Computers and Electronics in Agriculture 187 (2021) 106279

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

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

Original papers

Tomato plant disease detection using transfer learning with C-GAN synthetic images

Amreen Abbas, Sweta Jain, Mahesh Gour *, Swetha Vankudothu

Maulana Azad National Institute of Technology, Bhopal, MP 462003, India

ARTICLE INFO

Keywords:
Deep learning
Tomato plant disease detection
Conditional Generative Adversarial Network
Data augmentation
Pre-trained DesnseNet121 network
Synthetic Images

ABSTRACT

Plant diseases and pernicious insects are a considerable threat in the agriculture sector. Therefore, early detection and diagnosis of these diseases are essential. The ongoing development of profound deep learning methods has greatly helped in the detection of plant diseases, granting a vigorous tool with exceptionally precise outcomes but the accuracy of deep learning models depends on the volume and the quality of labeled data for training. In this paper, we have proposed a deep learning-based method for tomato disease detection that utilizes the Conditional Generative Adversarial Network (C-GAN) to generate synthetic images of tomato plant leaves. Thereafter, a DenseNet121 model is trained on synthetic and real images using transfer learning to classify the tomato leaves images into ten categories of diseases. The proposed model has been trained and tested extensively on publicly available PlantVillage dataset. The proposed method achieved an accuracy of 99.51%, 98.65%, and 97.11% for tomato leaf image classification into 5 classes, 7 classes, and 10 classes, respectively. The proposed approach shows its superiority over the existing methodologies.

1. Introduction

Agriculture has been the primary source of income for the majority of people in India. The extensive commercialization of agriculture has greatly affected our environment. One of the biggest concerns in the field of agriculture is the detection of plant diseases. Early detection of diseases helps in preventing the widespread of disease among other plants, thus preventing substantial economic losses. The impact of plant illness ranges from mild manifestation to the destruction of complete plantations that severely affect the agricultural economy (Savary et al., 2012).

Deep learning is a concept that combines data analysis with image processing resulting in very accurate results. Nowadays, deep learning is being extensively used in various fields like object detection (Redmon et al., 2016), signal and voice recognition (Abdel-Hamid et al., 2014), biomedical image classification (Gour et al., 2020; Gour and Jain, 2020) and segmentation (Gour et al., 2019). Deep learning is also widely being used in the field of agriculture, for the purpose of plant disease detection and classification. Among the deep learning techniques, Convolutional Neural Network (CNN) is considered as the best method. Various architectures of CNN like AlexNet (Krizhevsky et al., 2012), GoogLeNet (Al-Qizwini et al., 2017), etc. are being used for the detection and classification of plant diseases.

The performance of CNN models heavily depends on the available dataset for training. Generally, these models show improved results and high generalizability on the sufficiently large dataset. The datasets that are presently available for tomato plant disease typically do not contain enough images in diverse conditions, which is a necessity for making high accuracy models. Provided the dataset being tiny, the model may overfit and perform badly on real environment test data. Hence, various data augmentation techniques like affine transformation and perspective transformation are being used to enhance the dataset (Gour et al., 2020). When the training images are insufficient, and image transformation techniques are unable to change the outcomes, notably, Generative Adversarial Networks (GANs) are used for producing counterfeit images.

In this research work, we have developed a deep learning based framework to recognize the tomato plant diseases by investigating tomato leaf images. This method would help farmers in classification of diseases affecting tomato cultivation by simply taking an image of diseased leaves, instead of going after costly expert analysis. In the proposed method, we generated synthetic images using Conditional Generative Adversarial Network (C-GAN) (Mirza and Osindero, 2014) and augmented these images to the dataset for training purpose.

* Corresponding author.
E-mail address: maheshgour0704@gmail.com (M. Gour).

https://doi.org/10.1016/j.compag.2021.106279
Received 22 September 2020; Received in revised form 10 April 2021; Accepted 13 June 2021
Available online 29 June 2021

Thereafter a DenseNet model (Huang et al., 2017) was trained on tomato plant images and tomato synthetic images (generated by the C-GAN model) for the detection of tomato disease from the leaf images.

The paper is organized in the following manner: Section 2 contains the related work about existing methods, Section 3 presents the methodology , which includes the C-GAN model description and the description of the DenseNet model, Section 4 consists of the summary of the dataset followed by the experimental setup and results along with the performance comparison with the existing methos and pre-trained CNN models. Conclusions are presented in Section 5.

2. Related work

In agriculture, plant diseases cause havoc resulting in economic loss. Various researchers have put forth different techniques to put a check on these infections. Over the recent couple of years, some researchers have been using deep learning techniques like convolutional neural networks for disease detection and classification. Akila and Deepan (2018) have proposed a deep learning model, in which they have used Faster Region-based Convolutional Neural Network (R-CNN) (Ren et al., 2015), Region-based Fully Convolutional Network (R-FCN) (Dai et al., 2016), and Single Shot Detector (SSD) (Liu et al., 2016) for plant disease detection. Their dataset consists of images of fruit plants, vegetable crops, cereal crops, and cash crops. They have applied several image augmentation techniques like image rotations, brightness adjustment, perspective transformation, and affine transformation.

Adhikari et al. (2018) have proposed a model to detect plant disease automatically, especially for the tomato plant using image processing. The dataset consisted of images of three types of tomato plant diseases, including Gray spot, Late Blight, and Bacterial Canker. Some of the images were extracted from the Internet, and some were captured from the firm using camera devices. The designed CNN model consisted of 24 convolutional layers and two fully connected layers. The structure is designed based on the standard YOLO (You Only Look Once) model (Redmon et al., 2016). The overall accuracy of the model was 89% on the PlantVillage dataset and 76% when they used their own dataset. Fuentes et al. (2017) have proposed a system that is capable of distinguishing nine sorts of maladies and bugs in tomato plants. It detects disease class as well as the location of the disease. Images of plants in fields were taken using camera devices. Three detectors, namely R-FCN, Faster R-CNN, and SSD were combined with feature extractors like VGG-16 and ResNet50 Network. When Faster R-CNN was used with VGG-16 (Simonyan and Zisserman, 2014), the model had a mAP (mean Average Precision) of 83%, and when it was used with ResNet-50, the model achieved a mAP of 75.37%. Contrary to it, when SSD was used with ResNet-50, the model had a mAP of 82.53%, and when R-FCN was used with ResNet-50, the model achieved a mAP of 85.98%.

