INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

BWENet: Detection and Grading of Bacterial Wilt Using Deep Convolutional Neural Network

Desalegn Ashebir[1]*, Getahun Tadesse[2]

1 Department of Software Engineering, Mizan Tepi University, Ethiopia
2 Department of computer Science, Mizan Tepi University, Ethiopia

OPEN ACCESS

Received: 19-03-2022
Accepted: 05-05-2022
Published: 22-06-2022

Citation: Ashebir D, Tadesse G (2022) BWENet: Detection and Grading of Bacterial Wilt Using Deep Convolutional Neural Network . Indian Journal of Science and Technology 15(22): 1100-1111. https://doi.org/10.17485/IJST/v15i22.634

* **Corresponding author**.

desalegn629@gmail.com

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: To design a Bacterial Wilt Enset (BWENet) model that can detect and grade the level of bacterial wilt disease in Enset using a deep convolutional neural network. **Methods:** Convolutional neural network is used for detection of bacterial wilt and 4-way Softmax is used for grading bacterial wilt into a specific level (normal, early-stage, infected stage, and completely wilted Enset). About 1600 images were used to evaluate the performance of the model out of which 70% is for training, 15% for validation and 15% for testing. We collected dataset from kefa, Sheka agricultural research center and some selected areas in SNNPR. The proposed model was trained using these images and data augmentation techniques were applied to generate more images. **Findings:** BWENet model achieved an identification test grading accuracy of 90.53% bacterial wilt disease. BWENet model is found to be faster to train and has a smaller model size as compared to State-of-the-art models like Alex Net, Google net, and VGG19. **Novelty:** we are proposing a new CNN architecture to grade bacterial wilt disease severity measurement. By our experiments, we have shown the kernel size selection, poling layer selection, and segmentation achieved better accuracy results.

Keywords: Deep learning; Feature Learning; Segmentation; Grading; Concatenation

1 Introduction

Agriculture is the backbone of Ethiopia's economy. Around 80-85 percent of Ethiopians rely on agriculture, with more than 20 percent of those relying on Enset crop production in the country.[1,2]. Enset (Ensete ventricosum, Musaceae) is a multi-objective traditional crop cultivated widely in the highlands of Ethiopia, especially in the south and southwestern part of the country. Food scarcity is a serious challenge in underdeveloped countries like Ethiopia. States in Ethiopia finds harder to keep food resources enough to battle famine. The plant is drought-resistant staple food cultivated by farmers[1,3,4]. Currently, about one-fifth of the Ethiopian populations (20 million) depend on this crop mainly in the southern region and adjoining places especially Oromia and Gambella Regions[5]. The crop has several needed qualities which make it better than other crops and desirable in facing food scarcity. Enset gives the maximum

yield per unit area, thus supporting the country's heavily populated areas. It is also a significant source of raw material for agriculture-based manufacturing. Enset foods can be preserved for long periods. However, the Production and productivity of Enset are affected by many diseases, mainly by bacterial wilt. Enset bacterial wilt is caused by Xanthomas Campestral. Musacearum is the main constraint of Enset production in Ethiopia, although other biotic, as well as abiotic factors, compromise productivity. The disease was first reported about 52 years ago in Ethiopia on Enset, which is a family of bananas [2,5].

According to several reports, the productivity of Enset is declining continuously from time to time in several parts of the country due to bacterial wilt. Once the plant is attacked by the disease, it affects the entire farm and usually causes a maximum yield loss.[5,6]. Normally, Bacterial wilt disease is one of the most common widely spread diseases at a destructive level with high incidence and it is a great concern for farmers living in the southern part of the country [6].According to Ethiopia Central Statistical Agency(ECSA), Agricultural Sample Survey bacterial wilt is the most difficult disease in Enset production [1,6]. According to ECSA Agricultural Sample Survey, there are three levels of bacterial wilt disease in Enset comprising of stages like Early-stage, infected stage, and completely wilted Enset. To detect and grade the level of bacterial wilt, it is usually essential to look at the Enset carefully; examine the Enset leaves. The manual detection and grading of Enset diseases require a careful and conservative examination through expertise. Traditional ways and due to limited research attention given to Enset crop production and the naked eye visualization of experts is the main old-style approach adopted for the recognition and identification of Enset disease. This requires continuous monitoring of experts in a large number of Enset productions. The farmer may have to go distance to communicate expert making it expensive and time-consuming.

However, little research has been done on Enset plant disease identification. The first study used an Ethiopian Enset disease diagnosis model, and they considered two types of disease: bacterial and Fusarium wilt. They used a traditional machine learning algorithm (SVM), which requires manual feature extraction and is time-consuming and expensive. They didn't measure the severity of a specific disease. Another study looked into deep learning-based Enset plant disease identification, which included bacterial and Fusarium wilt, and used the VGG16 state-of-the-art CNN architecture. This architecture has a large number of parameters and leads to high memory usage, and they didn't measure the severity level of the disease, identify the stage (severity). Grading early disease increase the high-quality yield of Enset product. No research work has explored to detect the grade of this disease. To grade this bacterial wilt we have come into a deep learning approach and developed a model that easily detect the grade of the bacterial wilt [4,7,8]. In this paper, we proposed to fill the above research gap by grading normal, early-stage, infected stage, and completely wilted Enset on bacterial wilt disease and developed a model that improved high performance and less memory usage using deep learning techniques particularly in convolutional neural networks.

