Journal of Agricultural Sciences
(Tarim Bilimleri Dergisi)

J Agr Sci-Tarim Bili
e-ISSN: 2148-9297

jas.ankara.edu.tr

JAS

Diagnosis of Tomato Plant Diseases Using Pre-trained Architectures and A Proposed Convolutional Neural Network Model

Dilara GERDAN KOC*, Caner KOC, Mustafa VATANDAS

Ankara University, Faculty of Agriculture, Department of Agricultural Machinery and Technologies Engineering, Ankara, Turkey

ARTICLE INFO
Research Article
Corresponding Author: Dilara GERDAN KOC, E-mail: dgerdan@ankara.edu.tr
Received: 24 Jun 2021 / Revised: 9 Nov 2022 / Accepted: 11 Nov 2022 / Online: 25 Mar 2023

Cite this article

GERDAN KOC D, KOC C, VATANDAS M (2023). Diagnosis of Tomato Plant Diseases Using Pre-trained Architectures and A Proposed Convolutional Neural Network Model. *Journal of Agricultural Sciences (Tarim Bilimleri Dergisi),* 29(2):627-638. DOI: 10.15832/ankutbd.957265

ABSTRACT

Tomatoes are of the most important vegetables in the world. Presence of diseases and pests in the growing area significantly affect the choice of variety in tomato. The aim of this study is to diagnose tomato plant diseases faster and with higher degrees of accuracy. For this purpose, deep learning was used to diagnose some diseases in tomatoes, including bacterial spot, early blight, leaf mold, septoria leaf spot, target spot, mosaic virus, and yellow leaf curl virus were analyzed CNN models. A CNN model with a 2D convolutional three layers, one flatten layer approach and several Keras models, including DenseNet201, InceptionResNetV2, MobileNet, Visual Geometry Group 16 architectures were proposed. The experimental results showed that the accuracy scores were 99.82%, 92.12%, 92.75%, 91.50% and 84.12% training accuracy, respectively. The proposed CNN model provided the opportunity for rapid diagnosis for approximately 14.9 minutes. The results obtained in this study can be used in robotic spraying and harvesting operations.

Keywords: Automated diagnose, CNN modification, Deep learning, Smart farming

1. Introduction

Tomatoes are among the most widely produced and consumed foods in the world (*Solanum lycopersicum* L.). According to Turkish Statistical Institute (TSI) 2021 data, 13,095,258 tons of tomatoes were produced (TSI 2022) and they form an important agricultural product in Turkey (FAO 2019).

The cultivation of tomatoes is commonly carried out under greenhouse and in field conditions. In order to increase efficiency in production, many parameters such as climate conditions, diseases and pests should be taken into consideration. Tomato plant diseases can be divided into two groups according to their factors, physiological and pathogens. Physiological factors are typically caused by climatic conditions such as excessive irrigation and malnutrition, sunburn, cracking in fruits, and rot. Viral diseases cause damage such as black spots, brown spots, curl and deformity on the leaves. Early diagnosis of plant diseases is critical in preventing enormous economic and agricultural loss. Every disease in tomato manifests itself in different ways. Tomato early blight disease (*Alternaria solani*) occurs as spots on the leaves, stems and fruits that can be seen in every phase of the plant. Tomato mildew (*Phytophthora infestans*) disease starts as pale green spots which later turn brown and then black. Tomato leaf mold (*Cladosporium fulvum* = *Fulvia fulva*) is seen as yellow spots on the leaves, and brown mold occurs in the lower parts of these spots in the later stages (Griffiths et al. 2018). Viruses in tomato plants can cause diseases (Nitzany, 1960), one of which is the *Tomato mosaic virus* disease (ToMV) the symptoms of which manifest as light green and yellow irregular mosaic spots. The misshapen fruit causes damages such as the formation of smaller fruit than normal. Tomato yellow leaf curl virus disease, on the other hand, manifests itself in the form of shrinkage and swelling in the infected leaves, inward curving, stunting and deformity in the plant (Sade et al. 2020). According to Richard et al. (2017), the bacterial spot disease (*Xanthomonas vesicatoria*) can affect any part of the plant, including the stem, leaf, and fruit. The symptoms of this disease (*Pseudomonas syringae* pv. tomato) include the forming of stains on all above-ground organs of the plant (Abramovitch et al. 2006).

This symptom begins in the seedling period, and many brown-black spots appear on the leaves and stems of the seedlings. These spots cause drying of the entire seedling over time. Various methods are used to combat these diseases and include cultural measures if the disease is detected early, and chemical methods, in other words pesticides, are applied in the future. The early detection of diseases and determining and distinguishing exactly what factor caused the disease is critical in order to prevent devastation to large numbers of crops. Rapid diagnosis is crucial in preventing serious economic losses for tomato cultivators.

A wide variety of techniques are used for disease diagnosis and classification in tomatoes. Deep learning is one such technique and is based on Artificial Neural Networks, using multiple layers of neurons to extract attributes from raw data, and is designed to simulate the working mechanism of the human brain. Although deep learning, also known as deep neural networks or hierarchical learning, emerged in 2006 as a new field of machine learning, its foundations date back to 1940 (Deng & Yu 2014).

Diagnosing plant diseases via deep learning is popular diagnostic method for many researchers. A higher disease diagnosis rate can be obtained if this dataset is used by comparing it with images from the real environment. A further way to increase the success rate in disease diagnosis is to use hyperspectral/multispectral imaging techniques in deep learning studies. In particular, the diagnosis of diseases in their early stages is possible through the use of these technologies. Disease detection in the early stages means less pesticide application, greater economic gain, and less environmental pollution. In recent years, a number of algorithms have been used to determine plant diseases through the deep learning method (Dhakal & Shakya 2018). Ferentinos (2018) used a database consisting of 87,848 photographs taken in different conditions, both in the laboratory environment and in the field. In the database created, 58 diseases belonging to 25 plant species were examined. The trials highlighted that the VGG CNN model architecture, with a success rate of 99.53%, had the highest classification success. Another study in which the VGG19 model classified with the highest accuracy with a rate of 97.86% was carried out by Turkoglu & Hanbay (2019). Studies on real-time detection of plant diseases and the development of appropriate automation systems have made significant progress in recent years.

