Deep Learning Based Plant Disease Detection for Smart Agriculture

Conference Paper · December 2019

DOI: 10.1109/GCWkshps45667.2019.9024439

5 authors, including:

Laha Ale
Harvard University
12 PUBLICATIONS **288** CITATIONS
SEE PROFILE

Alaa Sheta
Southern Connecticut State University
198 PUBLICATIONS **2,837** CITATIONS
SEE PROFILE

Ning Zhang
University of Windsor
404 PUBLICATIONS **8,995** CITATIONS
SEE PROFILE

Deep Learning based Plant Disease Detection for Smart Agriculture

Laha Ale*, Alaa Sheta†, Longzhuang Li*, Ye Wang‡, Ning Zhang*
*Texas A&M University-Corpus Christi, Corpus Christi, TX, USA
†Computer Science Department, Southern Connecticut State University, CT, USA
‡Harbin Institute of Technology (Shenzhen), Shenzhen, China

***Abstract*—Deep learning is a promising approach for fine-grained disease severity classification for smart agriculture, as it avoids the labor-intensive feature engineering and segmentation-based threshold. In this work, we first propose a Densely Connected Convolutional Networks (DenseNet) based transfer learning method to detect the plant diseases, which expects to run on edge servers with augmented computing resources. Then, we propose a lightweight Deep Neural Networks (DNN) approach that can run on Internet of Things (IoT) devices with constrained resources. To reduce the size and computation cost of the model, we further simplify the DNN model and reduce the size of input sizes. The proposed models are trained with different image sizes to find the appropriate size of the input images. Experiment results are provided to evaluate the performance of the proposed models based on real-world dataset, which demonstrate the proposed models can accurately detect plant disease using low computational resources.**

***Index Terms*—Plants Disease, Corp Disease, Deep learning, Detection, Digital Epidemiology, Convolutional Neural Network, Classification**

I. Introduction

The food supply is essential for everyone's daily lives. We are increasingly aware that polluted and pathogenic plants are responsible for a large proportion of food poisoning by pathogenic microorganisms. With the advance of technologies including genetic engineering [1], fertilizer and agricultural machinery [2], yields of corps have been increased tremendously over the decades. However, food security is still a significant threat to humankind. The food security is more vulnerable in underdeveloped countries, and a large number of the population threatened by starvation. Corp diseases are the main culprit that farmers fail to harvest and threaten food security. Plant disease devouring about 10% of global food production, whereas a significantly large population is living without enough food [3]. We are witnessing a significant fraction of the population live with undernourished, and it had been increased to about 821 million in 2017, from around 804 million in 2016. The situations are even worse in South America and most countries of Africa.

A. Why We Need Detection Methods?

Plant disease is one of the significant threats to food security. Plant diseases not only affect agriculture and population growth but also affect the economy. Automatic and accurate estimation of plant disease severity is essential to food safety, disease management, and predicting loss of return. This encourages a demand for the detection, identification, and treatment of plant diseases at an early stage, which requires an understanding of the causes of plant diseases as well as the introduction of new methods of disease identification. However, non-expert farmers are usually unaware of non-native diseases; therefore, they need to contact experts to find out any abnormal appearance or symptoms on plants. In some developing countries, farmers may need to travel long distances to contact experts, which can be very costly and time-consuming. These issues stimulate research and expansion to automate the process of diagnosing plant diseases. A plant disease detection model is urgently needed, which can run on Internet of Things (IoT) devices such as smartphones and other mobile devices with limited computing capabilities.

B. Challenges of Plant Disease Detection

To develop a robust plant disease detection system, we need a large number of sample images of corps that have diseases. It was challenging to collect such a large number of sample in the past. Fortunately, almost every person has a smartphone equipped with a high-resolution camera, and smartphones can be used to collect and detect corp diseases. Plantvillage dataset [4] published a large number of images of corps with diseases. The dataset is well-labeled and adopted in many plant disease detection research. In addition, the farmers demand a lightweights detection system can run on their smartphones and detect plant diseases with a and eliminate the disease at the early stage so that they can increase yields.