Our novelty is to apply threshold segmentation before feeding the raw data into the network and filter size selection during training and validation of the model. Finally, build the BWENet CNN model for bacterial wilt disease detection and grading. In general, as with most Digital image processing applications, detecting bacterial wilt diseases have pre-processing, segmentation, and classification [3,4,8] Pre-processing is the process of improving the quality of digital photos by enhancing or adjusting them, whereas segmentation is the process of locating the bacterial wilt zone in the image [9]. Feature extraction is used to compute salient features characterizing the bacterial wilt disease [9,10]. Finally, pattern recognition techniques are applied to grade bacterial wilt [9,10]. The rest of the paper is structured as follows: Section 2 presents related works on the detection and grading of plant diseases. The proposed model is presented in section 2 while the experimental setup along with test results and outcomes were presented in Sections 3, Section 4 concludes with future works. And finally reference in section 5

1.1 Related Works

[1]according to Diagnosis Model for Ethiopian Enset Diseases, Image Processing and Machine Learning Techniques were used for manual or handcrafted base feature extraction that leads to time-consuming and is expensive. The researcher used 92 image datasets. It was reported that the accuracy is 94.4%. In [3] proposed Detection of Bacterial Wilt on Enset Crop Using Deep Learning Approach identification of bacterial wilt. The study didn't measure the severity level of the disease health and disease class is not considered. It is better to grade the severity level of disease for effective disease management. Mesay et al. [4] proposed a deep learning approach to identify Ginger Disease. With the support of domain specialists from several farms, they used 7,014 ginger photos. They merely detected normal and diseased plants, did not identify the level of disease, did not quantify the level of disease at each stage of plant development, and finally reached a test accuracy of 95.2%. As per [11] the use of a Convolutional Neural Network to identify Carabao Mango Leaf Disease has been proposed. 2080 leaf images were used for four classes of disease diagnosis and a recognition rate of 81.1 %. In **[10]** the authors proposed Plant Disease Classification Using Deep Bilinear CNN fine-tune VGG and pruned ResNet and utilize them as feature extractors and connect them to fully connected dense networks. Finally, the larger model obtained an accuracy score of 94.98% for 38 distinct classes. They didn't consider a comparison of other state of the art CNN architecture and they didn't evaluate the model. This may affect the performance of the model since it needs specific species of plant disease. In [12] authors suggested Image Analysis Techniques for Grading

Ethiopian Coffee Beans: Features Extraction and Dataset Preparation. A total of 700 images were used, with 100 images for each class. They were divided into seven classes, with morphological and color texture-based feature extraction. When altering the dataset, this feature extraction is time-consuming and expensive because it is handcrafted. [13] Proposed a deep learning method for detecting mobile botnets. They classified botnet apps and conventional apps using 342 static app features. They validated their study on 6,802 real apps containing 1,929 and 4,873 clean sample botnets from the publicly available ISCX botnet dataset, and they achieved an accuracy of 0.981 and a better result by applying SVM, but when we implement binary classification SVM, we can see the result is almost the same as CNN, because they only considered two classes, and they don't clearly show which filter size applied in order to extrapolate the results. [14] Suggested a method for detecting botnet attacks on BASHLITE and Mirai, on nine commercial IoT devices, utilizing CNN and LSTM. They reviewed and conducted using the N-BaIoTdataset, achieving detection accuracies of 90.88 percent and 88.61 percent for Dominion and Ennio brands, respectively, and overall accuracy of 89.64 percent for nine IoT devices infected by ten attacks.

[15] Proposed the Lightweight based CNN method to identify tomato leaf disease, they used tomato leaf disease datasets and divided them into 8;1:1 ratios, resulting in a total of 19,510 images from 10 distinct disease classes and one healthy class. They compared the proposed model to other state-of-the-art CNN models in terms of performance and computational complexity, and at the end, the model achieved an average accuracy of 99.34%. They did not assess the severity of each condition. According to [16] the Benchmark- Reference Model identified Botnet, Decision Tree, Random Forest, Naive Bayes Gaussian, Support Vector Machine, and K-Neighbors to evaluate them. They used two public datasets of real botnet traffic—CIC-AWS-2018 and ISOT HTTP Botnet—and the Grid Search technique to optimize the parameter. Due to the usage of handmade feature extraction and the necessity whenever the dataset changes due to standard machine learning algorithms, they were imbalanced data samples; Weak feature extraction and/or selection procedures. Finding a set of hyper parameters that gives an accurate CNN most of the time used manually searching and time consuming, hyper parameter Optimization problem slide and down the result. Hyper parameter optimization is a problem that identifies a good model of hyper parameter to over cum this problem we will applied manual searching [17]. Hyper parameter highly affected the performance of deep CNN selecting those parameter is better to improve the model so we will train and validate the model using try about different batch size, epoch, learning rate [17] [18]

In this paper, we introduced a new model using CNN architecture that is different from the previous state of the art CNN architectures where different layers or operations are stacked on top of each other linearly. Furthermore, instead of learning features from raw images as done with most CNN models, our model learns texture features extracted from images which significantly improved the overall performance of the system. Various researchers also done transfer learning and directly train the state of the art CNN models with large amount of data leading to computationally expensive and time consuming. Even worse is nearly all researches are aimed at only detection of the disease, They don't measure the severity level of the disease Hence, developing a model for grading Enset bacterial wilt severity will have crucial impact on the agricultural production. To fill this research gap we introduced a new architecture by including additional pre-processing and segmentation that best suit for small amount of data. The proposed BWENet model for bacterial wilt includes three stages of the disease levels like early stage, infected stage, and completely wilted Enset.

