BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 70(2), 2022, Article number: e140689
DOI: 10.24425/bpasts.2022.140689

Early detection of major diseases in turmeric plant using improved deep learning algorithm

DEVISURYA V.*, DEVI PRIYA R., and ANITHA N.

Department of Information Technology, Kongu Engineering College, Perundurai, India

Abstract. Turmeric is affected by various diseases during its growth process. Not finding its diseases at early stages may lead to a loss in production and even crop failure. The most important thing is to accurately identify diseases of the turmeric plant. Instead of using multiple steps such as image pre-processing, feature extraction, and feature classification in the conventional method, the single-phase detection model is adopted to simplify recognizing turmeric plant leaf diseases. To enhance the detection accuracy of turmeric diseases, a deep learning-based technique called the Improved YOLOV3-Tiny model is proposed. To improve detection accuracy than YOLOV3-tiny, this method uses residual network structure based on the convolutional neural network in particular layers. The results show that the detection accuracy is improved in the proposed model compared to the YOLOV3-Tiny model. It enables anyone to perform fast and accurate turmeric leaf diseases detection. In this paper, major turmeric diseases like leaf spot, leaf blotch, and rhizome rot are identified using the Improved YOLOV3-Tiny algorithm. Training and testing images are captured during both day and night and compared with various YOLO methods and Faster R-CNN with the VGG16 model. Moreover, the experimental results show that the Cycle-GAN augmentation process on turmeric leaf dataset supports much for improving detection accuracy for smaller datasets and the proposed model has an advantage of high detection accuracy and fast recognition speed compared with existing traditional models.

Key words: artificial intelligence; computer vision; turmeric leaf diseases detection.

1. INTRODUCTION

This agricultural research is aimed towards an increase in productivity and food quality with increased profit and less expenditure. The world will need 50 percent more food by 2050. Agriculture is a five trillion-dollar global industry and now the industry is switching to artificial intelligence (AI) technologies. With the help of AI, analyzing real-time data such as temperature, soil conditions or weather conditions is much easier for farmers to increase the productivity of their farms. Precision agriculture also uses AI technology in diseases and pest detection and aid in finding poor nutritional plants. Sensors with AI technology can detect weeds and decide the herbicides for applying in the field [1]. AI is also used in creating seasonal forecasting models to increase agricultural productivity. From drones, AI-enabled cameras can capture images from farms and analyze the data for problem identification, suggesting potential improvements [2].

Turmeric herb is common in South and East India, but it is cultivated mostly in Malaysia and India as an annual crop [3]. The crop productivity may be affected due to various diseases like leaf spot, leaf blotch, and rhizome rot. Appropriate detection and prevention methods must be identified to increase the production rate. Early detection of diseases and control measures will reduce the usage of pesticides and safeguard environmental conditions.

*e-mail: swathisurya11@gmail.com

Manuscript submitted 2021-02-20, revised 2021-09-28, initially accepted for publication 2021-10-12, published in April 2022.

Traditional leaf diseases detection methods cannot produce good results in large-scale planting. Machine learning and image processing technology provide tremendous advantages over traditional recognition methods and manual diagnosis [4]. First, images are captured from crop fields. After pre-processing, infected areas of the leaf are extracted from the image. From the segmented images, features are extracted. Using these features and suitable classification techniques, plant leaf diseases are identified. Training with a huge number of datasets takes more time and requires additional memory. Using artificial intelligence (AI) models, these traditional problems are addressed. The development of AI models has enabled the collection of vast amounts of data, the design and programming of algorithms, the recovery of laws from data, and the construction of models employing modern hardware.

2. BACKGROUND

Deep learning, which was discovered in 2012, is the branch of the machine learning model. Based on this model, many network models were developed for image processing applications [5, 6]. Compared with traditional plant leaf disease detection methods, a convolutional neural network (CNN) uses an end-to-end structure for image pre-processing and feature extraction to simplify the detection process [7, 8]. Pre-trained images are trained in CNN, which not only saves time and labor, and allows for real-time judgment for a huge number of datasets, significantly reducing disease-related losses.

In recent years, researchers have proposed more sophisticated network structures like RCNN (region-convolutional

neural network), Faster-RCNN, and SPP-Net (spatial pyramid pooling-networks) for accurate object detection [9–11]. Even though these methods have high detection accuracy, the computing power and memory resources of the embedded platform are very limited. In mobile object detection, miniatured deep neural networks like YOLO (you only look once) [12] and SSD (single shot multi-box detector) [13] have been proposed which use high GPU to achieve high detection accuracy. Many research works are going on plant leaf disease detection using both single and double object detection methods. A recent review using deep learning techniques [14] has found that using deep learning techniques for image processing provides better accuracy results and is highly effective in detecting diseases. Mehra *et al.* [15] used machine learning approaches such as artificial neural networks (ANNs), classification and regression trees, and random forests to solve the problem of forecasting the pre-planting risk of Stagonospora nodorum blotch (SNB) in winter wheat (RFs). They created risk assessment models that could help with disease management decisions before the wheat crop is planted. Ferentinos [16] proposed a deep learning model to identify plant diseases. Their results have shown that the proposed model achieves a high success rate, and it is suggested to extend further for operating in real cultivation conditions. Moreover, they have carried out extensive experiments to identify and diagnose diseases for a banana plant at an early stage. Huang *et al.* [17] recommended two-sub models to classify plant types and diagnose diseases by effectively eliminating background inferences from images, which produced better results in both classification and disease recognition. Argüeso *et al.* [18] introduced few-shot learning (FSL) algorithms to overcome the issues of acquisition and annotation of large image datasets. As well as they have compared fine tuning methodology of FSL algorithms with traditional fine-tuning algorithms and concluded that the FSL algorithms work well because of the presence of Siamese networks and triplet loss. Sujatha *et al.* [19] analysed existing machine learning and deep learning methods used for identification of plant diseases and summarized that deep learning methods performed well and achieved better accuracy results when compared to the machine learning method. Mewada and Patoliaya [20] used support vector machine for automated plant disease classification which provide the accuracy of 97%. Only very few research works have been carried out on turmeric plant. Kuricheti and Supriya [21] use k-means algorithm for image segmentation and SVM classifier for feature extraction is used. Moreover, they have developed GUI system to detect turmeric diseases and control spreading of diseases which enhances the crop production. Rajasekaran *et al.* [22] proposed CNN based on VGG-16 architecture to detect diseases at an early stage and protect turmeric plants from rapid spread of diseases. Since only few research works have been carried out on turmeric plants, there is a wider scope for research works on this plant using deep learning techniques.

