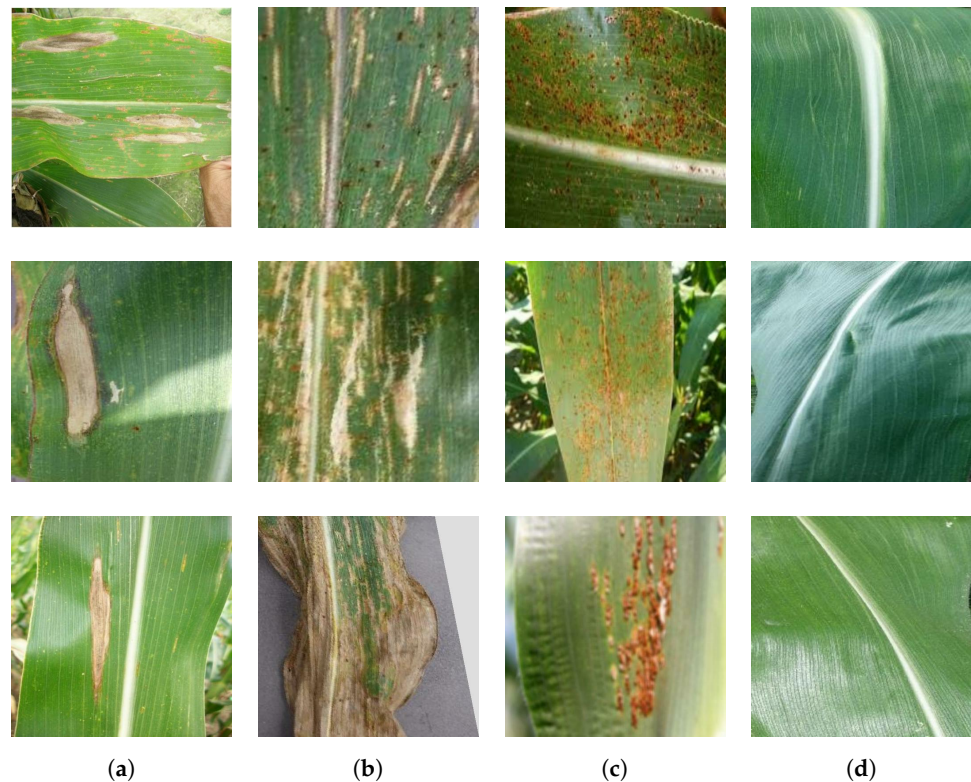


Figure 4. Example of maize leaf images from the PlantVillage dataset. The dataset contains the following classes: (a) Northern Corn Leaf Blight *Exserohilum turcicum*, (b) Gray Leaf Spot *Cercospora zae-maydis*, (c) Common Corn Rust *Puccinia sorghi* and (d) healthy maize leaf.

3.2. Proposed Methodology

In this section, we present the architecture and the methodology of our proposed maize leaf disease classification and quantification framework. Unlike other works, the framework presented here accepts raw RGB maize leaf images and outputs the predicted class as well as a severity estimate. Below are the steps taken to achieve the objective. A diagrammatic view of this framework is shown in Figure 5.

- Step 1: We train a wide variety of well-known deep learning models for the maize leaf disease classification problem. Then, we compare the performance of the deep learning models and extract the class activation heatmaps from the prediction layers of the CNN models.
- Step 2: We extract the class activation heatmap as a 2D matrix from the convnet layers and determine the area affected by the disease (region of interest).
- Step 3: We use these regions of interest to estimate image leaf disease severity.

3.2.1. Architecture of the Proposed CNN

Here, we define our custom CNN model. Most state-of-the-art CNNs are very deep, and are hence more complex. For example, VGG16 has 16 layers, while ResNet50 has 50 layers, and so on. In order to investigate the influence of CNN depth on classification accuracy, two separate CNN pipelines are implemented. The first consists of a shallow CNN (only three convolutional layers) and is trained from scratch, while the second uses deep CNN architectures (up to 250 convolutional layers) and employs transfer learning to speed up network training.

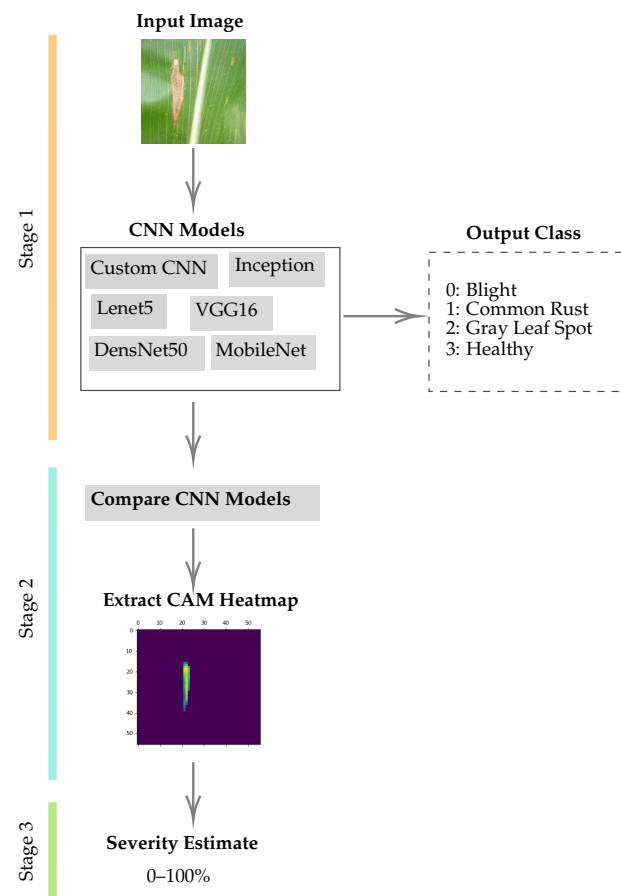


Figure 5. Research method of maize leaf disease classification and severity estimation using deep CNN features. As shown in this diagram, simultaneous maize leaf disease classification and quantification is the major component of our work, which comprise the following: CNN training, CAM heatmap extraction, and thresholding. First, several deep learning models are trained on the augmented PlantVillage maize leaf disease dataset. In Stage 2, the CAM extracted from the convnet layers from Stage 1 is obtained and eventually used for disease quantification. Although not illustrated in this diagram, the CAM can be treated as a 2D heatmap of pixels with different intensities. Bright pixels represent regions that have been identified by the CNN to be of great importance (regions of interest (ROI)), while darker regions correlate to a lack of features. In Stage 3, we use an adaptive thresholding technique that extracts the ROI from the heatmap of Stage 2 and represents that area as a percentage of the total leaf area.

Below summarizes the network architecture of the shallow CNN shown in Figure 6.

1. **Input image data:** The images in our dataset have different sizes of width $\times$ height. However, the pre-processing stage of the CNN models resizes the images to a 3D matrix of $224 \times 224 \times 3$. We have three channels since we are using RGB images.
2. **First hidden layer:** The first hidden convolutional layer is called *conv2d_1*. It has 32 feature maps of size 3×3 and a ReLu activation function.
3. **Second hidden layer:** The second hidden layer is called *conv2d_2*. It has 32 feature maps of size 3×3 and a ReLu activation function.
4. **Third hidden layer:** Referred to as *conv2d_3*, it has 64 filters of size 3×3 .
5. **Pooling layer:** The pooling layer is called *MaxPooling2D* and is added after every convolutional layer.
6. **Dropout layer:** A dropout layer is added to reduce overfitting; in this model, it has been set to exclude 30% of the neurons.
7. **Flatten layer:** This layer is added to convert the 2D feature maps to a 1D vector.
8. **First fully connected layer:** This has 128 neurons with a ReLu activation function.