Ashqar and Abu-Naser (2018) have developed a CNN model to identify five diseases of tomato plants namely bacterial spot, early blight, septoria leaf spot, leaf mold and yellow leaf curl virus. Their model consisted of four convolutional layers. ReLU (Rectified Linear Unit) activation function was used, and each convolutional layer was followed by a max-pooling layer. They have used 9,000 images from the PlantVillage dataset. Their model achieved an accuracy of 99.84% when it was trained on full-color images and an accuracy of 95.54% when it was trained on gray-scale images. Zhang et al. (2018) have proposed a CNN model to distinguish 8 types of tomato leaf illness by utilizing transfer learning. The pre-trained models used in the study were AlexNet, GoogLeNet, and ResNet. The dataset was taken from an open access repository (PlantVillage) containing plant images. The model achieved the highest accuracy of 97.28% when the ResNet network was used with SGD (Stochastic Gradient Descent).

Similarly, Durmuş et al. (2017) have used AlexNet and SqueezeNet to classify the tomato plant images into 10 classes (9 diseases and 1 healthy). On the PlantVillage dataset, the AlexNet model achieved an accuracy of 95.65%, while the SqueezeNet model achieved an accuracy of 94.3%. Karthik et al. (2020) have used two deep learning architectures on PlantVillage dataset to detect 3 diseases in tomato plants, namely, early blight, late blight, and leaf mold. In the first architecture, feed-forward CNN is used along with residual learning and in the second architecture, they have used CNN along with the attention mechanism and residual learning. The attention-based residual CNN architecture presented a highest accuracy of 98%. Agarwal et al. (2020) have developed a CNN model, consisting of 3 convolutional, 3 max-pooling, and 2 fully connected layers to detect 10 classes (9 diseases and 1 healthy) in tomato plants that achieved an overall accuracy of 91.2% on the PlantVillage dataset. Elhassouny and Smarandache (2019) have developed a smart mobile application to classify 9 diseases of tomato plant using MobileNet (Howard et al., 2017). The application used 7,176 tomato leaf images from PlantVillage dataset for the disease classification. They have achieved an accuracy of 90.3%. Widiyanto et al. (2019) developed a CNN model to identify 4 diseases in tomato plants namely Late blight, Septoria leaf spot, Mosaic virus and Yellowleaf curl virus along with healthy leaves. The author trained the model on 1,000 images per class obtained from PlantVillage dataset. The overall classification accuracy achieved for 5 classes was 96.6%.

Table 1
Summary of the literature of studies for plant diseases detection.

Paper	Methods	Dataset	Plants/ classes	Accuracy
Adhikari et al. (2018)	YOLO	internet images and PlantVillage	3 classes of diseases in tomato plants	76%
Fuentes et al. (2017)	SSD, R-FCN, Faster R-CNN with VGG and Residual Network	5,000 field images	9 classes of diseases and pests in tomato	85%
Ashqar and Abu-Naser (2018)	ANN	Tomato PlantVillage (9,000 images)	5 classes of tomato leaf diseases	99.84%
Zhang et al. (2018)	AlexNet, GoogLeNet and ResNet	Tomato PlantVillage (5,550 images)	8 classes of tomato leaf diseases	97.28%
Karthik et al. (2020)	Residual learning and Attention mechanism	Tomato PlantVillage (95,999 images)	3 classes of tomato plant diseases	98%
Agarwal et al. (2020)	CNN Network	Tomato PlantVillage (17,500 images)	10 classes of tomato plant diseases	91.20%
Durmuş et al. (2017)	AlexNet and SqueezeNet	Tomato PlantVillage	10 classes of tomato plant diseases	95.65%
Elhassouny and Smarandache (2019)	MobileNet	Tomato PlantVillage	10 classes of tomato plant diseases	90.3%
Widiyanto et al. (2019)	CNN	Tomato PlantVillage	5 classes of tomato plant diseases	96.6%

Table 1 represents the summary of the different deep learning approaches that have been used in the literature for the detection and classification of tomato plant diseases.

3. Methodology

In this section, we will discuss the proposed method. The Block diagram of the proposed method is shown in Fig. 1. The proposed approach can be divided into two parts; in the first part, synthetic images have been generated using C-GAN for data augmentation. In the second part, a pre-trained DenseNet121 model has been fine-tuned on

