Plant Disease Recognition Using Optimized Deep Convolutional Neural Networks

Ali Ghofrani[1], Rahil Mahdian Toroghi[1](✉), and Hamid Behnegar[2]

[1] Iran Broadcasting University (IRIBU), Tehran, Iran
{alighofrani,mahdian}@iribu.ac.ir
[2] Department of Computer Engineering, Khatam University, Tehran, Iran
h.behnegar@khatam.ac.ir

Abstract. In this paper, the problem of recognizing the plant's diseases and pests using deep learning methods has been addressed. This work can be implemented on a client-side or integrated with IoT concept, in order to be employed efficiently in smart farms. Nearly 40% of global crop yields each year are lost due to pests. By considering the global population growth, the agricultural food will run out of its resources very soon and this will endanger the lives of many people. A pretrained EfficientNet deep neural network architecture with student noise has been optimized, both in volume and the parameter number, and has been involved in this setup. Two different approaches have been adopted. First, achieving the highest accuracy of recognition using the optimum algorithms in development step. Second, preparation of the system as a microservice model in order to be integrated with other services in a smart agriculture deployment. Using an efficient number of parameters and inference time, it has become doable to implement this system as a service in a real world scenario. The dataset used in the training step is the plant village data. By implementing the model on this dataset, we could achieve the accuracy of 99.69% on test data, 99.85% on validation data, and 99.78% on training data, which is remarkably competitive with the state-of-the-art.

Keywords: Plant disease recognition · Convolutional neural network · EfficientNet · Smart farm

1 Introduction

The ever-increasing growth of the population of our planet, necessitates the food supplier organizations concern this critical problem for a sustainable future of the world due to the limited resources. On the other hand, the agricultural productivity will be decreased by growing the plant diseases and pests.

One development strategy would be migrating toward smart farms in order to improve the productivity through incorporating the smart irrigation, planting and other agricultural procedures. This will diminish the environmental damages, as well as the productivity losses which is normal in conventional farming techniques.

A crucial issue for treating a plant is to diagnose the disease early enough. One challenging point for large countries (e.g. Iran) is vastness of the lands and non-uniform distribution of herbalists in different provinces of the country. This causes a long delay and the golden time for treatment of the plant that suffers from a disease will be running late. Thus, irrecoverable damages would happen to the plants.

In recent years, owing to the rapid advancements in computer vision and artificial intelligence, it has become possible to integrate different engineering fields in order to be able to deal with a broader category of multi-disciplinary applications.

In this research work, our effort has been dedicated to develop an intelligent plant disease diagnosis system using the modern deep neural network architectures. This server-based system is able to get the image of the sick plant even on the farm, diagnose the disease and using the pretrained system which has been already used the knowledge of the herbalists for a wide range of diseases, determine a proper remedy for it. Thus, the whole integrated smart setup helps us diagnose and perform the plant treating during the golden time, without the presence of a herbalist.

2 Related Works

A handful of techniques have been proposed so far, in order to tackle the problem of plant's disease detection. Some approaches leverage image processing techniques and extracting features, such as LBP (Local Binary Pattern) and HBBP (Brightness Bi-Histogram Equalization) [17]. Other methods, extract features using HoG (Histogram of Gradients) followed by SVM classifiers [9,18]. The state-of-the-art of these methods has been achieved, through using the Otsu's classifier [13,19].

In recent years, by advancement of the computer vision field through leveraging the deep learning techniques, methods based on deep convolutional neural networks have been widely proposed for both plant's disease diagnosis, as well as classification of the healthy and infected plants [3,7].

3 The Proposed Framework

3.1 Dataset

Despite the potential capabilities of deep neural networks, one drawback of these systems is that they need a huge amount of data in order to train their few million network parameters. Providing such amount of data in many problems is practically a big challenge, by itself. In this work, we elaborate presenting different models in order to deal with various types of applications (Fig. 1).