9. Output layer: This has four neurons and a softmax activation function.

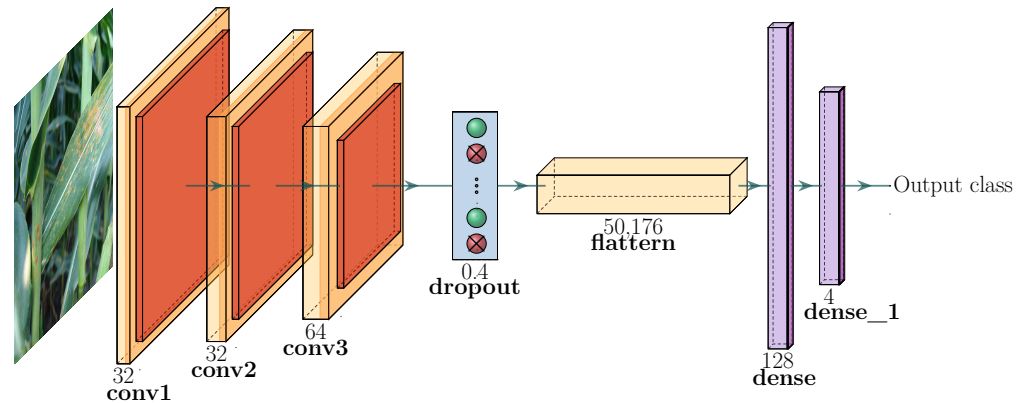


Figure 6. Architecture of the shallow CNN model for simultaneous maize leaf disease classification and quantification. From the left, the maize leaf image is processed by three convolutional layers that learn the disease features. A dropout layer is added to reduce overfitting before converting the feature maps into a 1D array through flattening. Finally, the fully connected layers consist of two dense layers (with a ReLu and softmax transfer function, respectively) to yield the class probabilities. This architecture produces about 6,451,812 trainable parameters. The neural network was drawn using the PlotNeuralNet LaTeX package [50].

3.2.2. Transfer Learning

Finally, transfer learning is applied to several pre-trained deep CNNs using the Keras and Tensorflow Python packages. The following parameters are used to train the network: base learning rate, 0.0001; momentum, 0.9; number of epochs, 500; and 80% training and 20% test samples. The PlantVillage maize leaf dataset had been augmented using several transforms (Section 3.1) to increase the size of the dataset to 46,896 images.

3.2.3. State-of-the-Art Thresholding Techniques

In what follows, we review some of the state-of-the-art thresholding techniques that have been developed for general image processing tasks.

Bernsern's Thresholding

The Bernsern's thresholding technique calculates an adaptive threshold for each heatmap position (u, v) using the maximum and minimum values of a sliding window. If $I_{max}(u, v)$ and $I_{min}(u, v)$ are the minimum and maximum values, respectively, found in an $n \times m$ sliding window, then the Bernsern's threshold, $T_{u,v}$ is given as [51]:

$$T_{u,v} = \frac{I_{min}(u, v) + I_{max}(u, v)}{2} \quad (6)$$

Equation (6) depends on the value of a local contrast, $c(u, v)$, as long as $c(u, v) = I_{max}(u, v) - I_{min}(u, v)$ is above a certain limit c_{min} . However, if the limit exceeds $c(u, v)$, then the pixels are assumed to belong to a single class [51].

Niblack's Thresholding

Unlike the previous method, Niblack's thresholding technique calculates an adaptive threshold for each heatmap position (u, v) using the mean $m(u, v)$ and the standard deviation $\sigma(u, v)$ of the pixels within an $n \times m$ sliding window. If k is an arbitrary constant that can take any value between 0 and 1, then Niblack's threshold $T_{u,v}$ is given as [52]:

$$T_{u,v} = m(u, v) + k \times \sigma(u, v) \quad (7)$$

From Equation (7), it is clear that the value of the threshold is affected by k and the size of the sliding window.

Sauvola's Thresholding

As shown in Equation (8), Sauvola's technique also uses the mean and the standard deviation to determine the adaptive threshold, just like Niblack's method. However, Sauvola's method adapts the threshold based on the contrast of the pixels in the sliding window [53].

$$T_{u,v} = m(u,v) \left[1 + k \left(\frac{\sigma(u,v)}{R} - 1 \right) \right] \quad (8)$$

where $m(u,v)$ is the mean, $0.2 \leq k \leq 0.5$ and $R = \max(\sigma(u,v))$ [53].

3.2.4. Visualizing the Class Activation Map (CAM)

Class activation mapping (CAM) is a technique for identifying the discriminative regions used by a particular class [54]. GradCam is a modification of CAM that seeks to address the limitations of CAM, particularly (i), the need for CAM to have a Global Average Pooling layer in the architecture, and (ii) the drawback of visualizing only the heatmap of the final layer. We need to compute the gradient of y^c with respect to the feature maps A of the last convolutional layer, i.e., $\frac{\delta y^{(c)}}{\delta A_{ij}^{(k)}}$. These gradients flowing back are global-average-pooled to obtain weights w_k^c , as shown in Equation 9.

$$w_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\delta y^{(c)}}{\delta A_{ij}^{(k)}} \quad (9)$$

where w_k^c captures the importance of feature map K for target class c .

The GradCam heatmap in Figure 7, is a weighted combination of feature maps with a ReLU.

$$L_{GradCam}^c = \text{ReLU} \left(\sum_{k=1}^K w_k^c A^k \right) \quad (10)$$

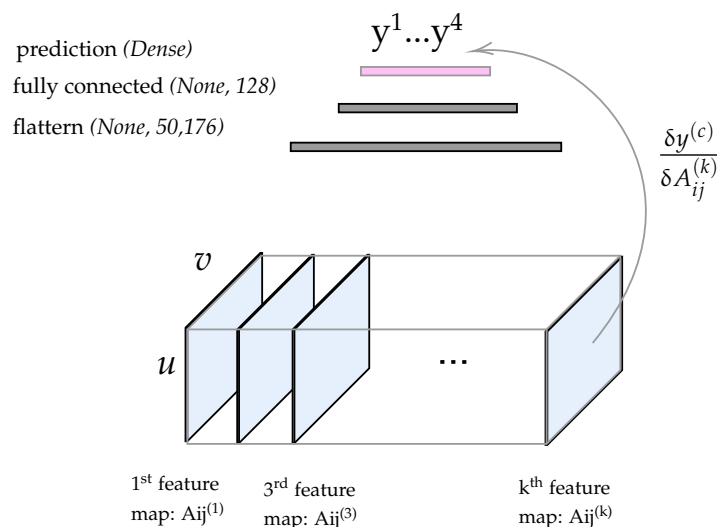


Figure 7. GradCam visualization. The last Conv layer can be thought of as the features of a classification model: $y^c = f(A^1, \dots, A^{512})$.