Our paper has the following contributions in identifying the disease and grading it. We show that applying threshold segmentation improves the learning capability of deep learning systems. Experimental results has shown that the use of segmented images provide better accuracy in the detection and grading of bacterial wilt that use of raw image data as inputs to the deep learning models and is computationally effective. The filter sizes are selected systematically based on the characteristic features on each grade of bacterial wilt. Selection is based on the length and breadth of broken Enset leaf. Each leaf size is from 2 pixels and 10 pixels. Those pixel sizes are determined empirically from the dataset. The aforementioned pixel sizes can be detected using 1x1 to 5x5 filter size. A 3x3 filter size at a time can learn features covered by 5 pixels whereas 5 x 5 filters at a time can learn features covered by 9 pixels. A characteristic feature that cannot be learned by 3x3 filter size can be learned by a filter size of 5x5. 1* 1 filter size also Create a linear projection of a stack of feature maps. The projection created by a 1×1 can act like channel-wise pooling and be used for dimensionality reduction. The projection created by a 1×1 can also be used directly or be used to increase the number of feature maps in a model, so our model computationally effective only take 3:45 hour to train the model and 90.53% test accuracy.

Performance improvement**:** The proposed BWENet model achieves better accuracy. The model Weighs very less and trains faster as compared to previous models**,** BWENet improves performance by 6% better than Alex Net, 4% better than VGG16, 2% better than Google net.

2 Material and methods

2.1 Pre-processing

The image obtained may have been exposed to various environmental variables and may have been impacted by noise. As a result, the needed information may not be visible. By reducing noise and changing the contrast while keeping the information, preprocessing techniques help to improve the image quality. For preprocessing, it was chosen to apply median filtering and histogram equalization based on the sample images obtained. As median filtering kept the edge information, which was critical to our work, it was the best strategy for boosting by reducing noise. To improve the contrast needed to distinguish bacterial wilt from the background, histogram equalization was used. The picture data of bacterial wilt image is then resized or rescaled to reduce complexity. Images are resized to a standard size of (224x224), similar to the size used by most of the state-of-the-art CNN architectures.

Another frequently faced problem is the number of images in the training set often results in increasing the size. To deal with these issues, we outline a technique that uses augmentation that transforms the images in the training to increase the ability of the model to recognize different versions of an image. Data augmentation: Random transformations (such as translation, rotation, shearing, and resizing) are applied to increase the size of our dataset. Data augmentation is typically applied to the training dataset, and not to the validation or for testing.

2.2 Segmentation

We used threshold segmentation in our paper. To choose the "interesting" area of the image, we first converted the image to gray scale, and then blurred it to de-noise, and then created a mask based on the threshold value of 125 and used the binary mask to select it. Finally, we fed the split image into the network. Normally in threshold, we try different threshold values to separate background from region of interest from image. Simple iterative algorithm for manual threshold (T) selection in a given image is shown in equation 1.

A. Choose the initial threshold value
B. Segment the image into 2 regions G1 and G2 using the threshold value
C. Find the mean of pixels of region G1 and G2 ($\mu 1$ and $\mu 2$ respectively)
D. $T = (\mu 1 + \mu 2)/2$ and repeat step 2
Continue the process the $|T_i\text{-}T_{i+1}|$ <= T where T is some value defined by the user (1)

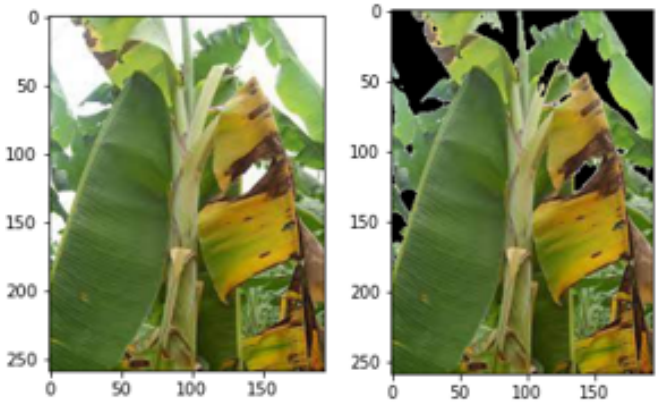

Fig 1. Applying segmentation on the selected image

2.3 Classification

Classification of bacterial wilt is made by learning texture features extracted from Enset leaf region. The process of classification composed of training, validation, detection and grading phases requires the following operations: convolution, pooling, filter concatenation, neuron dropping out, flattening, and Softmax classification. Besides, batch normalization, learning rate scheduling, stochastic gradient descent optimization, and data augmentation techniques were applied.