Some of the deep learning architectures currently in use have been successfully trained to accurately identify diseases. It is difficult to keep its generalisability because high accuracy must be addressed to the output, which may differ when implemented to real items (PC, tablet, mobile phone, various camera systems, etc.). The goal of this research is to develop a simpler convolutional neural network for diagnosing tomato plant diseases. Various deep learning models [Visual Geometry Group 16 (VGG16), MobileNet, DenseNet, Inception-ResNet] and proposed methods were investigated for this purpose. The proposed model's results were also compared to existing state-of-the-art methods.

2. Material and Methods

2.1. Dataset

In recent years, it is possible to access many data sets created by experts. In this study, the widely used PlantVillage augmented data set was used. PlantVillage augmented dataset (https://www.kaggle.com/vipoooool/new-plant-diseases-dataset) which contains 38 class of vegetable and fruit diseases (Hughes & Salathe 2015; Geetharamani & Pandian 2019). In total, 18,440 images of 7 types of diseased (bacterial spot, early blight, leaf mold, septoria leaf spot, target spot, mosaic virus, yellow leaf curl virus) and healthy leaves were selected for trials (Figure 1).

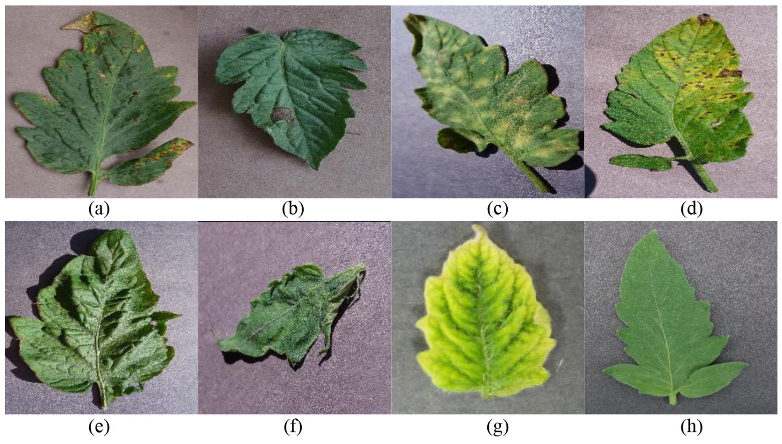


Figure 1- Bacterial spot (a), Early blight (b), Leaf mold (c), Septoria leaf spot (d), Target spot (e), Mosaic virus (f), Yellow leaf curl virus (g) and healthy (h)

This study's train and test image numbers of diseased and healthy leaves are provided in Table 1.

Table 1- Training and test dataset

Diseases	*Training images*	*Test images*
Bacterial spot	1702	425
Early blight	1920	480
Leaf mold	1882	470
Septoria leaf spot	1745	436
Target spot	1827	457
Mosaic virus	1790	448
Yellow leaf curl virus	1961	490
Healthy	1926	481

2.2. Convolutional neural network based models

Deep learning is a sub-field of machine learning and has been used in many areas in recent years. In today's engineering applications, the emphasis is on finding solutions for object recognition problems that require complex solutions. CNN is the most commonly used deep learning algorithm for object recognition applications. Generally, the basic architecture of the deep learning concept is accepted as CNN. The CNN architecture consists of convolution, pooling, fully connected, dropout and classification layers. In the convolution layer, the filtering-size reduction process is generally performed. A smaller size filter is determined for the existing object and the filtering process is performed by subjecting it to certain processes. This process continues until the system is trained. Like the convolutional layer, the pooling layer is also used for dimension reduction. In this way, the focus is on more important features. There are two different pooling techniques commonly used in CNN models, maximum and average pooling. Fully connected layers are often found towards the end of the CNN architecture and can be used to optimize goals such as classification scores. Dropout will improve learning performance by removing some connections within the network. In the fully connected layer, the visual that passes through the convolution and pooling layer and is in matrix form is transformed into a vector. After receiving the CNN input data, the training process begins and at the end of this training, it produces a final output to make a comparison with the correct result (Lawrence et al. 1997; Krizhevsky et al. 2017; LeCun et al. 2015).

In the study, pre-trained VGG16, MobileNet, DenseNet, Inception-ResNet models and a proposed CNN model were used to diagnose tomato plant diseases.

2.2.1. VGG16

VGG16, previous AlexNet (Krizhevsky et al. 2017) derivatives, focuses on smaller sizes and steps in the first convolution layer. Although VGG16 deep learning architecture provides a higher accuracy performance than AlexNet, because of the number of parameters it uses a large amount of memory. On the other hand, smaller filters were used compared to AlexNet. This architecture uses fixed 3x3 dimensional filters with variable numbers of 64, 128, and 256 filters in all convolution layers. VGG16 is a simple CNN model with GPU support that achieved 89% success in the ImageNet Large Scale Visual Recognition Challenge competition in 2014 (Simonyan & Zisserman 2014). The double or triple convolution layers are followed by the communing layers. On the input layer, the image is 224 x 224 x 3 pixels. The last layer is the classification layer. The architectural structure of VGG16 is given in Figure 2. This model consists of a total of 16 layers with 13 convolution layers, 3 fully connected layers, pooling, Relu, Dropout and Softmax layers.