Many models like YOLO and YOLO-V3 are effectively used in plant disease detection. You only look once (YOLO) method unifies object localization and classification into a regression problem. YOLO versions do not need a region proposal network and they perform regression directly to detect an object in the image. YOLO-V3 detects small objects and has high detection and accuracy rate. Liu & Wang [23] used the YOLO-V3 model to detect tomato plant diseases in order to save time and effort for processing images and extracting features from images. They have implemented the model in real-world cultivations and detected the diseases quickly and accurately. Moreover, the proposed model also detects the pests which incurred a high loss of tomato plant production. From the existing studies, there are still considerable limitations in the identification of plant diseases and the diagnostic procedure. Many researchers, however, need an effective solution for plant disease identification. From the literature, it is found that no research work has attempted to detect the diseases of turmeric plants using an improved YOLO-V3 model. As other observations would seem to suggest, there is still a cost and time-effective solution for a turmeric plant. The major leaf diseases are leaf spot, leaf blotch, and rhizome rot are shown in Fig. 1.

This proposed work aims to achieve high detection speed and limited computational power with fewer memory resources. This study employed the YOLOV3-tiny model, a simplified version of the YOLO-V3 model that can be trained with the minimum GPU models and has the potential to be used in portable devices. For accurate detection, the residual network is used in the YOLOV3-tiny model which adds 1×1 and 3×3 convolutional layers which leads to an improved YOLOV3-tiny model (IY3TM).

The rest of the paper is organized as follows: Section 3 discusses the processing of datasets which includes data acquisi-

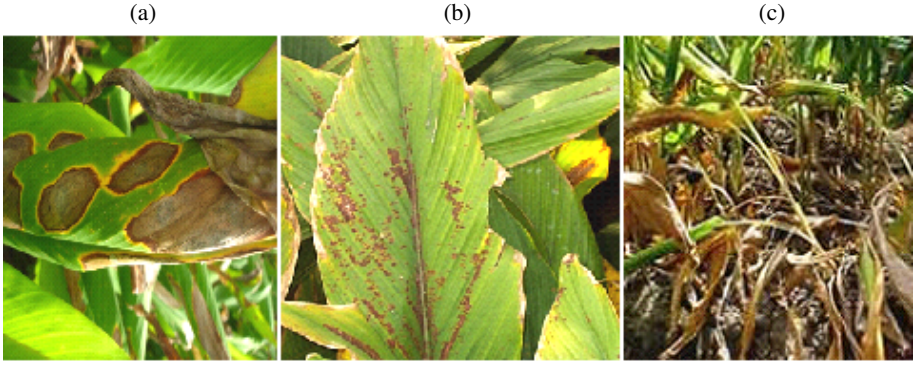

Fig. 1. (a) Leaf spot, (b) leaf blotch, and (c) rhizome rot diseases

tion and augmentation. Section 4 explains the proposed IY3TM model. Section 5 discusses the experimental results. The conclusion and future work of the proposed work are discussed in the last section.

3. DATASET CONSTRUCTION AND PRE-PROCESSING

3.1. Image acquisition

Image acquisition for this study was conducted using mobile devices with good resolution capacity during different periods of time. Images of turmeric leaves were randomly captured at four different times: morning, afternoon, evening, and on rainy days. The illumination conditions included back-lighting, front-lighting, side-lighting, and scattered lighting. A total of 1,600 images with 400 images for each category such as morning, afternoon, evening, and rainy days were consolidated. The viewing direction is positioned parallel to sunlight to simulate the front lighting. To get the back lighting, a device was set as antiparallel to sunlight. The perpendicular position was set to get the side lighting. To simulate scattered lighting, images are taken from cloudy conditions. This original image contains 4 class categories of leaves, namely, healthy leaves, leaf spot disease, leaf blotch disease, and rhizome rot disease. Each category contains 100 images for each class.

The overall dataset is divided into a raw training dataset of 80% of the original image and a testing dataset of 20% of the original image. Once the training model is completed, the user will test the model with the test set to predict and evaluate the performance.

3.2. Data augmentation

Deep learning methods need more datasets for accurate object detection. Sun *et al.* [24] stated that a huge dataset could improve the object detection accuracy rate and increase performance. In this work, 1,600 images are taken from a real-time environment. In order to expand the training image dataset, data augmentation is applied to artificially enlarge the dataset. As mentioned in [25], network performance can be improved by using data augmentation methods such as brightness transformation, image rotation, and histogram equalization. In this paper, traditional augmentation techniques like image rotation, image colour, image brightness transformation, and motion blur transformation are used. CycleGAN, the deep learning model for data augmentation is also used for finding better detection accuracy. The various augmentation methods used in this paper are discussed as follows:

Image rotation

Actual pictures are rotated 90° and 180° to expand the dataset. Mirroring of images is also done. The neural network performance can be improved by using a larger dataset with image rotation.

Image colour

The human vision ability to detect the colour invariance of the leaves under changing light conditions, but imaging technology lack this capability. Various lighting circumstances can result in a specific image and true colour divergence. To reduce the effect of light on colour rendering, the Gray world method is utilised. This approach is based on the gray-world hypothesis, which states that an image with a significant number of colour changes will have the same grey value if the average value of the components R, G, and B is the same. The grey world method implies that the average light reflection from objects is normally a fixed value, similar to grey. In the training set for colour invariance, this colour balance algorithm is applied to images.