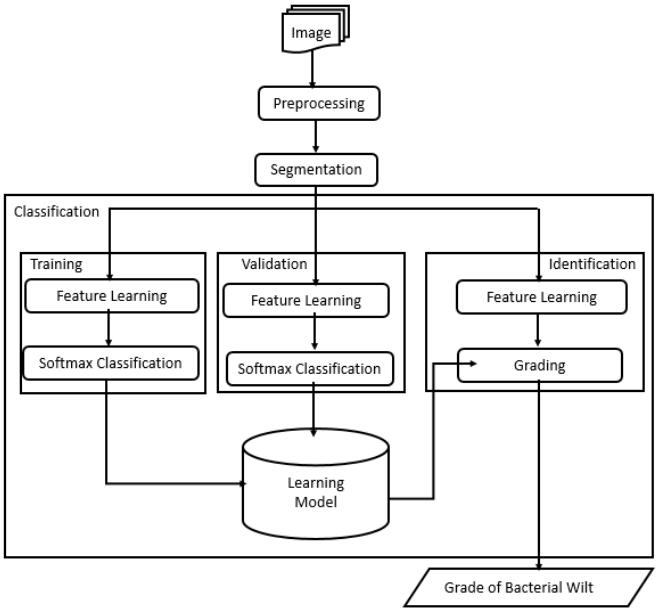

Fig 2. High-level architecture of the proposed system

2.3.1 Training phase

For learning the most characteristics or learnable features, nine convolution layers, five pooling layers, and one flattening layer were placed on top of each other in the feature learning for the training phase. The layout of the layers is depicted in Figure 3. The stride size is set to 2; the first convolution layer (Conv1a) down samples the input feature map by 2. Then, to learn more representative features, a 3x3 convolution operation (Conv1b) is used. Then, for down sampling, max-pooling (MaxPool1a) is used. To learn and construct additional representative features, these processes (convolution followed by max-pooling) are repeated (Conv2a, Conv2b, MaxPool2a). To employ a convolution operation with multiple filter sizes at a single layer, the convolution module is applied three times (ConvMod1a, ConvMod1b, and ConvMod1c). Each convolution operation's filter size is concatenated in convolution modules. For down sampling, a pooling module (PoolMod1a) is used. Then we use the convolution module twice (ConvMod2a, ConvMod2b) and the pooling module once (PoolMod2a). The suggested system's building blocks are the convolution and pooling modules, which are utilized to manage the system's width. These modules, as shown in Figure 3 and Table 1, add a new level of structure to CNN design, which differs from the basic CNN architectures that stacks distinct layers or operations on top of each other linearly.[8] [18] . Finally, a 7x7 average pooling (AvgPool1a) is performed, followed by a flattening operation to reduce the dimension of the feature map, the output of which is fed into the Softmax classifier.

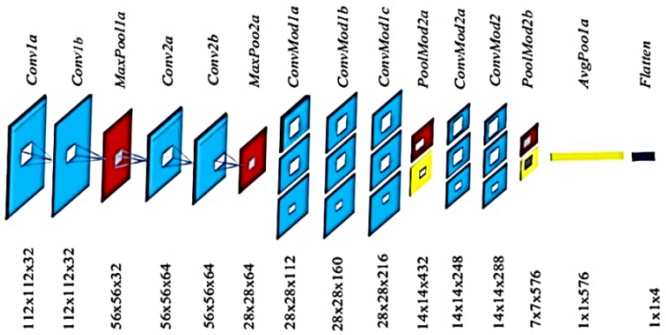

Fig 3. Architecture of featurelearning

Table 1. Description of the proposed classification model

	Name	Filter Size/stride	Output Size	#1x1 Filters	#3x3 Filters	#5x5 Filters	Parameters
Convolution	Conv1a	5x5/2	112x112x32			32	2.4K
Convolution	Conv1b	3x3/1	112x112x32		32		9.2K
Max Pooling	MaxPool1a	1x2/2	56x56x32				
Convolution	Conv2a	5x5/1	56x56x64			64	18.5K
Convolution	Conv2b	3x3/1	56x56x64		64		37K
Maximum Pooling	Maxpool2a	2x2/2	28x28x64				
Convolution Module	ConvMod1a		28x28x112	64	32	16	48K
Convolution Module	ConvMod1b		28x28x160	96	48	16	104K
Convolution Module	ConvMod1c		28x28x216	128	64	24	209K
Pooling Module	PoolMod1a		14x14x432				
Convolution Module	ConvMod2a		14x14x248	160	64	24	577K
Convolution Module	ConvMod2b		14x14x288	192	64	32	389K
Pooling Module	PoolMod2a		7x7x576				
Average Pooling	Avg Pool1a	7x7/1	1x1x576				
Dropping out	Dropout		1x1x576				
Flattening	Flatten		1x1x4				
Classification	SoftMax		1x1x4				
Total 1.4 Million							

Convolution layer: Convolution layers are made up of two parts: a convolution operation and an activation function. In the case of convolution modules, filter concatenation is also performed. The first convolution layer receives a 224x224x3 picture that has been filtered with a threshold segmentation filter. The number of filters, filter size, stride size, and zero Padding are all required for the convolution procedure. We used 32, 64, 96, 128, 160, and 192 filters in our model. Different numbers of convolution layers and filters are examined, and the ones that achieve the best results are chosen. On a single layer, a 5x5, 3x3, and 1x1 filter sizes are applied. Two (2x2) and one (2x1) stride sizes were used (1x1). When the stride length is two dimensions is reduced by half vertically and horizontally. In some cases, stride size is set to one with the same padding.

$$S(i,j) = (I * K)(i,j) = \sum_m * \sum_n I(m,n).(K(i+m, j+n) \tag{2}$$

Where i and j are an image I coordinates, and m and n are kernel K coordinates.

