Improving Tomato Disease Classification Using BR-TomatoCNN: An Efficient Model Utilizing Bottleneck Residuals

U. Shruthi [1,2,*], V. Nagaveni [1], and Sunil G. L. [3]

[1] Department of Computer Science and Engineering, Acharya Institute of Technology Affiliated to Visvesvaraya, Technological University, Bengaluru, India
[2] Department of Artificial Intelligence and Machine Learning, RNS Institute of Technology, Bengaluru, India
[3] Department of Data Science, RNS Institute of Technology, Bengaluru, India
Email: shruthi.u23@gmail.com (U.S.); nagaveniveerakyatharayappa@gmail.com (V.N.); sunilgl.gls@gmail.com (S.G.L.)
*Corresponding author

***Abstract*—Tomatoes represent a globally significant and commercially valuable crop, yet they are susceptible to a multitude of diseases that can significantly reduce their production and quality. To address this critical issue, we have introduced the BR-TomatoCNN, a novel lightweight Convolutional Neural Network (CNN) model that uses Bottleneck Residuals (BR) to increase the classification accuracy of tomato diseases. This research includes a comprehensive examination of how various optimizers influence the proposed model's performance using evaluation metrics such as accuracy, loss, precision, recall, and F1–Score. A dataset consisting of nine distinct tomato disease classes collected from the Plant Village repository and the Powdery Mildew disease class was prepared with the help of farmers and experts. That was used to train the proposed model achieved remarkable results of 99.82% accuracy and an F1–Score of 1.00. These findings not only underscore the BR-TomatoCNN's capability to accurately identify tomato diseases but also position it as a superior alternative to existing methodologies and pre-trained models. Our study underscores the significance of exploring a new approach, such as utilizing bottleneck residuals to improve the accuracy of the classification model. BR-TomatoCNN promises to play a pivotal role in disease management in the agricultural sector by facilitating early disease detection. This advancement in technology has the potential to enhance tomato crop yields and overall produce quality.**

***Keywords*—convolution neural network, bottleneck residuals, image classification, plant disease detection**

I. Introduction

The agriculture industry has increasingly prioritized the utilization of machine intelligence technology as a means to tackle the issue of food demand. Agricultural activity consists of three distinct stages: pre-harvesting, harvesting, and post-harvesting. The agricultural and food sectors place significant importance on post-harvesting operations such as fruit grading, humidity detection, quality detection, and temperature identification. Machine intelligence models can be utilized to execute these activities. The act of farmers choosing commercial crops has the potential to augment production levels and strengthen food security.

The tomato (Solanum lycopersicum) holds the distinction of being the most widely consumed and economically consequential vegetable crop globally. Tomato plants, meanwhile, exhibit vulnerability to a multitude of diseases resulting from fungal, bacterial, viral, and environmental factors. If these diseases are not promptly diagnosed and controlled, they can result in substantial reductions in yield and deterioration in quality. The timely and precise identification of tomato diseases is crucial for the efficient administration and mitigation of these conditions. Nevertheless, conventional approaches to disease identification, such as expert eye examination or laboratory tests, are laborious, expensive, and reliant on personal judgment. Machine intelligence technologies are necessary for the early detection of diseases in tomato crops. However, the cost of the required computing equipment may be prohibitive for farmers. Hence, there exists a pressing necessity to establish a proficient and precise framework for categorizing and diagnosing tomato plant ailments, to expedite timely intervention and mitigate agricultural yield losses. The objective of this research is to tackle the crucial problem of identifying tomato plant diseases by creating a strong and precise categorization system, which will ultimately be advantageous for tomato growers and the agricultural sector as a whole. Researchers have a challenging problem in developing machine intelligence-based plant disease detection systems for mobile phones with limited processing capacity, aiming to aid farmers.

The major contributions of this research work are:

- A novel lightweight deep learning model of size 0.87 MB has been developed to detect 10 different types of tomato plant diseases. This model is computationally efficient, making it appropriate for

Manuscript received December 25, 2023; revised February 11, 2024; accepted March 11, 2024; published August 7, 2024.

use on devices with limited resources, such as smartphones, through mobile applications. The minimal specification for a viable smartphone is a minimum of 4 GB of Random Access Memory (RAM) and 16 GB of internal storage, a high-performance CPU, and compatibility with either the Android or the iPhone Operating System (iOS**)**.

- The Bottleneck Residuals (BR) in the BR-TomatoCNN model help to reduce the computational complexity of the model while maintaining its predictive performance.
- To explore the attention mechanism to handle specific characteristics of tomato diseased leaf images, such as variations in color, shape, or size.
- Use various optimizers, like Stochastic Gradient Descent (SGD), Adam, Root Mean Squared Propogation (RMSProp), Adagrad, Adadelta, Adamax, and Adabelief, to find the best one with the BR-TomatoCNN model.
- The proposed model is tested on nine different types of tomato plant diseases from the plant village dataset. One more diseased class with Powdery Mildew gained the best accuracy, precision, and F1–Score compared to five Convolutional Neural Network (CNN) models that the author had already trained on the same dataset.

This article is structured with subsequent sections, starting with a literature survey on plant disease detection discussed in Section II. Section III outlines the materials and methodology used in our research, which includes a description of the dataset, pre-processing, and BR-TomatoCNN model. Section IV focuses on the details of the experiments carried out and the discussion of the research contributions, followed by conclusions.

II. Related Work

Machine learning algorithms utilize historical information on agricultural yield, weather, soil conditions, and other relevant factors to predict future crop yields [1]. Targeted weed control measures, enabled by machine learning algorithms trained to recognize various weed species, can lead to a reduction in the use of herbicides [2]. The accurate classification of fruits based on features such as size, shape, color, and ripeness remain crucial for sorting and grading in fruit processing facilities [3]. Machine learning techniques are also applied in plant disease diagnosis and other agricultural applications. Researchers have developed classifiers for plant disease recognition using machine-learning algorithms, but they face a basic flaw as they require external methods for feature extraction, and their performance tends to degrade with an increasing number of data points [4]. These challenges can be overcome by employing deep learning methods. In deep learning models like CNN, features are automatically learned from raw pixel data through convolutional layers. These learned features are hierarchical, enabling them to capture complex patterns and hierarchies.

Deep learning algorithms play a crucial role and find widespread application in various fields. The authors have developed pre-trained CNN architectures for the classification of plant-diseased leaf images. Rangarajan *et al.* [5] employed a transfer learning approach on pre-trained deep-learning models such as AlexNet and VGG16 to identify tomato crop maladies in seven classes, with six being diseased and one healthy. The author asserts that AlexNet outperforms VGG16 in terms of both execution speed and accuracy. Ferentinos [6] created and compared multiple CNN models to detect maladies in 25 plant categories, encompassing 58 classes and 87,848 leaf images, including both healthy and diseased leaves. VGGNet achieves higher accuracy compared to AlexNet, GoogleNet, and Overfeat. Mohanty *et al.* [7] proposed a model, which is trained on the Inception model for 26 classes in the plant village dataset to achieve an accuracy of 98.36%. Saleem *et al.* [8] evaluated pre-trained CNN architectures on 38 classes of plant village datasets, noting better performance on the Xception architecture. Zaki *et al.* [9] developed the MobileNet model trained on a smaller number of images, with a maximum accuracy of 95%. All pre-trained architectures are designed with a larger number of layers for the ImageNet dataset and the weights are saved. The use of CNN models can help mitigate the problems caused by larger models trained on datasets with fewer classes.