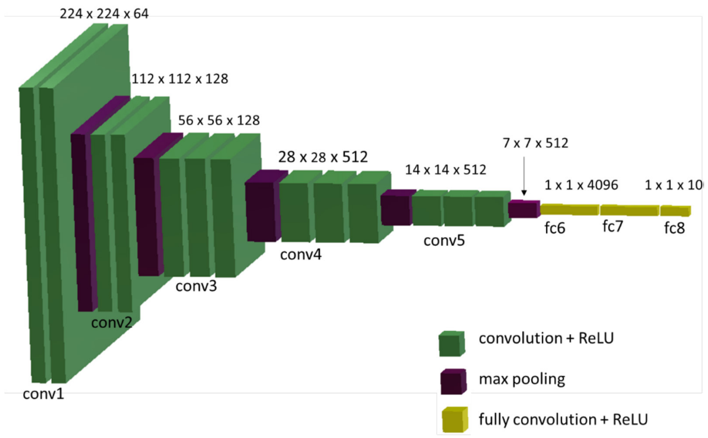


Figure 2- VGG16 architecture (Simonyan & Zisserman 2014)

2.2.2. MobileNet

MobileNet (Howard et al. 2017) has been developed to run vision applications on embedded and mobile platforms. The algorithm is based on depth-wise separable convolutional layers; these deeply separable layers are 1 x 1 convolutional layers that separate layers and are called point convolutions (Figure 3). Standard convolution filters the inputs and combines them into a new output in one step. The deeply separable convolutional layer, on the other hand, is composed of two sub-layers as in-depth convolution and point convolution. For both layers, batchnorm and ReLU non-linearities were used. Exception of the final fully connected layer, 3x3 convolution layer was followed by the batch norm and ReLU (Howard et al. 2017).

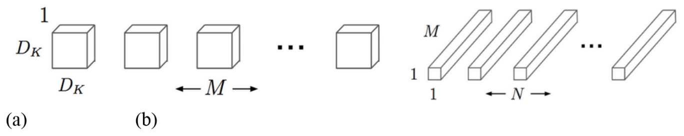


Figure 3- Depthwise convolution (a), pointwise convolution (b) (Howard et al. 2017)

2.2.3. DenseNet

Densely Connected Convolutional Networks (DenseNet) forward connects each layer to the other layers. In DenseNet architecture, each layer uses the properties of all previous layers as input and its own properties in the layer are given as input to the next layers. The advantage of DenseNet architectures is that they allow feature propagation and feature reuse, thereby reducing the number of parameters. One of the advantages of DenseNet is that the model parameters are smaller than those of ResNet. The architecture consists of dense blocks, composite function, pooling layers, bottleneck layers and compression (Figure 4) (Huang et al. 2017). This study used DenseNet201 architecture.

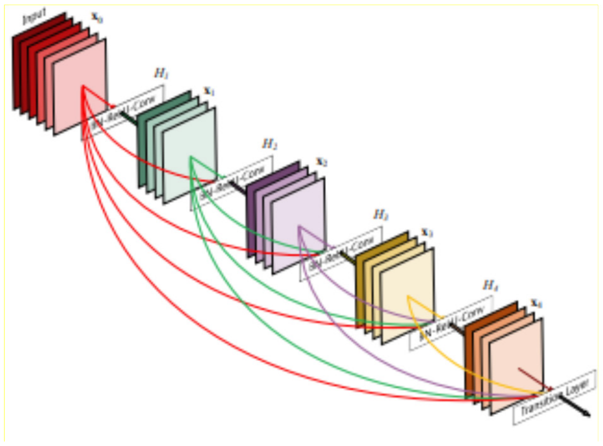


Figure 4- DenseNet architecture (Huang et al. 2017)

2.2.4. Inception-ResNet

Inception-ResNet, the structure s created by the parallel realization of 1×1, 3×3, 5×5 filters and 3×3 maximum joint operation in the convolution layers of Inception modules. Due to parallel operations, the size of the output and the number of parameters increase in complexity. To overcome this problem, 1×1 convolution layers are added before the parallel Naive Inception convolution layers to achieve size reduction. The pooling of feature is the process of centering on attributes. This is similar to the process of reducing the width and height dimensions in a normal maximum joint operation. In addition, the 1×1 convolution layers are followed by the ReLU process as the activation function. The residual layer was optimized by changing the size of the first convolution process to 1×1. ReLU is a structure based on taking a direct sum from the activation function output and transferring the previous activation value to the output even under conditions where learning stops at the convolution outputs.

2.3. Proposed CNN model

The CNN model consists of 4 convolution layers, MaxPooling, Batch Normalization, 2 Dense layers, and Dropout. While ReLU activation was used for each CNN output estimate, Softmax activation was used for the output of the final sequential estimation model sum. Softmax is usually used as the activation code at the end of the network; it is used in the process of label estimation on the data submitted to the trained network, after all stages are completed. Softmax makes it more effective and accurate in multiple classification studies (Tang 2013). The mathematical model for Softmax is presented in equations 1, 2, and 3.

$$a_i = \sum h_k W_{ki}, \qquad (1)$$

$$p_i = \frac{\exp(a_i)}{\sum_j^{10} \exp(a_j)} \qquad (2)$$

h: be the activation of the penultimate layer nodes

W: the weight

given by a, is

The predicted class î would be

$$\hat{\imath} = \arg\max p_i$$

$$\hat{\imath} = \arg\max a_i \qquad (3)$$

The Adam model and a Keras optimizer was used with a learning rate of 0.0001. ReduceLROnPlateau and EarlyStopping have served to terminate the training when the model reach the learning epoch. In order to obtain a view of the statistics of the model, to monitor the metrics and to record them periodically at various stages of the training. The purpose of the training pause aims to minimize the loss. Certain parameters need to be adjusted to minimum loss such as monitor, patience, and verbose. When training is paused during periods of no improvement, the final epoch before the accuracy rate begins to decline is displayed. The parameters related to the model are summarized in Table 2.

Table 2- Proposed CNN model parameters

1. Conv2D layer	*2. Conv2D layer*	*3. Conv2D layer*	*Flatten layer*
Output shape	Output shape	Output shape	Output shape
62, 62, 128	29, 29, 256	12, 12, 512	18,432
MaxPooling2D	MaxPooling2D	MaxPooling2D	1. Dense and Dropout
31, 31, 128	14, 14, 256	6, 6, 512	512
Batch normalization	Batch normalization	Batch normalization	2. Dense and Dropout
31, 31, 128	14, 14, 256	6, 6, 512	256
Activation function	Activation function	Activation function	Activation function
ReLu	ReLu	ReLu	Softmax
Total	Total parameters: 3,877,000	Trainable parameters: 3,875,208	Non-trainable parameters: 1,792

In the first convolution layer, the input level is 128x128x3, in the second convolution layer is 256x256x3, and in the third convolution layer is 512x512x3. The flatten layer was composed of two dense layers and a dropout. By the end of the model summary, the total parameters were 3,877,000, with 3,875,208 trainable parameters and 1,792 non-trainable parameters.