Activation: ReLU activation function is carried out in the activation layer throughout our model. ReLU activation function returns zero if the values in the input layer are negative, else it returns the existing value. **Filter concatenation**: The convolution and pooling modules execute filter concatenation. We employed 1x1, 3x3, and 5x5 filter sizes at the same layer in the convolution module. The filter diameters are chosen systematically based on the distinct characteristics of each grade of bacterial wilt on the Enset leaf. **Pooling layer**: A total of Maximum and average pooling layers will form the pooling layers. In the same way, the filter concatenation is done in convolution layers; filter concatenation is done in pooling modules.

$$ReLU = F(X) = Max(0, X) \tag{3}$$

2.3.2 Validation Phase

The validation phase is mainly concerned with optimizing the learning model by increasing accuracy or decreasing loss. In our case, the network at the first epoch is trained with the learning rate of 10^{-3}, while at the last epoch (epoch 100) the network is trained with 10−3=100 (which is equal to 10^{-5}). As learning started to slow dramatically around epoch 100, we stopped the training phase.

Stochastic gradient descent optimization: In each run, samples are chosen at random from the training data to update parameters during optimization. We set the momentum to 0.9 to speed up the gradient decline and reduce oscillation. Batch normalization: It normalizes the activations of the given input volume before being applied to the next layer in the network.

In our case, the network in each epoch is trained with 35 (70% of the training dataset, 1120 divided by the batch size of 32) examples per mini-batch.

Dropping out: At p = 0.1 following the activation layer and between two convolution layers, it combats fitting problems and improves the performance of our model. A 4-way Softmax classifier is used for grading into a specific class (early stage, infected stage, and completely wilted Enset along with the normal Enset). The Softmax classifier in the training and validation phases helps to construct the learning model, which is further used for detecting and grading samples unseen in the training dataset.

$$Softmax(z_j) = \left(= \frac{e^z j}{\sum_{k=1}^{M} e^z k} \right) \tag{4}$$

2.4 Detection and Grading

Detection and grading are carried out by using the knowledge from the learning model, which is constructed by using training and validation of samples.

3 Results and Discussion

In this section, the dataset and implementation of the proposed model were described. Test results at the training and testing phases. The effects of segmentation are also evaluated. In addition, the test results were presented and compared with the state-of-the-art models.

3.1 Dataset

A total of 1600 images were taken for this study. All images were captured using a slit lamp camera and are in JPEG format having a pixel size of 4608 $\times$ 3570. The images were captured in a controlled environment with illumination on a constant white background by maintaining the camera at a distance of 15 cm and 45° angle with reference to the ground. Canon Digital camera with 16.2 MP was used.

3.2 Implementation

The proposed system is implemented with Keras (Tensor Flow as a backend) using Python programming language. The deep learning system that we developed is called BWENet. The model is trained for 100epochs, a batch size of 32, and a starting or initial learning rate of 0.001. Because we have trained in different parameters we achieved better accuracy. Hyperparameter are selected based on empirical optimization.

The SGD optimizer was chosen because there are indications that most published results use such an optimizer[18]. Momentum value of 0.9 has given better result after training and validating the model. Momentum and learning rate are closely related. The optimal learning rate is dependent on the Momentum and vice versa. In our case, a momentum of 0.9 has given better accuracy and the optimal choice for learning rate clearly depends on the momentum. Momentum of 0.99, 0.97and 0.95) displays signs of over fitting (the upward slant after the minimum loss) during trained our model before diverging (near the learning rate of 0.001). Momentum of 0.9 does not display over fitting and also accelerate our network training. We have trained and validated our model using 100, 200, 50, 150, 80, and 20 epochs. When we train using 159, 200 epochs our model shows over fitting i.e. our model perform quite well on the training data but have high error rates on the test data. On the other hand, when we train with 20, 50 epochs our model shows under fitting didn't well on the training data and also we get have high error rates on the train and test data. We determine the optimal number of epochs to train our model to be able to get good results in both the training and validation data. The right number of epochs depends on the inherent perplexity (or complexity) of dataset. We find that the model is still improving after all epoch's complete; we tried again with a higher value. We found that the model stopped improving way before the final epoch, tried again with a lower value as we may be overtraining. At epoch 100 we achieved better result and also controlling over fitting and under fitting[18].

When we train our model, learning rate (LR) is too small, over fitting occurred. Large learning rates help to regularize the training[15] but if the learning rate is too large, the training will diverge. Hence empirical short Runs to find learning rates that converge or diverge is possible first, in our case the learning rate are too small to make any progress at all. As the learning rate increases, eventually, the loss will begin to decrease, but, at some point, the learning rate will get too large, and the loss will stop decreasing and even begin increasing[11].