Many researchers have developed the CNN architecture for plant disease classification. Chen *et al.* [10] created a modified pre-trained VGG19 model that includes two Inception modules. They showed that it works better than other pre-trained CNN models by testing it on a dataset of images of rice and maize crops. Khan *et al.* [11] proposed an integrated Sage Maker into Amazon Web Services (AWS) DeepLens for plant evaluation in real-time using a transfer learning method in the cloud with scalability on AWS. This paradigm might not be usable by farmers in developing countries with limited resources. Agarwal *et al.* [12] proposed the CNN model with three layers of convolution and max-pooling layers. The CNN model trained on 39 classes of the plant village dataset in 5000 epochs to achieve the best result. Ahmed *et al.* [13] developed a mobile application using a three-convolution layer deep learning model trained on 38 classes of the plant village dataset for 10 epochs, which is obtained with 93.6% accuracy.

Patil *et al.* [14] have proposed a hybrid LSTM-CNN model that includes a Long Short-Term Memory (LSTM) layer added before the CNN layers and achieved 97.5% accuracy on a dataset of six classes of tomato diseases. Pal *et al.* [15] created the Inception-VGG and Kohonen-based models for finding diseases in multiple crops. They used the grabcut algorithm to fix problems with occlusion, which led to a 95.64% success rate. However, Kohonen networks may face scalability challenges with larger datasets. These studies collectively highlight the importance of model choice, preprocessing, and dataset size in plant disease detection. Shelke *et al.* [16] presented an innovative in plant leaf classification method that employs convolutional neural networks (CNN) for effective classification. The classifier has

demonstrated notable proficiency in the model. Gosh *et al.* [17] combined the power of CNN with PB3C, aiming for an enhanced and efficient leaf classification system. The optimization techniques are applied to exhibit a commendable trait of high convergence, indicating a promising feature for the model's performance. However, it is essential to note a significant drawback: the optimization process is susceptible to falling into local optima, posing a challenge that warrants careful consideration in the design and implementation of the classification system.

The above-studied existing models often have high computational and memory requirements, making them challenging to integrate with resource-constrained hardware devices, which are commonly used in agriculture. Many deep learning models are designed for image classification tasks, including those for plant disease recognition, and can have a large number of parameters. This leads to increased computational demands during both training and inference. Large models may not be feasible for deployment on low-power devices. These issues are to be addressed by developing a lightweight model that is more suitable for deployment on mobile devices and can facilitate real-time tomato plant disease classification in the field. Our research study is focused on a lightweight deep-learning model using bottleneck residuals with the best suitable optimizers for tomato plant disease classification.

III. Materials and Methods

Our research addresses the challenges of classifying tomato plant pathology using CNN models. The BR-TomatoCNN model uses the bottleneck residuals to make it possible to build a deeper CNN architecture without the need for more computing power. Deeper networks have the potential to capture more complex and abstract features, leading to improved classification accuracy. A structured flow of our research is illustrated in Fig. 1.

Initially, the Plant Village dataset is collected, and the data augmentation technique is used to enhance the diversity in training the model and to ensure fairness across all the classes. Model optimization employs the Adam optimizer to train the five pre-trained models and the proposed BR-TomatoCNN model. The ability of Adam Optimizer to adapt its learning rate makes it suitable for a variety of tasks, which is the driving force behind the choice. The continuous monitoring of training and validation results using the accuracy and loss of the metric to select the most promising CNN model. These results show that the BR-TomatoCNN model is superior to all the pre-trained models. To enhance the performance, it has been experimented with different optimizers by fine-tuning hyperparameters such as learning rate, moments of gradients, epsilon, and decay. This rigorous optimization process aims to maximize the BR-TomatoCNN model's accuracy in tomato disease classification. Ultimately, the novelty of our approach is demonstrated by analyzing the performance of the BR-TomatoCNN model with the Adadelta optimizer. This model demonstrates its effectiveness and uniqueness in tomato disease classification.

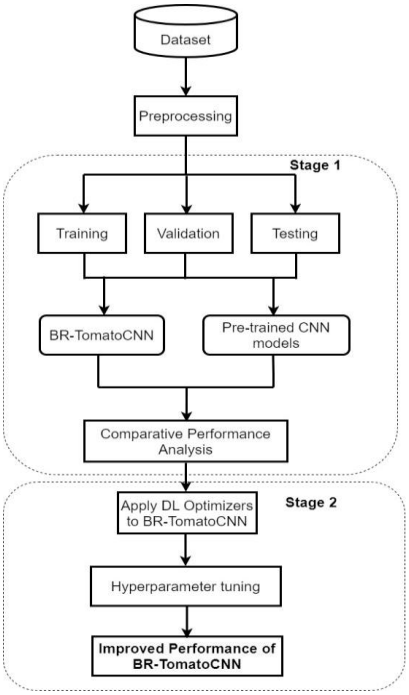


Fig. 1. The approach of this research.

A. Dataset and Pre-processing

The BR-TomatoCNN model is trained on a publicly available tomato plant disease leaf dataset from Plant Village [18]. That consists of nine classes of tomato diseases and healthy leaves. Three more classes are added to the available dataset: one is powdery mildew disease, and the other two are negative classes. Negative classes are non-tomato leaves, and non-leaf images are included to avoid miss-classification issues. Tomato powdery mildew disease: 173 images are collected using a mobile phone camera and included to identify more additional tomato diseases compared to existing models. The curated dataset consists of 10 tomato leaf diseased classes (class No. 1–10 is given in Table I), a healthy tomato leaf class (class No.11 in Table I), and 2 negative classes (class no. 12 and 13 in Table I). Sample images from each class considered in the dataset to train the proposed model are shown in Fig. 2.

TABLE I. Description of the Dataset

Class No.	Class Label	Number of Images
1	BacterialSpot	2,127
2	EarlyBlight	1,000
3	LateBlight	1,909
4	Leaf Mold	1,000
5	SeptoriaLeafSpot	1,771
6	SpiderMites	1,676
7	TargetSpot	1,404
8	YellowLeafCurlVirus	2,704
9	TomatoMosaicVirus	1,000
10	PowderyMildew	1,258
11	Healthy	1,591
12	NonTomatoLeaf	2,642
13	NonLeaf	1,143
	Total	**21,225**

Fig 2. Example images of different classes: (a). TargetSpot, (b). TomatoMosaicVirus, (c). YellowLeafCurlVirus, (d). BacterialSpot, e. Healthy, (f). EarlyBlight, (g). LateBlight, (h). LeafMold, (i). SeptoriaLeafSpot, (j). SpiderMites, (k). PowderyMildew, (l). NonTomatoLeaf, (m). NonLeaf.