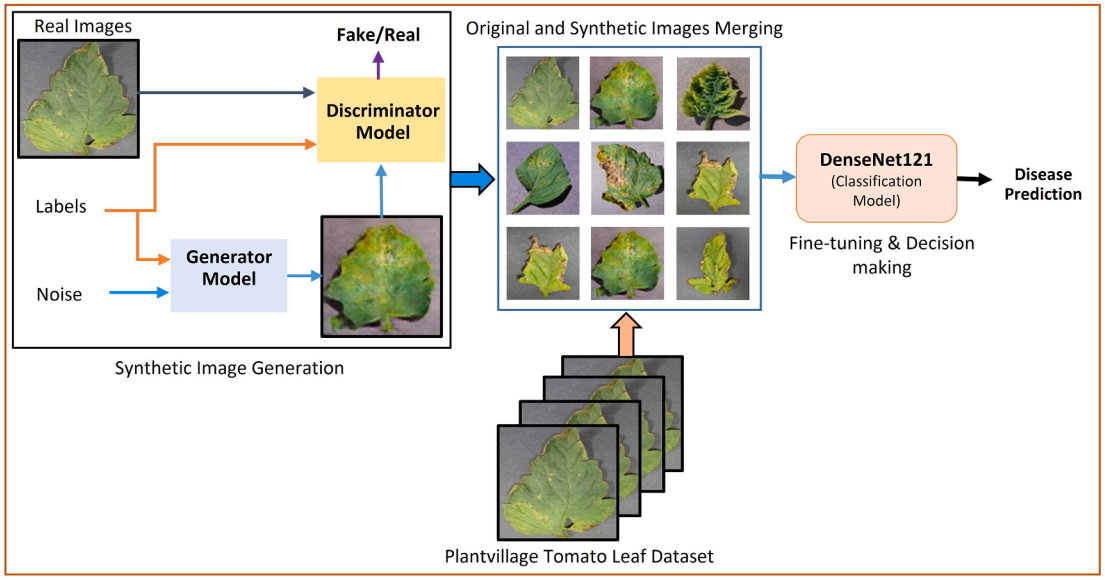


Fig. 1. Block diagram of the proposed method.

Table 2
Summary of C-GAN Discriminator model.

Layer Name	Type	Parameter/Filter size
Input	Input	[128,128,3]
Embedding	Embedding	[0,3,25]
Dense	Dense	sigmoid activation
Reshape	Reshape	[64,64,3]
Concatenate	Concatenate	[64,64,6]
Leaky ReLU	Leaky ReLU	max(0,x)
Conv2D	Conv2D	[32 32], [16 16], [8,8], [4 4], stride=[2 2], padding same
Flatten	Flatten	[0,2048]
Dropout	Dropout	[0,2048]

Table 3
Summary of C-GAN Generator model.

Layer Name	Type	Parameter/Filter size
Input	Input	[4,4,3]
Dense	Dense	sigmoid activation
Embedding	Embedding	[0,3,25]
Leaky ReLU	Leaky ReLU	max(0,x)
Reshape	Reshape	[4,4,256]
Concatenate	Concatenate	[4,4,259]
Conv2D	Conv2D	[32 32], [16 16], [8 8], [4 4], stride=[2 2], padding same

tomato leaf images for disease classification. The detailed description is given in the following subsections:

3.1. C-GAN model as synthetic image generator

In order to prevent the network from over-fitting, Conditional Generative Adversarial Network (C-GAN) (Mirza and Osindero, 2014) can be used as data augmentation technique to enhance the size of dataset. In GAN, traditional convolutional layers are applied to form an image matrix from random noise. GAN consists of a discriminator model and a generator model. The work of generator is to produce fake images and the work of discriminator is to distinguish between fake and real images. The generator and discriminator train simultaneously and try to outdo each other. The discriminator makes sure that the images generated by the generator are as close to the real images as possible.

The C-GAN model (Mirza and Osindero, 2014) consists of two adversarial networks: the generator model and discriminator model. The C-GAN discriminator model consists of an input layer, an embedding layer, and a dense layer followed by an input layer, reshape layer, and concatenate layer, which is in turn followed by four convolutional layers. Each convolutional layer is followed by a Leaky ReLU layer. The last Leaky ReLU layer is followed by a flatten layer, a dropout layer, and a dense layer. The model consists of a total of 771,454 trainable parameters. The network layers and parameters of discriminator model are represented in Table 2.

The C-GAN generator model consists of input layers and a dense layer followed by an embedding layer and a Leaky ReLU layer which is in turn followed by dense, reshape and concatenate layers. The concatenate layer is followed by four convolutional layers where each convolutional layer is followed by a Leaky ReLU layer. The last Leaky ReLU layer is followed by a final convolutional layer. The model consists of a total of 1,735,904 trainable parameters. The network layers and parameters of discriminator model are represented in Table 3. The C-GAN generator model (G) takes latent points and random noise as input and generates synthetic images while the work of discriminator model (D) is to differentiate between real images and synthetic images generated by the G model. Suppose I represents data and L represents the additional class label information where I and L are fed as input to the discriminative function and the input noise distribution in G model is given by $q_z(z)$. The D model tries to maximize the probability of assigning the labels correctly to the original images as well as synthetic images generated by the generator model represented by $logD(I|L)$ while the G model tries to minimize the generator loss represented by $log(1-D(G(z|L))$. The objective function of C-GAN is similar to a two player minimax game and represented as follows:

$$\min_G \max_D V(D,G) = E_{I\sim p_{\text{data}}(I)}[\log D(I|L)] + E_{z\sim p_z(z)}[\log(1-D(G(z|L))] \quad (1)$$

In the proposed approach, we have used C-GAN for generating synthetic tomato leaf images of various diseases. For generating synthetic images from the C-GAN model, we first trained it on the tomato images. Let given tomato disease leaf dataset $\mathscr{D} = \{(\mathbf{I}^{(n)}, \mathbf{L}^{(n)})\}_{n=1}^{N}$, where $I^{(n)}$ represents a given image and $L^{(n)} \in \{0, 1, \ldots, 9\}$, represents its corresponding label. To train C-GAN, a real tomato image $I^{(n)}$ with corresponding label $L^{(n)}$ is given as input to the discriminator model; simultaneously, a noise and a label $L^{(n)}$ is given to the generator model. Then the generator model generates tomato fake image I_f and now generated fake image I_f is also given to the discriminator model.