Image brightness transformation

The brightness transformation is a typical way of data augmentation used to improve the robustness of a network. As indicated in equation (1), multiplying the proportionate coefficient near 1.0 with the original RGB image can result in increased or decreased image brightness. The systematic annotation is based on visual inspection of the diseased leaf outline on the image and labelling the disease using a rectangular frame. If the luminosity is too high or low, the bounding boxes are difficult to draw. In increments of 0.1, coefficients k of 0.7–0.9 and 1.1–1.3 are set based on the target edge which accurately defines the diseases during manual annotation.

$$g(x,y) = \begin{cases} f(x,y) \times k, & \text{if } g(x,y) < 255, \\ 255, & \text{if } g(x,y) \geq 255, \end{cases} \tag{1}$$

where: $f(x,y)$ indicates the values in RBG original image and $g(x,y)$ indicates the values in RGB image after brightness change. If the value of multiplication is greater than 255, then it is again adjusted to 255 automatically.

Motion blur transformation

The distanced camera can make incorrect focusing and movement of the camera may lead to blur image. To obtain the blur image transformation, equations (2) and (3) are used in this study. Parameters L (length reflects the linear motion of the camera pixels) and θ (theta is the angle between horizontal line and direction of the motion of the camera) are set based on the application type.

$$g(x,y) = h(x,y) * f(x,y), \tag{2}$$

$$h(x,y) = \begin{cases} \dfrac{1}{L}, & \sqrt{x^2+y^2} \leq L \text{ and } \dfrac{y}{x} = \tan\theta, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

where: $g(x,y)$ is a motion-blurred image, $h(x,y)$ is a degenerate function and $f(x,y)$ is the original image. Images in the training set are converted into YOLO format for comparison with different algorithms. The length and width of the images are adjusted based on the aspect ratio. Bounding boxes are drawn and the categories are classified. ImgLabel is used for labelling each image in the training dataset. In order to prevent over-fitting in deep learning algorithms, positive images with inadequate or undefined pixel areas are avoided.

Image augmentation using Cycle-GAN deep learning model

The generative adversarial network (GAN) can learn the characteristics of incoming data and create similar data. There are two types of models: generative and discriminative models. The generative model aims to generate samples that get closer and closer to the true samples (G). By capturing the distribution of the genuine sample data, the generator can produce a sample that is comparable to the original training data with a noise z that follows a predefined distribution such as uniform or Gaussian distribution. The discriminator (D) is a binary classifier that determines how likely a sample is drawn from training data rather than generated data. The discriminator will output a high probability if the sample belongs to the true training data, and a low probability if it does not. The generator and discriminator alternately enhance their networks during training until Nash equilibrium is obtained. The generator can now generate samples with the same distribution as the original data, and the discriminator can detect the created samples with a 50% accuracy. The resulting generative model is used to create required new data.

Cycle-GAN, which is applied in this paper, can extract and transfer image features to change one type of image into another. This will be extremely useful for transferring the characteristics of diseased turmeric leaf images to healthy turmeric leaf images in order to transfer healthy turmeric leaf images to diseased turmeric leaf images. The Cycle-GAN model is designed to employ a generator to turn samples in X into samples in Y. There are two sample spaces, X and Y. To avoid loss invalidation caused by G, the cycle consistency loss is presented, which maps all images in X to the same image in Y. The mapping function F is assumed to be capable of transforming the y image in Y into the $F(y)$ image in X in this method. Meanwhile, a discriminator D_x for F is introduced. The Cycle-GAN loss function model is as follows:

$$L(G,F,D_X,D_Y) = L_{GAN}(G,D_Y,X,Y) + L_{GAN}(F,D_X,X,Y) + \mu L_{\text{cyc}}(FmG), \quad (4)$$

where: $L_{GAN}(G,D_Y,X,Y)$ is the loss of G and D_Y, the loss of F and D_X is $L_{GAN}(F,D_X,X,Y)$ and the cycle consistency loss is $L_{\text{cyc}}(FmG)$. The Cycle-GAN model expectation is as follows:

$$G^*,\ F^* = \arg \min_{G,F} \max_{D_X,D_Y} L(GFD_XD_Y). \quad (5)$$

Cycle-GAN can learn the features of diseased turmeric leaf plants and healthy turmeric plants by training process and generating diseased spots on the surface of healthy turmeric leaves. This makes us generate new images even though we have less amount of diseased turmeric leaf plants. Table 1 provides a dataset description of turmeric plant leaf disease detection with image augmentation. Various augmentation techniques on turmeric leaves are shown in Fig. 2.

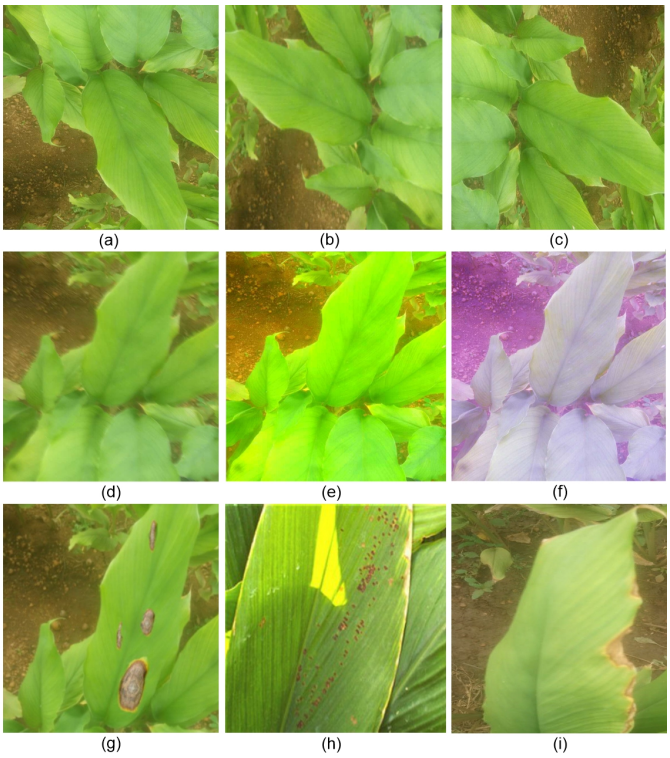

Fig. 2. (a) Original image, (b) rotated image (90°), (c) rotated image (180°), (d) motion blur transformation, (e) image brightness transformation, (f) image colour, (g, h, i) Cycle-GAN generated turmeric leaf diseases

4. THE PROPOSED ALGORITHM – IMPROVED YOLOV3-TINY MODEL

The proposed model is based on a convolutional neural network with a residual network structure. Turmeric leaves may have a smaller size of diseases on leaf surfaces. Going deeper