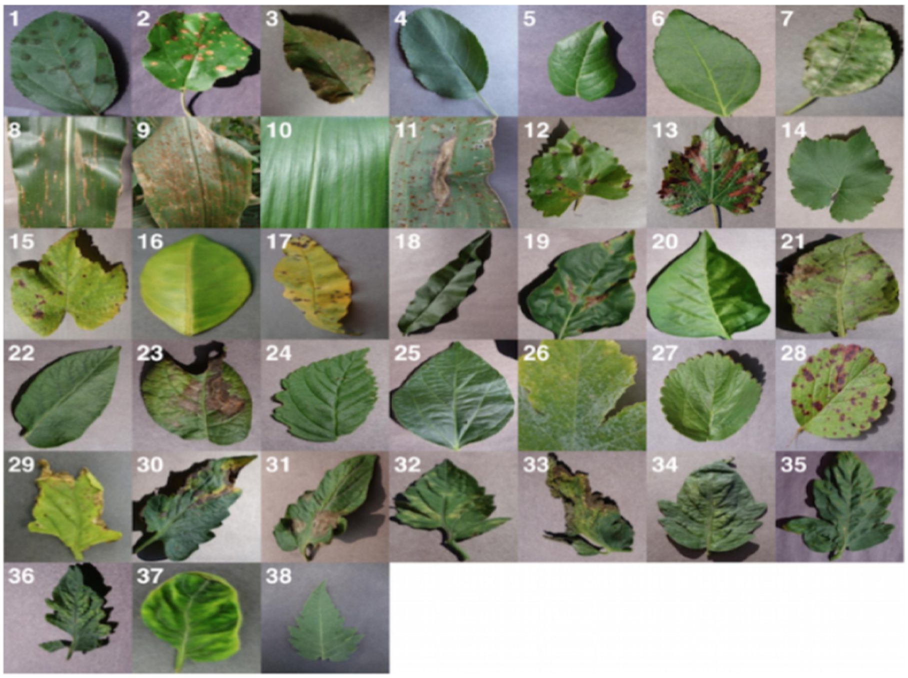

Fig. 1. Example of leaf images from the PlantVillage dataset, representing every crop-disease pair used. 1) Apple Scab, Venturia inaequalis 2) Apple Black Rot, Botryosphaeria obtusa 3) Apple Cedar Rust, Gymnosporangium juniperi-virginianae 4) Apple healthy 5) Blueberry healthy 6) Cherry healthy 7) Cherry Powdery Mildew, Podosphaera spp. 8) Corn Gray Leaf Spot, Cercospora zeae-maydis 9) Corn Common Rust, Puccinia sorghi 10) Corn healthy 11) Corn Northern Leaf Blight, Exserohilum turcicum 12) Grape Black Rot, Guignardia bidwellii, 13) Grape Black Measles (Esca), Phaeomoniella aleophilum, Phaeomoniella chlamydospora 14) Grape Healthy 15) Grape Leaf Blight, Pseudocercospora vitis 16) Orange Huanglongbing (Citrus Greening), Candidatus Liberibacter spp. 17) Peach Bacterial Spot, Xanthomonas campestris 18) Peach healthy 19) Bell Pepper Bacterial Spot, Xanthomonas campestris 20) Bell Pepper healthy 21) Potato Early Blight, Alternaria solani 22) Potato healthy 23) Potato Late Blight, Phytophthora infestans 24) Raspberry healthy 25) Soybean healthy 26) Squash Powdery Mildew, Erysiphe cichoracearum, Sphaerotheca fuliginea 27) Strawberry Healthy 28) Strawberry Leaf Scorch, Diplocarpon earlianum 29) Tomato Bacterial Spot, Xanthomonas campestris pv. vesicatoria 30) Tomato Early Blight, Alternaria solani 31) Tomato Late Blight, Phytophthora infestans 32) Tomato Leaf Mold, Fulvia fulva 33) Tomato Septoria Leaf Spot, Septoria lycopersici 34) Tomato Two Spotted Spider Mite, Tetranychus urticae 35) Tomato Target Spot, Corynespora cassiicola 36) Tomato Mosaic Virus 37) Tomato Yellow Leaf Curl Virus 38) Tomato healthy.

One useful dataset to deal with the plant's disease problem is called **Plant Village** [10]. This dataset has been publicly available and contains 54, 306 images of the plants, including the healthy ones, as well as the sick plants and have been classified into 38 classes with 14 different plant species and 26 different types of plant's diseases.