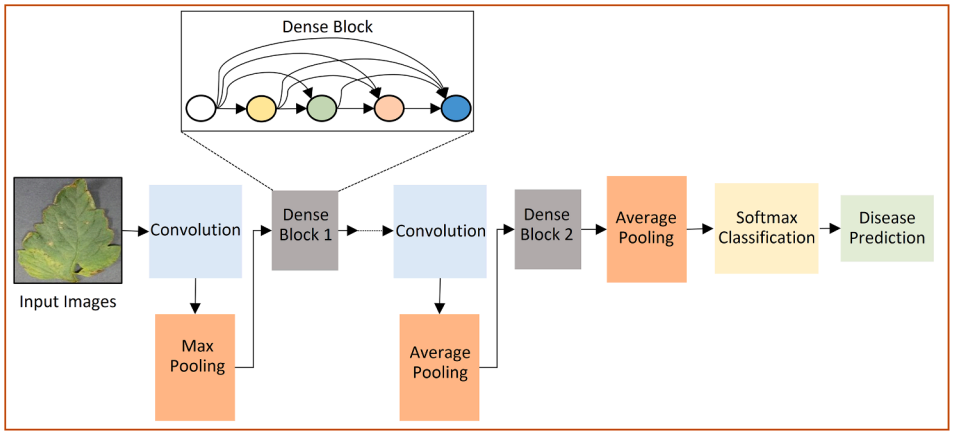


Fig. 2. DenseNet Architecture.

Table 4
Summary of DenseNet121 model.

Layer Name	Output size	Parameter/Filter size
Convolution	112 × 112	7 × 7 conv, stride 2
Pooling	56 × 56	3 × 3 max pool, stride 2
Dense block (1)	56 × 56	[1 × 1 conv, 3 × 3 conv × 6]
Transition layer (1)	56 × 56 28 × 28	1 × 1 conv 2 × 2 average pool, stride 2
Dense block (2)	28 × 28	[1 × 1 conv, 3 × 3 conv × 12]
Transition layer (2)	28 × 28 14 × 14	1 × 1 conv 2 × 2 average pool, stride 2
Dense block (3)	14 × 14	[1 × 1 conv, 3 × 3 conv × 24]
Transition layer (3)	14 × 14 7 × 7	1 × 1 conv 2 × 2 average pool, stride 2
Dense block (4)	7 × 7	[1 × 1 conv, 3 × 3 conv × 16]
Additional convolution layer	7 × 7	[3 × 3 conv, 3 × 3 conv]
Classification layer	1 × 1	7 × 7 global average pool 10D fully connected, softmax

Thereafter, the discriminator tries to distinguish between fake and real images. This way, C-GAN has been trained on the tomato leaf images. After successful completion of the training of C-GAN, we have obtained synthetic images of tomato leaves corresponding to each disease category.

3.2. Tomato plant disease detection with DenseNet model

The pre-trained Dense Convolutional Network (DenseNet) has been trained on CIFAR-10, CIFAR-100, SVHN, and ImageNet datasets (Huang et al., 2017). It has achieved promising results for object recognition tasks. In this study, we used the DenseNet121 model using transfer learning for tomato plant disease detection from tomato leaf images. The detailed description of DenseNet architecture is given as follows:

DenseNet Architecture: DenseNet is a deep architecture in which each layer is connected to every other layer in a feed-forward manner. DenseNet differs from ResNet (He et al., 2016) in the manner of a combination of features that take place at a layer before they are passed on to the next layers. In ResNet, the features are combined through summation while in DenseNet, the features are combined by concatenating them. Other convolutional networks have l connections for l layers whereas DenseNet has $l(l + 1)/2$ connections for l layers. In DenseNet, inputs of each layer are the feature-maps of previous layers, and subsequently, the feature-maps of a layer are used as inputs in the succeeding layers (Huang et al., 2017).

The architecture of DenseNet121 model is represented in Fig. 2 and the description of different layers of the model is represented in Table 4. It consists of 4 dense blocks that take 224 × 224 pixels image as input. The first convolution layer consists of 2000 convolutions of size 7 × 7 with stride 2, which is followed by a 3 × 3 max pooling layer with stride 2. The pooling layer is followed by three dense blocks with each dense block being followed by a transition layer. The fourth dense block is followed by a classification layer. The dense blocks consist of batch normalization, ReLU and convolutional layers. The transition layer consists of 1 × 1 convolution layer followed by 2 × 2 average pooling layer with stride 2.

DenseNet Fine-tuning: In order to fine-tune the pre-trained DenseNet model on the tomato leaf images, the top Fully-connected layer, and Softmax layer have been removed from the network, and two new Convolutional layers with ReLU activation function, an Average pooling layer, a Fully-connected layer, and a Softmax layer have been added in the network. The weights of the first 410 layers have been initialized with pre-trained imagenet weights, and the remaining layers' weights have been learned during the training. The model has been trained for 100 epochs with a learning rate of 0.0001 and a batch size of 32, with the weights being updated using Adam's optimizer (Kingma and Ba, 2014). We have experimentally found these are the best suitable values of hyperparameter for our application. For the training of the DenseNet model, generated synthetic images of tomato leaves and images from the tomato PlantVillage dataset (Hughes et al., 2015) have been combinedly used.

Table 5
Class-wise image distribution of the tomato PlantVillage dataset.

Category	Number of images
Tomato Yellow Leaf Curl Virus (YLCV)	3209
Tomato Bacterial Spot (Bctsp)	2127
Tomato Late Blight (TLB)	1909
Tomato Septoria leaf spot (SptL)	1771
Tomato Two Spotted Spider Mite (SpdM)	1676
Tomato Target Spot (TTS)	1404
Tomato Early Blight (TEB)	1000
Tomato Leaf Mold (LMld)	952
Tomato Mosaic Virus (MscV)	373
Tomato healthy (Hlth)	1591
Total	16012

4. Implementation and result

This section presents the implementation details of the proposed method and the results analysis. Section 4.1 contains description of the dataset, followed by experiment setup represented in Section 4.2. The results of the proposed method and performance comparison with existing methodologies is presented in Section 4.3.