Generally we optimize our model with a large learning rate (0.1), and then progressively reduce this rate, often by an order of magnitude (so to 0.01, then 0.001, 0.0001) that are recommended by various researcher. Finally we achieved beter result on

0.001. Dropout forces the units to be more robust, learning feature on their own, without depending on other units. In our case it can be thought of as simplified model resembling. The number of units to retain is controlled by a new hyper parameter; dropout removes units during training, reducing the capacity of the network. To compensate for that the number of units has to be adjusted by the dropout rate, i.e. the number of units has to be multiplied by 1/ (dropout rate). Recommended values for the dropout rate are less than 0.5 for the input layer and between 0.5 and 0.8 for hidden layers [15]. In our case we have introduced new model and applied dropout on input layer train with by added different dropout rate 0.1, 0.2, 0.3, 0.4, 0.5 and achieved better result and control overfitting and underfitting at P=0.1. Using dropout requires some adjustments to the hyper parameter

3.3 Performance metrics

The proposed model has been evaluated using accuracy because we equally distributed the dataset 400 for each class so that data imbalance does not occur. The data is partitioned into training, validation and testing datasets such that 70% of the data is used for training and 15% for validation and 15% for testing. This training, validation and testing split has based on the structure of our model and our dataset are equally distributed for each class (balanced dataset). Accuracy is used for measuring the performance of the proposed system. BWENet achieves higher accuracy than state-of-the-art models. It shows a change in improvement when compared with Alex Net VGG19 and Google Net detection and grading accuracy of trachoma by 6%, 4%and 2% respectively. Alex Net takes smaller time than BWENet and Google Net for the detection and grading of bacterial wilt. However, due to its higher model size, it is difficult to load the model while we try to test new bacterial wilt case

Table 2. Description of the classification model

Model	Accuracy Training	Testing	Model Size (MB)	Training Time (Hr)
Proposed model (BWENet)	93.25	90.53	18.4	3:45
Alex Net	87.18%	84.56%	466.7	3.00
VGG19	89.27%	86.48%	250	5:35
Google Net	91.58%	88.54%	62.5	6:55

As shown in Table 1, our model, BWENet, has 1.4 million parameters (14m), which is significantly less than Alex Net's 58.3 million parameters and Google Net's 6 million parameters. VGG19 model consists of 138 million parameters. As a result, while being more accurate, BWENet learns 40 times fewer parameters than Alex Net, 4 times fewer parameters than Google Net, and 98.5 times fewer than VGG19. The smaller size of BWENet makes it more efficient especially for computational resources such as high memory usage. The application of threshold segmentation boosts the accuracy of BWENet Hence, training the model with the significant threshold segmentation and Appling average and max pooling at the same time has a crucial impact on the optimization of the CNN model. VGG19 and Alex Net both utilize a single filter with a size of 7*7 and 3*3, respectively, and a max-pooling layer; as a result, the most representative feature may not be extracted and over fitting occurred and slow the accuracy of the model and the huge number of parameters leads to increased memory usage. However, our BWENet model employed a 5*5, 3*3, 1*1 filter size, which was chosen based on the shape of the wilted leaf. We enabled the extraction of the most representative high-level and low-level features, as well as the application of max and Average pooling at the same time, and concatenation of each to obtain a single feature and applied threshold segmentation before feeding the data into a model and this boosted the performance of our BWENet by 6%.

The performance of proposed BWENet model and state-of-the-art models are evaluated and compared and shown in the training loss and accuracy curve in Figure 4. Training and validation accuracy increases and validation loss decreases nearly linearly up to epoch 62. Between epochs 62 and 80 training and testing, accuracies decrease too much and run constantly while training and testing loss increases and run constantly. These two signs are typical indications of over fitting occurrence. However, both training and validation accuracies and training and validation losses go close until the final epochs. The gaps were relatively small throughout the curve except between epochs 62 and 80 test accuracy of 80.35%, before threshold segmentation.

As shown in the training loss and accuracy curve in Figure 5, the proposed bacterial wilt disease detection and grading model training accuracy is increased linearly and passes by 93.25% after epoch 76, while the validation accuracy oscillates a little and reaches 93% at some instances. The training loss also decreases intensely to 0.78, while validation loss oscillates and reaches 0.7. This experiment shows that 5*5, 3*3, 1*1 kernel filter and segmentation highly improve the learning capability of the proposed deep learning system.

As shown in the training loss and accuracy curve in Figure 6, VGG19 obtains 89.27% training and 86.48% testing accuracy on our data. It is lower (about 4%) than our model, which contains 93.25% training accuracy and 90.53% testing accuracy, the total time to train the model is around 5:35 hour, and the weight file size for VGG19 250 is much higher than 18.4. In the

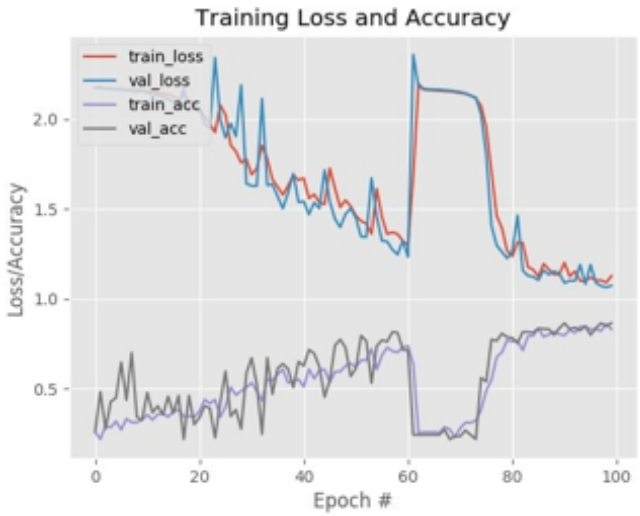

Fig 4. The proposed model (BWENet) training and validation curve before segmentation

Fig 5. Proposed model (BWENet) training and validation curve after segmentation

proposed bacterial wilt disease detection and grading, the size of the model cannot influence the accuracy or loss of the model.