In comparison to the number of images in all classes, the Tomato Mosaic Virus and Powdery Mildew classes have fewer images. Hence, to increase the number of images, the rotation and flip method of data augmentation is used in preprocessing. The class labels of the dataset and the number of images considered in the dataset for each class are given in Table I. The total number of images considered in the dataset is 21,225. Resize is applied to all images to 227 by 227 and scales image pixels between 0 and 255 in the pre-processing stage.

B. BR-Tomato CNN Model

Many researchers rely on a pre-trained model for plant disease identification that is trained for 1,000 classes of image data, and models have a large number of layers [19–21]. More layers in the model necessitate more storage and processing time for parameter management. Overfitting may occur when pre-trained models are applied to available datasets and lead to inaccurate results. A fundamental linear model without a hidden layer would result in underfitting and an incorrect conclusion. Consequently, a CNN model is more efficient when the proposed number of classes is smaller.

A novel BR-TomatoCNN lightweight model is proposed to classify tomato plant diseases using bottleneck residuals [22]. Lightweight models typically have fewer parameters, and the number of parameters in the proposed BR-TomatoCNN is 223,653. A lower parameter count reduces the model's size and memory footprint. Bottleneck residuals are additional layers to the BR-TomatoCNN model; they often have fewer parameters compared to traditional stacked layers. This reduces the model's complexity and helps to prevent overfitting. Especially when the dataset is limited, the proposed model is improved with generalization to unseen data. The BR-TomatoCNN model includes three basic parts: stem, body, and head, as shown in Fig. 3, and a layer description is given in Table II.

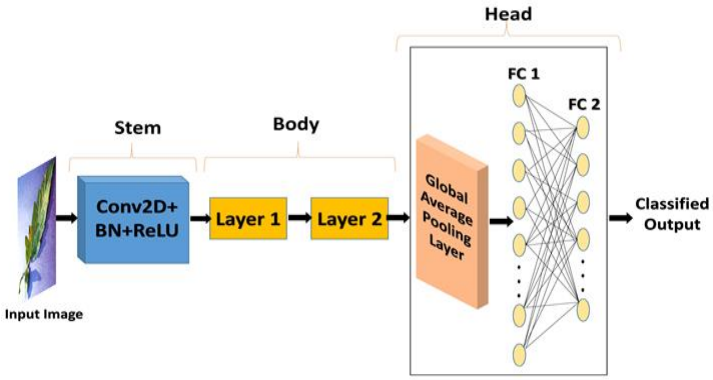


Fig. 3. The BR-TomatoCNN architecture.

TABLE II. SUMMARY OF THE PROPOSED MODEL'S MAIN LAYERS

Type of the Layer	Input Size	Output Size
Stem	227×227×3	114×114×32
Layer_1_Bottleneck_ Residual_Block_0	114×114×32	57×57×64
Layer_2_Bottleneck_Residual_ Block _1	57×57×64	29×29×128
Layer_2_Bottleneck_Residual_ Block _2	29×29×128	15×15×128
Global Average Pooling	15×15×128	128
Dense	128	512
Dense	512	13
Total parameters	223,653	

The stem layer plays a crucial role in shaping the hierarchy of feature extraction in the network and can significantly affect the model's performance and efficiency in tomato disease classification. In a BR-TomatoCNN model, the stem layer is used to do initial data preprocessing and feature extraction. This prepares the input image of a diseased tomato leaf for the next layers of the network. The stem portion consists of a convolution operation with 32 filters, each of size 3×3, and a downsampling with stride 2 to accomplish it. Then, Batch Normalization (BN) and ReLU activation will occur. The stem part can be expressed as:

$$\text{Stem} \leftarrow \text{ReLU}\,(\text{BN}\,(\text{Conv}\,(32, (3, 3), \text{Stride}=2)))$$

The body part is composed of two layers, each composed of bottleneck residuals. Layer 1 includes one bottleneck residual with 64 filters, and two bottleneck

residuals are adopted in Layer 2 with 128 filters. A smaller number of bottleneck residuals simplifies the model architecture. This reduction in complexity makes the model easier to train, interpret, and optimize, especially when dealing with limited computational resources. Reducing the model's complexity decreases the risk of overfitting. Overfitting occurs when a model learns to fit the training data too closely and performs poorly on unseen data. Fig. 4 shows the structure of the bottleneck residual block in the BR-TomatoCNN model.

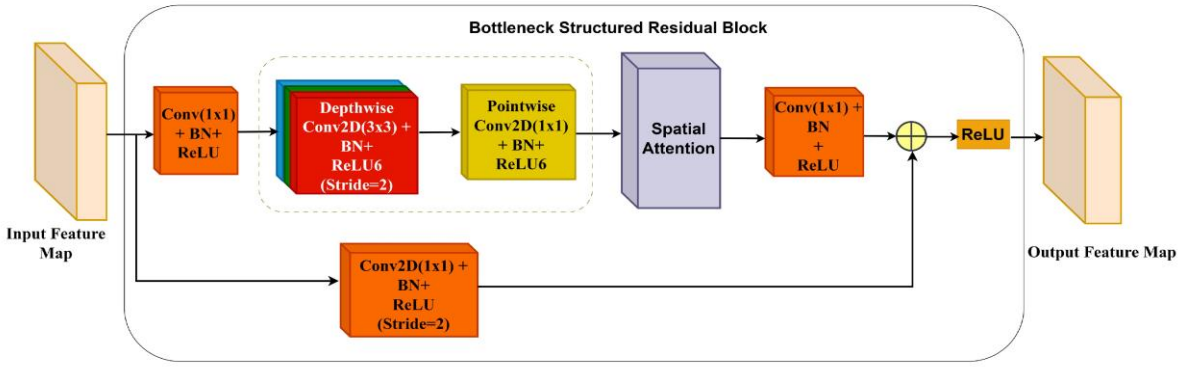


Fig. 4. Bottleneck structured residual block.

Steps 1 through 5 show the order of layer operations for the bottleneck residual block.

