Journal of Agricultural Engineering and Food Technology
p-ISSN: 2350-0085; e-ISSN: 2350-0263; Volume 10, Issue 1; January-March, 2023, pp. 18-21

http://www.krishisanskriti.org/Publication.html

Plant Disease Identification using Deep Learning and Python Based App

Mridul Kumar[1] and Soami Daya Krishnananda[2]

[1]Ph.D. Scholar, Department of Physics and Computer Science
Dayalbagh Educational Institute, Agra-282005 (U.P.)
[2]Professor, Department of Physics and Computer Science
Dayalbagh Educational Institute, Agra-282005 (U.P.)

Abstract—*The world population is set to reach 9 billion people by the end of year 2050. It is estimated that crop production needs to increase by 70% to feed this population. Plant diseases challenge the increased demand for the food supply, taking away 30% of the quantity and decreasing the crop quality. Rapid identification of these diseases can reduce both the loss in crop quality and quantity. However, identification of plant disease requires human expertise and examination of each plant individually, which is quite tedious. Furthermore, different human experts rate the disease differently, which further complicates the identification of the plant disease. Deep learning-based methods (such as convolutional neural network-based techniques) for the prediction of the plant disease have been lauded among the scientific community for their high classification accuracy. In this paper, we present a model for the identification of diseases in plant leaves which has been trained on openly available xPLNet dataset. The reported accuracy of xPLNet model is 94.13% on test data. Our model, on the other hand, produces 98.6% test accuracy with ~40% less trainable parameters. The accuracy of our model has been increased by using data augmentation technique. The final accuracy of our model on training data comes out to be 98.7%. For the real-time detection of the plant disease, a graphical user interface has been built using PyQt5, which accepts the clicked images of the plant leaves and displays the type of disease. This graphical user interface also works on Android devices, which makes plant disease identification even easier.*

INTRODUCTION

Crop disease are a major threat to the food security as most of them reduce the crop produce by 30% [1]. This becomes a big problem as the world population is set to reach 9 billion people by 2050. For feeding the world population the crop production should increase by 70% [2]. Therefore, crop disease identification becomes important for the mitigation of the loss caused by them. Experts are required for the identification of the plant disease who can distinguish the symptoms. However, this requires examination of each plant individually which is tedious and time-consuming. Therefore, techniques are required which can tell the type of the disease and the corrective measure that can be taken for the mitigation of the disease. Recently, deep learning based image classification is trending among scientific community because of its high accuracy of detection of the plant disease based on the visual symptoms [2]–[5]. Since these techniques use deep neural networks, to avoid the problem of overfitting, require large datasets for training which is not possible in most of the cases. Therefore, techniques such as image data augmentation is required for bringing the variety in the data by producing transforms and increase the classification accuracy [6]. Furthermore, deep neural network models are large in size and have high computation requirements which is not possible for on-the-go disease identification using smartphones [7], [8]. In this paper, we propose a lightweight model trained on openly available xPLNet dataset which contains pictures of plant leaves suffering from 8 different types of diseases [3]. For increasing the variety of the data available in the xPLNet dataset we have used image data augmentation which has helped us in increasing the testing accuracy by 4.4% from the benchmark accuracy of 94.13% as reported by Ghosal et. al. [3]. By using image data augmentation, we could decrease the trainable parameters by ~40% and could achieve higher accuracy which helps us in exporting a lightweight (6.5 MB in size) model. We have used this model in our Python based Graphical User Interface (GUI) which also works on Android devices by using the Pydroid app as backend and can be used for on-the-go plant disease identification.

MATERIALS AND METHODS

Dataset for training

We obtained the xPLNet dataset by filling up the Google form link available at Github, which is the primary GitHub repository for the same [3]. This dataset contains pictures of 9 different classes in which 8 different diseases have been included with 1 class of healthy plant leaves (see Table 1). This dataset has 65760 images in which 59184 images have been taken in training set and 6576 have been taken in test set. The further distribution of the images in respective classes can be found in Table 1. Initially, we used the model architecture that the authors have provided which has 901,653 trainable parameters in total. After testing their model we moved to create our own and improved it thereafter by hyperparameter variation and using image data augmentation.