In paper [10], the AlexNet and GoogleNet architectures have been used and the accuracy of 99.35% on disease diagnosis have been achieved.

Another dataset, namely **PlantDoc** [16], has been released on 2019 which contains 2, 598 images on 13 different plant species and 27 classes (10 healthy and 17 diseased classes). The main difference between this dataset and the Plant Village is that the images of the new dataset have been gathered in real life, whereas the old one has collected the images in a lab-controlled fashion.

3.2 The Proposed Method

In PlantVillage paper [10], AlexNet and GoogleNet architectures have been used in order to perform the classification task. However, both these architectures are so large and not efficient and unlike to be implementable on the client side.

In this paper, we have proposed two different models. First, we propose a model based on EfficientNet architecture to achieve a high accuracy and implementability on the client side as an internet-based service [20]. Second, we propose a model based on tiny MobileNet V2 [12], and the separable CNN logic which has been presented in *Xception* architecture [1]. This model yields a reasonably small number of controllable parameters in a light model and makes it suitable as an end system on the client's mobile handset (Fig. 2).

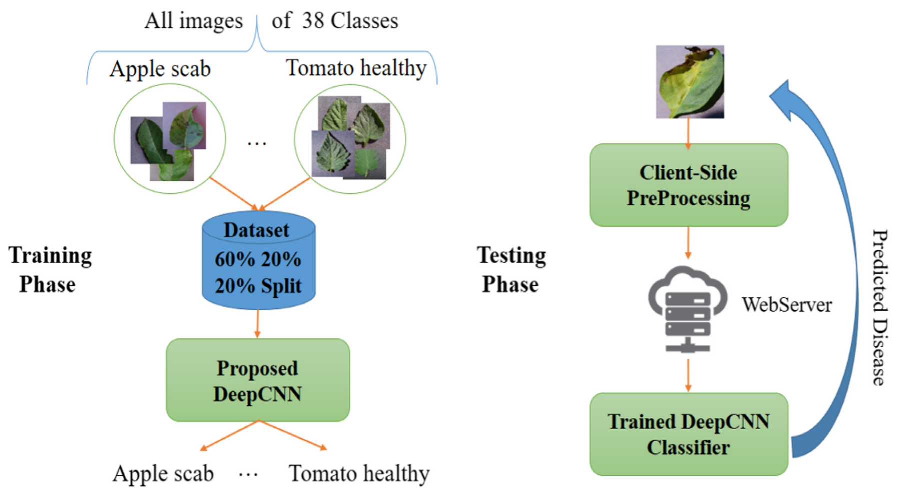

Fig. 2. Process flowgraph of the entire proposed architecture for: (left) training and (right) testing phases.

Our proposed architectures have been trained by Plant Village dataset, with 60% – 20% – 20% chunks for training, validation, and test steps, respectively.

Overall saying, the model is first trained using the Plant Village dataset. However, in a server-based implementation the input image being taken by the user is to be preprocessed to get suitable dimensions corresponding to the network input resolution and then sent to the server. In addition, as the second method an architecture with a relatively small number of parameters has been proposed that could be implemented on all kinds of smart mobile handsets directly with a slightly worse, yet acceptable accuracy.

3.3 EfficientNet Architecture

EfficientNet B0 is a CNN architecture being employed in our first proposed model [20], as depicted in Fig. 3.

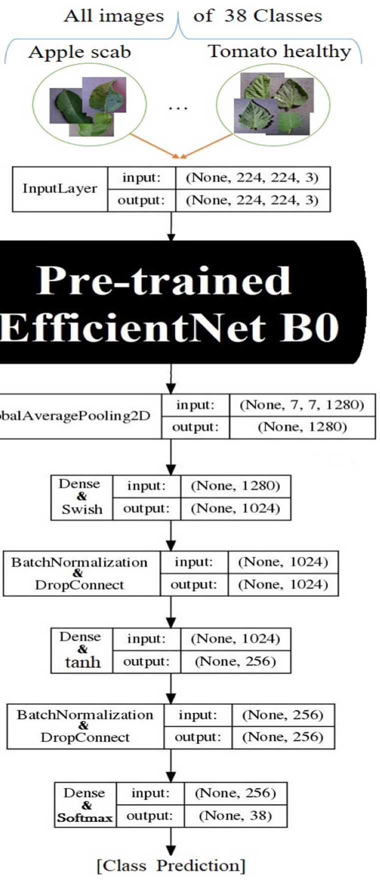

Fig. 3. The first proposed architecture based on EfficientNet B0. Transfer learning has been involved regarding the weights of the network.