Table 1
Turmeric leaf diseases dataset after image augmentation

Images captured time	Original data	Colour	Brightness	Rotation (90°, 180°, mirrored)	Motion blur transformation	Cycle GAN	Total
Morning	320 (80 images for each class)	320	320	960	320	320	2,560
After noon	320 (80 images for each class)	320	320	960	320	320	2,560
Evening	320 (80 images for each class)	320	320	960	320	320	2,560
Rainy day	320 (80 images for each class)	320	320	960	320	320	2,560
						Total	10,240

into the convolutional networks leads to having more features extracted and finding the diseases very easily. However, if the network is too deep, feature information will be lost when it is transferred between layers. Therefore, this model adds residual network structures in the 4th, 6th, and 7th convolutional layers which extract more features from the targeted image. The residual network uses a 1×1 convolutional layer and a 3×3 convolutional layer for exacting features. As the name suggests, the 1×1 convolution layer convolves the image with a 1×1 filter size with zero padding and stride of 1. Likewise, a 3×3 convolutional layer uses a 3×3 filter size. Going deeper into the network will increase the activation maps generation. The problem is further exacerbated when large-sized filters such as 5×5 or 7×7 filters are used in the convolution procedure, resulting in a huge number of parameters. That is why 1×1 and 3×3 convolutional layers are used. When we use lower convolutional layers, it slightly reduces the detection time than others. The feature map formed before inputting the structure is combined with the feature map obtained after the residual structure, and the shallow and deep information is sent to the next convolution layer at the same time to extract features. As a result, feature information loss when transferring between layers will be reduced, and network detection accuracy will improve. The structure of IY3TM is shown in Fig. 3.

YOLO model loss function is the combination of three parts: intersection over union (IoU) error, coordinate prediction error, and classification error. The loss function in YOLO is defined as follows:

$$\text{Loss} = \text{Error}_{\text{coord}} + \text{Error}_{\text{IOU}} + \text{Error}_{\text{cls}}\,, \tag{6}$$

where: $\text{Error}_{\text{coord}}$ (coordinate prediction error) is defined as follows:

$$\begin{aligned}\text{Error}_{\text{coord}} = \lambda_{\text{coord}} \sum_{i=1}^{S^2} \sum_{j=1}^{B} 1_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2\right] \\ + \lambda_{\text{coord}} \sum_{i=1}^{S^2} \sum_{j=1}^{B} 1_{ij}^{\text{obj}} \left[(w_i - \hat{w}_i)^2 + \left(h_i - \hat{h}_i\right)^2\right].\end{aligned} \tag{7}$$

The weight of the coordinate error is λ_{coord}, the number of grids in the image is S^2 and B is the count of bounding boxes produced by each grid. $\lambda_{\text{coord}} = 5$, $S = 7$, and $B = 9$ are chosen for the evaluation process based on the YOLO-V3 model initial parameters. $1_{ij}^{\text{obj}} = 1$ denotes that the object falls into the j_{th} bounding boxes in grid i, otherwise $1_{ij}^{\text{obj}} = 0$. The values ($\hat{x}_i$, $\hat{y}_i$, $\hat{w}_i$, $\hat{h}_i$) represent center coordinate, height, and width of the predicted bounding box and (x_i, y_i, w_i, h_i) are actual values.

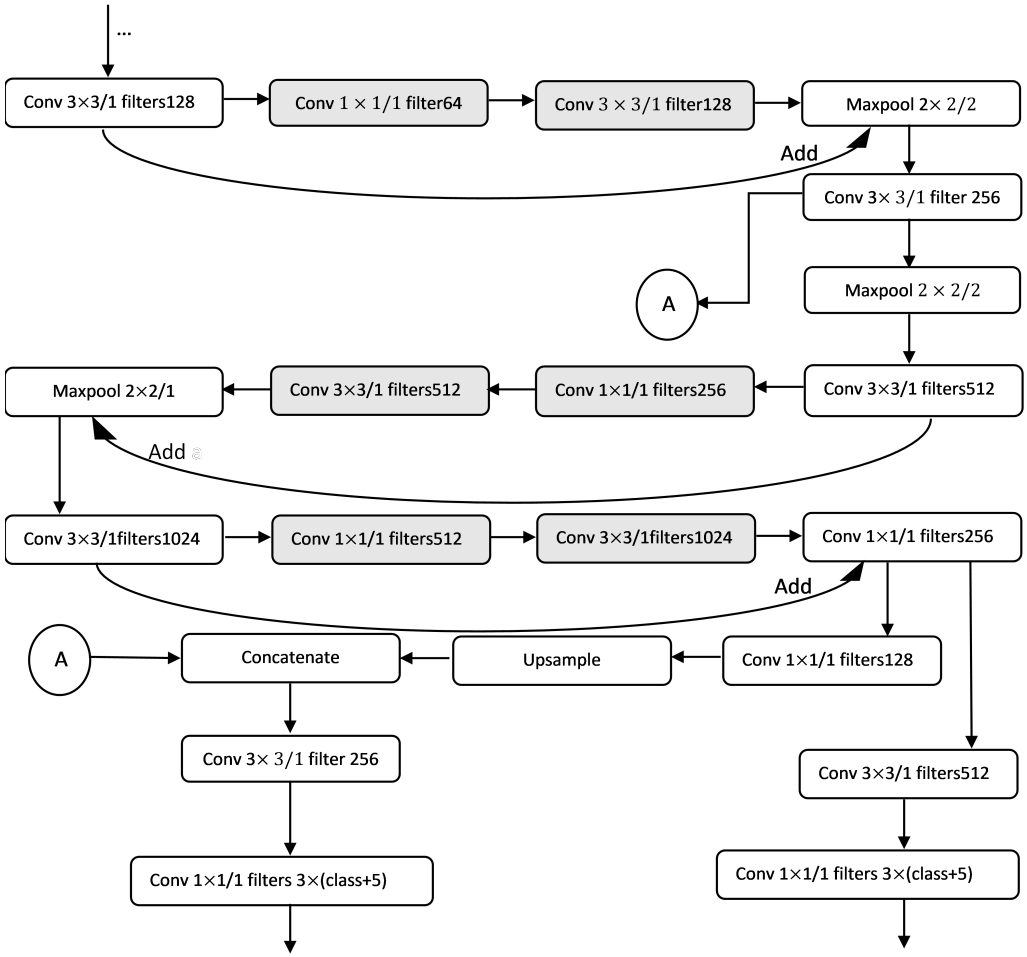

Fig. 3. Structure of IYMT3 model

The $\text{Error}_{\text{IOU}}$ (The IOU Error) is defined in equation (8) as follows:

$$\text{Error}_{\text{IOU}} = \sum_{i=1}^{S^2}\sum_{j=1}^{B} 1_{ij}^{\text{obj}} \left(C_i - \hat{C}_i\right)^2 + \lambda_{\text{noobj}} \sum_{i=1}^{S^2}\sum_{j=1}^{B} 1_{ij}^{\text{obj}} \left(C_i - \hat{C}_i\right)^2, \tag{8}$$

where: λ_{noobj} is the weight of IoU loss and its value is selected as 0.5 as the original parameters of the YOLO-V3 model. C_i is the actual confidence value and $\hat{C}_i$ is the predicted confidence.

$\text{Error}_{\text{cls}}$ (Classification error) is defined in equation (9) as follows:

$$\text{Error}_{\text{cls}} = \sum_{i=1}^{S^2}\sum_{j=1}^{B} \sum_{c\in\text{classes}} \left(p_i(c) - \hat{p}_i(c)\right)^2, \tag{9}$$

where: c represents the class of detected plant diseases. $p_i(c)$ refers to the actual probability of object in class c of grid i. $\hat{p}_i(c)$ refers to the actual value. Even though YOLO has a higher detection speed, it still has a smaller margin of error. To address this issue, YOLOV3 has an anchor box and K-means clustering algorithms for generating appropriate prior bounds.

The improved YOLOV3-tiny model (IY3TM) is modified using the darknet framework. The training platform is a laptop with an intel i3 quad-core CPU and 4GB of memory running on a Windows 10 with a 64-bit system. Images and corresponding labeled images are the set of data required for training the model. The labeled data contains normalized center coordinates, object type, height, and weight of the bounding box.

Considering the GPU memory limitation during the training process, the batch size was set to 8 and the network momentum was set to 0.9 with a weight decay of 0.0005 as per the original parameters used in the YOLO-V3 network [12]. For all layers of the network, a learning rate of 0.001 is applied. 10,000 training steps are used for better analysis. The initialization parameters of IY3TM are shown in Table 2. After 50,000 steps, the learning rate is reduced to 0.0001. Different turmeric leaf diseases are trained using the IY3TM model for images present in the training dataset. In this paper, to verify the performance of the algorithm, various experiments are conducted with the testing image set resolution of 512×512 pixels which is the same as the pixels value of the training dataset.

Table 2
Initialization parameters of IY3TM

Input image size	Momentum	Batch	Initial learning rate	Training steps	Decay
512×512	0.9	8	0.001	10,000	0.0005

5. EXPERIMENTATION AND DISCUSSION

The performance of the model is evaluated by precision (P), recall (R), detection speed, loss function, F1 score, and AUC (area under the curve) measures and are given as follows:

For binary classification problems, samples can be split into true positive (TP), false positive (FP), true negative (TN), and false negative (FN) based on the combinations of actual class and predicted class. For the classification outcomes, precision (P) and recall (R) are defined in equations (10) and (11), respectively

$$\text{P} = \text{TP}/(\text{TP}+\text{FP}), \tag{10}$$

$$\text{R} = \text{TP}/(\text{TP}+\text{FN}), \tag{11}$$

where: TP, FP, and FN refer to the number of plant diseases correctly identified, the number of plant diseases wrongly identified and the number of plant diseases missed, respectively. The P-R curve can be obtained using the precision ratio of the vertical axis and the recall ratio of the horizontal axis. AUC measures the sensitivity of the network which is defined in equation (12) as the area under the P and R curve. It indicates the global performance of the network

$$\text{AUC} = \int_0^1 \text{P}_\text{R}\,\text{dR}. \tag{12}$$

F1 score used for the model evaluation is given in equation (13)

$$\text{F1} = 2\cdot\text{P}\cdot\text{R}/\text{P}+\text{R}. \tag{13}$$

IoU is a parameter that evaluates the bounding box prediction precision. The overlap ratio between predicted and actual boundaries is used by IoU to verify detection performance. We compare the average detection time and evaluate the real-time performance of several models in this paper.

5.1. Test results using mix-up training and transfer learning

Zhang *et al.* [26] have introduced the mix-up training technique that generates new samples by mixing two different images of certain weights, which improves network generalization. The training model based on this method is stable and effective at minimizing the incidence of image variations. The traditional architectures do not use any pre-trained models and train all the parameters from the beginning stages. Transfer learning uses a pre-trained dataset for training some portion of the layer parameters of the proposed model. The transfer learning method significantly affects the detection speed compared with a traditional method. This study states that the training time is reduced when mix-up training data and transfer learning methods are combined. The detection results of different kinds of training methods are shown in Table 3. The F1 score of the mix-up

Table 3
Comparison of results using different training methods

Training methods	F1 score	AUC value	IoU
Original training method	0.8872	0.8756	0.8047
Transfer learning method	0.9162	0.8845	0.8154
Mix-up with transfer learning method	0.9234	0.9023	0.8262

with the transfer learning method is improved by 0.0362, the AUC value improved by 0.0267, and the IoU value improved by 0.0215 from the original training method.

5.2. Influence of Cycle-GAN augmentation method

To get better detection results, various augmentation techniques like colour, angle transformation, brightness, and Cycle-GAN deep learning methods are adopted for improving the quality of the dataset. For evaluating the impact of augmentation techniques on the IY3TM model, the F1 score and IoU values of complete datasets with each augmentation method are calculated and compared with that of the algorithm implementations without including the augmentation method and the results are shown in Table 4.

From the findings of the above experiments, the F1 score and IoU values are greatly increased when the Cycle-GAN method is included. It makes clear that the image data created by Cycle-GAN is important in enriching the training dataset diversity. Cycle-GAN generates samples containing diseases with different backgrounds, textures, and sizes which help to improve the consistency of the detection model when compared to traditional methods used for color, brightness, and angle transformation.