Although image processing methods cannot classify and label disease, it can facilitate plant disease researchers and farmers to identify the plant diseases more accurately. Besides, image processing methods adopted for disease detection can achieve acceptable performance, which however require human further detection and analysis [5]. Moreover, the performance using traditional image processing methods is limited.

In this paper, we propose deep learning models for plant disease detection. First, we propose a DenseNet [6] based transfer learning approach for plant disease detection,

978-1-7281-0960-2/19/$31.00 ©2019 IEEE

where we adopt DenseNet as the backbone and add a flatten and two fully-connected layers on DenseNet121 with 4 dense blocks and input size $256 \times 256 \times 3$. To further reduce the size or model, we propose a lightweight deep learning model that can run IoT devices with limited computing capacities. Moreover, we also investigate the model performance with respect to input sizes so that appropriate input size for different application scenarios can be selected. Experiment results are provided to evaluate the performance of the proposed models, based on a real-world dataset offered by [7].

The rest of the paper is organized as follows: Section II, a literature review about previous work is provided. Section III, presents the proposed deep learning models. Section IV provides the experiments setting and the experimental results. Finally, conclusion and future works are presented in section V.

II. Related Work

Image processing methods [5] have been adopted to solve the plant disease detection problem to help researcher, farmers, and other agricultural workers to identify diseases. Image preprocessing can be considered as a feature maps extract as presented in [8], where image preprocessing is adopted to segment the images and exact feature for a simple neural network and classify the plant diseases. A method with HSI (hue, saturation, intensity) is proposed in [9] to help farmers to recognize some diseases that they cannot distinct before processing. A texture statistics extraction process is presented in [10], where the authors mask the green-pixels, segment the images based on HSI, and then extract the texture of the plant diseases. The work in [11] focuses on the health color of the plant, and uses Otsu's method to compute value to mask the image, and removes the rest of values in the image to filter out the features of leaves diseases. However, image processing methods often require human efforts to diagnose plant diseases further since the methods cannot classify the disease automatically.

As one of the most efficient patterns recognition method, deep learning models have revolutionized in many computer version and artificial intelligence. Deep learning methods have been applied in many areas [12], [13], especially in object recognition and classification. Plant disease detection from crop leaves tasks is generally categorized into the computer vision tasks. We can input images or video streams of crop leaves to the system, which outputs the detected diseases. Therefore, it is natural to adopted deep neural network models to plant disease detection. A transfer deep learning approach to detect plant disease detection is presented in [7], and the authors introduce AlexNet![14], VGG [15], GoogleNet [16] and ResNet [17], [18] based models, which achieve considerably high prediction accuracy. Though transfer learning models can reduce the training effort and detect the plant disease, however with high computational cost incurred by high complexity architecture and additional features in the pre-trained models. On the other hand, training state of the art deep learning models such Inception-ResNet [19], DenseNet [6] and RetinaNet [20] from scratch is with high computational cost, and models may have the issues of over-fitting or fail to converge to the optimal due to insufficient training data. Therefore, a deep learning model with low computational cost is demanded in many application scenarios, which can be applicable to devices without much computational capacities, such as mobile devices and embedded system.

Moreover, the input image size has a significant impact on the computational cost. Image qualities and size vary from one dataset to another, depending on the methods to collect dataset. For example, the images captured by a drone fly over crops, professional cameras, and smartphones have significantly different image qualities. The input images have to resize before being fed to any deep learning models; the resized settings have substantial impact on model size and computational cost as well as its prediction performance, which needs to be carefully investigated as well.

III. Proposed Deep Learning Models

Deep learning methods were proposed to mimic human brains that can learn from examples and address the unprecedented problems. It became one of the hottest topics in computer and cognitive science research communities. Deep learning models are machine learning models construct by deep artificial neural networks (DNN) which originally inspired by neural in the human brain. Specifically, DNN models often consist of CNN layers, pooling layers, and fully connected neural network layers. Each CNN and NN layer contain numbers of neural nodes with its active functions and connected other nodes previous and follow layers. The training process is to adjust the parameters in all of neural to fit the data. The DNN models can achieve human-level patter recognition after well-trained. In this section, first, we will present and transfer learning model base on DenseNet [6] that can detect plant disease in a quite high accuracy such that can run on mobile edges and devices with relatively large appliances with more computation resources than a regular IoT device. And then, we proposed a light DNN model from scratch to detect plant diseases; and it can run on various IoT devices with constrained resources.