This network has been already trained on ImageNet [2], with input dimensions of $224 \times 224 \times 3$ for 1000 different classes of images and could be employed in similar applications using transfer learning, since it can seriously expedite the training process of the networks being involved. Figure 4, visualizes the final dense layer perception of EfficientNet B0 on ImageNet data. This figure implicates that the pretrained model on the ImageNet data could be logically justified as a suitable model to be transferred for learning the plant disease problem. This can be realized from the fact that visualizing the final model layers before clipping them for the training phase has strong similarities with the classes that are included in our problem. This helps our pretrained model using its original weights be converged quite faster than the model being initialized by the random noise weights.

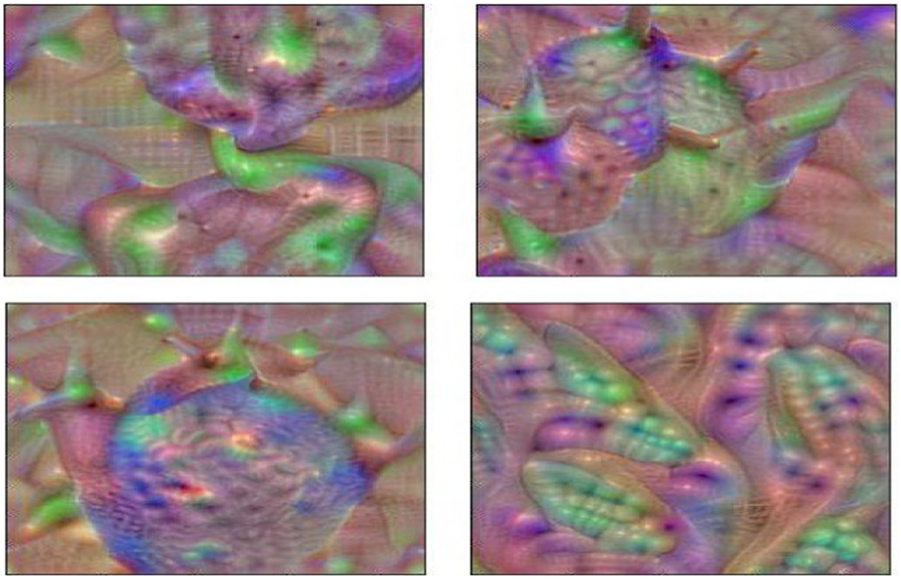

Fig. 4. Visualized final layer of the trained EfficientNet model on ImageNet. (Top-Left) 945: Bell pepper, (Top-right) 949: strawberry, (Bottom-left) 950: orange, (Bottom-right) 987: corn

Next step would be modifying the architecture appropriately to meet the requirements of our problem. After dropping off the FC (Fully Connected) layer of ImageNet classifier, we add our determined stack of layers to the model. This stack includes a CNN connected to a dense layer with 1024 neurons that are initiated with *Swish* activation functions [11], due to a better performance compared with the Leaky ReLU, and it would be connected to the next hidden layer. In order to avoid overfitting the dropconnect technique has been incorporated with 0.5 deactivation coefficient [21]. The batch-normalization has been also employed [8], and the layer is then connected to another fully-connected layer. Then, the dimension has been reduced to 256 and *Tanh* activation function has been used. This stack of batch-normalization and dropconnect, has been repeated until it boils down to 38 neurons with *Softmax* activation function in the output in order to perform a probabilistic multi-class classification. This stack has been pretrained and the model parameters have been weighted using *Xavier* Normal technique in order to make the model converge rapidly [6]. Then, the entire model has been trained on the Plant Village dataset using these pretrained weights.

Since a multi-class classification is to be performed, the categorical cross entropy has been used as the loss function and the proposed model is monitored through maximizing the accuracy over the validation data at each epoch.