2.4. Training -testing data and model evaluation

This study used a 80%:20% partition ratio on the models. 80 percent of the dataset was divided up into training sets. While 20% of the dataset was partitioned as test set for model verification. For this purpose, 14,753 images were used for training and 3,687 images were used for testing.

In order to evaluate the performance of the outputs, accuracy of true positive, false positive, false negative, and true negative were used. These values evaluate the performance of the classification model including the Accuracy, Precision, Recall and F1-score. True positive correctly predicts the positive class in the model. Likewise, true negative correctly predicts the negative class. False positive incorrectly predicts the positive class in the model while false negative incorrectly predicts the negative class. Accuracy is used to measure the success of the model using ratio of true prediction in all samples. Precision measures the percentage of outcomes correctly classified. Recall measures the proportion of actual positives that are identified correctly. F1-score value shows the harmonic average of Precision and Recall values (Klinkman et al. 1998; Cinar & Koklu 2022). The formulas for the metrics used are given below.

$$\text{Accuracy} = \frac{TP}{\text{TP+TN+FP+FN}} \quad (4)$$

$$Precision = \frac{TP}{TP+FP} \quad (5)$$

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

$$F1 = 2 * \frac{Precision*Recall}{Precision+Recall} \quad (7)$$

There have been chosen particular parameters for the proposed CNN model (Table 3).

Table 3- CNN training parameters

Parameter	***Value***
Batch size	32
Epoch	25
Momentum	0.9
Learning rate	0.0001
Metric	Categorical cross entropy
Patience*	2
Factor*	0.2
Verbose*	2
Optimization method*	Adam

*Specific parameters for Custom CNN model

2.5. Software

The codes were written using Python and were run on Google Colab. The computer used had a Windows 10 operating system, Intel® Core™ i7-10750H CPU 16 GB RAM with NVIDIA GeForce RTX 2060 graphics processor. The model used in the study was compiled with GPU support as TensorFlow 2.4.1 is compatible with Python 3.8.7, CUDA 11.1.0, CuDNN (deep neural networks library) 11.2.

TensorFlow as a platform was chosen because it has pre-trained models and a high-level neural network application programming interface integrated with Keras in versions 2.0 and above. Keras is an effective application programming interface for creating and training deep learning models (Gulli et al. 2019).

NumPy, SciPy, Pandas, Matplotlib, Scikit-learn, Flask, and Cython libraries were used for this study. Numpy, the open-source library of the Python programming language, provides data blocks that implement multidimensional mathematical vectors and matrices along with useful functions for mathematical operations. With Numpy, large volumes of data can be processed and advanced mathematical operations can be performed quickly and with less code. SciPy functions are created in the Numpy library. SciPy was selected for this study as it increases data processing capabilities. Pandas library was chosen as it performs fast mathematical operations and allows the processing of different types of complex data tables through its special data structures. Matplotlib was used for data visualization. Scikit-learn, one of the best-known machine learning libraries, can be used as an independent machine learning library and can successfully create various machine learning models. SciKit (SciPy Toolkit) is an extension developed separately for SciPy. It has been used to create machine learning models as well as providing efficient tools such as data preprocessing, supervised and unsupervised learning, validation and metrics (El-Amir & Hamdy 2020).

3. Results

Table 4 shows the results of comparing the proposed CNN model to other models to confirm the results. The sequential new model was built with various hyperparameters (Momentum, Learning rate, patience, verbose etc.). It was calculated using the tomato disease dataset. The results were obtained in a short period of time and were more efficient than other CNN models in multi-classification tasks. In terms of total parameter size and time, the method has provided an advantage in terms of obtaining fast and highly accurate results.

High accuracy rates were achieved in the disease classification study performed with 4 different pre-trained models and a proposed deep CNN approach method. The models were successful, but pre-trained models achieved similar results with previous studies. The result of experimental studies consisting of 25 iterations is shown. In this experimental study, training success is calculated as 99.82% at the highest level and a rapid running time obtained as 14.88 minutes. The performance of the proposed CNN model was compared with Keras pre-trained weight architectures such as MobileNet, DenseNet201, VGG16 and InceptionResNetV2 in Table 4. The proposed model diagnosed faster than from VGG16, MobileNet, DenseNet201, InceptionResNetV2 approximately 6, 3, 23, 11 times, respectively.

Table 4- Accuracy of CNN models

Models		*Training*		*Testing*		*Time*
Architectures	Input size	Loss	Accuracy	Loss	Accuracy	Minutes
VGG16	224, 224, 3	0.2239	0.9212	0.1835	0.9388	87.06
MobileNet	224, 224, 3	0.2135	0.9275	0.1901	0.9350	42.18
DenseNet201	224, 224, 3	0.2560	0.9150	0.1729	0.9434	339.72
InceptionResNetV2	299, 299, 3	0.4629	0.8412	0.4147	0.8569	162
Proposed CNN model	256, 256, 3	0.0084	0.9982	0.0587	0.9826	14.88

Among the performance criteria, the largest training and testing loss was seen in the InceptionResNetV2 model. The lowest losses were seen in the proposed CNN model. DenseNet201 was the slowest algorithm in terms of training and testing time. Following the proposed CNN model, DenseNet201 and VGG16 algorithms have high accuracy.

The performance metrics (Precision, Recall and F1-score) of the proposed CNN model is given in Table 5.