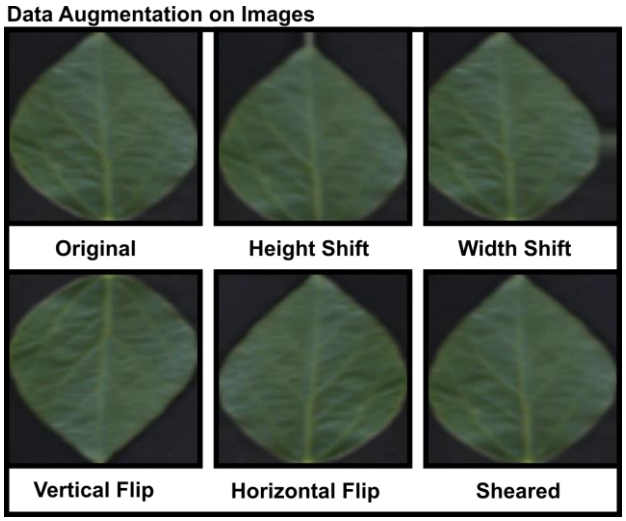

Figure 1: This figure shows the augmented images produced with the help of the image data generator

Table 1: Types of disease available in the dataset for the prediction.

Class	Type of Leaves	Number of Leaves	
		Train Set	Test Set
Class 0	Bacterial Blight	5953	623
Class 1	Frogeye Leaf Spot	5897	679
Class 2	Brown Spot	5934	642
Class 3	Healthy	11850	1302
Class 4	Herbicide Injury	5918	658
Class 5	Iron Deficiency	5937	639
Class 6	Potassium Deficiency	5872	704
Class 7	Bacterial Pustule	5906	670
Class 8	Sudden Death Syndrome	5917	659
Total		59184	6576
Grand Total		65760	

Hardware and Software

We used Nvidia Tesla V100 SXM2 GPU with 32 GB RAM memory for training the deep learning model. To expedite the training process, the GPU was used with CUDA hardware acceleration. For this purpose, TensorFlow GPU (v2.4.1) module in anaconda environment (v22.9.0) was used. To further simplify the process of modeling and training the neural network, Keras (v2.4.0) was used on top of TensorFlow GPU. All the preprocessing of the training and testing data was done using Python (v3.9.5).

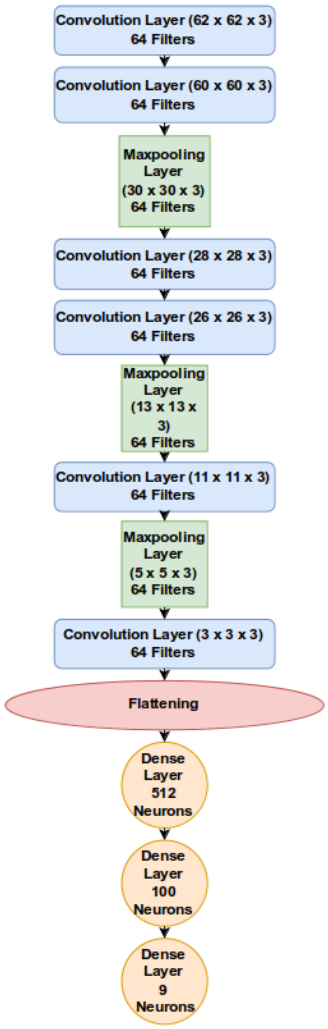

Figure 2: The architecture of the model that we have trained.

Preprocessing of Data

The input data was preprocessed to form images of 64 x 64 with 3 channels (RGB). Data augmentation was used for creating minor transforms in the images which neural network treats as a new data. This increased the size of the dataset for training and led to better results. Researchers use data augmentation (DA) when they are unable to collect variety of data for training the neural network. This leads to neural network being underfit or overfit depending upon the number of training epochs. It is to be noted that the data augmentation technique is generally applied only on the training data to make the neural network more generalized and robust and not on the testing data. We used Image Data Generator class from Keras module for creating transformations such as width shift, height shift, and horizontal flip etc. for the images in dataset (see Figure-1).

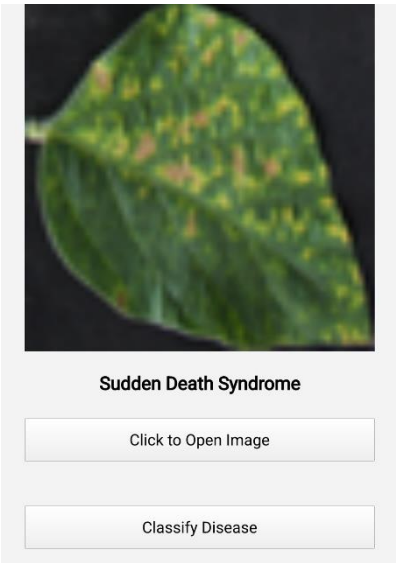

Fig. 3: Python GUI running on Android device with the help of Pydroid app. This app identifies the disease in the clicked pictures of the plant leaves.

