Plant Disease Detection and Classification Using Deep Neural Networks

Aravindhan Venkataramanan
Department of CSE, PESIT, Bangalore South Campus

Deepak Kumar P Honakeri
Department of CSE, PESIT, Bangalore South Campus

Pooja Agarwal
Department of CSE, PESU-EC Bangalore

***Abstract*—Plants play a vital role in the survival of all organisms on Earth. Due to this fact, it is veryimportant to ensure that measures are taken to detect and mitigate any diseases on plants. Plant diseases are a major factor for crop losses in agriculture. This paper presents a Deep Learning approach to detect and classify plant diseases by examining the leaf of a given plant. In this paper, the classification is performed in multiple stages to eliminate possibilities at every stage, hence providing better accuracy during predictions. A YOLOv3 object detector is used to extract a leaf from the input image. The extracted leaf is analyzed through a series of ResNet18 models. These ResNet18 models were trained using transfer learning. One layer identifies the type of leaf and the following layer checks for the possible diseases that could occur in the plant.**

Keywords - component; YOLO v3, ResNet18, Keras

I. INTRODUCTION

Image classification has been around a very long time and also been a popular field of research. In fact, it has been implemented in a majority of major application. In this paper, we have implemented the same to solve the problem of plant disease identification by analyzing the leaf from a plant using Convolutional Neural Networks (CNN).

A major factor that supports the existence of life on earth starts from deep down in the food chain, the plants. All these plants are prone to various diseases because they are exposed to the various conditions of nature. As shown in Fig. 1, the agricultural losses are huge because of these diseases. Detecting and curing these diseases at a very rudimentary stage helps save a lot of money and effort.

As a solution to this problem, we have devised a system that uses deep learning to analyze, detect and classify any disease that might have affected a plant by taking an image of the leaf. The processing pipeline goes as follows:

1. The leaf is detected in the given image and cropped out
2. The extracted leaf is then run through a classifier to identify which plant the leaf belongs to
3. The leaf is then checked for the disease class, if any, based on the result from the previous step

II. LITERATURE SURVEY

There has been a lot of research around classifying the disease in a plant using image processing. The researches include the use of various Machine Learning and Deep Learning strategies to complete the task.

Machine Learning approaches include image segmentation[11], shape-only features for plant leaf identification[12] , Support Vector Machines (SVM's)[13], using shape features and K-Nearest Neighbours (KNN)[14], K-means and Artificial Neural Networks (ANN)[15][16] and Probabilistic Neural Networks (PNN)[17][18].

Deep Learning based plant disease classification models includes the use of variety of CNN models such as AlexNet[1], GoogleNet[2], modified GoogleNet[2], LeNet[3], Caffe[4][5] and Deconvolutional Network[4], and VGGNet[6]. There have been implementations of ResNet model for many applications such as Paying more Attention[8], Large-Scale Plant Classification[9] and Imagenet classification by ResNet-50[10].

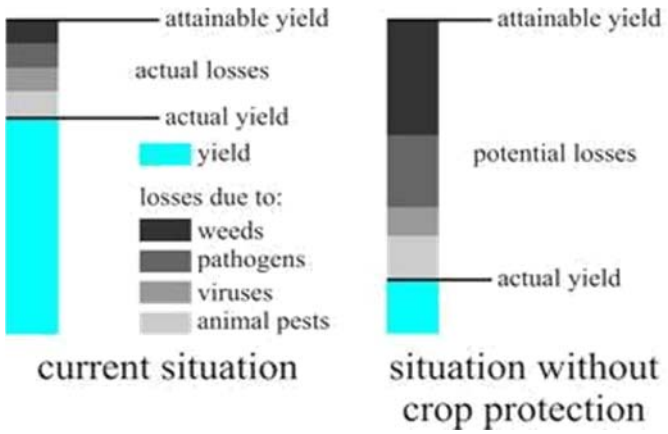


Fig. 1. CROP PROTECTION YIELD ANALYSIS

One of the research paper [1] has implemented an Apple Leaf Diseases Classifier which uses a combination of Alex Precursor and Cascade Inception architecture to classify leaf diseases. It also provides a good comparison between their model and the other models like SVM, Back Propagation (BP) Neural Network, AlexNet, GoogleNet, Resnet-20 and VGGNet-16 on the basis of performance, accuracy, convergence rate (with respect to epochs) and the computational resources required. These criteria were tested on the apple disease dataset comprising of 1053 images. The classifier that was built gave an accuracy of 97.62%

Wang-Su's[2] work makes use of the original GoogleNet and a modified version of GoogleNet to classify 3767 images into 8 classes. These images were taken from the Flavia dataset[28]. The modified version of GoogleNet contains 2x more inception modules compared to the standard GoogleNet. On comparing the two networks based on accuracy and performance. The modified version of GoogleNet performed marginally better compared to its counterpart.