3.2.5. Heatmap Extraction

Equation (10) produces a heatmap that shows how the model makes predictions. This heatmap is a rectangular array of real numbers $y^c = f(A^1, \dots, A^{512})$ that can be used to visualize the diseased parts of an image and hence derive an estimate of the severity level. In other words, the brighter parts of the heatmap indicate the importance of those parts with respect to disease mapping. However, this heatmap not only contains the required regions of interest (ROI) but also the interference regions, which vary in magnitude.

3.2.6. Thresholding the Heatmap of the CAM

The aim of this algorithm is to isolate the region of interest from the background. Thus, given a heatmap image I , fixed thresholding groups the pixels of the heatmap into sets H_0 and H_1 , where H_0 contains all the elements less than or equal to a particular threshold T , and H_1 contains all the elements greater than the threshold, as shown in Equation (11):

$$(i, j) \in \begin{cases} H_0, & \text{if } I(i, j) \leq T. \\ H_1, & \text{if } I(i, j) > T. \end{cases} \quad (11)$$

However, this thresholding cannot adequately extract the ROI from the heatmap, for example when confronted with pixels that contain varying levels of interference, or ROIs that have lower values than the fixed threshold.

Therefore, to efficiently separate the ROIs from interference, we first create an $n \times m$ sliding window that moves through the entire 2D heatmap image. The sliding window contains reference cells, guard cells and the cell under test. We then determine an adaptive threshold value $T(m, n)$ for each sliding window position based on the average intensity of the reference cells, and then we compare this value with the magnitude of the cell under test (CUT) $x(i, j)$. If the magnitude of the CUT is greater than the threshold, we declare it a region of interest. We test all the cells of the heatmap until we get a clearer picture of all ROIs. We use the guard cells to create a buffer zone around the CUT. By doing this, we avoid the ROIs from leaking into the reference cells, which could affect the value of the threshold. We estimate the interference levels from the reference cells or neighboring cells of the j th sliding window using Equation (12):

$$Z = \frac{1}{N} \sum_{i=1}^N x_i \quad (12)$$

If Z is the average intensity value of the reference cells, the adaptive threshold value of the j th sliding window is calculated using Equation (13):

$$T_{i,j} = \frac{\alpha}{N} \sum_{i=1}^N x_i \quad (13)$$

where

$T_{i,j}$ is the adaptive threshold for each sliding window position;

α is the threshold factor;

N is the number of samples in the sliding window;

$x_{i,j}$ is the reference cell.

Algorithm 1 describes the implementation of the proposed adaptive threshold method. It takes a 2D heatmap and the number of reference cells and guard cells as inputs, and it returns a 2D heatmap of the ROIs. Figure 8 shows a graphical description of the algorithm.

Algorithm 1 Disease Extent Detection from CNN Heatmaps

```

1: procedure CALCULATE(2D heatmap, number of reference and guard cells (N and M), cell
   under test (CUT).)
2:   Determine the number of reference and guard cells for each dimension.
3:   Determine the length of the heatmap for each dimension ( $L_i$  and  $L_j$ ).
4:   Create a new matrix  $b$  for storing the averaged values.
5:   for <each index  $i$  from  $(N_i+M_i)$  to  $L_i-(N_i+M_i)$ > do
6:     for <each index  $j$  from  $(N_j+M_j)$  to  $L_j-(N_j+M_j)$ > do
7:       Determine the average noise level across the reference cells.
8:       Add the offset to the average threshold value.
9:       if <CUT[ $i, j$ ] >  $b[i, j]$ > then
10:        Assign the original value.                                ▷ Detection result, (True or False).
11:       else
12:        Assign a value of zero.
13:   Suppress the edges to zero.
14:   Return  $b$                                                     ▷ Final heatmap showing the regions of interest.

```

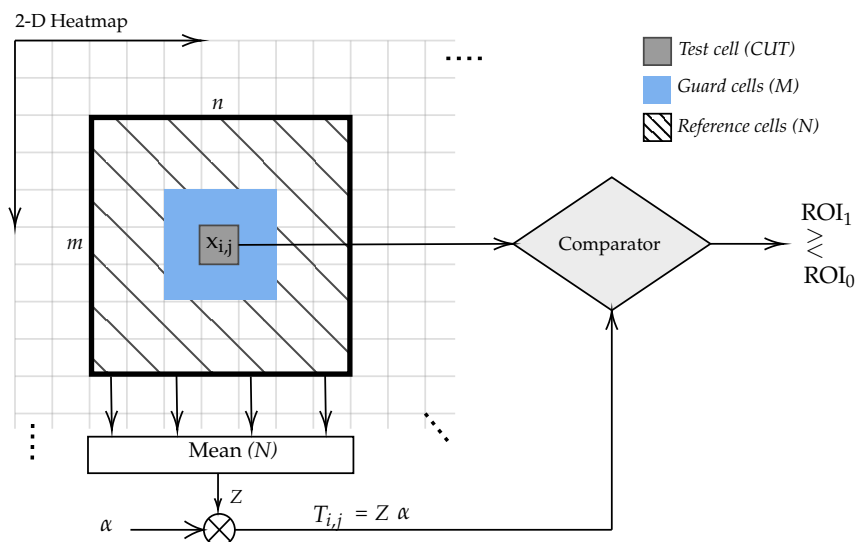


Figure 8. Adaptive threshold computation. The algorithm accepts a 2D heatmap image I and an $(n \times m)$ 2D sliding window that samples the entire region of the heatmap. The sliding window partitions the heatmap into three regions: (i) test cell, (ii) guard cells and (iii) reference cells. For each test cell or heatmap index $x_{i,j}$, the algorithm assigns guard cells around the test cell and calculates the average values of the pixels in the reference cells. The resulting mean is then multiplied by a fixed constant α to determine the threshold value for that test cell only. Finally, a comparator compares this threshold with the value of the test cell to determine the presence of an ROI before moving to another heatmap position.

4. Results

In this section, we conduct experiments to validate the feasibility of simultaneous maize leaf disease classification and quantification using a single deep learning model. Visualizing the CNN class activation heatmaps provides support for the hypothesis that the areas covered by bright pixels represent the areas affected by lesions. Therefore, by extracting the affected areas (ROIs) and expressing them as a ratio of the total leaf area, it is possible to obtain a severity estimate as a percentage.

4.1. Maize Leaf Disease Classification

The proposed study demonstrates the superior performance of transfer learning for retraining convolutional neural networks by comparing the accuracies of CNNs such as Inception V3, DenseNet 121, DenseNet 201, MobileNetV2, VGG16, EffcientNetB0,

ResNet50 and LeNet in the context of maize leaf disease classification. Table 1 shows the comparison results.

The results of the experiments indicate that with a large enough dataset, a shallow CNN trained from the ground up can achieve remarkable accuracy compared with mainstream convolutional neural networks. Our shallow CNN with only three convolutional layers manages to achieve high accuracy of 96% compared with the 99% obtained from DenseNet121 with 121 layers.

Table 1. State-of-the-art CNN models summary.