A. DenseNet Transfer Learning Model

In this subsection, we propose a DenseNet [6] based transfer learning approach for plant disease detection. We adopt DenseNet as the backbone of the transfer learning. With a unique design that each layer outputs to all the rest follow layers in the Dense Block, it is considered as a small network itself. The n^{th} layer in a Dense Block connected to all its previous layer in the Dense Block, as shown in Fig.1, specifically the n^{th} layer has connected to $\frac{n(n+1)}{2}$

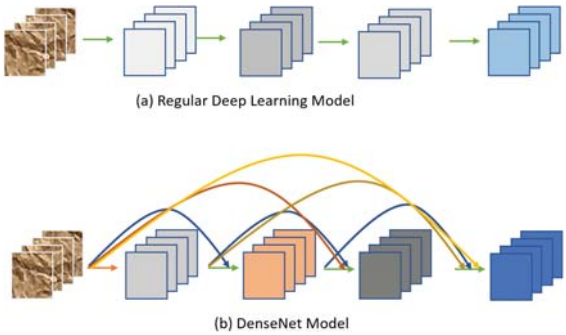

Fig. 1. Regular DNN and DenseNet

layers. The gradient decay issue has been easily solved as it connects all the layer to the last layer. Also, the layers which are away from the input layer still have a sense of the original input and feature maps extracted by all the previous layers that improve the prediction accuracy. Intuitively, it seems the DenseNet is extremely heavy to train and run the model to predict since all the layers are densely connected. However, the authors argue that dense connectivity pattern requires fewer parameters than the regular CNN because it does not need to relearn repetitive feature maps.

In this transfer learning model, we add a flatten and two fully-connected layers on DenseNet121 with 4 dense blocks and input size $256 \times 256 \times 3$. As a result, we have a total of $7,148,166$ parameters in the model. The total parameters of the proposed model is less than that of the other classical deep learning model, such as ResNet152 [17] ($60,419,944$ parameters), InceptionV3 [16] ($23,851,784$ parameters) and InceptionResNetV2 [19] ($55,873,736$ parameters). It can reach almost the same accuracy. The weights of the backbone of this model are initialized with ImageNet [21], as a transfer learning model. The l^{th} layer would receive concatenated of all of the feature maps of preceding layers, as below,

$$\mathbf{f}_\ell = C_\ell\left([\mathbf{f}_0, \mathbf{f}_1, \ldots, \mathbf{f}_{\ell-1}]\right) \tag{1}$$

where f_i denotes the feature maps from i^{th} layer in the same Dense Block. The backbone parameters will converge to the plant diseases detection as training on the target dataset, and extract the feature maps so that the added fully-connected layers can learn from the outputs feature and classify the plant diseases.

Indeed, classical deep learning models can detect plant diseases accurately than small models. However, those models can hardly run on mobile devices due to the high complexity and computational cost. Therefore, farmers may have to access the cloud server for using the models, which can be costly or even infeasible due to poor/no internet connections. The proposed model is relatively smaller than the other classical deep learning models and can be deployed on mobile edges or powerful IoT devices [22], which can mitigate the aforementioned issue. Further, we would like to propose a light model that can run on most mobile devices so that users can use the detection system anytime and anywhere.

B. Lightweight Deep Learning Model

In this subsection, we further propose a light deep learning model that can run IoT devices. There are two typical ways to make a DNN model smaller. First, we can simplify the DNN model architecture by reducing the number of layers and neurons in each layer. Second, the input size has a significant impact on the model and the number of parameters to train in the model; therefore, we investigate the model performance with respect to input sizes so that appropriate input size for different application scenarios can be selected. Suppose that we have an input size $W \times H \times M$, and we set both the stride and padding size as one, a regular CNN operation can be computed as follows:

$$\mathbf{O}_{k,l,n} = \sum_{i,j,m} \mathbf{S}_{i,j,m,n} \cdot \mathbf{F}_{k+i-1,l+j-1,m} \tag{2}$$

where $\mathbf{O}_{k,l,n}$ is output with size $W \times H \times N$, the kernel S with size $W_k \times H_k \times M \times N$. If we have another input size $W' \times H'$ where $W < W'$ and $H < H'$, we can find the cost of a CNN layer that connects to the input layers is quite different, and the input $W' \times H'$ raises much more computation cost than $W \times H$, as shown below:

$$\mathbf{O}_{k,l,n} = \sum_{i,j,m} \mathbf{S}_{i,j,m,n} \cdot \mathbf{F}_{k+i-1,l+j-1,m} \tag{3}$$

More importantly, the input sizes affect all the layers in the model as the matrixes computation goes through all the layers in the model.