Training of Model

The architecture of the model that we have developed can be seen in Figure-2. It is based on the convolutional neural network (CNN) layers which are highly effective against classification of images. The total number of trainable parameters in the model are 534,449 which are ~40% less than the original model [3]. For applying the convolution operations, the size of kernel was chosen as 3×3. For the reduction of the size of the images while training Max-pooling layers were also used with kernel size 2×2. For reducing the chances of overfitting, dropout layers have been used with the dropout rate of 50%. Adaptive momentum (Adam) was used as the optimizer for updating the weights, categorical cross entropy was used as the loss function, and accuracy was used as the metric for checking the efficiency of the model. While training the model, the image data generator creates transforms in the images. These transformed images are fed to the fit generator which operates model in forward direction and finally calculates the error with the help of loss function which is fed back in the neural network. On the basis of this error, the optimizer updates the weights.

Python GUI Application

For making the plant disease detection easier a GUI application was coded using PyQt5 on Python. This app allows loading of the plant leave images with a push of a button. These images are then sent to the trained neural network to see if the plant leaves are healthy or sick. Finally, the result is shown on the GUI application interface.

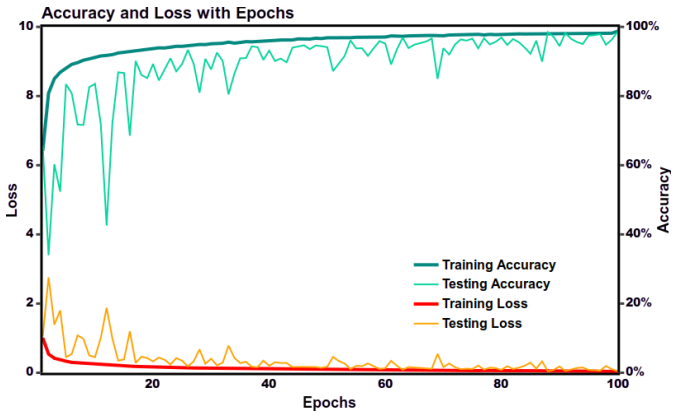

Fig. 4 Variation of accuracy and loss with the number of epochs for both training and testing.

Detected Class

Actual Class	0	1	2	3	4	5	6	7	8
0	602	3	2	2	0	0	0	14	0
1	6	664	0	0	1	0	0	7	1
2	0	0	640	1	0	0	0	1	0
3	0	0	1	1301	0	0	0	0	0
4	4	3	1	0	649	0	0	1	0
5	0	1	0	0	0	637	1	0	0
6	0	1	0	6	1	1	694	1	0
7	30	1	0	1	0	1	0	637	0
8	0	0	0	0	0	0	0	0	659

Fig. 5: Confusion matrix for the testing data. The highest confusion was observed between Bacterial blight and Bacterial pustule that is class 0 and 7.

RESULTS AND DISCUSSION

The neural network was trained for 100 epochs with batch size 64 and the training and testing accuracies consistently increased with number of epochs (see Figure-4). The final accuracy of the neural network comes out to be 98.75% on the training data and 98.6% for the testing data (see Figure-4). Our test accuracy is 4.4% higher than the model reported by Ghosal *et. al.* [3]. Confusion matrix for both the training and testing phases (see Figure-5) were created for checking the false-positive rate of the neural network (see Figure-5). The highest false-positive rate is observed for bacterial blight and bacterial pustule because of the visible similarity between the two which leads to false identification. On test data, our model identifies bacterial blight as bacterial pustule in 14 samples and bacterial pustule as bacterial blight in 30 samples which is quite low in comparison to Ghosal et. al., who report it in 72 and 118 samples, respectively. After training and testing of the

model, it was exported to a Hierarchical Data Format (HDF or h5). The final size of the model was 6.5 MB because of which we could load this model in our smartphone and used it for the disease identification on the plant leaf images on the internet (see Figure-3). In our real-world testing using the smartphone camera and the python app for disease identification. We noticed that the model in model of the cases is unable to identify potassium deficiency which occurs as the yellowing of the plant leaves. This can be attributed to the fact that potassium is a macronutrient and helps plants in fighting the disease [9], [10]. In the absence of such an important nutrient, plants are more likely to suffer from other diseases which the model could identify. Our model could identify sudden death syndrome in all the cases. However, the model is confused between brown spot and frogeye leaf spot in some samples because of similarity of visible cues.