Table 5- The performance metrics of the proposed CNN model

	Precision	*Recal*	*F1-score*	*Accuracy*
Bacterial spot	1.00	0.99	0.99	99.78
Early blight	0.95	0.98	0.97	99.1
Leaf mold	0.98	1.00	0.99	99.7
Septoria leaf spot	0.99	0.94	0.96	99.19
Target spot	0.98	0.96	0.97	99.27
Yellow leaf curl virus	1.00	1.00	1.00	99.95
Mosaic virus	1.00	1.00	1.00	99.92
Healty	0.98	1.00	0.99	99.67

When Table 4 is examined, the model with the lowest loss in the testing was the DenseNet201 architecture with a value of 0.1729 fallowing the proposed CNN model. In addition, the DenseNet201 proved to be the most successful architecture with the least loss based on the results of the training. InceptionResNetV2 was found to be the architecture with the highest loss. The plot accuracy-epochs for the models used in the experiments are given in Figure 5.

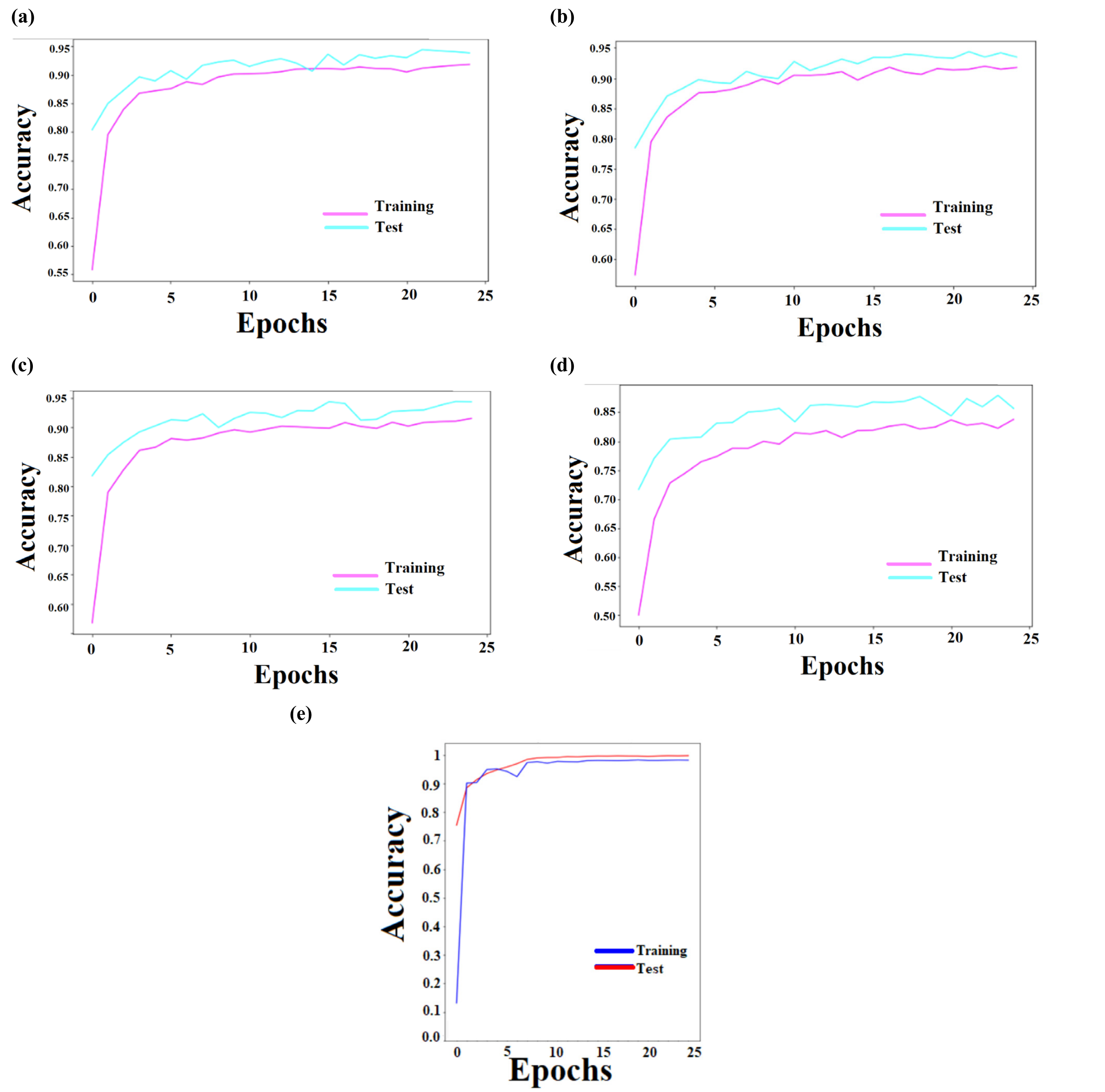
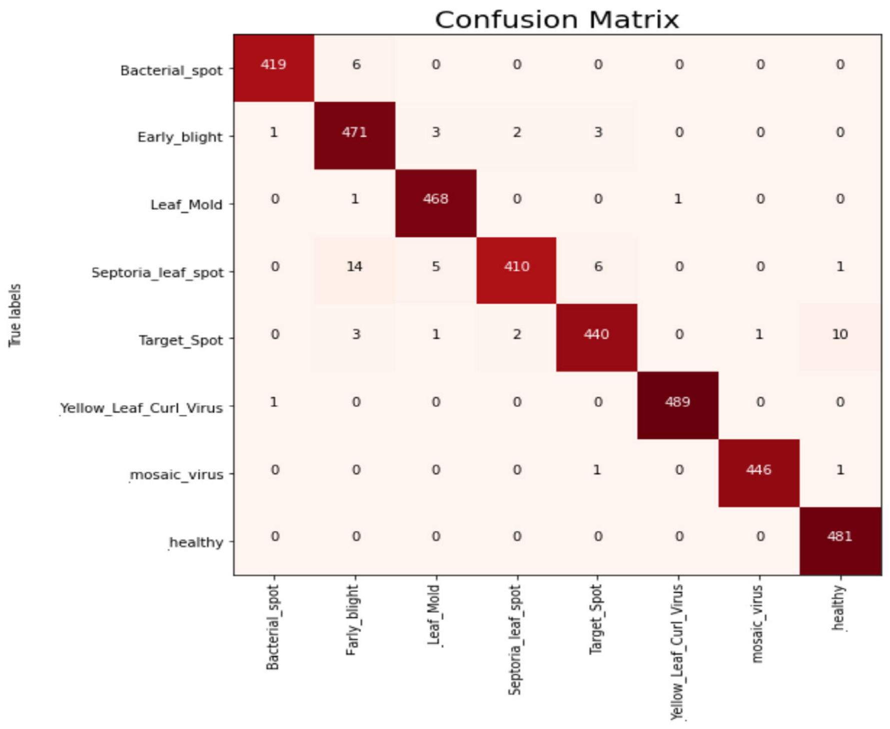