A paper by Sue Han Lee[4] makes use of Caffe framework and Back-Propogation mechanism in training the model.One of the approaches used in understanding the internal working of the CNN model and visualizing the selected filters was done using Deconvolutional Network. The Deconvolutional Network provides a function that enables us to see the feature map at each layer by deconvoluting and unpooling down to input image pixel. The models were trained on two datasets, MalayaKew Dataset(original) and addition of local leaf data to original dataset. Experimental results show that the model trained on the modified dataset yielded only a small increase in the accuracy of the model when compared to the model trained on the original dataset.

In another paper that proposes a plant identification system [6], the standard VGGNet-16 architecture was taken and modified by adding more convolution layers for learning the combined species and organ features resulting the high-level fusion architecture model. The high-level fusion architecture model model and Finetuned VGGNet-16 were trained on PlantClef2015 dataset. According to the experimental results, the High-Level fusion architecture didn't perform better compared to the fine tuned VGGNet-16.

Ignacio's[9] white paper discusses the implementation of a large-scale plant classification model which uses the ResNet-50 architecture. The performance of the model was tested on the Portuguese flora dataset[19], iNaturalist dataset and Google Search Image results.

III. PROPOSED METHOD

A. Dataset

The dataset used in this study is called the PlantVillage Dataset was obtained from SP Mohanty's Git-Hub repository. The dataset comprised of raw images, data distribution for SVM and other useful data. The raw images consisted each of colour, grayscale and segmented images. Each category resulting close to 55k images which includes 38 different classes. Altogether there are 38 classes present in the dataset, our models were trained on 29 classes, and selection was done by selecting classes with atleast one disease and one healthy class each, so that the model learns to distinguish the disease. Coloured (RGB) images were used in this study to train and classify the diseases.Hence the selected dataset comprises of 29 classes and 36k images. Hierarchical division is done on the selected dataset as shown in Table I and resulting into 8 superficial classes. Now the dataset is split into training and validation sets with 80% and 20% of the dataset respectively. Thus, we have 28938 and 7210 images for training and validation sets respectively. Fig. 2 shows images from a few classes of the dataset.

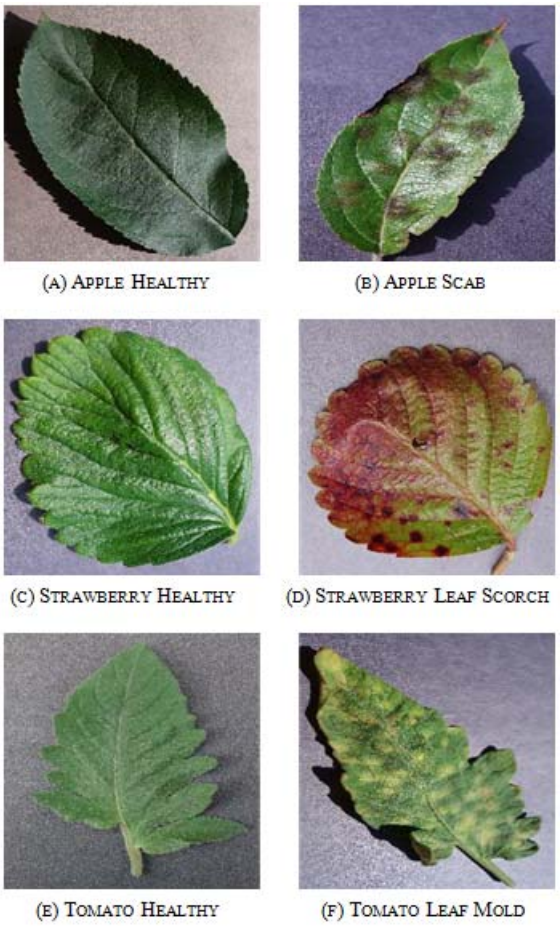


Fig. 2. IMAGES FROM THE DATASET

Table I. DATASET BREAKUP

Leaf Category	Images
Apple	3171
Cherry	1906
Grape	4062
Peach	2657
Pepper	2475
Potato	2152
Strawberry	1565
Tomato	18170
Total	36148