In this work, we propose a lightweight model, as shown in Figure 2. The proposed architecture has eight layers of the backbone network and an input layer. Specifically, we have an input layer to take the raw feature data, five CNN layers to extract the feature maps, with two fully-connected layers along with a flatten layer to perform final classification based on the extracted feature map. The main parameter settings are also given in Figure 2. First, for the input layer, W, H, and 3 are the width, height, and channels, respectively. Second, the three number on CNN layers correspond to the number of convolution filters, the number of rows and columns in each convolution kernel, respectively. Third, 32 and 38 and numbers of nodes in the fully-connected layer, whereas the flattening layer has no parameter. Besides, to improve the efficiency of the model, we adopt ReLU [23] as the activation function in the convolutional operations. The outputs and number of parameters of each layer are given in the Table.I when the input size is $128 \times 128 \times 3$. Note that the output shapes and numbers of parameters depend on the input size of the first layer; therefore, other input sizes will produce different outputs and number of parameters, as we discussed above.

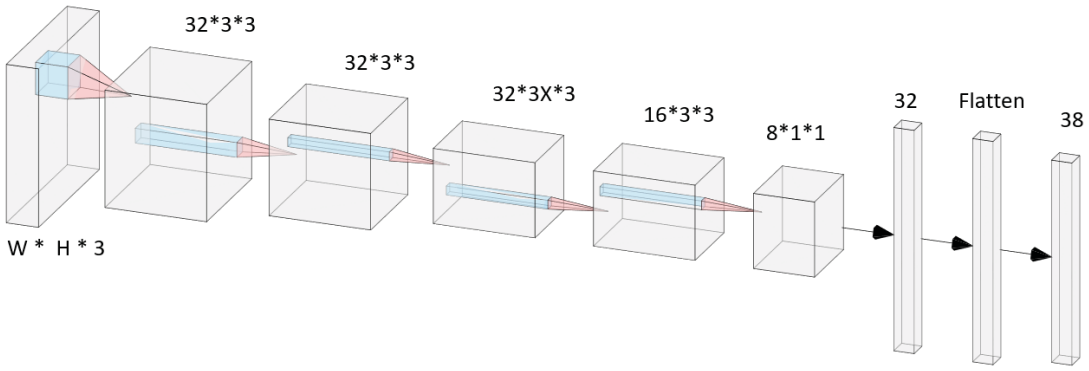

Fig. 2. Lightweight Deep Learning Model for Plant Disease Detection

TABLE I
PROPOSED MODEL

Layer (type)	Output Shape	Total Param #
Input	$128^2 \times 3$	0
Conv2D	$42^2 \times 32$	896
Conv2D	$14^2 \times 32$	9,248
Conv2D	$4^2 \times 32$	9,248
Conv2D	$2^2 \times 16$	2,064
Conv2D	$2^2 \times 8$	136
Dense	$2^2 \times 32$	288
Flatten	$8^2 \times 320$	0
Dense	$8^2 \times 1280$	4902

IV. Experiment Results

In this section, we present the experiment setting and results for both the transfer learning model with input size $256 \times 256 \times 3$, and the light model with various input sizes. We have the following settings:

1) we adopt the most popular optimization function, Adam as the optimization functions for both transfer learning model and light model.
2) we set the learning rate at 0.001 and 50 epochs.
3) we shuffle the data and split the data into 20% as the validation set and 80% as the training set with randomly generated indexes from 21917 samples.
4) we have augmented the dataset before inputting the data into the models.