Step 1: The model takes the input feature map from Stem and applies a 1×1 convolution with a stride of 2, which reduces the spatial dimensions of the feature map and introduces some non-linearity. Followed by batch normalization, a ReLU activation function is applied, and the result is stored in X_{skip}. This skip connection is used in training networks by allowing gradients to flow more easily during backpropagation and is expressed as:

$$X_{skip} \leftarrow \text{ReLU (BN (Conv (Stem, 1×1, stride=2)))}$$

Step 2: In this layer, the input feature maps are taken from Stem and apply a 1×1 convolution with a stride of 1, which performs a linear transformation of the input features. The batch normalization is applied to normalize the output of the convolutional layer, and the ReLU activation function introduces non-linearity. The result is stored in X_1, which is used as input to subsequent layers in the model. It is given as:

$$X_1 \leftarrow \text{ReLU (BN (Conv (Stem, 1×1, stride=1)))}$$

Step 3: To reduce the number of multiplication operations, depthwise separable convolution is used in conjunction with batch normalization and ReLU6 activation [23]. The two layers of the depthwise separable convolution operation are depthwise and pointwise. Applying a 3×3 filter and a stride of 2, along with batch normalization and ReLU operations, results in the depthwise convolution (X_2). Applying a 1×1 filter with a stride of 1 and then performing batch normalization and ReLU operations results in the pointwise convolution (X_3). The depthwise convolution is used in the development of the BR-TomatoCNN model, which can be used in mobile and resource-constrained applications due to its efficiency and ability to capture spatial features. Pointwise convolution is often used to perform channel-wise adjustments and helps reduce or expand the number of feature channels to capture different levels of abstraction in the image data. The depthwise separable convolution is expressed as:

$$X_2 \leftarrow \text{ReLU6 (BN (DeptwiseConv}(X_1, 3\times3, \text{stride=2})))$$

$$X_3 \leftarrow \text{ReLU6 (BN (PointwiseConv}(X_2, 1\times1, \text{stride=1})))$$

Step 4: The ElementMultiply is an element-wise multiplication operation that is applied between the SpatialAttention feature maps and the X_3 feature maps, and the result is stored in X_4 and given as:

$$X_4 \leftarrow \text{ElementMultipy } \{\text{SpatialAttention}, X_3\}$$

The spatial attention operation is included in the bottleneck residuals that give more importance to the diseased regions of tomato leaves in the feature map while suppressing other regions. The process of spatial attention block (X_4) is adopted in the bottleneck-structured residual block, as shown in Fig. 5. Spatial attention is used to focus more on the information on diseases in the leaf, and it includes global average pooling and two convolution layers. The output features of the first convolution layer constitute 25% of the spatial attention block's input. The output features of the second convolution layer are equal to the input features of the spatial attention block. The spatial attention operation is expressed as:

$$\text{SpatialAttention} \leftarrow \sigma\,(\text{Conv (ReLU (Conv (GlobalAvgPooling } (X_4)))))$$

Step 5: The result of 1×1 convolution applied to the feature map X_4, followed by batch normalization, is stored in X_5, which is added to the features from the earlier layer (X_{skip}), followed by the ReLU activation function, and the obtained resultant feature maps are

stored in X_6. This sequence of operations, designed to extract and combine features while allowing information from earlier layers to influence the final output, is expressed as:

$$X_5 \leftarrow BN\ (Conv\ (X_4,\ 1\times1,\ stride=1)$$

$$X_6 \leftarrow ReLU(X_5+Xskip)$$

The head part of the model is the classification phase, which takes the features extracted from the previous layer (X_6) and performs global average pooling to reduce the spatial dimensions. The fully connected layers (FC_1 and FC_2) are applied to transform the features, and finally, the Softmax activation function is used to produce the class probabilities in X_{Output}. The head part of the BR-TomatoCNN model is expressed as follows:

$$X_7 \leftarrow GlobalAvgPooling(X_6)$$

$$X_8 \leftarrow FC_1(X_7,\ 512,\ ReLU)$$

$$X_{OutPut} \leftarrow FC_2(X_8,\ 13,\ Softmax)$$

where, FC=Fully Connected, FC_1 and FC_2 generally expressed as FC (inputFrom, #nodes, activation).

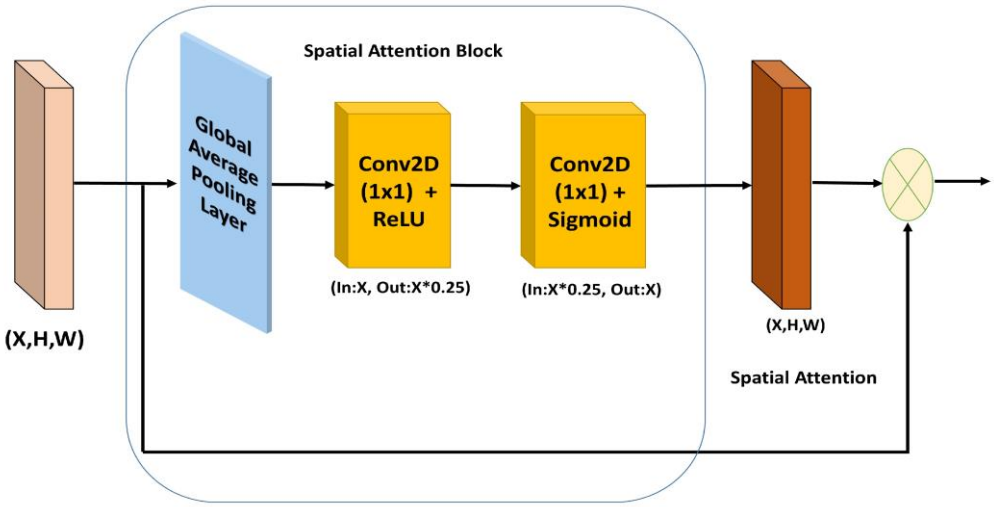


Fig. 5. Spatial attention block.

The bottleneck residual in the proposed model is improved by training speed, model efficiency, and the ability to extract intricate features from tomato plant leaf images. These benefits contribute to the development of an accurate and efficient tomato disease detection system, ultimately aiding farmers and agricultural experts in managing plant diseases effectively.

C. Comparison of BR-TomatoCNN with Pretrained Models

CNN models have been extensively utilized for image classification in a variety of fields, including the classification of tomato plant diseases. In previous studies, researchers have utilized pre-trained architectures such as AlexNet [24], VGGNet [25], GoogleNet/InceptionNet [26], ResNet [27], and EfficientNet [28] to classify tomato plant diseases [29]. These pre-trained models include significant layers and filters in each layer because they were developed for 1000 classes of ImageNet data. The cost of these pre-trained models is considerable since they require high storage memory and take a long time to execute. Furthermore, their implementation on edge devices can be challenging.

The comparison of the proposed BR-TomatoCNN model with pre-trained models is as follows:

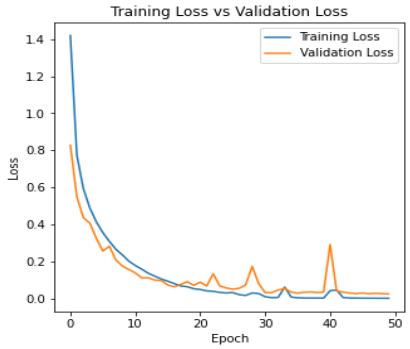
- ResNet: ResNet introduced the residual connections to address the vanishing gradient problem. It uses skip connections to add the input to the output of multiple layers. Effective in training deep neural networks and avoiding degradation issues. This can be computationally expensive and memory-intensive.
- MobileNet and MobileNetV2: An improvement over MobileNet, it introduces inverted residuals and linear bottlenecks for better performance. Improved accuracy over MobileNet is still efficient for mobile and edge devices. This may not match the accuracy of larger models like ResNet in some scenarios.
- RegNetX and RegNetY: Introduces the concept of "network design space" and scaling to achieve a balance between model size and performance. Flexible design space allows for efficient scaling for various tasks. The specific architecture and scaling may not be suitable for all use cases.
- BR-TomatoCNN: The BR-TomatoCNN incorporates bottleneck residual blocks, which typically consist of depthwise convolution and spatial attention mechanisms. The bottleneck structure reduces computational costs, and spatial attention can enhance model performance by focusing on diseased regions.