When a dataset is processed with Cycle-GAN and other augmentation methods, the F1 value is increased to 0.834 while the normal dataset without augmentation has a lesser F1 score. This experiment shows that using augmentation methods will increase the detection accuracy by 8.4%. More than 4% of detection accuracy is improved by using the Cycle-GAN augmentation technique.

5.3. Influence of data category

The IY3TM neural network is used to train images for the impact of dataset categories: morning, afternoon, evening, and rainy day. They also combined the images of all categories and used them to train the model. The P-R curves of the models using the training methods are shown in Fig. 5. The F1 scores are shown in Table 5 of the corresponding categories. Based on the results, the F1 score of the morning image dataset category has higher detection accuracy than other categories.

This indicates that the lighting effect of the sun also affects the detection ability of the model. From this result, collecting a real-time dataset for turmeric leaf diseases detection during the morning time is highly recommended for better results compared with other conditions.

Table 5
F1 score of different categories of turmeric leaf images

Categories	Morning	Afternoon	Evening	Rainy-day	Mix-up image
F1 score	0.8504	0.8323	0.8284	0.8012	0.8356

5.4. Comparison of different deep learning methods with IY3TM model

The model is compared with various versions of YOLO models and with Faster R-CNN with VGG16 net. Figure 4 illustrates the loss of all YOLO-based models. Figure 5 shows the P-R curves for all the models outlined previously. Figures 6–8 demonstrate the detection results for all models in various diseases. The IoU, F1 scores, and average detection time of the comparison models are shown in Table 6.

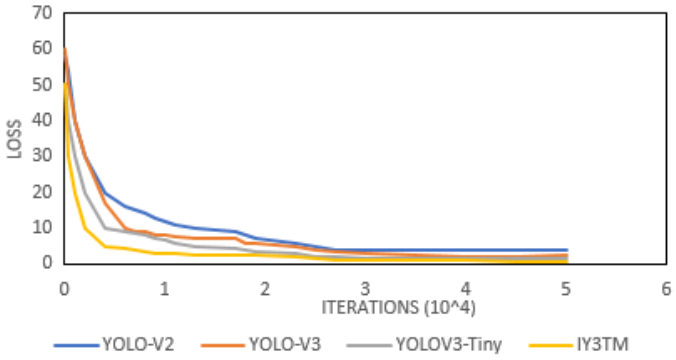

Fig. 4. Loss curves of the four YOLO models

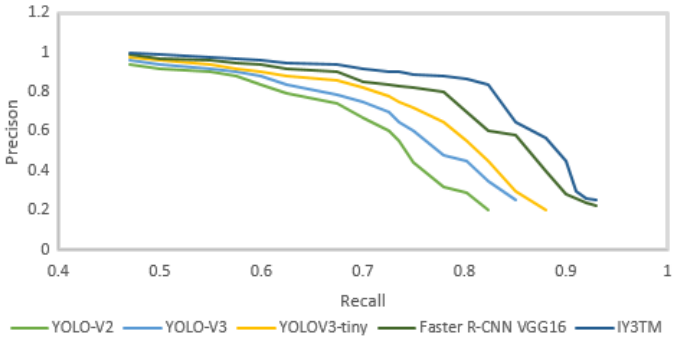

Fig. 5. *P-R* curves for detection deep learning models

Table 4
Influence of Cycle-GAN augmentation method

Categories	Complete dataset without augmentation methods	With image rotation only	With image color method	With image brightness transformation	With motion blur transformation	With Cycle-GAN augmentation method	Complete dataset with all augmentation methods
F1 Score	0.745	0.756	0.769	0.754	0.785	0.796	0.834
IoU	0.846	0.886	0.894	0.875	0.891	0.895	0.905

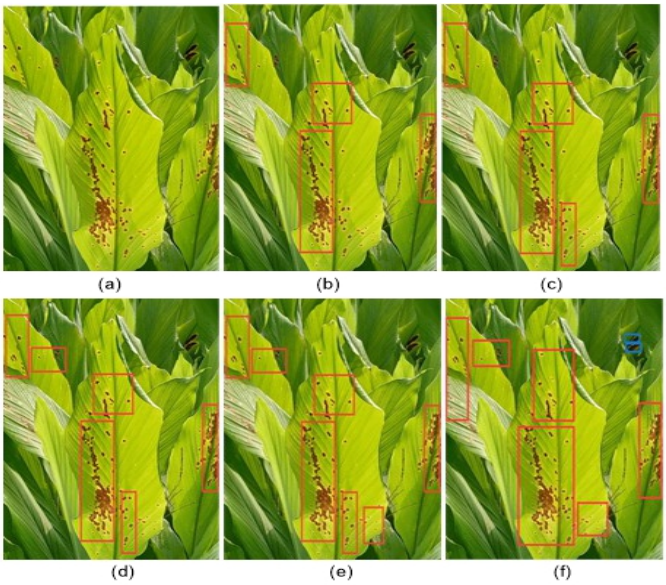

Fig. 6. (a) image of leaf blotch disease, (b) detection results of YOLO-V2, (c) YOLO-V3, (d) YOLOV3-tiny, (e) faster RCNN with VGG16 (f) IY3TM method

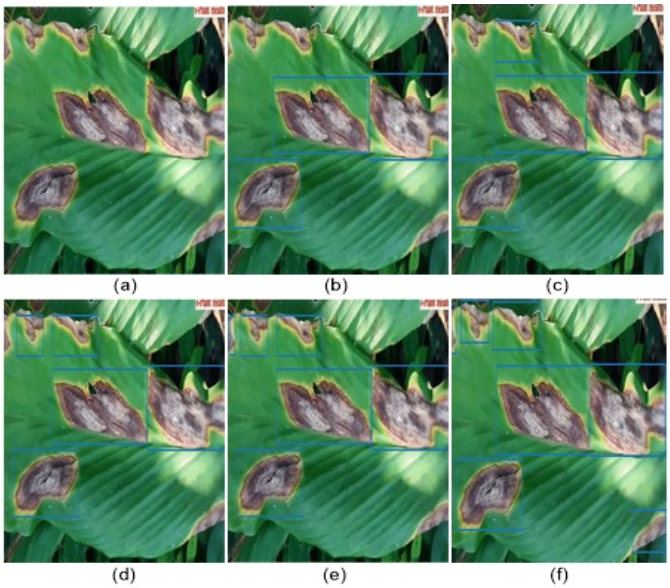

Fig. 7. (a) Image of leaf spot, (b) detection results of YOLO-V2, (c) YOLO-V3, (d) YOLOV3-tiny, (e) faster RCNN with VGG16 (f) IY3TM method