CNN	ImageNet Accuracy	Layers	Params (M)
EfficientNetB0	77.1%	132	5.3
InceptionV3	77.9%	189	23.9
Lenet		7	5.6
DenseNet121	75.0%	242	8.1
DenseNet201	77.3%	402	20.2
MobileNet	70.4%	55	4.3
VGG16	71.3%	16	138.4
Custom CNN	–	7	6.5

This paper visualizes the class activation maps of the convolutional layers to estimate disease severity. The class activation heatmap has pixels of varying intensities. Bright pixels represent areas that contribute more to the decision of the CNN, while darker pixels represent regions that do not carry any features related to the predicted class.

The adaptive thresholding algorithm adopted in this paper is not an entirely new concept. In fact, it was introduced in the field of radar target detection to identify objects buried in background noise [55]. We adopted the same concept due to the need to extract regions of interest that are often surrounded by background noise in class activation heatmaps. Background noise in this context refers to high-intensity pixels that are not part of the ROI. Fixed thresholding algorithms do not work well for these kind of problems, as the intensity levels of the ROI vary from pixel to pixel.

The adaptive thresholding algorithm's performance is compared with that of the well-known adaptive thresholding techniques in image processing. From the results, we can conclude that the proposed adaptive thresholding framework, borrowed from radar signal processing, performs much better than Bernsen's, Niblack's or Sauvola's thresholding. This is because thresholding relies on determining the average intensity of a region while isolating the cell being tested.

The performance of the proposed adaptive thresholding method is highlighted in Figure 9. Here, we demonstrate its ability to extract the ROI from three maize leaf disease classes: Blight, Common Rust and Gray Leaf Spot. Consequently, leaf disease severity is then obtained by expressing the area of the ROI as a percentage of the total area of the leaf.

4.2. Adaptive Thresholding Performance

Figure 10 shows the performance of the various thresholding techniques on extracting the ROI from the noisy CAM heatmap. A good thresholding algorithm must extract the entire ROI while eliminating the background pixels. Any residual background pixels affect the performance of leaf disease quantification. To determine the performance, we manually extract the ROI from Figure 10b and use this as a standard for comparing the thresholding algorithms. Next, we quantify the background noise that remains in the heatmap after applying the adaptive thresholding techniques. Results show that our adaptive thresholding method can remove up to 98.81% of the background noise in the heatmap, while Bernsen's thresholding removes 76.09%, Niblack's method removes 92.12%, and Sauvola's method performs better with 98.64%.

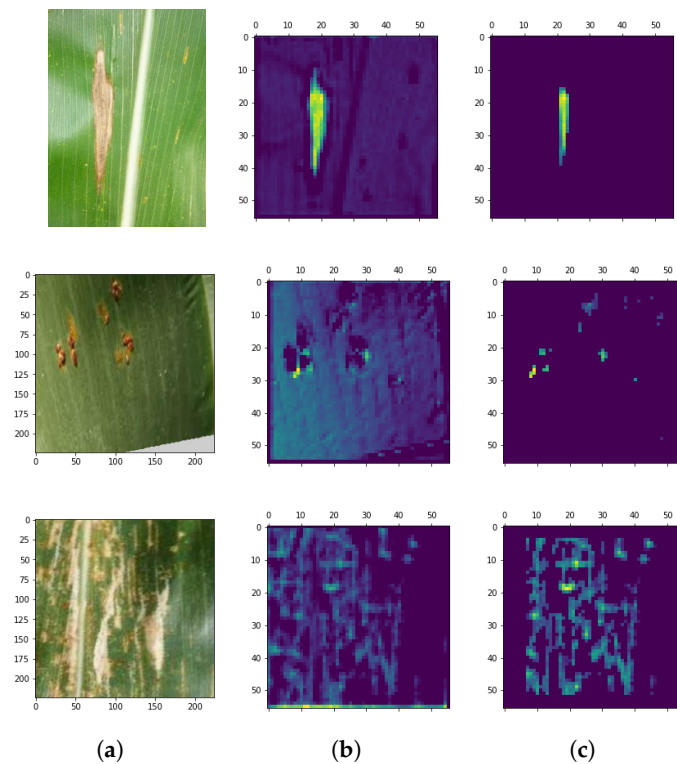


Figure 9. Examples of maize leaf images from the PlantVillage dataset. The dataset contains the following classes: (a) Raw RGB color images—Top: Blight; Middle: Common Rust; and Bottom: Gray spot. (b) Covnet image, deep feature visualization. The heatmap output of Covnet contains the ROI as well as the background. (c) Adaptive threshold performance. Result of adaptive thresholding. Here, adaptive thresholding successfully removes the background from the heatmap, and the result is a clear ROI from which we can calculate its area to determine the overall disease severity.

4.3. Leaf Disease Quantification

Once the ROI is successfully extracted, the final step is to determine the area of the ROI and express it as a percentage of the total leaf area. A simple workaround to this problem is to count the non-zero (ROI) and the zero (background) pixels from the heatmap. It then follows that:

$$\text{Diseased area} = \frac{\text{Area of ROI}}{\text{Total Leaf area}} \times 100\% \quad (14)$$

To assess the performance of leaf disease quantification, we use ImageJ software, a Java-based image processing tool to analyze image dataset [56]. We manually segment the diseased area in ImageJ, as shown in Figure 11, and calculate its area. We then compare the area obtained by manual ROI extraction from the heatmap to that obtained by automatic ROI extraction (using the best-performing adaptive thresholding technique). The results are shown in Table 2.

The results in Table 2 show a slight decrease in the area calculated from the heatmap as compared with the area of the original RGB leaf image. This happens because Gradcam does not highlight the entire area covered by the leaf Blight disease. Further, our adaptive thresholding technique estimates a lower area of the ROI because it does not completely remove the background noise from the heatmap. Nonetheless, this presents an important step toward leaf disease quantification. We believe we can improve performance by varying the value of the constant k in our algorithm.

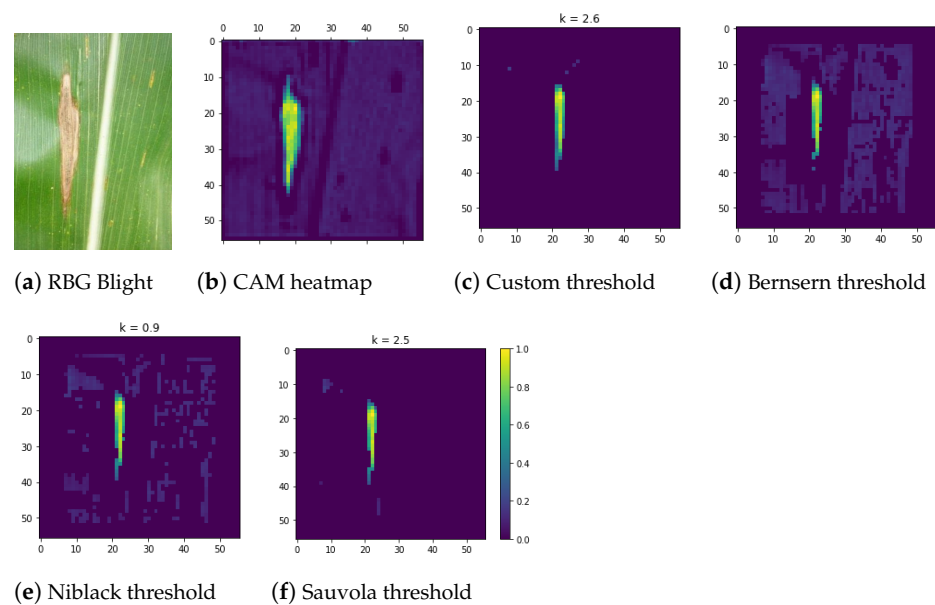


Figure 10. Results of various thresholding techniques on the CAM heatmap of an image of a Blight-infected leaf: (a) raw RGB color image of a leaf with Common Blight disease; (b) conv2D CAM heatmap visualization; (c–f) Custom, Bernsen’s, Niblack’s and Sauvola’s thresholding, respectively, to extract the ROI from the heatmap of (b). The main idea behind the algorithms is that the area of the extracted ROI correlates to the area of the disease in the original RGB image. Therefore, the area of the ROI as a percentage gives us an estimate of disease severity. Here, it is clear that our adaptive thresholding technique successfully removes the background from the heatmap, and the result is a clear ROI that can then be used for disease quantification.