IV. Result and Discussion

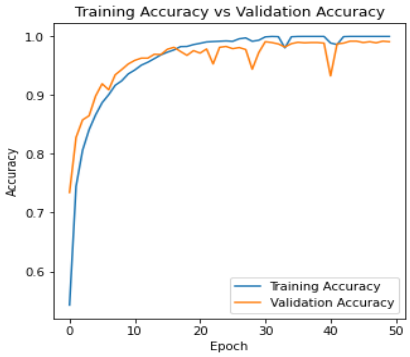
Keras and TensorFlow libraries are used to implement CNN models. To speed up the work with TensorFlow, the cuDNN 8.6 libraries are installed. All of these experiments are run on an NVIDIA GeForce RTX 3090 GPU with 24GB of memory and a 1.7 GHz boost clock speed, as well as a Core i9 processor and 128GB of RAM in the central processing unit. The metrics used to measure the performance of pre-trained CNN models and the BR-TomatoCNN model in multi-class classification are the loss function, accuracy, specificity, precision, recall, and F1–Score [30–32]. To find the error in the CNN model, the categorical cross-entropy loss function is employed.

A. Performance of BR-TomatoCNN Model

In the initial stage of our research, experiments are conducted on a dataset of infected tomato leaves, which is provided in Section II-A. Initially, the dataset is divided into three sections: training for 80%, validation for 10%, and testing for 10%. The BR-TomatoCNN and pre-trained classifiers are trained for 50 epochs with the Adam optimizer. After each epoch, the model is validated using the validation dataset, and the final trained model is evaluated using unseen images. The BR-TomatoCNN model resulted in a higher accuracy of 99.02% and a lower error rate of 0.0373 on the testing dataset, as shown in Fig. 6. Precision, recall, and F1–Scores have reached 0.99. BR-TomatoCNN outperforms in comparison of the results of five pre-trained CNN models, as given in Table III. Fig. 7 demonstrates that the BR-TomatoCNN model outperforms all pre-trained models.



(a)



(b)

Fig. 6. Performance of BR-TomatoCNN using Adam optimizer: (a) Training loss vs Validation loss, (b) Training loss vs Validation accuracy.

TABLE III. CNN Models Testing Results

Models	Loss	Accuracy(%)	Precision	Recall	F1–Score
ResNet50	0.0809	98.27	0.98	0.98	0.98
MobileNet	0.2511	92.40	0.92	0.92	0.92
MobliNet V2	0.3383	89.65	0.90	0.90	0.90
RegNetX	0.1185	97.09	0.97	0.97	0.97
RegNetY	0.0482	98.32	0.98	0.98	0.98
BR-TomatoCNN	0.0373	99.02	0.99	0.99	0.99

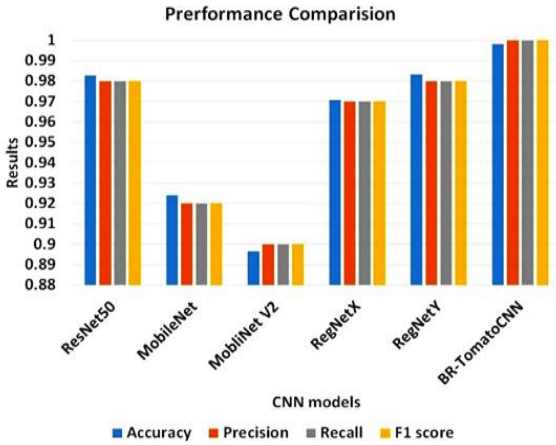


Fig. 7. Visualization of the BR-TomatoCNN model's performance analysis.

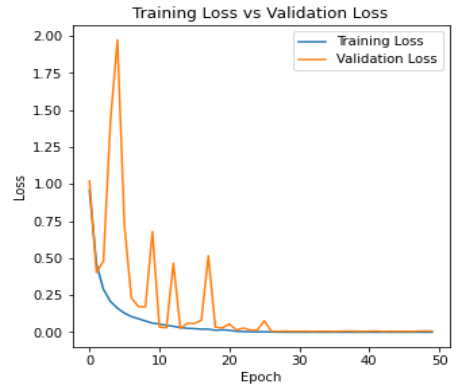
B. Enhanced Performance of BR-TomatoCNN Model

In the second part of our study, seven different deep-learning optimizers are used on the BR-TomatoCNN model to check how well they work. These are SGD, RMSProp, Adagrad, Adamax, Adadelta, and Adabelief. The Adamax and Adadelta optimizers with hyperparameter tuning yielded superior results in comparison to the Adam optimizer used in the initial phase of our research. When a learning rate of 0.5, a rho of 0.95, and an epsilon of 1e-06 are used with the Adadelta optimizer on a testing dataset, the proposed model does better. It can achieve a lower error rate of 0.008, a higher accuracy of 99.82%, and 1.0 precision, 1.0 recall, and 1.0 F1–Score, as shown in Table IV.

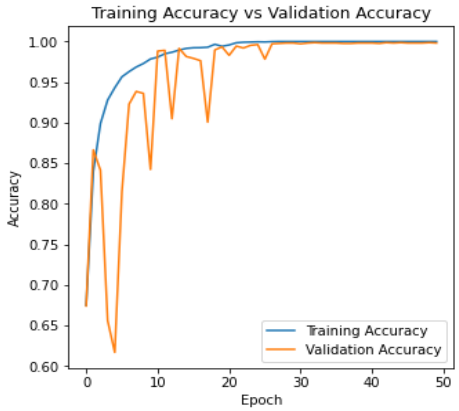
TABLE IV. BR-TomatoCNN model with Optimizer's Performance Report on Test Dataset

Optimization Techniques	Loss	Accuracy (%)	Precision	Recall	F1–Score
SGD	0.032	98.69	0.99	0.99	0.99
RMSprop	0.048	98.97	0.99	0.99	0.99
Adam	0.037	99.02	0.99	0.99	0.99
Adamax	0.017	99.49	0.99	0.99	0.99
Adadelta	0.008	99.82	1.00	1.00	1.00
Adagrad	0.057	98.23	0.98	0.98	0.98
AdaBelief	0.082	97.95	0.98	0.98	0.98

The BR-TomatoCNN model with the Adadelta optimizer is the optimal model for classifying tomato plant diseases. Fig. 8 depicts the graph of loss and accuracy on training versus validation outcomes for 50 epochs of the BR-TomatoCNN model. After epoch number 27, the model becomes stable, and the difference between the accuracy of training and validation is reduced.