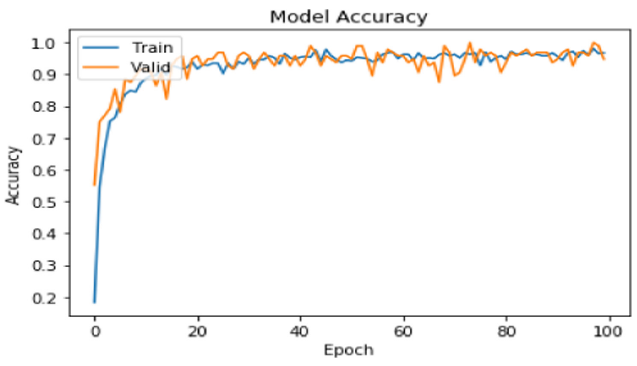

Fig. 5. Training and validation **accuracy** of the proposed model, based on EfficientNet.

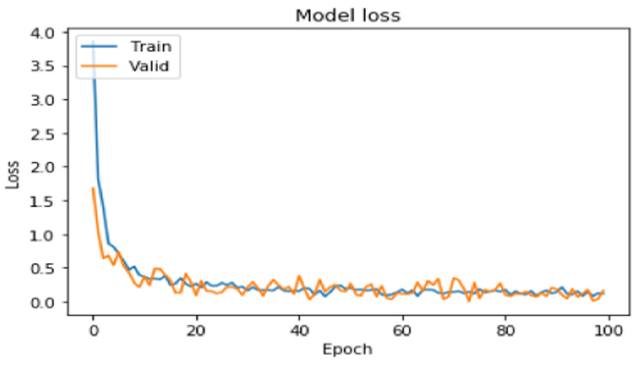

Fig. 6. Training and validation **loss** of the proposed model, based on EfficientNet.

As it is obvious from Fig. 5 and 6, the validation accuracy and loss are slightly better than that of training accuracy and loss, respectively. That is due to using dropconnect during the training phase which prevents our model from overfitting.

The final proposed architecture contains 5, 988, 002 parameters which are to be trained. Moreover, the final model needs about 68 MBytes to be saved.

The proposed model outperforms both the GoogleNet and AlexNet-based models being tested on Plant Village dataset from both the accuracy aspect and having the smaller size. This highlights the idea of leveraging the proposed model and implementing it as a web service. The results of the average training, validation and test accuracy and loss for the proposed model is shown in Table 1.

Table 1. Accuracy and Loss of Training, Validation and testing of the model.

	Training	Validation	Testing
Accuracy	99.78	99.85	99.69
Loss	0.0363	0.0089	0.0403

3.4 MobileNet V2 - Lite Version

In addition to the previously proposed architecture that could be implemented as a web service, we further elaborated a low weight model which could be deployed and executed on mobile device processors, to the cost of a slight degradation on the accuracy level as a trade-off with the model size.

In order to implement this model the separable CNN logic (i.e., the Point-wise, Depth-wise blocks [4]) has been leveraged which has already been employed in Xception architecture [1], and the MobileNet V2 has been optimized to meet the requirements of our problem. To achieve that, we have omitted the dense layer with 1000 neurons and connected our stack of blocks to the final CNN layer. Similar to the previously proposed architecture, we have used the ImageNet pretrained model for CNN as the initial weights and the Xavier Normal technique has been employed for training the weights on the remaining parts of the architecture [5].

The input resolution size has been resized to 96×96 and the kernel numbers have been reduced to one third, as well. Instead of using Swish activation function which entails a heavy burden for the processor, the *Leaky ReLU* has been used. furthermore, dropconnect technique has been involved in place of dropout to be more efficient [21], and the batch size has been increased form 48 to 512. Hence, a group-normalizer has been used instead of batch-normalization [22]. Finally, similar to the previously proposed model the stack of fully connected layers has been attached to the CNN final layer. The entire model has been depicted in Fig. 7.

The total number of parameters for the proposed model has been reduced to 279, 169 and the model size has become 3, 92 MBytes, as well. Needless to say that after performing the postprocessing methods such as quantization and pruning the model size will boil down to about 270 KBytes.