Figure 5- Plot line of train and test loss and accuracies (a) VGG16, (b) MobileNet, (c) DenseNet201, (d) InceptionResNetV2, (e) proposed CNN model

The accuracy and loss graphs of the proposed CNN model are given in Figure 5. The iteration value in the experiments was determined as 25 epochs. However, due to the callbacks, the training was completed in the 24th epoch, with an accuracy of 99.82% (Figure 5). When this result is compared with other studies in the literature, the model used has made a classification with a higher accuracy. The confusion matrix graph obtained for the CNN model is given in Figure 6. As shown in the confusion matrix, the diagnosis of the disease was found with high degrees of accuracy from the images tested. Incorrectly classified images may be the result of early symptoms of selected disease types which have a tendency to show similar symptoms (Blancard 2012).



True \ Predict	Bacterial_spot	Early_blight	Leaf_Mold	Septoria_leaf_spot	Target_Spot	Yellow_Leaf_Curl_Virus	mosaic_virus	healthy
Bacterial_spot	419	6	0	0	0	0	0	0
Early_blight	1	471	3	2	3	0	0	0
Leaf_Mold	0	1	468	0	0	1	0	0
Septoria_leaf_spot	0	14	5	410	6	0	0	1
Target_Spot	0	3	1	2	440	0	1	10
Yellow_Leaf_Curl_Virus	1	0	0	0	0	489	0	0
mosaic_virus	0	0	0	0	1	0	446	1
healthy	0	0	0	0	0	0	0	481

Figure 6- Confusion Matrix graph of the CNN model

Any misreading in the confusion matrix were shown in Figure 7. Similar symptoms observed in the leaf during the early stage are thought to be the cause of this. According to Figure 7, when early blight disease is predicted, it resembles bacterial spot disease about 82% of the time, while in another image of the same disease, it resembles leaf mold 58% of the time and 35% of the time. Target spot disease is thought to resemble Septoria leaf spot 88% of the time while resembling mosaic virus 17% of the time. The model discovered that the disease appeared to resemble Leaf mold in another image with the same condition 76% of the time. Early blight, target spot, leaf mold, Septoria leaf spot, and early period diseases of the mosaic virus are diseases with inaccurate predictions. This is believed to be the result of the similar symptoms that were highlighted above and displayed on the leaf in the early period.

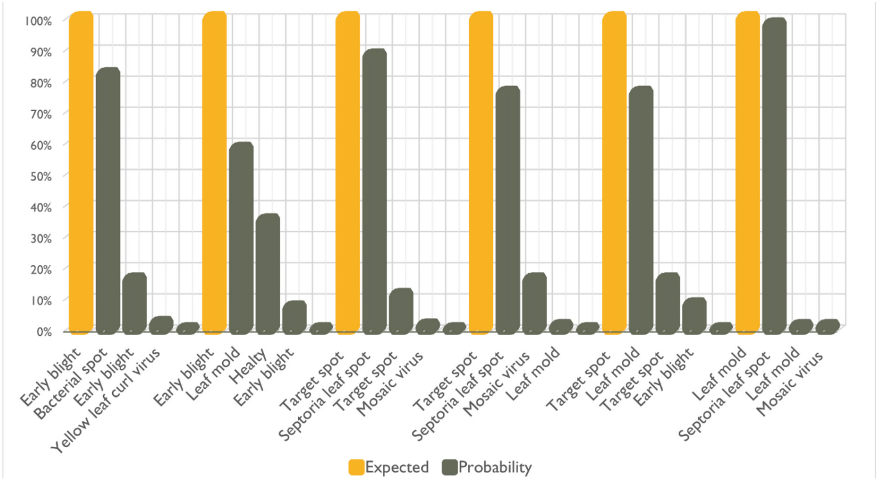


Figure 7- Wrongly predicted diseases

4. Discussion

Several studies have been conducted to improve success rates in the classification and diagnosis of fruit and vegetable diseases that affect fruits and vegetables. In Table 6, only studies related to the determination of tomato diseases are given. Table 6 also gives the number of images in the data set, the diseases that were diagnosed, the models used and their accuracy rates.