A. Software and Hardware Settings

Although most of the deep learning models require High-Performance Computing (HPC) machines to run the training process due to the complexity of deep learning models, our models are considerably small compared to the classical deep learning architectures, and we can run them on our regular computers. We set up the training environment on Windows 10 desktop with hardware including GeForce GTX 1070 Ti, RAM 32GB and Intel(R) Core(TM) i7-8700 CPU; and software including Python 3.7.3, CUDA 10.0, and Tensorflow 1.13.1 and Keras.

B. Data Augmentation

Data augmentation is an essential component of our model. Deep learning introduces a large number of parameters, which indicates that the model demands a colossal amount of data to train it. However, collecting and creating such well-label data requires considerable efforts and time. Data augmentation can be performed to generate many additional copies by adding some reasonable noise and transform the training dataset. Specifically, the images can be shifted, sheared, rotated, and flipped from horizontally and vertically; and it can generate many data by processing the data combination of those methods.

Data augmentation introduces several benefits. First, it can prevent the models from over-fitting because of the models learning limited data over time. Second, by adding noise to the data, the models learn how to address the noises and learn key features of the data so that the data can deal with unseen data more robustly. Third, the data augmentation can help the model to improve the prediction accuracy by learning the data from different perspectives.

C. Experimental Results

Fig.3 shows that the transfer learning model can achieve very high accuracy respect to epochs (around 98% and it could be improved further if we train the model using more epochs). The backbone of the models is DenseNet121, whose weights are initialized with ImagNet weights. In this experiment, none of the layers in the model has been freeze from training with plant disease dataset. The model can achieve relatively high accuracy. As we can see from Fig.3, the model converges considerably fast, and it tends to converge around 50 epochs. Considering the performance and complexity, this model can run on the mobile edge severs.

We also consider the performance for deep learning model with different image resolutions. As we can see from their accuracy shown in Fig.4 and Fig.5, and loss shown in Fig.6 and Fig.7, the models with different input sizes mostly converge after 50 epochs. Again, the performance may be improved if the number of epochs increases.