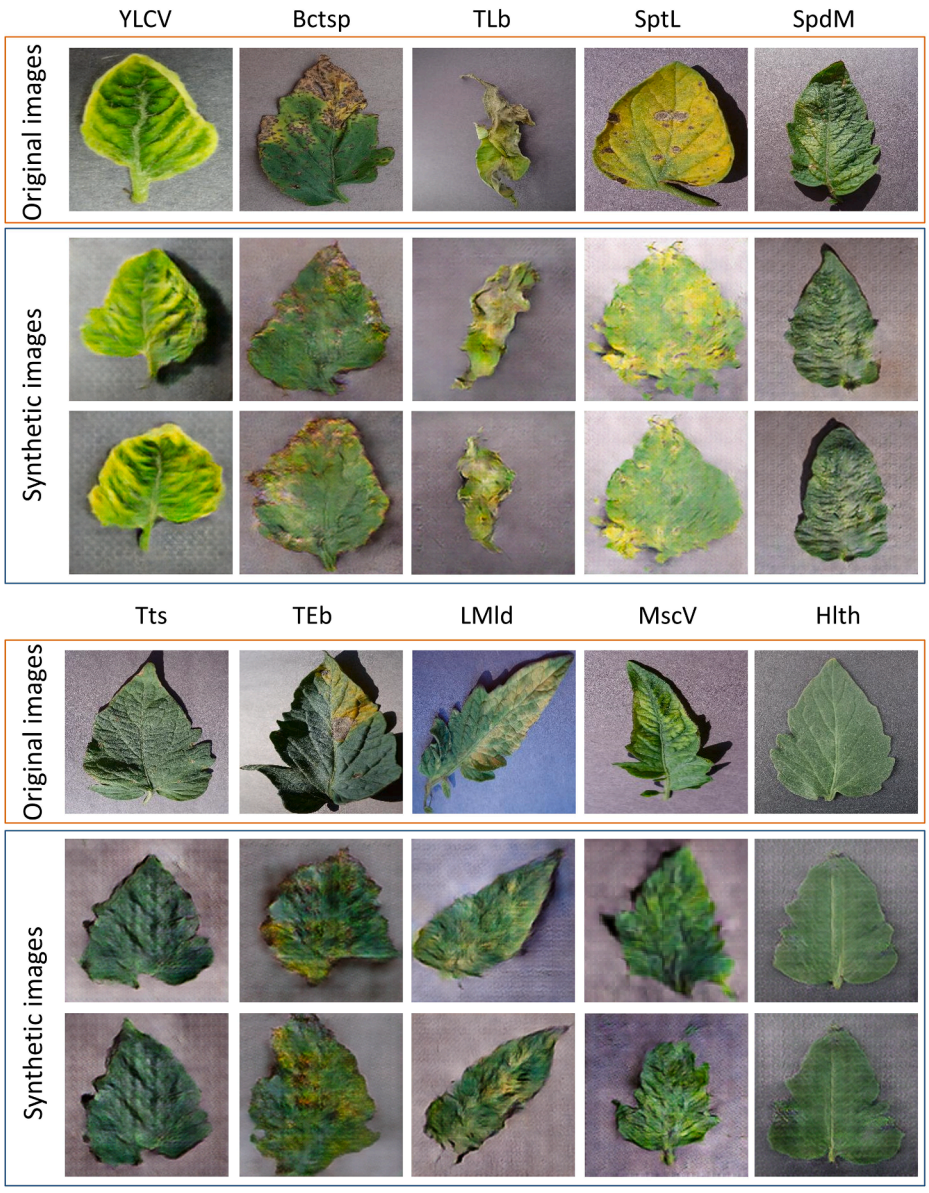


Fig. 3. Tomato original leaf images and synthetic leaf images generated by the C-GAN model: here row 1 represents the original tomato leaf images of five classes (Yellow Leaf Curl Virus, Bacterial Spot, Late Blight, Septoria leaf spot and Two Spotted Spider Mite), row 2 and row 3 represent synthetic images of corresponding classes; row 4 represents the original tomato leaf images of next five classes (Target Spot, Early Blight, Leaf Mold, Mosaic Virus and Healthy), row 5 and row 6 represent the synthetic images of respective classes.

4.1. Dataset

In order to evaluate the performance of the proposed approach, a publicly available tomato PlantVillage dataset (Hughes et al., 2015) has been used in the experiments. PlantVillage dataset consists of 16,012 images of tomato plant leaves of ten classes. Out of ten classes, nine categories are of tomato plant leaf diseases, and one category is for healthy leaves. The classes of the PlantVillage dataset and class-wise image distribution is represented in Table 5. We have used abbreviations of class names, which are also shown in Table 5. All the images of the dataset have been resized to 224 × 224 for faster computations. We have divided the dataset into training set, validation set and test set in the ratio of 60:10:30 with the consideration of no overlapping between the three sets. The training set and the validation set have been used during the network training, and the test set has been used for performance assessment of the model.

4.2. Experiment setup and evaluation metrics

In this study, two sets of experiments have been performed. In the first set of experiments, the C-GAN model has been trained on the training set for 300 epochs to generate synthetic leaf images of tomato for each of the ten categories. The weights of the generator and discriminator model were updated after each epoch to generate synthetic images as close to real images as possible. At the end of the training of the network, we have generated 4,000 synthetic images of tomato leaves from the C-GAN model. In the second set of experiments, the pre-trained DenseNet model has been trained on the original training set as well as on the combination of the training set (from the PlantVillage) and generated synthetic tomato leaf images.

To evaluate the performance of the proposed approach, we have used various performance metrics like accuracy, precision, recall, F1-score, and confusion matrix.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

Table 6
PSNR values between original images and between original and GAN images for 10 classes.