Figure 11. ROI extraction with ImageJ software.

Table 2. Leaf disease quantification.

Sample Image	% Area of the Diseased Region
ROI from the raw RGB leaf image	4.87%
ROI extracted manually from the heatmap	4.32%
ROI extracted from the heatmap using our adaptive thresholding technique	3.09%

The results also confirm the influence of batch normalization in deep learning. In essence, batch normalization can improve the accuracy and convergence rates by good margins, as shown in Table 3 and Figure 12. In Table 3, batch normalization increased the accuracy of the shallow CNN by 1%, while Figure 13 shows a steeper learning curve, and the corresponding confusion matrices are shown in Figure 14.

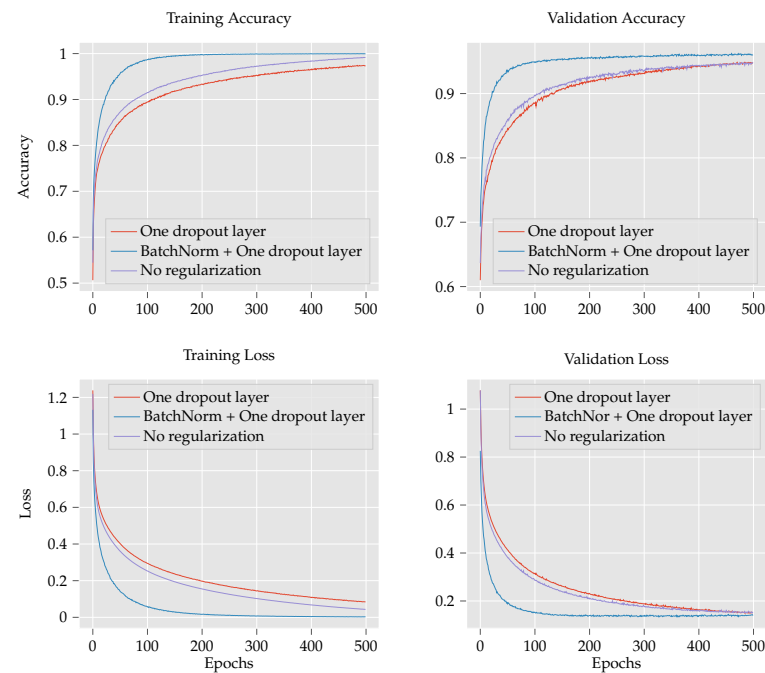


Figure 12. Training and validation learning curves for the custom CNN with various parameters: (i) one dropout layer, (ii) batch normalization + one dropout layer, and (iii) no regularization. All the training and validation loss curves decrease to a point of stability—representing a good fit. The plots show the expected behaviour. Namely, the model with batch normalization and a dropout layer learns the problem quickly as compared to the models with no regularization. In particular, the first model gets up to 90% accuracy in about 25 epochs compared to 100 epochs when no regularization or batch normalization is used.

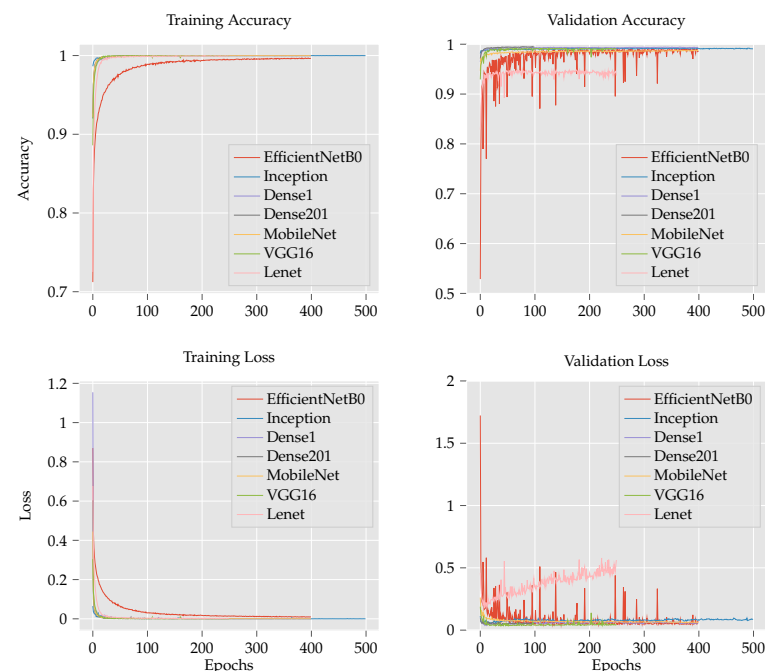


Figure 13. Training and validation learning curves of state-of-the-art CNNs: EfficientNetB0, Inception, DenseNet101, DenseNet201, MobileNet, VGG16 and Lenet. All the training and validation loss curves decrease to a point of stability—representing a good fit. All CNNs except EfficientNetB0 reach 99% accuracy in about 10 epochs.