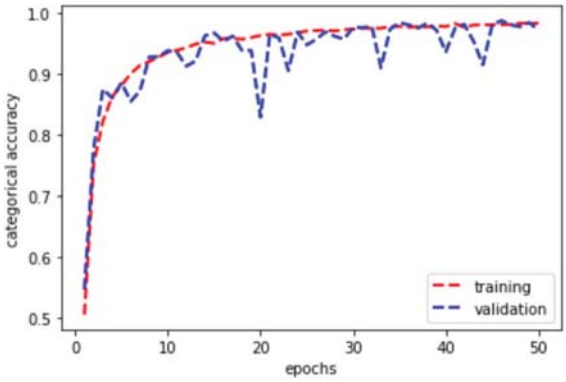

Fig. 3. Transfer Learning Model Accuracy

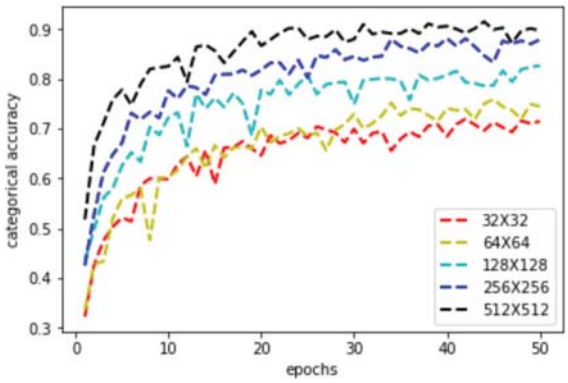

Fig. 5. Validation Categorical Accuracy

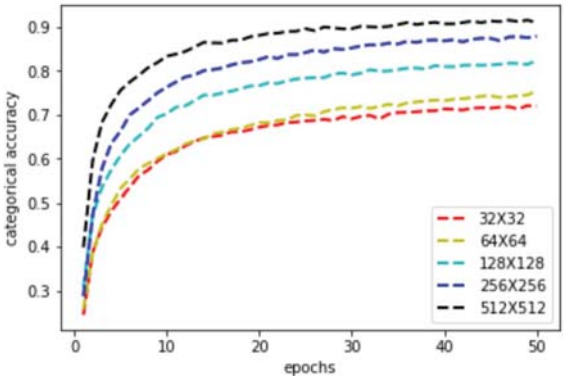

Fig. 4. Training Categorical Accuracy

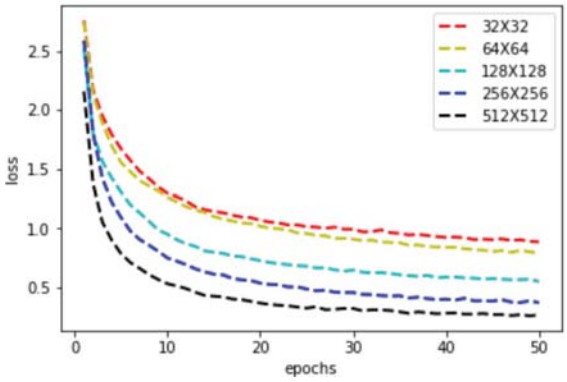

Fig. 6. Loss

The accuracy is various respect to input resolutions from around 70% to 90% respect to 50 epochs. By resizing input image into low resolutions may lose some of the features and compromise the performance. On the other hand, high resolution conveys more information about the features but raise the computational cost exponentially. Additionally, the training learning curves are considerably smoother than the validation curves. The reason is that the models tend to be fit the training dataset more, and training dataset have more samples than validation samples to reduce the noise. Additionally, we compare the models in terms of loss and validation loss respect to epochs, as shown in Fig.6 and Fig.7. Loss results measure the error of the models during training and validation process.

Table.II shows the comparison of the performance and time-consuming of the proposed lightweight deep learning model with different input resolutions. The relative change cost computed as:

$$C = log_{10}(\frac{value - min * 0.99}{min}), \quad (4)$$

where min corresponds to the performance or time when the input size is 32×32, and current cost denoted as $value$. As shown in Fig.8, the training time exponentially grows with respect to the input sizes, whereas the performance

TABLE II
DNN MODEL COMPARISON WITH DIFFERENT INPUT SIZES

Input Sizes	Train Accuracy	Val Accuracy	Train Time
32x32	71.98%	71.49%	2:08:12
64x64	81.12%	74.47%	2:15:54
128x128	86.01%	85.26%	2:45:35
256x256	87.96%	87.92%	5:26:15
512x512	90.00%	89.70%	18:06:16

almost remains unchanged. Therefore, some application scenarios that do not require extremely high accuracy but prefer less training and inference time can choose images with relatively low resolution rather than high resolution. Besides, the proposed lightweight deep learning model is considerably small, but can achieve very high accuracy. Therefore, it could be considered for IoT devices where computing resource is limited. If the application requires significant accuracy and can run on high-performance computers, high-resolution input or transfer learning models can be considered.

V. Conclusions

In this work, we have proposed deep learning methods for plant disease detection. Specifically, we have proposed