(a)



(b)

Fig. 8. Performance of BR-TomatoCNN model using Adadelta optimizer: (a) Training loss vs Validation loss, (b) Training loss vs validation accuracy.

The confusion matrix is used to predict the results of the BR-TomatoCNN model on the testing dataset, as shown in Fig. 9.

The specificity, precision, recall, and F1–Score between the positive predictive value and the true positive rate are used to rate the performance of each class. These metrics are shown in Table V. The BR-TomatoCNN model reduces the number of false negatives and achieves excellent precision.

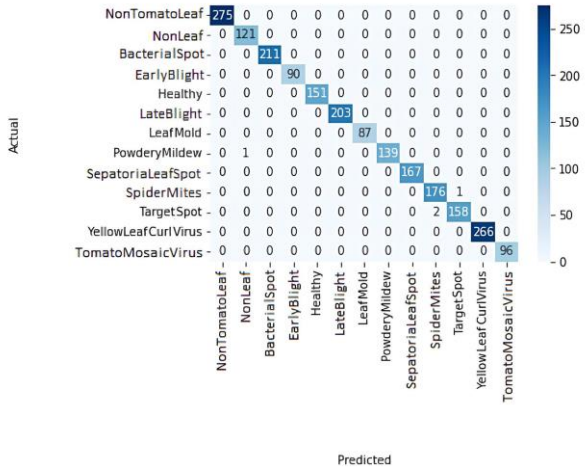


Fig. 9. Confusion Matrix of BR-TomatoCNN model with Adadelta optimizer.

TABLE V. BR-TOMATOCNN MODEL WITH ADADELTA OPTIMIZER PERFORMANCE REPORT OF ALL THE CLASSES

Class Label	Specificity	Precision	Recall	F1–Score
NonTomatoLeaf	1.00	1.00	1.00	1.00
NonLeaf	0.99	0.99	1.00	1.00
BacterialSpot	1.00	1.00	1.00	1.00
EarlyBlight	1.00	1.00	1.00	1.00
Healthy	1.00	1.00	1.00	1.00
LateBlight	1.00	1.00	1.00	1.00
LeafMold	1.00	1.00	1.00	1.00
PowderyMildew	1.00	1.00	0.99	1.00
SeptoriaLeaf Spot	1.00	1.00	1.00	1.00
SpiderMites	0.99	0.99	0.99	0.99
TargetSpot	1.00	0.99	0.99	0.99
YellowLeafCurlVirus	1.00	1.00	1.00	1.00
TomatoMosaicVirus	1.00	1.00	1.00	1.00
Average Accuracy				**1.00**
Macro Average		**1.00**	**1.00**	**1.00**
Weighted Average		**1.00**	**1.00**	**1.00**

C. Discussion

The authors have developed a novel BR-TomatoCNN that incorporates bottleneck residual blocks with a limited number of parameters. This approach effectively decreases both the computational complexity and training time. The CNN model proposed incorporates three bottleneck residuals within layers consisting of 64 and 128 filters. This framework presents a robust approach for the timely identification of tomato plant diseases. By integrating the advantages of depth, hierarchical feature extraction, and interpretability, this approach has the potential to provide precise crop disease control solutions. The utilization of bottleneck residuals has resulted in enhanced network efficiency and reduced computational expenses. The utilization of a depthwise convolution layer and pointwise convolution operations within the residual layer results in a decrease the number of multiplication operations, hence leading to a drop in both the number of calculations and the training time. The combination of global average pooling and two convolutional layers forms a spatial attention mechanism that specifically highlights the unhealthy regions of a tomato leaf image while concealing the remaining areas. The performance of the BR-TomatoCNN model has been enhanced through parameter reduction, mitigation of overfitting issues, and enhancement of the model's capacity to acquire significant features. Furthermore, the suggested methodology exhibits not only high accuracy and computational efficiency but also has strong generalization capabilities when applied to novel and unfamiliar data. This is crucial for the detection of diseases in tomato plants since it allows the model to precisely identify diseases in various growing conditions and geographic areas.

The proposed BR-TomatoCNN model has been evaluated in two phases using a dataset that has been developed and trained for 50 epochs. This approach yields the highest level of accuracy in classifying 10 distinct types of tomato plant diseases, with no instances of misclassification. The BR-TomatoCNN model achieves an accuracy of 99.02% in the initial stage, outperforming the results of five pre-trained models using Adam Optimizer. The performance of the proposed

approach is improved in the second stage by experimentation with several optimizers, such as SGD, RMSProp, Adamax, Adagrad, Adadelta, and Adabelief. The Adadelta optimizer on the improved BR-TomatoCNN model results in an accuracy of 99.82%, which is better than the approaches suggested in [33–37].

This paper presents a case study on the BR-TomatoCNN model for tomato plant disease classification through experiments. Experiments are conducted using the dataset described in Section III-A. Images are randomly selected from each class for the splitting of the training, validation, and testing datasets. The results of each experiment are given in Table VI.

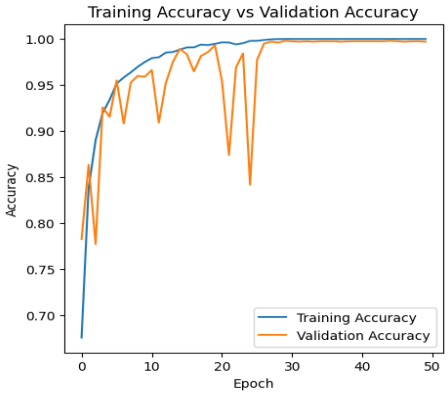
TABLE VI. CASE STUDY ON BR-TOMATOCNN MODEL

Experiment No.	Accuracy (%)			F1–Score
	Training	Validation	Testing	
1	100	99.71	99.71	1.00
2	100	99.76	99.66	1.00
3	100	99.61	99.86	1.00

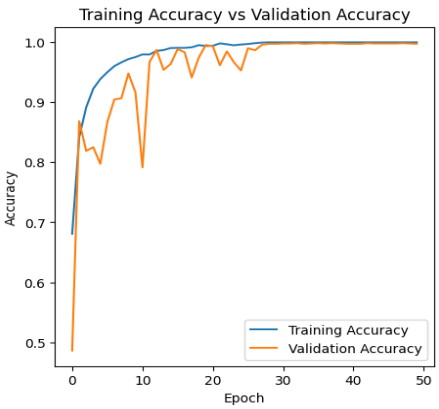
The results demonstrate that the BR-TomatoCNN model is successful at identifying and classifying tomato diseases with high accuracy across all three experiments carried out, as shown in Fig. 10. To verify the robustness of the algorithm, three data sets are derived from the main dataset described in Section III-A. A random selection of image data members from the main dataset allows for the split-up of the dataset. This is meant to ensure that the derived datasets truly represent variety in the data. BR-TomatoCNN models Experiment 1 performance is as shown in Fig. 10(a). It can be concluded that the validation result of the BR-TomatoCNN model has become stable after the 27th epoch and has achieved a testing accuracy of 99.71% at the 50th epoch. Similarly, as shown in Fig. 10(b), the model has become stable after the 28th epoch and has achieved an accuracy of 99.66% in Experiment 2. In Experiment 3, as shown in Fig. 10(c), the model has become stable after the 27th epoch and has achieved an accuracy of 99.86%. In comparison with the testing results of each experiment on the BR-TomatoCNN model, it is proven that the model is robust.