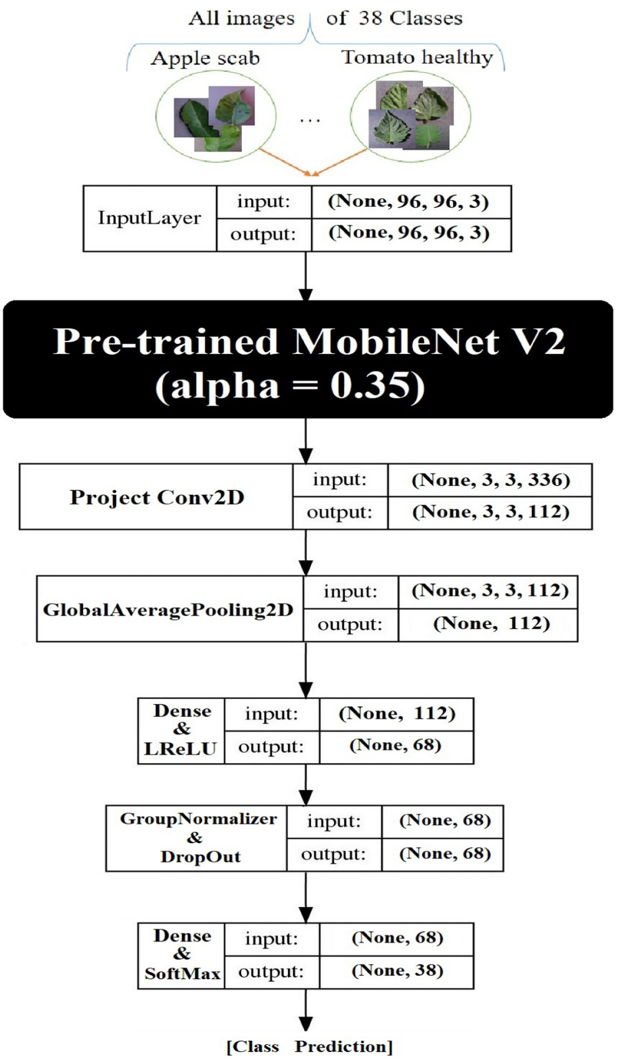

Fig. 7. Our proposed architecture based on lite mobileNet V2. Transfer learning has been involved for the weights of the network.

Despite the simplifications imposed to the model architecture, the model could still achieve the reasonable accuracy of 95, 64% on the test dataset. The accuracy and loss of the model on training and validation datasets have been depicted in Fig. 8 and Fig. 9. Moreover, the results of the average training, validation and test accuracy and loss for the simplified proposed model is shown in Table 2.

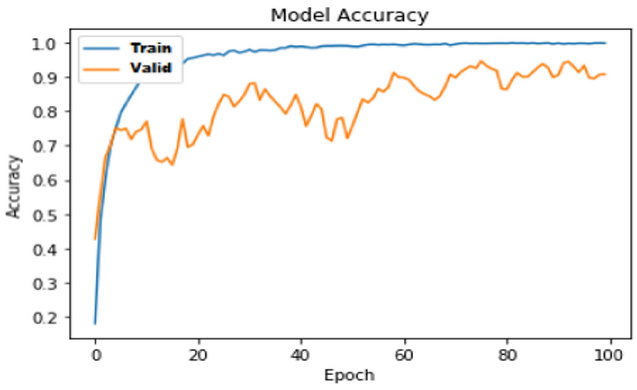

Fig. 8. Training and validation accuracy results for the proposed architecture based on lite-mobilenetV2

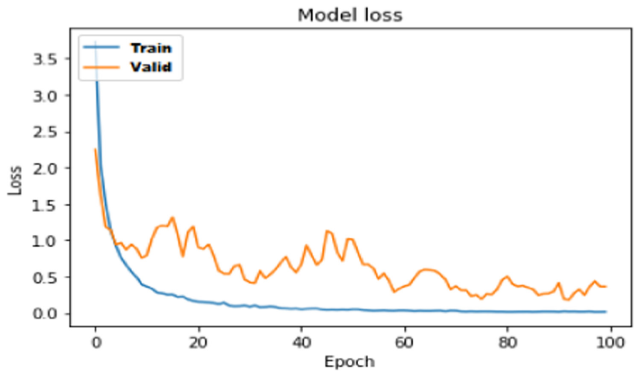

Fig. 9. Training and validation losses for the proposed architecture based on lite-mobilenetV2

Table 2. Accuracy and Loss of Training, Validation and testing of the simplified model.