Table 6
F1 scores, IoU, and average detection time for all models

Models	F1 score	IoU	Average detection time (s)
YOLO-V2	0.745	0.807	2.18
YOLO-V3	0.786	0.835	2.15
YOLOV3-Tiny	0.821	0.864	2.24
Faster R-CNN with VGG16	0.801	0.875	2.89
IY3TM	0.834	0.905	2.21

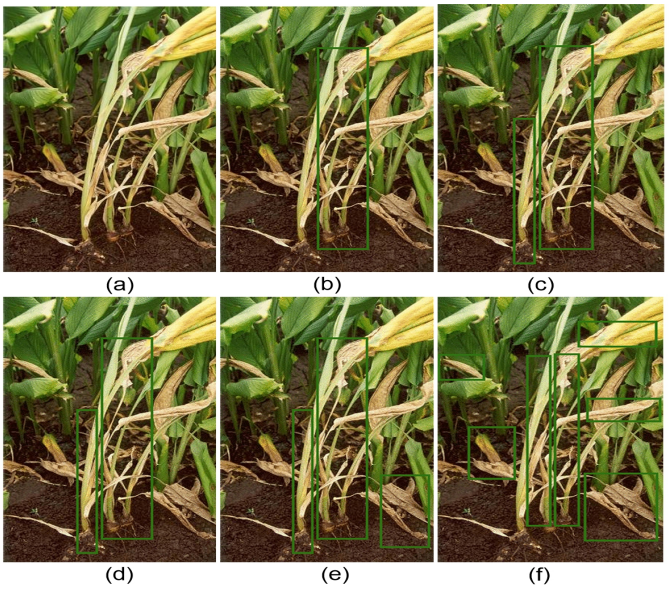

Fig. 8. (a) Rhizome rot disease (b) Detection results of YOLO-V2; (c) YOLO-V3 (d) YOLOV3-tiny (e) Faster RCNN with VGG16 (f) IY3TM method

5.5. Influence of the quantity of experimental data

In this paper, the size of the dataset for IY3TM is analyzed. 10, 50, 100, 200, 400, 600, 800, and 1200 images are selected randomly from the turmeric leaf diseases dataset of each category of an image captured at four times (morning, afternoon, evening, and on rainy days) and the dataset of 50, 250, 500, 1000, 2000, 3000, 4000 and 6000 images are formed. From these experimental results, one can conclude that the size of the training set increases the performance accuracy of the proposed model. The performance of the model is increased rapidly until 1800 images. After that, the speed of enhancement is slightly reduced and beyond that value, the training dataset does not have any significant improvement. The diagrammatic view of F1 scores for different sizes of datasets is shown in Fig. 9. An accuracy curve for the proposed model is given in Fig. 10.

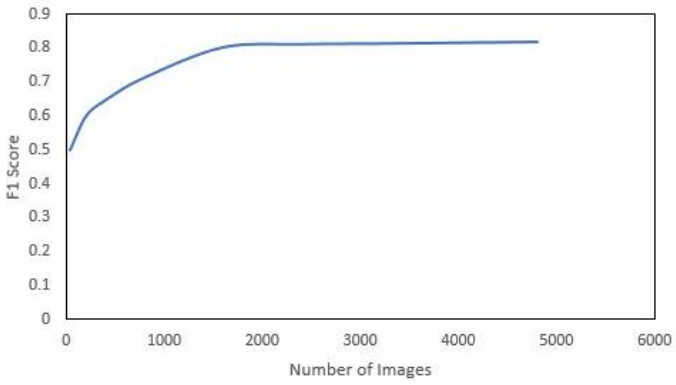

Fig. 9. *F*1 scores for different sizes of datasets

Based on the information provided above, it is understood that the IY3TM model has higher convergence and faster speed than other YOLO models. The resultant loss in the YOLO-V2

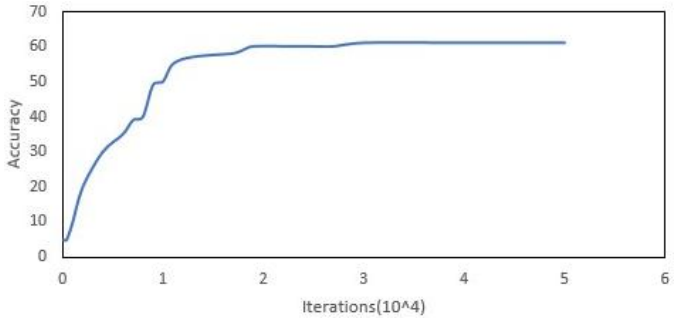

Fig. 10. Accuracy curve for IY3TM model

model is approximately 4.05, while YOLO-V3 and YOLOV3-tiny have a loss value around 2.05 and 1.67 respectively. The loss in the IY3TM model is about 0.85 which is 0.82 lower than the YOLO-tiny model. Based on this observation, it concludes that the performance of the IY3TM model is improved significantly. All other models began to saturate after 3500 iterations but the IY3TM model continues up to 5000 iterations, after that there is no decrease in convergence. In detection accuracy, the improved YOLOV3-tiny model has higher accuracy detection than the other 4 models.

The $F1$ score and IoU value of the proposed model are 0.834 and 0.905, respectively, which is greater than other models. This shows that the detection of bounding boxes for turmeric leaf diseases detection is higher for the proposed model compared with other models. The average detection time of the proposed model is 2.21s which is approximately the same as other YOLO models and lesser than the faster R-CNN VGG16 model. Moreover, all the above experiments show that the detection accuracy of the IY3TM model is higher when compared with the other four models for the turmeric leaf diseases detection dataset.