Table 6- Comparison of accuracy rates of tomato diseases using deep learning

Authour/ authors	*Model architecture*	*Images*	*Disease/diseases*	*Accuracy*
Brahimi et al. (2017)	AlexNet and GoogleNet	14.828	Yellow leaf curl virus, tomato mosaic virus, target spot, spider mites, septoria spot, leaf mold, lateblight, earlyblight, bacterial spot	98.66%, 99.18%*
Fuentes et al. (2017)	Faster R-CNN with VGG-16, ResNet-50, ResNeXt-50, R-FCN (ResNet-50 as feature extractor)	5.000	Gray mold,canker, leaf mold, plague, leaf miner, whitefly, low temperature, nutritional excess or deficiency, powdery mildew.	83%, 75.37%, 71.1%, 85.98%*
Rangarajan et al. (2018)	AlexNet and VGG16	13.262	Late blight, leaf mold, two-spotted spider mite attack, target spot, mosaic virus disease, yellow leaf curl virus disease	97.49%*, 97.23%
Tm et al. (2018)	AlexNet, GoogleNet and LeNet	18.160	Septoria leaf spot, yellow leaf curl	A highest validation accuracy of 94.8%* (with 30 epochs) a high 99.3%* of training accuracy was obtained LeNet)
Zhang et al. (2018)	AlexNet(SGD-Adam), GoogLeNet (SGD-Adam), and ResNet (SGD-Adam)	1000	Corynespora leaf spot disease, early blight, late blight, leaf mold disease, septoria leaf spot, two-spotted spider mite, virus disease, yellow leaf curl disease	95.83%, 13.86%, 95.66%, 94.06%, 96.51%*, 94.39%
Wang et al. (2019)	Faster R-CNN (VGG-16 ResNet-50 ResNet-101) and Mask R-CNN (MobileNet ResNet-50 ResNet-101)	286	Malformed fruit, blotchy ripening, puffy fruit, dehiscent fruit, blossom-end rot, sunscald, virus disease, gray mold, ulcer disease, anthracnose)	86.09%, 88.41%, 88.53%, 88.39%, 98.52%, 99.64%*
Mkonyi et al. (2020)	VGG16, VGG19, and ResNet50	2.145	Tuta absoluta	91.9%*
Verma et al. (2020)	AlexNet, SqueezeNet, Inception V3	2342	Tomato lLate blight (Early, middle and end stage)	90.43%, 93.40%*, 90.76%
Zhang et al. (2020)	Faster RCNN, Faster RCNN-mobile, Faster RCNN-res101 (combined k-means analysis)	4.178	Powdery mildew, blight, leaf mold fungus, and mosaic virus	97.01%, 97.31%, 98.54%*
Al-gaashani et al. (2022)	MobileNetV2 and NASNetMobile	1.152	Bacterial spot, yellow leaf curl virus, septoria leaf spot, leaf mould, healthy, late blight	97%, 97%
Tarek et al. (2022)	ResNet50, InceptionV3, AlexNet, MobileNetV1, MobileNetV2 and MobileNetV3	16,004	Target spot, septoria leaf spot, two spotted spider-mite, yellow leaf curl virus, bacterial spot, leaf mold, mosaic virus, late and early blight.	99.80%, 99.62%, 96.68%, 99.49%, 98.93%,, 99.81%, 98.99%
Our proposed model	VGG16, MobileNet DenseNet201 InceptionResNetV2 Custom CNN model	18.440	Bacterial spot, early blight, leaf mold, septoria leaf spot, target spot, mosaic virus, yellow leaf curl virus	92.12%, 92.75%, 91.50%, 84.12% 99.82%

*Bold numerical characteristics are the best performance of the study

When all the studies were examined, it was seen that pre-trained models gave the most successful results. The highest accuracy found in the literature is the 99.64% result achieved by Wang et al. (2019). Brahimi et al. (2017) achieved a 99.18% accuracy result in their 2017 study. In addition, among the data sets used for tomato disease detection in the literature, deep learning applications have been conducted with the highest number of samples in this study.

The proposed model achieved high accuracy rates in the classification of tomato plant diseases. The models used were successful, but pre-trained models produced comparable results in previous studies. The results of 25 iterations of experimental studies are shown. This experimental study was completed in 14.88 minutes with a success rate of 99.82%. When this result is compared to previous studies in the literature, the model used produced a more accurate classification.

5. Conclusions

The study attempted to use sequential CNN estimation to identify diseases found in tomatoes. A simpler CNN model is focused on faster and higher accuracy results, with the goal of providing better quality and efficient products with fewer errors. DenseNet201, InceptionResNetV2, MobileNet, VGG16, and Custom CNN models achieved training success rates of 92.12%, 92.75%, 91.50%, 84.12%, and 99.82%, respectively.

Artificial intelligence is being used in agriculture to identify diseases and pests, which allows for accurate and highly efficient monitoring and screening of thousands of products produced in horticulture and field crops. The findings of this study can be applied to crop growth monitoring and robotic spraying operations.

Data availability: Data are available on request due to privacy or other restrictions.

Authorship Contributions: Concept: D.G.K., C.K., M.V., Design: D.G.K., C.K., M.V., Data Collection or Processing: D.G.K., Analysis or Interpretation: D.G.K., Literature Search: D.G.K., C.K., Writing: D.G.K., C.K., M.V.

Conflict of Interest: No conflict of interest was declared by the authors.

Financial Disclosure: The authors declared that this study received no financial support.

References

Abramovitch R B, Anderson J C & Martin G B (2006). Bacterial elicitation and evasion of plant innate immunity. *Nature Reviews Moleculer Cell Biology* 7: 601-611. doi.org/10.1038/nrm1984

Babu S, Maravarman M, & Pitchai R (2022). Detection of Rice Plant Disease Using Deep Learning Techniques. *Journal of Mobile Multimedia* 18(3): 757-770. doi.org/10.13052/jmm1550-4646.18314

Blancard D (2012). Tomato diseases: identification, biology and control: a colour handbook. *CRC Press.* pp. 688. doi.org/10.1201/b15145

Brahimi M, Boukhalfa K & Moussaoui A (2017). Deep learning for tomato diseases: classification and symptoms visualization. *Applied Artificial Intelligence* 31(4): 299-315. doi.org/10.1080/08839514.2017.1315516

Canziani A, Paszke A & Culurciello E (2016). An analysis of deep neural network models for practical applications. *arXiv* doi.org/10.48550/arXiv.1605.07678

Cinar I & Koklu M (2022). Identification of Rice Varieties Using Machine Learning Algorithms. *Journal of Agricultural Sciences* 28(2): 307-325. doi.org/10.15832/ankutbd.862482

Deng L & Yu D (2014). Three Classes of Deep Learning Networks in Deep learning: methods and applications. *Foundations and trends in signal processing* 7(3-4): 197-387. doi.org/10.1561/2000000039

Dhakal A & Shakya S (2018). Image-based plant disease detection with deep learning. *International Journal of Computer Trends and Technology* 61(1): 26-29. doi.org/10.14445/22312803/ijctt-v61p105