Class	PSNR b/w original images (dB)	PSNR b/w original and C-GAN images (dB)
Yellow leaf curl virus	28.047	28.117
Septoria leaf spot	27.894	28.320
Two spotted spider mite	28.013	28.505
Bacterial spot	27.987	28.304
Leaf mold	27.901	27.924
Mosaic virus	27.894	27.920
Target spot	27.919	28.026
Early blight	28.294	27.895
Late blight	27.861	27.749
Healthy	27.897	28.741
Mean	**27.970**	**28.150**

$$F1-score = \frac{2*TP}{2*TP+FP+FN} \tag{5}$$

where TP = True Positive, TN = True Negative, FP = False Positive and FN = False Negative.

In order to evaluate the quality of the synthetic images generated by C-GAN, we have used image quality measures like Peak Signal-to-Noise Ratio (PSNR) and Inception Score (IS).

$$PSNR = 20.\log_{10}(MAX_I) - 10.\log_{10}(MSE) \tag{6}$$

$$MSE = \frac{1}{mn}\sum_{0}^{m-1}\sum_{0}^{n-1}[I(i,j) - K(i,j)]^2 \tag{7}$$

where MAX_I is the maximum possible pixel value of the image I and MSE is the mean squared error.

Inception score uses the pre-trained Inception-v3 (Szegedy et al., 2016) model to classify the generated images. The model is used to classify a large number of generated images by predicting the probability of the image belonging to each class. These predictions are summed up into the inception score. The Kullback–Leibler (KL) divergence is calculated for each image as the conditional probability multiplied by the log of the conditional probability minus the log of the marginal probability. The KL divergence is then summed over all images and averaged over all classes and the exponent of the result is calculated to give the final inception score.

$$IS = exp(E_{y\sim p_g} D_{KL}(p(z|y)\|p(z))) \tag{8}$$

where $y \sim p_g$ denotes that y is an image sampled from g, $p(z|y)$ is conditional class distribution, p(z) is marginal class distribution and $D_{KL}(p(z|y)\|p(z))$ is the KL divergence between the two distributions.

4.3. Results and discussion

This section presents performance evaluation of C-GAN model and classification model.

4.3.1. Performance of C-GAN

The qualitative performance of C-GAN can be visually examined in Fig. 3, where it shows the original tomato leaf images from the dataset as well as the synthetic tomato leaf images generated by the C-GAN model. It can be seen in Fig. 3 that the generated synthetic images are looking closely similar to the real images. The PSNR values between original images and between original and synthetic images is represented in Table 6. The comparable values of PSNR show that the quality of images generated by C-GAN is very close to the quality of real images. The mean and standard deviation of inception scores have been calculated on original images as well as on the C-GAN synthetic images, which are represented in Table 7. The inception score of synthetic images is also very similar to the score of original images.

Table 7
Inception score of original images and C-GAN images for 10 classes.

Image type	Mean Inception score
Original images	2.558 ± 0.189
C-GAN images	2.835 ± 0.228

Table 8
Accuracy, Precision, Recall, and F1-score for the model with and without augmentation.

Model	Accuracy	Precision	Recall	F1-score
5 classes				
DenseNet121	98.16%	0.83	0.97	0.98
DenseNet121 + Synthetic images	99.51%	0.99	0.99	0.99
7 classes				
DenseNet121	95.08%	0.94	0.94	0.94
DenseNet121 + Synthetic images	98.65%	0.98	0.99	0.98
10 classes				
DenseNet121	94.34%	0.95	0.92	0.93
DenseNet121 + Synthetic images	97.11%	0.97	0.97	0.97

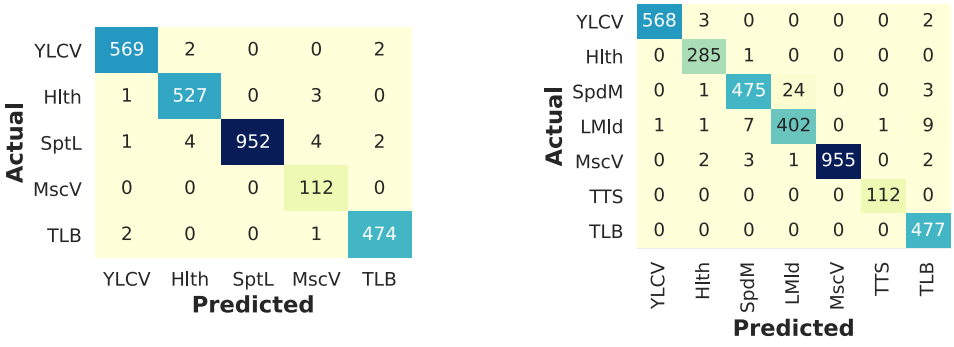
Table 9
Precision, Recall and F1-score for different disease classes of tomato plant.

Class	Precision	Recall	F1-score
Yellow leaf curl virus	0.97	0.97	0.97
Septoria leaf spot	0.97	0.98	0.97
Two spotted spider mite	1.00	0.97	0.98
Bacterial spot	1.00	0.97	0.98
Leaf mold	0.96	0.97	0.96
Mosaic virus	0.89	0.98	0.93
Target spot	1.00	0.99	0.99
Early blight	0.94	0.99	0.97
Late blight	0.99	1.00	0.99
Healthy	0.95	0.87	0.91
Mean	**0.97**	**0.97**	**0.97**