When numerous diseases occur in a plant, the proposed model may have the limitation that it can only identify the most likely diseases. The diversity of the training dataset can be increased by incorporating images from different sources, under different lighting conditions, and capturing various stages of tomato growth. The techniques for online learning or incremental training for a model can be explored to adapt to new data over time, especially in dynamic agricultural environments.

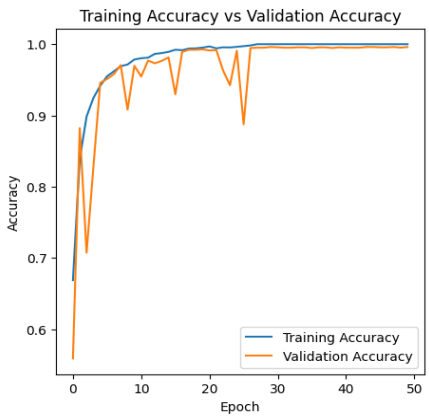
To integrate domain expertise into the development process, we intend to collaborate with specialists in agriculture and tomato farming to interface the BR-TomatoCNN model with an edge device. Because resources are limited in some areas, especially in agriculture, where computers are often hard to come by, it is important to lower the processing needs of deep neural networks. To address this, we need to employ network compression techniques such as fine-tuning [38], quantization [39], pruning [40], knowledge distillation [41], weight clustering [42], and even frameworks like OpenVINO [43], with the help of which we can optimize the models. Among various compression strategies, network pruning stands out as a commonly utilized method. This approach follows a traditional three-stage pipeline where large models are initially trained, redundant weights are subsequently pruned while retaining essential ones, and fine-tuning is performed to optimize accuracy. When the BR-TomatoCNN model undergoes pruning to reduce its size, extraneous connections and neurons are eliminated based on their weight magnitude, which is indicative of their importance level. Such an approach holds significant potential to improve the model's characteristics and practical utility in the agricultural field.



(a)



(b)



(c)

Fig. 10. Training versus Validation Accuracy: (a) Experiment 1, (b) Experiment 2, (c) Experiment 3.

V. Conclusion

This research presents a unique model called BR-TomatoCNN, which incorporates bottleneck residuals to improve the classification of tomato diseases. The present study aims to meet the growing need for intelligent systems in the field of agriculture by offering a highly effective and precise solution for the automated diagnosis of diseases affecting tomato plants. The inclusion of bottleneck residuals in the model's design significantly enhances training stability, gradient flow, and the model's capacity to identify disease-related challenging characteristics. The experiment is carried out in two phases using a dataset comprising 10 distinct categories of tomato disease classification. During the preliminary phase, the outputs of the BR-TomatoCNN model are analyzed using the results acquired from the five pre-trained models. The experimental results exhibit promise and provide evidence for the efficacy of the BR-TomatoCNN model in the classification of tomato diseases. The model has demonstrated superior performance compared to the pre-trained models, with an excellent accuracy rate of 99.02% on unseen data. During the second step, the BR-TomatoCNN model's performance is improved by implementing several optimization techniques. The utilization of the Adadelta optimizer has yielded noteworthy accomplishments, exhibiting the highest accuracy of 99.82% and an F1–Score of 1.00 when applied to previously unexplored data.

The findings of this study highlight the significance of careful optimizer selection in determining the overall performance of the model. The algorithm's robustness has been validated using three derived datasets. In their future research, the scientists intend to investigate the efficacy of the model under different climatic circumstances and datasets encompassing a wide range of tomato disease severity levels. Furthermore, it is important to take into account the possibility of scaling and implementation in real-world agricultural contexts to optimize the practical implications of the model.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

U.S. played a central role in this study, contributing to the conceptualization, methodology, validation, formal analysis, investigation, allocation of resources, and the preparation of the original draft. U.S is also responsible for meticulously reviewing and editing the manuscript and creating data visualizations. V.N. provided valuable supervision and oversaw the project administration, ensuring that the research proceeded smoothly and efficiently. S.G.L. contributed significantly to data curation and played a vital role in preparing the initial draft of the manuscript and enhancing the quality of the final document; all authors had approved the final version.

Acknowledgment

We extend our appreciation to the farmers for providing access to their agriculture farms and the Department of Horticulture, Government of Karnataka, for providing expertise to identify Powdery Mildew diseased leaves.

References

[1] G. L. Sunil, V. Nagaveni, and U. Shruthi. "A review on Prediction of crop yield using machine learning techniques," in *Proc. IEEE Region 10 Symposium (TENSYMP)*, Mumbai, India, 2022.

[2] Z. G. Wu, Y. J. Chen, B. Zhao, X. B. Kang, and Y. Y. Ding, "Review of weed detection methods based on computer vision," S*ensors*, vol. 21, no. 3647, pp. 1–23, 2021.

[3] U. Shruthi, K. S. Narmadha, E. Meghana, D. N. Meghana, K. P. Lakana, and M. P. Bhuvan. "Apple varieties classification using light weight CNN model," in *Proc. International Conf. On Circuits, Control, Communication and Computing (I4C)*, Bengaluru, India, 2022, pp. 68–72.

[4] U. Shruthi, V. Nagaveni, and B. K. Raghavendra, "A review on machine learning classification techniques for plant disease detection," in *Proc. International Conf. On Advanced Computing & Communication Systems (ICACCS)*, Coimbatore, India, 2019, pp. 281–284.

[5] R. A. Krishnaswamy, R. Purushothaman, and A. Ramesh, "Tomato crop disease classification using pre-trained deep learning algorithm," *Procedia Computer Science*, vol. 133, pp. 1040–1047, 2018.

[6] P. F. Konstantinos, "Deep learning models for plant disease detection and diagnosis," C*omputers and Electronics in Agriculture*, vol. 145, pp. 311–318, 2018.

[7] P. M. Sharada, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," *Frontiers in Plant Science*, vol. 7, 1419, pp. 1–10, 2016.

[8] M. H. Saleem, J. Potgieter, and K. M. Arif, "Plant disease classification: A comparative evaluation of convolutional neural networks and deep learning optimizers," *Plants*, vol. 9, 1319, pp 1–16, 2020.

[9] S. Z. M. Zaki, M. A. Zulkifley, M. M. Stofa, N. A. M. Kamari, and N. A. Mohamed, "Classification of tomato leaf diseases using MobileNet V2," *IAES International Journal of Artificial Intelligence*, vol. 9, no. 29, pp. 290–296, 2020.