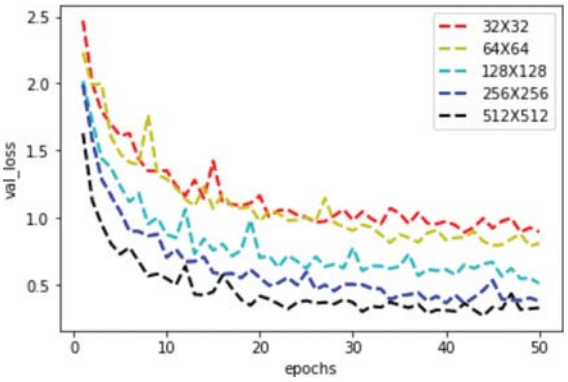

Fig. 7. Validation Loss

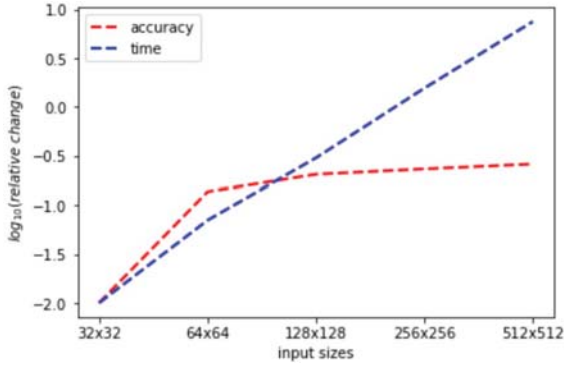

Fig. 8. Logarithmic Scale Compare of Relative Changes

a transfer learning model and a simple deep learning model to recognize and classify plant disease detection. Compared with the classical deep learning models, the proposed models are considerably light while maintaining acceptable prediction performance. Subsequently, we have tested our light model with different image qualities and compared the performance with different image input sizes so as to find balance the complexity raised by image resolution and performance. For the future work, we will study depthwise separable convolution and inverted residuals and linear bottlenecks to reduce the size of the model and convolution operations, without compromising the feature map extract performance. Moreover, we will develop the plant disease detection applications to facilitate deep learning aided smart agriculture.

References

[1] R. Mittler and E. Blumwald, "Genetic engineering for modern agriculture: Challenges and perspectives," *Annual Review of Plant Biology*, vol. 61, no. 1, pp. 443–462, 2010, pMID: 20192746.

[2] F. Caffaro, M. Roccato, M. M. Cremasco, and E. Cavallo, "Falls from agricultural machinery: Risk factors related to work experience, worked hours, and operators behavior," *Human Factors*, vol. 60, no. 1, pp. 20–30, 2018.

[3] R. N Strange and P. Scott, "Plant disease: A threat to global food security," *Annual review of phytopathology*, vol. 43, pp. 83–116, 02 2005.

[4] D. P. Hughes and M. Salathé, "An open access repository of images on plant health to enable the development of mobile disease diagnostics through machine learning and crowdsourcing," *CoRR*, vol. abs/1511.08060, 2015.

[5] V. Singh and A. Misra, "Detection of plant leaf diseases using image segmentation and soft computing techniques," *Information Processing in Agriculture*, vol. 4, no. 1, pp. 41 – 49, 2017.

[6] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, 2016.

[7] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," in *Front. Plant Sci.*, 2016.

[8] S. D. Khirade and A. B. Patil, "Plant disease detection using image processing," in *2015 International Conference on Computing Communication Control and Automation*, Feb 2015, pp. 768–771.

[9] M. P. Sanjay B. Dhaygude, "Agricultural plant leaf disease detection using image processing," *Building climate resilience for food security and nutrition. Rome, FAO*, vol. Licence: CC BY-NC-SA 3.0 IGO., 2018.

[10] S. Arivazhagan, R. N. Shebiah, S. Ananthi, and S. V. Varthini, "Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features," *Agric Eng Int: CIGR Journal*, vol. 15, No.1 211, 03 2013.

[11] J. K Patil, R. Kumar, B. Vidyapeeth, C. O E Kolhapur, B. Vidyapeeth, and P. , "Advances in image processing for detection of plant diseases," *International Journal of Application or Innovation in Engineering Management (IJAIEM)*, 04 2019.

[12] L. Ale, N. Zhang, H. Wu, D. Chen, and T. Han, "Online proactive caching in mobile edge computing using bidirectional deep recurrent neural network," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5520–5530, 2019.

[13] L. Ale, X. Fang, D. Chen, Y. Wang, and N. Zhang, "Lightweight deep learning model for facial expression recognition," *Proceedings of IEEE BigdataSE*, 2019.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 1097–1105.

[15] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, 2014.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[18] H. Agh Atabay, "Deep residual learning for tomato plant leaf disease identification," *Journal of Theoretical and Applied Information Technology*, vol. 95, pp. 6800–6808, 12 2017.

[19] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, inception-resnet and the impact of residual connections on learning," *CoRR*, vol. abs/1602.07261, 2016.

[20] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *CoRR*, vol. abs/1708.02002, 2017.

[21] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[22] N. Zhang, R. Wu, S. Yuan, C. Yuan, and D. Chen, "Rav: Relay aided vectorized secure transmission in physical layer security for internet of things under active attacks," *IEEE Internet of Things Journal*, 2019, DOI: 10.1109/JIOT.2019.2919743.

[23] A. F. Agarap, "Deep learning using rectified linear units (relu)," *CoRR*, vol. abs/1803.08375, 2018.