4.3.2. Performance of classification model

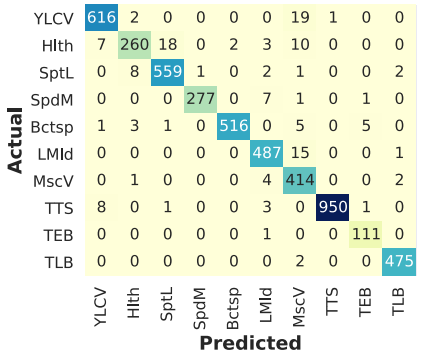
We have evaluated the performance of proposed model for 5-class classification (YLCV, SptL, MscV, TLB and Hlth classes), 7-class classification (YLCV, SpdM, LMld, MscV, TTS, TLB and Hlth classes) and 10-class classification (YLCV, SptL, SpdM, Bctsp, LMld, MscV, TTS, TEB, TLB and Hlth classes) tasks. Table 8 represents the classification performance of the proposed method on the PlantVillage dataset and augmented dataset (PlantVillage + Synthetic images dataset). The proposed method achieved a classification accuracy of 99.51%, 98.65%, and 97.11% for 5-class classification, 7-class, and 10-class classification tasks, respectively. It can be observed from Table 8, DenseNet121 model with synthetic images have shown performance improvement in accuracy, precision, recall, and F1-score for all classes, as compared to the original dataset. This improvement in classification performance clearly indicates that data augmentation using the C-GAN model has prevented the network from over-fitting and helped the network to be more generalized. The class-wise performance of the proposed approach for 10-class classification task on augmented dataset with synthetic images is represented in Table 9.

The confusion matrix and Receiver Operating Characteristic (ROC) Curves of the proposed method for 5-class, 7-class and 10-class classification are shown in Fig. 4 and Fig. 5, respectively. The proposed



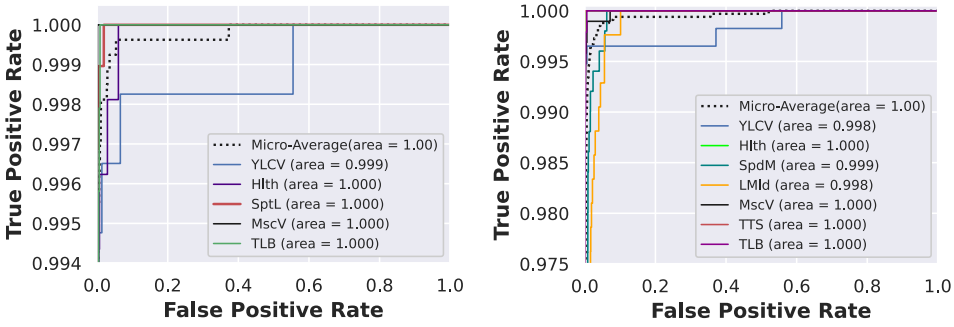
(a) 5 classes

(b) 7 classes



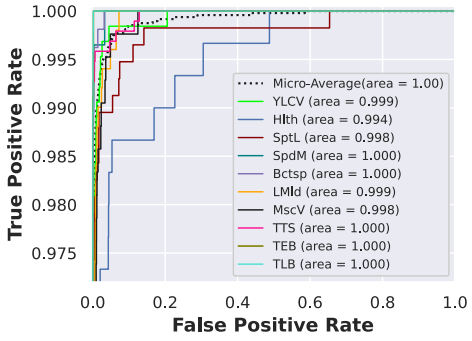
(c) 10 classes

Fig. 4. Confusion matrix of tomato plant disease classification using proposed method.



(a) 5 classes

(b) 7 classes



(c) 10 classes

Fig. 5. Receiver Operating Characteristic (ROC) curves of tomato plant disease classification using proposed method.

Table 10
Comparison of performances of different Pre-trained Networks.

Method(s)	PlantVillage	PlantVillage + Synthetic images
VGG19	89.60	90.90
ResNet50	76.90	80.00
Inception-V3	82.10	84.90
Xception	92.20	95.80
MobileNet	91.90	94.60
DenseNet169	93.03	95.15
DenseNet201	93.71	95.86
DenseNet121	**94.34**	**97.11**

Table 11
Comparison of result with existing models.

Classification task	Author (s)	Method(s)	Accuracy
5 Classes	Widiyanto et al. (2019)	CNN model	96.60%
	Proposed	**DenseNet, C-GAN**	**99.51%**
7 Classes	Rangarajan et al. (2018)	AlexNet, VGG16	97.49%
	Proposed	**DenseNet, C-GAN**	**98.65%**
10 Classes	Agarwal et al. (2020)	CNN network	91.20%
	Durmuş et al. (2017)	AlexNet and SqueezeNet	95.65%
	Elhassouny and Smarandache (2019)	MobileNet	90.30%
	Proposed	**DenseNet, C-GAN**	**97.11%**

method produces very little false positive and false negative, as observed from the confusion matrix.

4.3.3. Performance comparison

Performance comparison of the proposed DenseNet121 model with pre-trained networks, namely VGG19 (Krizhevsky et al., 2012), ResNet50, Inception-V3 (Szegedy et al., 2016), Xception (Chollet, 2017), and MobileNet (Howard et al., 2020) for 10-class classification, is shown in Table 10. All the models were trained first on the original PlantVillage dataset and then on augmented dataset consisting of synthetic images generated by the C-GAN model. The DenseNet121 model gave the best results among all the pre-trained models with an accuracy of 97.11% on the augmented dataset. The results of all the models clearly depict that an augmented dataset consisting of synthetic images generated by the C-GAN model has given better accuracy as compared to the original dataset.

In the past, several studies have been proposed for the tomato plant disease detection from tomato leaves images. We have compared our proposed method with some of the studies that are proposed on the tomato PlantVillage dataset. The performance comparison with existing methods is represented in Table 11. The studies presented in the table are mainly developed for three kinds of classification tasks, such as 5-class, 7-class and 10-class classification. For 5 class classification task, the performance of our approach is better than the performance in (Widiyanto et al., 2019). Likewise for 7 class classification task, the performance of our approach is comparable to the performance achieved in (Rangarajan et al., 2018). Similarly, for 10 class classification task, the proposed method has shown its superiority over the methods in (Agarwal et al., 2020; Durmuş et al., 2017; Elhassouny and Smarandache, 2019).