As shown in the training loss and accuracy curve in Figure 7, Google Net obtains 91.58% training and 88.54 % testing accuracy on our data. It is lower (about 2%) than our model, which contains 93.25% training accuracy and 90.53% testing accuracy, the total time to train the model is around 6:55 hours, and the weight file size for Google Net is 62.5 which is much higher than 18.4 MB in proposed model.

The challenge in the experiments is over fitting and oscillation in the training and validation loss or accuracy. It is due to a random sample from our dataset: the dataset at each evaluation step is different, and so is the validation loss. In addition, at each epoch, there are 35 (70% of the training dataset, 1120 divided by the batch size, 32) iterations. At each iteration different samples were taken, trained, and tested, thereby oscillation or fitting occur. It also happens due to the result of the stochastic gradient descent overshooting in the optimal direction or overshooting of areas of the lower loss the comparison is done based on the following parameters. **Accuracy of the model:** a model that record higher accuracy is better for detecting and grading trachoma. **Loss of the model:** a model that records a lower loss value is better for detecting and grading new trachoma cases. **Size of the model**: a model with a smaller size is better at processing the dataset due to the smaller number of parameters trying to learn. **Time taken to train the model**: a model with the fastest (shortest) time is preferred for classifying the testing data.

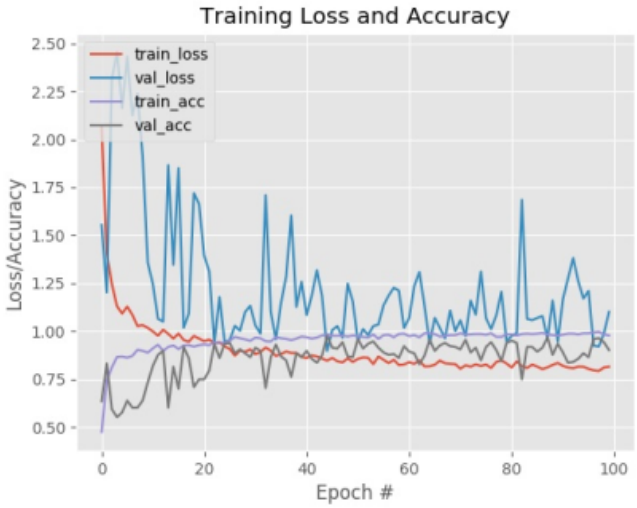

Fig 6. VGG19 model training and validation curve

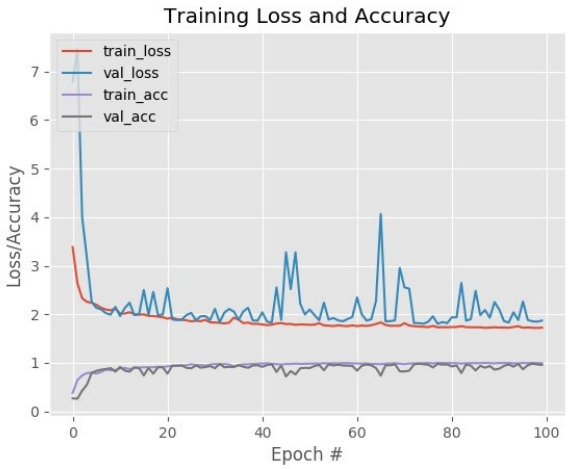

Fig 7. Google Net model training and validation curve

The application of the segmentation of threshold in turn boosts the accuracy of BWENet by 6%. This makes BWENet have a detection and grading accuracy of 90.53%.

Table 3. Comparison with related works

	Architecture	Datasets	Accuracy (%)	Summary
(1)	Multi-class SVM	43 Enset leaf images Belongs to 3 class	92.44%	handcrafted based on feature extraction, didn't measure the Level of disease
(3)	VGG16	4896 images belong to 3 class	96.6%	large number of parameter leads To High memory usage, didn't measure the Level of disease
(4)	CNN	7,014 ginger images belong To 2 class health & disease,	95.2%	only identify disease and health, didn't detect specific diseases and levels of disease

Continued on next page

Table 3 continued

(13)	CNN	1,929 botnets , Only 2 class	98.1%	only considered two class large no of parameter consume large memory
(15)	Lightweight based CNN	19,510 images belong to 10 class	99.34%	didn't Measure, Level of disease, improved performance using large dataset, but higher Parameter
Our model BWENet	BWENet	1600 image (4 class)	90.53%	Measure level of diseases Measure level of diseases with small amount of dataset, higher performance, Less memory usage, fewer parameter

4 Conclusion

A limited study has been done on the automatic detection of Enset plant disease illnesses based on symptoms seen on the leaves. Therefore, this paper aimed to propose a deep convolutional neural network model BWENet for grading the severity of bacterial wilt Enset crop disease. We evaluated the BWENet model through extensive experiments with a 1600 Enset leaf image. The model outperforms several state-of-the-art CNN architectures evaluated on the same dataset. A result of accuracy: 90.53% indicate that our proposed BWENet CNN-based model can be used to grade new, previously unseen Enset plant disease more accurately than the other models. For future work, we will aim to improve the model training process by automating the search and selection of the key influencing parameters (i.e. number of filters size, number of filters, and number of fully connected (dense) layers) that jointly result in the optimal performing CNN model. Furthermore, increasing the size of data, using different pre-trained models, and capturing different angles of the leaf may improve the grading model's performance, and considering other types of disease may be a better model to identify disease. By expanding this research, we hope to have a valuable impact on sustainable development and strengthen the Enset plant value chain.