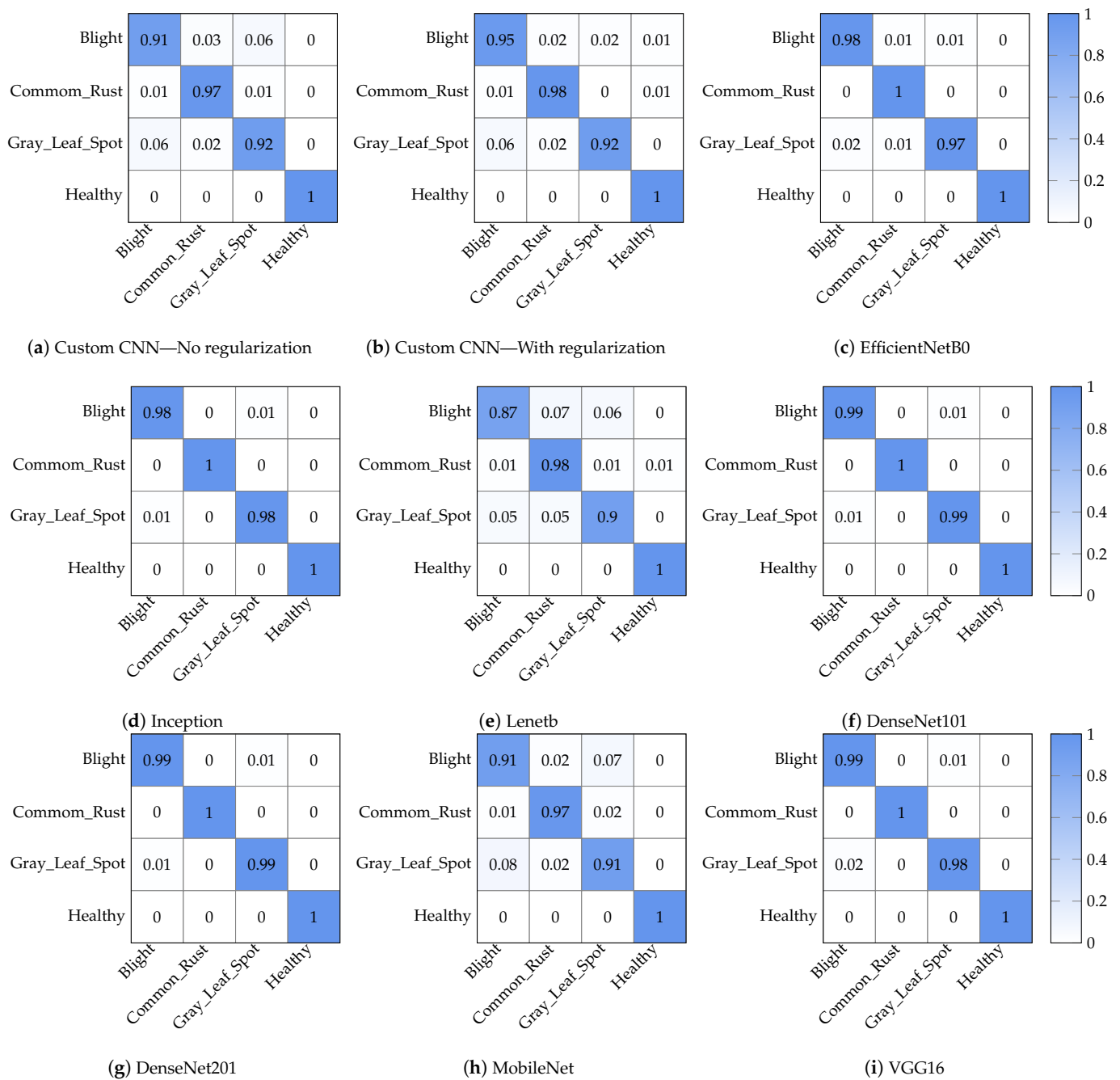


Figure 14. Confusion matrices of state-of-the-art CNN models obtained using our augmented PlantVillage maize leaf disease dataset.

Table 3. Comparison of performance of custom CNN on maize leaf disease dataset against state-of-the-art methods (specificity = 2345 and NPV = 0.2).

Method	Class Label	Precision	Recall	F1-Score	Overall Accuracy
Inception V3	Blight	0.99	0.98	0.98	99%
	Common Rust	0.99	1.0	1.0	
	Gray Leaf Spot	0.99	0.98	0.98	
	Healthy	1.0	1.0	1.0	
DenseNet 121	Blight	0.99	0.99	0.99	99%
	Common Rust	1.00	1.00	1.00	
	Gray Leaf Spot	0.99	0.99	0.99	
	Healthy	1.00	1.00	1.00	

Table 3. Cont.

Method	Class Label	Precision	Recall	F1-Score	Overall Accuracy
DenseNet 201	Blight	0.99	0.99	0.99	99%
	Common Rust	1.00	1.00	1.00	
	Gray Leaf Spot	0.99	0.99	0.99	
	Healthy	1.00	1.00	1.00	
MobileNet	Blight	0.98	0.98	0.98	99%
	Common Rust	0.99	1.00	0.99	
	Gray Leaf Spot	0.98	0.98	0.98	
	Healthy	1.00	1.00	1.00	
VGG16	Blight	0.99	0.97	0.98	99%
	Common Rust	1.00	1.00	1.00	
	Gray Leaf Spot	0.97	0.99	0.98	
	Healthy	1.00	1.00	1.00	
EfficientNetB0	Blight	0.98	0.98	0.98	99%
	Common Rust	0.98	1.00	0.99	
	Gray Leaf Spot	0.98	0.97	0.98	
	Healthy	1.00	1.00	1.00	
LeNet	Blight	0.94	0.87	0.90	94%
	Common Rust	0.90	0.98	0.93	
	Gray Leaf Spot	0.93	0.90	0.92	
	Healthy	0.98	1.00	0.99	
Custom CNN (One dropout layer)	Blight	0.92	0.92	0.92	95%
	Common Rust	0.96	0.96	0.96	
	Gray Leaf Spot	0.93	0.90	0.92	
	Healthy	0.97	1.00	0.99	
Custom CNN (Batch Normalization)	Blight	0.93	0.95	0.94	96%
	Common Rust	0.96	0.98	0.97	
	Gray Leaf Spot	0.97	0.92	0.94	
	Healthy	0.98	1.00	0.99	
Custom CNN (No Regularization)	Blight	0.92	0.91	0.92	95%
	Common Rust	0.94	0.97	0.96	
	Gray Leaf Spot	0.93	0.92	0.92	
	Healthy	0.99	1.00	1.00	

5. Discussion and Conclusions

In the collected literature, severity analysis has been largely based on qualitative assessment, with authors switching between nominal and ordinal scales of measurement, i.e., low-severity, medium-severity and high-severity. There is no quantitative framework for assigning a percentage to the problem of maize leaf disease quantification. In this paper, we present an important step towards simultaneous crop disease classification and quantification. We analyzed maize disease data using a variety of deep convolutional neural network models together with deep CNN feature visualization. The ultimate goal is to limit the number of data features and image processing stages while obtaining a discrete measure of disease severity. To this end, we employed a combinations of techniques from leaf image augmentation, transfer learning and CAM visualization (extracting the ROI) to applying an adaptive thresholding method for estimating the leaf area affected by the lesions. In comparison to other methods on the same classification problem, the proposed method obtained a competitive level of accuracy for classifying maize leaf diseases. The key motivation for our approach is that it can work in low-resource environments, e.g., when data are scarce or on low-computational-power systems, but most importantly, it can quantify crop diseases without the need of expert knowledge.

Additionally, our study used the most-common data augmentation techniques and obtained good accuracies of over 99% for most CNN models. Lenet was the worst performing model, with an accuracy of 94%. Our proposed shallow CNN model that uses only three convolutional layers managed to obtain an accuracy of 95% without any regularization. This demonstrates that with a larger dataset, shallow CNNs can produce impressive accuracies comparable with some deeper CNNs. All the benchmark CNN models used

in this work rely on multiple deep convolutional layers to obtain high performance on our dataset. Additionally, when shallow CNN was used together with regularization, we obtained a slight improvement in performance (96% accuracy). Our method of extracting the ROI specifies an adaptive threshold for each CUT based on the average intensity values of the reference cells in the 2D sliding window. Problems caused by busy backgrounds and specular lighting are well-known in the literature, but in this study, we report no impact of such factors on the performance of our algorithm. This work analysed the performance of several CNN architectures: Inception V3, DenseNet 121, DenseNet 201, MobileNet, VGG16, EfficientNetB0, LeNet, ResNet50 and our proposed shallow CNN.