5. Conclusion

Deep learning-based approaches have shown promising results in plant disease detection. In this study, we have proposed yet another deep learning-based method for tomato plant disease detection from tomato leaf images. In the proposed approach, synthetic images have been generated using Conditional Generative Adversarial Networks for data augmentation purposes. Thereafter a pre-trained DenseNet121 model is fine-tuned on the original tomato leaf images and synthetic images. The proposed data augmentation technique improves network generalizability and prevents it from the over-fitting problem. The proposed model achieves an accuracy of 98.16%, 95.08%, 94.34%, on the original PlantVillage dataset for 5-class classification, 7-class, and 10-class classification tasks, respectively, and it achieves an accuracy of 99.51%, 98.65%, 97.11% with the original PlantVillage + synthetic images dataset for 5-class classification, 7-class, and 10-class classification tasks, respectively. In addition, our experiment results show the proposed method's superiority over the existing methods.

In future work, we wanted to extend this method for disease identification and classification to various parts of the plant, like fruits, stems, and branches. We also wanted to identify the different phases of the plant disease.

CRediT authorship contribution statement

Amreen Abbas: Methodology, Software, Writing - original draft. **Sweta Jain:** Supervision, Writing - review & editing, Investigation. **Mahesh Gour:** Conceptualization, Methodology, Software, Writing - review & editing, Visualization. **Swetha Vankudothu:** Software, Writing - original draft.

Declaration of Competing Interest

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

References

Abdel-Hamid, O., Mohamed, A.-R., Jiang, H., Deng, L., Penn, G., Yu, D., 2014. Convolutional neural networks for speech recognition. IEEE/ACM Trans. Audio, Speech, Language Process. 22 (10), 1533–1545.

Adhikari, S., Saban Kumar, K., Balkumari, L., Shrestha, B., Baiju, B., 2018. Tomato plant diseases detection system using image processing. In: 1st KEC Conference on Engineering and Technology, Lalitpur, vol. 1, pp. 81–86.

Agarwal, M., Singh, A., Arjaria, S., Sinha, A., Gupta, S., 2020. Toled: Tomato leaf disease detection using convolution neural network. Procedia Comput. Sci. 167, 293–301.

Akila, M., Deepan, P., 2018. Detection and classification of plant leaf diseases by using deep learning algorithm. Int. J. Eng. Res. Technol 6, 2–7.

Al-Qizwini, M., Barjasteh, I., Al-Qassab, H., Radha, H., 2017. Deep learning algorithm for autonomous driving using googlenet. In: 2017 IEEE Intelligent Vehicles Symposium (IV), IEEE, 2017, pp. 89–96.

Ashqar, B.A., Abu-Naser, S.S., 2018. Image-based tomato leaves diseases detection using deep learning.

Chollet, F., 2017. Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1251–1258.

Dai, J., Li, Y., He, K., Sun, J., 2016. R-fcn: Object detection via region-based fully convolutional networks. In: Advances in Neural Information Processing Systems, pp. 379–387.

Durmuş, H., Güneş, E.O., Kırcı, M., 2017. Disease detection on the leaves of the tomato plants by using deep learning. In: 2017 6th International Conference on Agro-Geoinformatics. IEEE, pp. 1–5.

Elhassouny, A., Smarandache, F., 2019. Smart mobile application to recognize tomato leaf diseases using convolutional neural networks. In: 2019 International Conference of Computer Science and Renewable Energies (ICCSRE), IEEE, pp. 1–4.

Fuentes, A., Yoon, S., Kim, S.C., Park, D.S., 2017. A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. Sensors 17 (9), 2022.

Gour, M., Jain, S., 2020. Stacked convolutional neural network for diagnosis of covid-19 disease from x-ray images, arXiv preprint arXiv:2006.13817.

Gour, M., Jain, S., Agrawal, R., 2019. Deeprnnetseg: Deep residual neural network for nuclei segmentation on breast cancer histopathological images. In: International Conference on Computer Vision and Image Processing. Springer, pp. 243–253.

Gour, M., Jain, S., Sunil Kumar, T., 2020. Residual learning based cnn for breast cancer histopathological image classification. Int. J. Imaging Syst. Technol.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778. https://doi.org/10.1109/CVPR.2016.90.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2020. Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861.

Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708.

Hughes, D., Salathé, M., et al., 2015. An open access repository of images on plant health to enable the development of mobile disease diagnostics, arXiv preprint arXiv: 1511.08060.

Karthik, R., Hariharan, M., Anand, S., Mathikshara, P., Johnson, A., Menaka, R., 2020. Attention embedded residual cnn for disease detection in tomato leaves. Appl. Soft Comput. 86, 105933.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems 25. Curran Associates, Inc., pp. 1097–1105

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C., 2016. Ssd: Single shot multibox detector. In: European Conference on Computer Vision. Springer, pp. 21–37.

Mirza, M., Osindero, S., 2014. Conditional generative adversarial nets, arXiv preprint arXiv:1411.1784.

Rangarajan, A.K., Purushothaman, R., Ramesh, A., 2018. Tomato crop disease classification using pre-trained deep learning algorithm. Procedia Comput. Sci. 133, 1040–1047.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection, in. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788.

Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99.

Savary, S., Ficke, A., Aubertot, J.-N., Hollier, C., 2012. Crop losses due to diseases and their implications for global food production losses and food security.

Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2016. Rethinking the inception architecture for computer vision, in. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2016. Rethinking the inception architecture for computer vision, in. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826.

Widiyanto, S., Fitrianto, R., Wardani, D.T., 2019. Implementation of convolutional neural network method for classification of diseases in tomato leaves. In: 2019 Fourth International Conference on Informatics and Computing (ICIC). IEEE, pp. 1–5.

Zhang, K., Wu, Q., Liu, A., Meng, X., 2018. Can deep learning identify tomato leaf disease? Adv. Multimedia.