6. CONCLUSIONS

Depending on the features of the turmeric leaf dataset, two convolutional layers of 3×3 and 1×1 are included to improve the YOLOV3-Tiny model. The IY3TM model enhances feature propagation, improving network performance and promoting feature reuse. Images are captured at different conditions and augmented to test the proposed model. The proposed IY3TM model when compared with other deep learning models like YOLO-V2, YOLO-V3, YOLOV3-tiny, and faster RCNN with VGG-16, it is observed that the IY3TM model has better performance than other object detection models. The F1 score, detection speed, and the AUC values are used for evaluation for these models on turmeric leaf diseases detection. For the same datasets, the F1 value of the IY3TM model with Cycle-GAN augmentation is higher than other models. Overall, the experimental results state that the IY3TM model performs much better than other models for turmeric leaf diseases. In the future, the work can be extended for other vital diseases in turmeric leaf and other important crops.

REFERENCES

[1] A.-K. Mahlein, "Plant Disease Detection by Imaging Sensors – Parallels and Specific Demands for Precision Agriculture and Plant Phenotyping", *Plant Disease*, vol. 100, no. 2, pp. 241–251, Sep. 2015, doi: 10.1094/PDIS-03-15-0340-FE.

[2] V. Puri, A. Nayyar, and L. Raja, "Agriculture drones: A modern breakthrough in precision agriculture", *J. Stat. Manage. Syst.*, vol. 20, no. 4, pp. 507–518, 2017.

[3] V.O. Ayodele, O.M. Olowe, C.G. Afolabi, and I.A. Kehinde, "Identification, assessment of diseases and agronomic parameters of Curcuma amada Roxb (Mango ginger)", *Curr. Plant Biol.*, vol. 15, pp. 51–57, Nov. 2018, doi: 10.1016/j.cpb.2018.10.001.

[4] J.G. Barbedo, "Factors influencing the use of deep learning for plant disease recognition", *Biosyst. Eng.*, vol. 172, pp. 84–91, 2018.

[5] C. Szegedy *et al.*, "Going Deeper With Convolutions", *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 19, doi: 10.1109/CVPR.2015.7298594.

[6] S. Xie, R. Girshick, P. Dollár, Z. Tu and K. He, "Aggregated Residual Transformations for Deep Neural Networks", in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 59875995, doi: 10.1109/CVPR.2017.634.

[7] K. O'Shea and R. Nash, "CAn Introduction to Convolutional Neural Networks", *arXiv:1511.08458 [cs]*, Dec. 2015. [Online]. Available: http://arxiv.org/abs/1511.08458 (Accessed Dec. 30, 2020).

[8] S. Albawi, T.A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network", in *2017 International Conference on Engineering and Technology (ICET)*, Aug. 2017, pp. 1–6. doi: 10.1109/ICEngTechnol.2017.8308186.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580587, doi: 10.1109/CVPR.2014.81.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015, doi: 10.1109/TPAMI.2015.2389824.

[11] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection", *arXiv:1506.02640 [cs]*, May 2016. [Online]. Available: http://arxiv.org/abs/1506.02640 (Accessed: Jul. 12, 2021).

[13] W. Liu *et al.*, "SSD: Single Shot MultiBox Detector", in *Computer Vision – ECCV 2016*, Cham, 2016, pp. 21–37, doi: 10.1007/978-3-319-46448-0_2.

[14] R.N. Jogekar and N. Tiwari, "A review of deep learning techniques for identification and diagnosis of plant leaf disease", *Smart Trends in Computing and Communications: Proceedings of SmartCom 2020*, 2020, pp. 435–441.

[15] L.K. Mehra, C. Cowger, K. Gross, and P.S. Ojiambo, "Predicting pre-planting risk of Stagonospora nodorum blotch in winter wheat using machine learning models", *Front. Plant Sci.*, vol. 7, p. 390, 2016.

[16] K.P. Ferentinos, "Deep learning models for plant disease detection and diagnosis", *Comput. Electron. Agric.*, vol. 145, pp. 311–318, 2018.

[17] S. Huang, W. Liu, F. Qi, and K. Yang, "Development and Validation of a Deep Learning Algorithm for the Recognition of Plant Disease", in *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, 2019, pp. 19511957, doi: 10.1109/HPCC/SmartCity/DSS.2019.00269.

[18] D. Argüeso *et al.*, "Few-Shot Learning approach for plant disease classification using images taken in the field", *Comput. Electron. Agric.*, vol. 175, p. 105542, 2020.

[19] R. Sujatha, J.M. Chatterjee, N.Z. Jhanjhi, and S.N. Brohi, "Performance of Deep Learning vs Machine Learning in Plant Leaf Disease Detection", *Microprocess. Microsyst.*, vol. 80, p. 103615, 2021.

[20] H.K. Mewada and J.J. Patoliya, "IoT based Automated Plant Disease Classification using Support Vector Machine", *Int. J. Electron. Telecommun.*, vol. 67, no. 3, pp. 517–522, 2021.

[21] G. Kuricheti and P. Supriya, "Computer Vision Based Turmeric Leaf Disease Detection and Classification: A Step to Smart Agriculture", in *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, 2019, pp. 545–549.

[22] C. Rajasekaran, S. Arul, S. Devi, G. Gowtham, and S. Jeyaram, "Turmeric Plant Diseases Detection and Classification using Artificial Intelligence", in *2020 International Conference on Communication and Signal Processing (ICCSP)*, 2020, pp. 1335–1339.

[23] J. Liu and X. Wang, "Tomato diseases and pests detection based on improved Yolo V3 convolutional neural network", *Front. Plant Sci.*, vol. 11, p. 898, 2020.

[24] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era", in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 843–852.

[25] Y. Tian, G. Yang, Z. Wang, H. Wang, E. Li, and Z. Liang, "Apple detection during different growth stages in orchards using the improved YOLO-V3 model", *Comput. Electron. Agric.*, vol. 157, pp. 417–426, 2019.

[26] H. Zhang, M. Cisse, Y.N. Dauphin, and D. Lopez-Paz, "mixup: Beyond Empirical Risk Minimization", *International Conference on Learning Representations*, Feb. 2018. [Online]. Available: https://openreview.net/forum?id=r1Ddp1-Rb (Accessed: Sep. 28, 2021).