[10] J. Chen, J. X. Chen, D. F. Zhang, Y. D. Sun, and Y. A. Nanehkaran, "Using deep transfer learning for image-based plant disease identification," C*omputers and Electronics in Agriculture*, vol. 173, 105393, 2020.

[11] A. Khan, U. Nawaz, A. Ulhaq, and R. W. Robinson, "Real-time plant health assessment via implementing cloud-based scalable transfer learning on AWS deeplens," *Plos One*, vol. 15, no. 12, pp. 1–23, 2020.

[12] M. Agarwal, S. K. Gupta, and K. K. Biswas, "Development of efficient CNN model for tomato crop disease identification," *Sustainable Computing: Informatics and Systems*, vol. 28, 100407, 2020.

[13] A. A. Abdelmoamen and G. H. Reddy, "A mobile-based system for detecting plant leaf diseases using deep learning," *AgriEngineering*, vol. 3, no. 3, pp. 478–493, 2021.

[14] M. A. Patil and M. Manohar, "Plant leaf disease classification using optimal tuned hybrid LSTM-CNN model," *SN Computer Science*, vol. 4, no. 6, 710, 2023.

[15] P. Arunangshu and V. Kumar, "AgriDet: Plant leaf disease severity classification using agriculture detection framework," *Engineering Applications of Artificial Intelligence*, vol. 119, 105754, 2023.

[16] A. Shelke and N. Mehendale, "A CNN-Based android Application for plant leaf classification at remote locations," *Neural Computing and Applications*, vol. 35, no. 3, pp. 2601–2607, 2023.

[17] S. Ghosh, A. Singh, and S. Kumar, "PB3C-CNN: An integrated PB3C and CNN based approach for plant leaf classification," *Inteligencia Artificial*, vol. 26, no. 72, pp. 15–29, 2023.

[18] J. A. Pandian and G. Geetharamani, "Data for: Identification of plant leaf diseases using a 9-layer deep convolutional neural network," *Mendeley Data*, vol. 1, 2019.
[19] X. Li and L. Rai, "Apple leaf disease identification and classification using ResNet models," in *Proc. 2020 IEEE 3rd International Conference on Electronic Information and Communication Technology (ICEICT)*, 2020, pp. 738–742.
[20] S. Vallabhajosyula, V. Sistla, and V. K. K. Kolli, "Transfer learning-based deep ensemble neural network for plant leaf disease detection," *Journal of Plant Diseases and Protection*, vol. 129, no. 3, pp. 545–558, 2022.
[21] V. Suryawanshi, S. Adivarekar, K. Bajaj, and R. Badami, "Comparative study of regularization techniques for VGG16, VGG19 and ResNet-50 for plant disease detection," in *Proc. International Conference on Communication and Computational Technologies*, 2023, pp. 771–781.
[22] X. Zhang, Y. Sun, Y. Wang, Z. Li, N. Li, and J. Su, "A novel effective and efficient capsule network via bottleneck residual block and automated gradual pruning," *Computers and Electrical Engineering*, vol. 80, 106481, 2019.
[23] G. A. Howard, M. L. Zhu, B. Chen, D. Kalenichenko, W. J. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint, arXiv:1704.04861, 2017.
[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, 2012.
[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint, arXiv:1409.1556, 2014.
[26] C. Szegedy, V. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in *Proc. AAAI Conf. on Artificial Intelligence*, 2017.
[27] K. M. He, X. G. Zhang, S. Q. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. International Conf. on Computer Vision* and *Pattern Recognition*, Las Vegas, USA, 2016, pp. 770–778.
[28] M. X. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proc. International Conf. on Machine Learning*, 2019, vol. 97, pp. 6105–6114.
[29] U. Shruthi, V. Nagaveni, C. S. Arvind, and G. L. Sunil, "Tomato plant disease classification using deep learning architectures: A review," in *Proc. International Conf. on Advances in Computer Engineering and Communication Systems*, *Algorithms for Intelligent Systems*, Singapore, 2022, pp. 153–169.
[30] M. Grandini, E. Bagli, and G. Visani, "Metrics for multi-class classification: An overview," arXiv preprint, arXiv: 2008.05756, 2020.
[31] M. Hossin and M. N. Sulaiman, "A review on evaluation metrics for data classification evaluations," *International Journal of Data Mining and Knowledge Management Process*, vol. 5, no. 2, pp. 1–11, 2015.
[32] U. Shruthi and V. Nagaveni, "TomSevNet: A hybrid CNN model for accurate tomato disease identification with severity level assessment," *Neural Computing and Applications*, vol. 36, no. 10, pp. 5165–5181, 2024.
[33] H. Durmuş, E. O. Güneş, and M. Kırcı, "Disease detection on the leaves of the tomato plants by using deep learning," in *Proc. International Conf. on Agro-geoinformatics*, Fairfax, USA, 2017.
[34] A. Mohit, A. Singh, S. Arjaria, A. Sinha, and S. Gupta, "ToLeD: tomato leaf disease detection using convolution neural network," *Procedia Computer Science*, vol. 167, pp. 293–301, 2020.
[35] M. A. Alzahrani and F. W. Alsaade, "Transform and deep learning algorithms for the early detection and recognition of tomato leaf disease," *Agronomy*, vol. 13, no. 5, 1184, 2023.
[36] E. Suryawati, R. R. Sustika, R. S. Yuwana, A. Subekti, and H. F. Pardede, "Deep structured convolutional neural network for tomato diseases detection," in *Proc. International Conf. on Advanced Computer Science and Information Systems*, Yogyakarta, Indonesia, 2018, pp. 385–390.
[37] S. Gnanavel, G. W. Sathianesan, V. S. Murugan, A. J. Reddy, P. Jayagopal, and M. Elsisi, "Detection and classification of tomato crop disease using convolutional neural network," *Electronics*, vol. 11, no. 21, 3618, 2022.
[38] B. Chu, V. Madhavan, B. Oscar, J. Hoffman, and T. Darrell, "Best practices for fine-tuning visual classifiers to new domains," in *Proc. Computer Vision–ECCV*, 2016, vol. 14, pp. 435–442.
[39] S. U. Hussain and B. Triggs, "Visual recognition using local quantized patterns," in *Proc. European Conference on Computer Vision*, 2012, pp. 716–729.
[40] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," arXiv preprint, arXiv:1810.05270, 2018.
[41] J. P. Gou, B. S. Yu, S. J. Maybank, and D. C. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021.
[42] D. Lior and D. Horn, "The weight-shape decomposition of density estimates: A framework for clustering and image analysis algorithms," *Pattern Recognition*, vol. 81, pp. 190–199, 2018.
[43] V. V. Zunin, "Intel openvino toolkit for computer vision: Object detection and semantic segmentation," in *Proc. 2021 International Russian Automation Conference (RusAutoCon)*, IEEE, 2021, pp. 847–851.