	Training	Validation	Testing
Accuracy	99.62	95.82	95.46
Loss	0.0363	0.0089	0.2234

In a practical implementation of the tiny MobileNet version (with 270 KBytes) two samples of the test data have been given to the model. Figure 10 illustrates the sample infected leaves along with their saliency maps [15], as well as their corresponding heat-maps [14]. As it is depicted in Fig. 10 the model attention has been correctly placed on the infected parts of the leaves, as expected by a human expert [14].

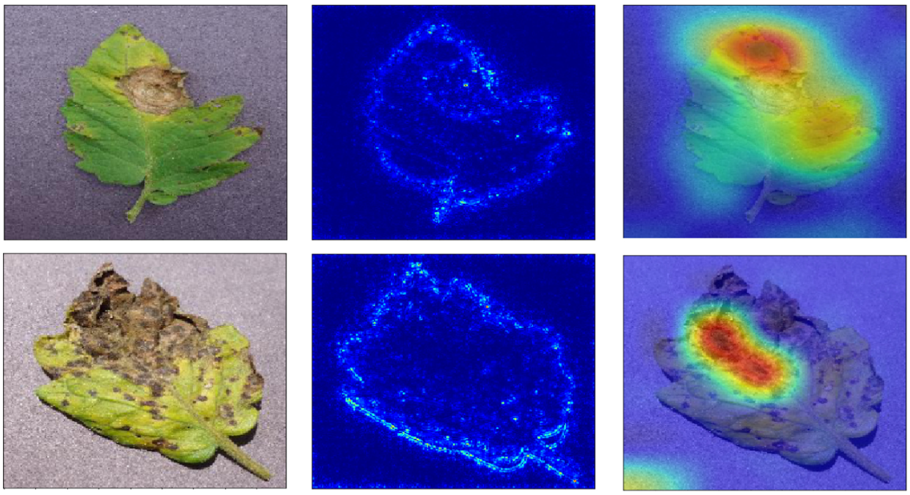

Fig. 10. (Left to right) 1st Column: RGB images of the sick plants, 2nd column: Saliency maps corresponding to the RGB samples, 3rd column: Model attention of the corresponding RGB samples

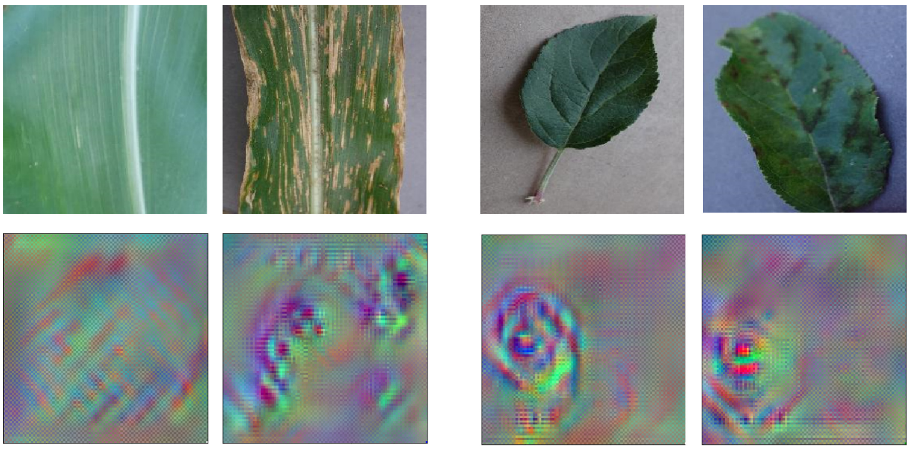

Fig. 11. (Top row: Left to right) Corn maize healthy, Corn maize with Cercospora leaf spot disease, Apple healthy, Apple Scab. (Bottom row: Left to right) Visualization of the corresponding dense layer of “Corn maize healthy”, Visualization of the corresponding dense layer of “Corn maize with Cercospora leaf spot disease”, Visualization of the corresponding dense layer of “Apple healthy”, Visualization of the corresponding dense layer of “Apple scab”, all for lite MobileNet V2.