El-Amir H & Hamd M (2020). A Gentle Introduction in Deep Learning Pipeline, Apress, Berkeley, CA, USA, 2020. doi.org/10.1007/978-1-4842-5349-6_1

FAO (2019). Web Page: http://www.fao.org/faostat/en/#data/QC/visualize, Accessed on: 28.04.2021

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 (Basel)* 17(9): 2022. doi.org/10.3390/s17092022

Geetharamani G & Pandian A (2019). Identification of plant leaf diseases using a nine-layer deep convolutional neural network. *Computers & Electrical Engineering* 76: 323-338. doi.org/10.1016/j.compeleceng.2019.04.011

Griffiths S, Mesarich C H, Overdijk E J, Saccomanno B, De Wit P J & Collemare J (2018). Down-regulation of cladofulvin biosynthesis is required for biotrophic growth of Cladosporium fulvum on tomato. *Molecular Plant Pathology* 19(2): 369-380. doi.org/10.1111/mpp.12527

Gulli A, Kapoor A & Pal S (2019). Deep learning with TensorFlow 2 and Keras: regression, ConvNets, GANs, RNNs, NLP, and more with TensorFlow 2 and the Keras API. Packt Publishing Ltd.

Howard A G, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T & Adam H (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv.* doi.org/10.48550/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. *arXiv.* doi.org/10.48550/arXiv.1608.06993

Hughes D P & Salathe M (2015). An open access repository of images on plant health to enable the development of mobile disease diagnostics. *arXiv.* doi.org/10.48550/arXiv.1511.08060

Klinkman M S, Coyne J C, Gallo S & Schwenk T L (1998). False positives, false negatives, and the validity of the diagnosis of major depression in primary care. *Arch Fam Med* 7(5): 451-461. doi.org/10.1001/archfami.7.5.451

Krizhevsky A, Sutskever I & Hinton G E (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM* 60(6): 84-90. doi.org/10.1145/3065386

Lawrence S, Giles C L, Tsoi A C & Back AD (1997). Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks* 8(1): 98-113. doi.org/10.1109/72.554195

LeCun Y, Bengio Y & Hinton G (2015). Deep learning. *Nature* 521(7553): 436-444. doi.org/10.1038/nature14539

Mkonyi L, Rubanga D, Richard M, Zekeya N, Sawahiko S, Maiseli B & Machuve D (2020). Early identification of Tuta absoluta in tomato plants using deep learning. *Scientific African* 10: e00590. doi.org/10.1016/j.sciaf.2020.e00590

Nitzany F A (1960). Transmission of tobacco mosaic virus through tomato seed and virus inactivation by methods of seed extraction and seed treatments. *Ktavim* 10: 63-67.

Rangarajan A K, Purushothaman R & Ramesh A (2018). Tomato crop disease classification using pre-trained deep learning algorithm. *Procedia Computer Science* 133: 1040-1047. doi.org/10.1016/j.procs.2018.07.070

Richard D, Boyer C, Lefeuvre P, Canteros B I, Beni-Madhu S, Portier P & Pruvost O (2017). Complete genome sequences of six copper-resistant Xanthomonas strains causing bacterial spot of solaneous plants, belonging to X. gardneri, X. euvesicatoria, and X. vesicatoria, using long-read technology. *Genome Announc* 5(8): e01693-e0169316. doi.org/10.1128/genomeA.01693-16

Roy A M & Bhaduri J (2021). A deep learning enabled multi-class plant disease detection model based on computer vision. *AI* 2(3): 413-428. doi.org/10.3390/ai2030026

Sade D, Sade N, Brotman Y & Czosnek H (2020). Tomato yellow leaf curl virus (TYLCV)-resistant tomatoes share molecular mechanisms sustaining resistance with their wild progenitor Solanum habrochaites but not with TYLCV-susceptible tomatoes. *Plant Sci* 295: 110439. doi.org/10.1016/j.plantsci.2020.110439

Saleem M H, Potgieter J & Mahmood Arif K (2019). Plant disease detection and classification by deep learning. *Plants(Basel)* 8(11): 468. doi.org/10.3390/plants8110468

Simonyan K & Zisserman A (2014). Very deep convolutional networks for large-scale image recognition. *arXiv.* doi.org/10.48550/arXiv.1409.1556

Tm P, Pranathi A, SaiAshritha K, Chittaragi N B & Koolagudi S G (2018). Tomato leaf disease detection using convolutional neural networks. In 2018 Eleventh International Conference on Contemporary Computing (IC3) (pp. 1-5). IEEE. doi.org/10.1109/IC3.2018.8530532

Turkoglu M & Hanbay D (2019). Plant disease and pest detection using deep learning-based features. *Turkish Journal of Electrical Engineering & Computer Sciences* 27(3): 1636-1651. doi.org/10.3906/elk-1809-181

TSI (2022). Crop Production Statistics. Retrieved in August, 19, 2022 from https://data.tuik.gov.tr/Bulten/Index?p=Crop-Production-Statistics-2021-37249

Verma S, Chug A & Singh AP (2020). Application of convolutional neural networks for evaluation of disease severity in tomato plant. *Journal of Discrete Mathematical Sciences and Cryptography* 23(1): 273-282. doi.org/10.1080/09720529.2020.1721890

Wang Q, Qi F, Sun M, Qu J & Xue J (2019). Identification of Tomato Disease Types and Detection of Infected Areas Based on Deep Convolutional Neural Networks and Object Detection Techniques. *Computational Intelligence and Neuroscience* 2019: 15. doi.org/10.1155/2019/9142753

Zhang K, Wu Q, Liu A & Meng X (2018). Can Deep Learning Identify Tomato Leaf Disease? *Advances in Multimedia* 2018: 10. doi.org/10.1155/2018/6710865.

Zhang Y, Song C & Zhang D (2020). Deep Learning-based Object Detection Improvement for Tomato Disease. *IEEE Access* 8: 56607-56614. doi.org/10.1109/access.2020.2982456

Copyright@Author(s) - Available online at https://dergipark.org.tr/en/pub/ankutbd.
Content of this journal is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.