In the future, the current findings of this work can be extended to: (i) Disease classification problems in other crops where yield is largely affected by crop disease, thus affecting the livelihoods of farmers, such as beans, bananas and cassava. (ii) Develop new algorithms for simultaneous multi-label leaf disease classification and quantification. Developing a robust method for quantifying leaf diseases with multiple symptoms is the next logical step. (iii) Validate the proposed CNN model on standard image processing datasets such as ImageNet and CiFar.

Author Contributions: Conceptualization, H.D.M. and G.O.; Formal analysis, C.N.; Methodology, F.M.; Supervision, D.O. and A.N.; Writing—review & editing, G.O. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The augmented PlantVillage dataset is available from the authors.

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

Abbreviations

The following abbreviations are used in this manuscript:

CAM	Class Activation Map
CNN	Convolutional Neural Network
CUT	Cell Under Test
DL	Deep Learning
ML	Machine Learning
ROI	Region of Interest
SVM	Support Vector Machine

References

1. Food and Agriculture Organization of the United Nations. The Future of Food and Agriculture: Trends and Challenges. Available online: <http://worldcat.org> (accessed on 20 February 2022).
2. Barburiceanu, S.; Meza, S.; Orza, B.; Malutan, R.; Terebes, R. Convolutional Neural Networks for Texture Feature Extraction. Applications to Leaf Disease Classification in Precision Agriculture. *IEEE Access* **2021**, *9*, 160085–160103. [\[CrossRef\]](#)
3. Zhang, X.; Qiao, Y.; Meng, F.; Fan, C.; Zhang, M. Identification of Maize Leaf Diseases Using Improved Deep Convolutional Neural Networks. *IEEE Access* **2018**, *6*, 30370–30377. [\[CrossRef\]](#)
4. Eraslan, G.; Avsec, Ž.; Gagneur, J.; Fabian, J. Deep learning: New computational modelling techniques for genomics. *Nat. Rev. Genet.* **2019**, *20*, 389–403. [\[CrossRef\]](#) [\[PubMed\]](#)
5. Tetila, E.C.; Machado, B.B.; Menezes, G.K.; Junior, A.S.O.; Alvarez, M.; Amorim, W.P.; Belete, N.A.; Silva, G.G.; Pistori, H. Automatic Recognition of Soybean Leaf Diseases Using UAV Images and Deep Convolutional Neural Networks. *IEEE Geosci. Remote Sens. Lett.* **2020**, *17*, 903–907. [\[CrossRef\]](#)
6. Taylor, B.; Marco, V.S.; Wolff, W.; Elkhatib, Y.; Wang, Z. Adaptive deep learning model selection on embedded systems. *ACM Sigplan Not.* **2018**, *53*, 31–43. [\[CrossRef\]](#)
7. Lane, N.D.; Warden, P. The Deep (Learning) Transformation of Mobile and Embedded Computing. *Computer* **2018**, *51*, 12–16. [\[CrossRef\]](#)
8. Dhingra, G.; Kumar, V.; Joshi, H.D. Study of digital image processing techniques for leaf disease detection and classification. *Multimed. Tools Appl.* **2018**, *77*, 19951–20000. [\[CrossRef\]](#)
9. Bock, C.H.; Chiang, K.S.; Del Ponte, E.M. Plant disease severity estimated visually: A century of research, best practices, and opportunities for improving methods and practices to maximize accuracy. *Trop. Plant Pathol.* **2021**, *47*, 25–42. [\[CrossRef\]](#)

10. Olivoto, T.; Andrade, S.; Del Ponte, E.M. Measuring plant disease severity in R: Introducing and evaluating the pliman package. *Trop. Plant Pathol.* **2022**, *47*, 95–104. [\[CrossRef\]](#)
11. Owomugisha, G.; Ernest, M. Machine Learning for Plant Disease Incidence and Severity Measurements from Leaf Images. In Proceedings of the 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), Anaheim, CA, USA, 18–20 December 2016; pp. 158–163. [\[CrossRef\]](#)
12. Sethy, P.K.; Negi, B.; Barpanda, N.K.; Behera, S.K.; Rath, A.K. Measurement of Disease Severity of Rice Crop Using Machine Learning and Computational Intelligence. In *Cognitive Science and Artificial Intelligence*; SpringerBriefs in Applied Sciences and Technology; Springer: Singapore, 2017. [\[CrossRef\]](#)
13. Qiaokang, L.; Shao, X.; Yucheng, H.; Gianmarc, C.; Dan, Z.; Wei, S. PD2SE-Net: Computer-assisted plant disease diagnosis and severity estimation network. *Comput. Electron. Agric.* **2019**, *157*, 518–529. [\[CrossRef\]](#)
14. Kim, D.H.; Baddar, W.J.; Jang, J.; Ro, Y.M. Multi-Objective Based Spatio-Temporal Feature Representation Learning Robust to Expression Intensity Variations for Facial Expression Recognition. *IEEE Trans. Affect. Comput.* **2019**, *10*, 223–236. [\[CrossRef\]](#)
15. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826. [\[CrossRef\]](#)
16. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely Connected Convolutional Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2261–2269. [\[CrossRef\]](#)
17. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520. [\[CrossRef\]](#)
18. Simonyan, K.; Andrew, Z. Very deep convolutional networks for large-scale image recognition. In Proceedings of the ICLR 2015 Conference, San Diego, CA, USA, 7–9 May 2015. [\[CrossRef\]](#)
19. Tan, M.; Quoc, L. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the 36th International Conference on Machine Learning, PMLR 97, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114. Available online: <https://proceedings.mlr.press/v97/tan19a.html> (accessed on 11 August 2021).
20. He, K.; Zhang, S.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [\[CrossRef\]](#)
21. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [\[CrossRef\]](#)
22. Available online: <https://github.com/spMohanty/PlantVillage-Dataset> (accessed on 9 February 2020).
23. Zhuang, F.; Qi, Z.; Duan, k.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. A Comprehensive Survey on Transfer Learning. *Proc. IEEE* **2020**, *109*, 43–76. [\[CrossRef\]](#)
24. Panigrahi, K.P.; Das, H.; Sahoo, A.K.; Moharana, S.C. Maize Leaf Disease Detection and Classification Using Machine Learning Algorithms. In *Progress in Computing, Analytics and Networking*; Advances in Intelligent Systems and Computing; Das, H., Pattnaik, P., Rautaray, S., Li, K.C., Eds.; Springer: Singapore, 2020; Volume 1119. [\[CrossRef\]](#)
25. Alehegn, E. Ethiopian maize diseases recognition and classification using support vector machine. *Int. J. Comput. Vis. Robot.* **2019**, *9*, 90–109. [\[CrossRef\]](#)
26. Pantazi, X.E.; Moshou, D.; Tamouridou, A.A. Automated leaf disease detection in different crop species through image features analysis and One Class Classifiers. *Comput. Electron. Agric.* **2019**, *156*, 96–104. [\[CrossRef\]](#)
27. Jayme, G.; Arnal, B. A review on the main challenges in automatic plant disease identification based on visible range images. *Biosyst. Eng.* **2016**, *144*, 52–60. [\[CrossRef\]](#)
28. Shanwen, Z.; Xiaowei, W.; Zhuhong, Y.; Liqing, Z. Leaf image based cucumber disease recognition using sparse representation classification. *Comput. Electron. Agric.* **2017**, *134*, 135–141. [\[CrossRef\]](#)
29. Ali, H.; Lali, M.I.; Nawaz, M.Z.; Sharif, M.; Saleem, B.A. Symptom based automated detection of citrus diseases using color histogram and textural descriptors. *Comput. Electron. Agric.* **2017**, *138*, 92–104. [\[CrossRef\]](#)
30. Ordonez, F.J.; Roggen, D. Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors* **2016**, *16*, 115. [\[CrossRef\]](#)
31. DeChant, C.; Wiesner-Hanks, T.; Chen, S.; Stewart, E.L.; Yosinski, J.; Gore, M.A.; Nelson, R.J.; Lipson, H. Automated Identification of Northern Leaf Blight-Infected Maize Plants from Field Imagery Using Deep Learning. *Phytopathology* **2017**, *107*, 1426–1432. [\[CrossRef\]](#)
32. Feng, J.; Yang, L.; Yu, C.; Di, C.; Gongfa, L. Image recognition of four rice leaf diseases based on deep learning and support vector machine. *Comput. Electron. Agric.* **2020**, *179*, 105824. [\[CrossRef\]](#)
33. Mohammad, S.; Wahyudi, S. Convolutional neural network for maize leaf disease image classification. *Telkomnika* **2020**, *18*, 1376–1381. [\[CrossRef\]](#)
34. Zhou, C.; Zhou, S.; Xing, J.; Song, J. Tomato Leaf Disease Identification by Restructured Deep Residual Dense Network. *IEEE Access* **2021**, *9*, 28822–28831. [\[CrossRef\]](#)
35. Liu, B.; Yu, X.; Zhang, P.; Yu, A.; Fu, Q.; Wei, X. Supervised Deep Feature Extraction for Hyperspectral Image Classification. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 1909–1921. [\[CrossRef\]](#)