CONCLUSION

With the help of image data augmentation, the accuracy of the earlier reported model could be increased by 4.4 % even when decreasing the number of trainable parameter by 40%. The decrease in the trainable parameters makes the exported model lightweight and helped in exporting the model to smartphone for on-the-go plant disease identification. This model can also be used with unmanned aerial vehicle for a large-scale plant disease identification.

Visible symptoms of plant disease occur after a significant loss to the plant health has already happened, however, it has been reported that plants already start their self-defense mechanism even before the actual symptoms are seen [11]–[13]. Plant disease identification techniques which rely on the visible cues of the disease lack early detection and prediction elements in them. Therefore, a technique which can detect plant stress early targeting the plant self-defense mechanism would decrease the loss of crop significantly.

ACKNOWLEDGMENTS

The authors would like to thank Mr. Manish Kumar for providing the access to the Nvidia DGX server for model training and testing. Authors would also like to thank Mr. Zeeshan for his help in editing the manuscript.

REFERENCES

[1] E.-C. Oerke, “Crop losses to pests,” *The Journal of Agricultural Science*, vol. 144, no. 1, Art. no. 1, 2006.

[2] D. Hughes, M. Salathé, and others, “An open access repository of images on plant health to enable the development of mobile disease diagnostics,” *arXiv preprint arXiv:1511.08060*, 2015.

[3] S. Ghosal, D. Blystone, A. K. Singh, B. Ganapathysubramanian, A. Singh, and S. Sarkar, “An explainable deep machine vision framework for plant stress phenotyping,” *Proceedings of the National Academy of Sciences*, vol. 115, no. 18, Art. no. 18, 2018.

[4] S. P. Mohanty, D. P. Hughes, and M. Salathé, “Using deep learning for image-based plant disease detection,” *Frontiers in plant science*, vol. 7, p. 1419, 2016.

[5] H. Durmuş, E. O. Güneş, and M. Kırcı, “Disease detection on the leaves of the tomato plants by using deep learning,” in *2017 6th international conference on agro-geoinformatics*, IEEE, 2017, pp. 1–5.

[6] C. Shorten and T. M. Khoshgoftaar, “A survey on image data augmentation for deep learning,” *Journal of big data*, vol. 6, no. 1, pp. 1–48, 2019.

[7] D. Singh, N. Jain, P. Jain, P. Kayal, S. Kumawat, and N. Batra, “PlantDoc: A dataset for visual plant disease detection,” in *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD*, 2020, pp. 249–253.

[8] M. Chohan, A. Khan, R. Chohan, S. H. Katpar, M. S. Mahar, and others, “Plant disease detection using deep learning,” *International Journal of Recent Technology and Engineering*, vol. 9, no. 1, pp. 909–914, 2020.

[9] M. Wang, Q. Zheng, Q. Shen, and S. Guo, “The critical role of potassium in plant stress response,” *International journal of molecular sciences*, vol. 14, no. 4, Art. no. 4, 2013.

[10] M. Hasanuzzaman *et al.*, “Potassium: a vital regulator of plant responses and tolerance to abiotic stresses,” *Agronomy*, vol. 8, no. 3, p. 31, 2018.

[11] M. F. Machinandiarena, M. C. Lobato, M. L. Feldman, G. R. Daleo, and A. B. Andreu, “Potassium phosphite primes defense responses in potato against Phytophthora infestans,” *Journal of plant physiology*, vol. 169, no. 14, pp. 1417–1424, 2012.

[12] F. Cheng and Z. Cheng, “Research progress on the use of plant allelopathy in agriculture and the physiological and ecological mechanisms of allelopathy,” *Frontiers in plant science*, vol. 6, p. 1020, 2015.

[13] C. Das *et al.*, “Allelopathic potentialities of leachates of leaf litter of some selected tree species on gram seeds under laboratory conditions,” *Asian J. Exp. Biol. Sci*, vol. 3, no. 1, Art. no. 1, 2012.