Furthermore, as depicted in Fig. 11 the discrepancies between the visualized outputs of the neurons corresponding to the sample images has been clearly illustrated. The visualized outputs belong to the dense layers corresponding to the input class of images.

4 Conclusion

The gist of this paper has been to recognize the plant's disease using deep learning techniques. To achieve this, we have proposed two models based on convolutional neural networks. The first proposed model uses an EfficientNet B0 CNN as a core block which has been customized to meet the requirements of our problem and has been optimized to be properly usable as a server-based application. The second model involves MobileNet V2 as a core block and simplifies it to be deployable on a mobile device. The first model outperforms the state-of-the-art in the accuracy sense, and the second model in spite of simplicity performs reseanably well for diagnosing the plant's diseases and pests on Plant Village test data samples. Specifications of the proposed systems qualifies them for being employed as a microservice model in an integration with other required services in a smart farm and other agricultural activities.

Acknowledgement. This work is heavily owing to Sarvban startup company in agricultures and specifically expresses the deepest thanks to Erfan MirTalebi and Aref Samadi for their friendly cooperation and cordially sharing their knowledge with us.

References

1. Chollet, F.: Xception: deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1251–1258 (2017)
2. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)
3. Ferentinos, K.P.: Deep learning models for plant disease detection and diagnosis. Comput. Electron. Agric. **145**, 311–318 (2018)
4. Ghofrani, A., Toroghi, R.M., Ghanbari, S.: Realtime face-detection and emotion recognition using MTCNN and minishufflenet v2. In: 2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI), pp. 817–821. IEEE (2019)
5. Ghofrani, A., Toroghi, R.M., Tabatabaie, S.M.: Attention-based face antispoofing of RGB camera using a minimal end-2-end neural network. In: 2020 International Conference on Machine Vision and Image Processing (MVIP), pp. 1–6. IEEE (2020)
6. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 249–256 (2010)
7. Hanson, A., Joel, M., Joy, A., Francis, J.: Plant leaf disease detection using deep learning and convolutional neural network. Int. J. Eng. Sci. **5324** (2017)
8. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)

9. Islam, M.A., Yousuf, M.S.I., Billah, M.: Automatic plant detection using hog and LBP features with SVM. Int. J. Comput. (IJC) **33**(1), 26–38 (2019)
10. Mohanty, S.P., Hughes, D.P., Salathé, M.: Using deep learning for image-based plant disease detection. Frontiers Plant Sci. **7**, 1419 (2016)
11. Ramachandran, P., Zoph, B., Le, Q.V.: Swish: a self-gated activation function, vol. 7. arXiv preprint arXiv:1710.05941 (2017)
12. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv 2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510–4520 (2018)
13. Saradhambal, G., Dhivya, R., Latha, S., Rajesh, R.: Plant disease detection and its solution using image classification. Int. J. Pure Appl. Math. **119**(14), 879–884 (2018)
14. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 618–626 (2017)
15. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034 (2013)
16. Singh, D., Jain, N., Jain, P., Kayal, P., Kumawat, S., Batra, N.: Plantdoc: a dataset for visual plant disease detection. In: Proceedings of the 7th ACM IKDD CoDS and 25th COMAD, pp. 249–253 (2020)
17. Singh, K., Kumar, S., Kaur, P.: Automatic detection of rust disease of lentil by machine learning system using microscopic images. Int. J. Electr. Comput. Eng. **9**(1), 660 (2019)
18. Sullca, C., Molina, C., Rodríguez, C., Fernández, T.: Diseases detection in blueberry leaves using computer vision and machine learning techniques. Int. J. Mach. Learn. Comput. **9**(5) (2019)
19. Sun, G., Jia, X., Geng, T.: Plant diseases recognition based on image processing technology. J. Electr. Comput. Eng. **2018** (2018)
20. Tan, M., Le, Q.V.: Efficientnet: rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946 (2019)
21. Wan, L., Zeiler, M., Zhang, S., Le Cun, Y., Fergus, R.: Regularization of neural networks using dropconnect. In: International Conference on Machine Learning, pp. 1058–1066 (2013)
22. Wu, Y., He, K.: Group normalization. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 3–19 (2018)