36. Barbedo, J.; Garcia, A. Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification. *Comput. Electron. Agric.* **2017**, *153*, 46–53. [\[CrossRef\]](#)
37. Bock, C.H.; Pethybridge, S.J.; Barbedo, J.G.; Esker, P.D.; Mahlein, A.K.; Del Ponte, E.M. A phytopathometry glossary for the twenty-first century: Towards consistency and precision in intra-and inter-disciplinary dialogues. *Trop. Plant Pathol.* **2022**, *47*, 14–24. [\[CrossRef\]](#)
38. Bock, C.H.; Barbedo, J.G.A.; Del Ponte, E.M.; Bohnenkamp, D.; Mahlein, A. From visual estimates to fully automated sensor-based measurements of plant disease severity: Status and challenges for improving accuracy. *Phytopathol. Res.* **2020**, *2*, 9. [\[CrossRef\]](#)
39. Arnal Barbedo, J.G. Digital image processing techniques for detecting, quantifying and classifying plant diseases. *Springerplus* **2013**, *47*, 14–24. [\[CrossRef\]](#)
40. Barbora, S.; Vlastimil, K.; Rostislav, Z. Computer-assisted estimation of leaf damage caused by spider mites. *Comput. Electron. Agric.* **2006**, *53*, 81–91. [\[CrossRef\]](#)
41. Sengar, N.; Malay, K.D.; Carlos, M.T. Computer vision based technique for identification and quantification of powdery mildew disease in cherry leaves. *Computing* **2018**, *100*, 1189–1201. [\[CrossRef\]](#)
42. Saxena, D.K.; Jhanwar, D.; Gautam, D. Classification of Leaf Disease on Using Triangular Thresholding Method and Machine Learning. In *Optical and Wireless Technologies. Lecture Notes in Electrical Engineering*; Tiwari, M., Maddila, R.K., Garg, A.K., Kumar, A., Yupapin, P., Eds.; Springer: Singapore, 2022; Volume 771. [\[CrossRef\]](#)
43. Bakar, M.N.; Abdullah, A.H.; Rahim, N.A.; Yazid, H.; Misman, S.N.; Masnan, M.J. Rice leaf blast disease detection using multi-level colour image thresholding. *J. Telecommun. Electron. Comput. Eng.* **2018**, *10*, 1–15.
44. Sinha, A.; Shekhawat, R.S. Detection, Quantification and Analysis of Neofabraea Leaf Spot in Olive Plant using Image Processing Techniques. In Proceedings of the 2019 5th International Conference on Signal Processing, Computing and Control (ISPCC), Solan, India, 10–12 October 2019; pp. 348–353. [\[CrossRef\]](#)
45. Yadav, A.; Dutta, M.K. An Automated Image Processing Method for Segmentation and Quantification of Rust Disease in Maize Leaves. In Proceedings of the 2018 4th International Conference on Computational Intelligence and Communication Technology (CICT), Ghaziabad, India, 9–10 February 2018; pp. 1–5. [\[CrossRef\]](#)
46. Mahmud, A.; Esakki, B.; Seshathiri, S. Quantification of Groundnut Leaf Defects Using Image Processing Algorithms. In *Advances in Intelligent Systems and Computing, Proceedings of the International Conference on Trends in Computational and Cognitive Engineering, Online, 21–22 October 2021*; Springer: Singapore, 2021. [\[CrossRef\]](#)
47. Barbedo, J.; Garcia, A. An automatic method to detect and measure leaf disease symptoms using digital image processing. *Plant Dis.* **2014**, *98*, 1709–1716. [\[CrossRef\]](#) [\[PubMed\]](#)
48. Eaganathan, U.; Prasanna, S.; Sripriya, D. Various approaches of color feature extraction in leaf diseases under image processing: A survey. *Int. J. Eng. Technol.* **2018**, *7*, 712–717.
49. Wilhelm, B.; Burge, M.J. *Principles of Digital Image Processing Advanced Methods. Undergraduate Topics in Computer Science*; Springer: London, UK, 2013. [\[CrossRef\]](#)
50. Haris, I. PlotNeuralNet. Saarland University. Available online: <https://github.com/HarisIqbal88/PlotNeuralNet> (accessed on 10 December 2021).
51. Bernsen, J. Dynamic thresholding of gray level images. In Proceedings of the International Conference on Pattern Recognition, Paris, France, 27–31 October 1986; pp. 1251–1255.
52. Niblack, W. *An Introduction to Digital Image Processing*; Prentice-Hall: Englewood Cliffs, NJ, USA, 1986; pp. 115–116.
53. Sauvola, J.J.; Seppanen, T.; Haapakoski, S.; Pietikainen, M. Adaptive document binarization. In Proceedings of the 4th International Conference on Document Analysis and Recognition, Ulm, Germany, 18–20 August 1997; pp. 147–152.
54. Sun, K.H.; Huh, H.; Tama, B.A.; Lee, S.Y.; Jung, J.H.; Lee, S. Vision-Based Fault Diagnostics Using Explainable Deep Learning with Class Activation Maps. *IEEE Access* **2020**, *8*, 129169–129179. [\[CrossRef\]](#)
55. Richards, M.A. *Fundamentals of Radar Signal Processing*; McGraw-Hill Professional: New York, NY, USA, 2005.
56. Abramoff, M.D.; Magalhaes, P.J.; Ram, S.J. Image processing with Image. *J. Biophotonics* **2004**, *11*, 36–42.