References

1) Ganore KA, Tigistu G. Ethiopian Enset Diseases Diagnosis Model Using Image Processing and Machine Learning Techniques. *International Journal of Intelligent Information Systems*. 2020;9(1):1–5. doi:10.11648/j.ijiis.20200901.11.
2) Aytenfsu M, Haile B. Distribution and Management of Bacterial Wilt (Xanthomonas Campestris pv . Musacearum) of Enset (Ensete Ventricosum) in Ethiopia : a Review. *International Journal of Research in Agriculture and Forestry*. 2020;7(2):40–53. Available from: https://www.ijraf.org/papers/v7-i2/5.pdf.
3) Afework YK, Debelee TG. Detection of Bacterial Wilt on Enset Crop Using Deep Learning Approach. *International Journal of Engineering Research in Africa*. 2020;51:131–146. Available from: http://dx.doi.org/10.4028/www.scientific.net/JERA.51.131.
4) Yigezu MG, Woldeyohannis MM, Tonja AL. Early Ginger Disease Detection Using Deep Learning Approach. *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*. 2022;411:480–488. doi:10.1007/978-3-030-93709-6_32.
5) Haile B, Fininsa C. Spatial Distribution of Enset Bacterial Wilt (Xanthomonas campestris Pv . musacearum) and its Association with Biophysical Factors in Southwestern Ethiopia Enset (Ensete ventricosum (Welw .) Cheesman) is a monocarpic herbaceous plant. *Ethiopian Journal of Agricultural Sciences*. 2020;30(3):33–55. Available from: https://www.ajol.info/index.php/ejas/article/view/198450.
6) Zerfu A, Gebre SL, Berecha G, Getahun K. Assessment of spatial distribution of enset plant diversity and enset bacteria wilt using geostatistical techniques in Yem special district, Southern Ethiopia. *Environmental Systems Research*. 2018;7(1). doi:10.1186/s40068-018-0126-9.
7) Merga IF, Tripathi L, Hvoslef-Eide AK, Gebre E. Application of Genetic Engineering for Control of Bacterial Wilt Disease of Enset, Ethiopia's Sustainability Crop. *Frontiers in Plant Science*. 2019;10. doi:10.3389/fpls.2019.00133.
8) Deng R, Tao M, Xing H, Yang X, Liu C, Liao K, et al. Automatic Diagnosis of Rice Diseases Using Deep Learning. *Frontiers in Plant Science*. 2021;12. Available from: https://doi.org/10.3389/fpls.2021.701038.
9) Lee H, Song J. Introduction to convolutional neural network using Keras ; an understanding from a statistician Introduction to convolutional neural network using Keras ; an understanding from a statistician. *Communications for Statistical Applications and Methods*. 2019. Available from: https://doi.org/10.29220/CSAM.2019.26.6.591.
10) Rao DS. Plant disease classification using deep bilinear cnn. *Intelligent Automation & Soft Computing*. 2022;31(1):161–176. Available from: https://www.techscience.com/iasc/v31n1/44287.
11) Orbien C. Identification of Carabao Mango Leaf Disease using Convolutional Neural Network. *Journal of Advanced Research in Dynamical and Control Systems*. 2020;12(01-Special Issue):152–158.
12) Subramanian KS, Vairachilai S, Gebremichael T. Features extraction and dataset preparation for grading of ethiopian coffee beans using image analysis techniques. Information and Communication Technology for Intelligent Systems. *Information and Communication Technology for Intelligent Systems*. 2019;106:287–298. doi:10.1007/978-981-13-1742-2_28.
13) Yerima SY, Alzaylaee MK. Mobile Botnet Detection: A Deep Learning Approach Using Convolutional Neural Networks. *Int Conf Cyber Situational Awareness, Data Anal Assessment, Cyber SA*. 2020;2020:15–19. Available from: https://doi.org/10.48550/arXiv.2007.00263.
14) Alkahtani H, Aldhyani THH. Botnet Attack Detection by Using CNN-LSTM Model for Internet of Things Applications. *Security and Communication Networks*. 2021;2021:1–23. Available from: https://doi.org/10.1155/2021/3806459.
15) Bhujel A, Kim NEE, Arulmozhi E, Basak JK, Kim HTE. A Lightweight Attention-Based Convolutional Neural Networks for Tomato Leaf Disease Classification. *Agriculture*. 2022;12(2):228–228. Available from: https://doi.org/10.3390/agriculture12020228.

16) Ramos KSH, Monge MAS, Vidal JM. Benchmark-based reference model for evaluating botnet detection tools driven by traffic-flow analytics. *Sensors (Switzerland)*. 2020;20(16):1–31. Available from: https://doi.org/10.3390/s20164501.
17) Aszemi NM, Dominic PDD. Hyperparameter Optimization in Convolutional Neural Network using Genetic Algorithms. *International Journal of Advanced Computer Science and Applications*. 2019;10(6):269–278. Available from: http://dx.doi.org/10.14569/IJACSA.2019.0100638.
18) Indolia S, Goswami AK, Mishra SP, Asopa P. Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach. *Procedia Computer Science*. 2018;132:679–688. Available from: https://doi.org/10.1016/j.procs.2018.05.069.