B. Data Preprocessing and Augmentation

Image augmentation plays a key role in building an effective image classifiers. Though datasets may contain anywhere from hundreds to a couple of thousand training examples, the variety might still not be enough to build an accurate model. Some of the many image augmentation options are flipping the image vertically/horizontally, rotating through various angles and scaling the image. These augmentations help increase the relevant data in a dataset.

The size of each image in the PlantVillage dataset is found to be 256x256 pixels. The data processing and image augmentation are done using the Keras deep-learning framework. The augmentation options used for training are as follows:

Rotation - To rotate a training image randomly over various angles

Brightness - Helps the model to adapt to variation in lighting while feeding images of varying brightness during training

Shear - Adjust the shearing angle of the image in clockwise/counter-clockwise direction

Zoom - Provides the input image scaled by various factors

Vertical/Horizontal Flip - The image is randomly flipped about the vertical/horizontal axis.

C. *System Overview*

The whole process is divided into 3 stages as in Fig. 3:

1. An input image is initially taken, A You Only Look Once (YOLOv3)[20] object detector is run over the input image to obtain the coordinates of bounding boxes around leaves present in the image, if any. The detector divides the input image into a grid and then analyzes every cell to identify features of the target object. The adjacent cells where the features are detected with high confidence are then put together to produce the output of the model.
2. The leaves are then cropped out of the image using the OpenCV library using the given coordinates. These extracted images are passed as input to a CNN Classifier which classifies the input into the 8 classes of plants from the dataset.
3. 8 CNN classifiers are trained to identify the diseases of each of the 8 plant classes. The result from stage 2 is used to call the classifier that has been trained to classify the different diseases for that plant. If there are none, the leaf would be classified as 'Healthy'.

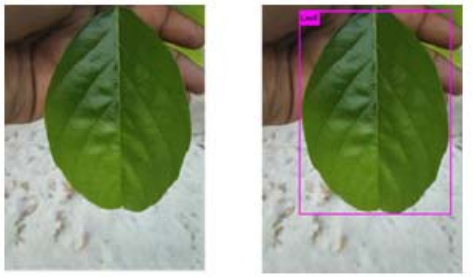
Fig. 3. BLOCK DIAGRAM OF THE IDENTIFICATION SYSTEM

Every stage in this end-to-end system was trained on the same dataset to provide better accuracy and consistency.

IV. EXPERIMENTATION& RESULTS

A. *YOLOv3 Object Detector*

An object detector that uses the YOLOv3 algorithm was trained to detect leaves in a given image. The images were manually annotated using an annotation tool. This had to be done in order to suit the use-case. This detector returns the (x,y) coordinates of two opposite corners of the bounding box for an object. The input Fig. 4a produces an output Fig. 4b obtained when the bounding box is drawn onto the image. These coordinates are then used to crop the object from the input image.



(A) INPUT IMAGE (B) LEAF DETECTOR OUTPUT

Fig. 4. YOLOV3 OBJECT DETECTOR RESULTS

B. *Simple Classifier Approach*

The classifier network was trained using a structure built upon the network from [3]. This network had to undergo few minor changes which included the scaling of the input size image from 256x256 px to 96x96 px due to constraints on graphics memory, changes also included an addition of a dense layer prior to the output layers and choosing Adam[21] as an optimizer. All the layers use either of Rectified Linear Unit (ReLU) or Softmax[23] activation functions. The resulting structure of the network is as shown in Table II.

Table II. STRUCTURE OF SIMPLE CLASSIFIER

Layer (Type)	Output Shape	Kernel Size / Function
Convolution	92x92x30	5x5
Max Pooling Padded	88x88x10	5x5
Convolution	84x84x30	5x5
Max Pooling Padded	80x80x30	5x5
Dense	1024	ReLU
Dense	700	ReLU
Dense	700	ReLU
Dense	8	Softmax

After training the model from scratch for 20 epochs, It was found that the model converged at an accuracy of 50% to 50.5%. Fig. 5 shows a plot of the accuracy of the model at the end of every epoch.

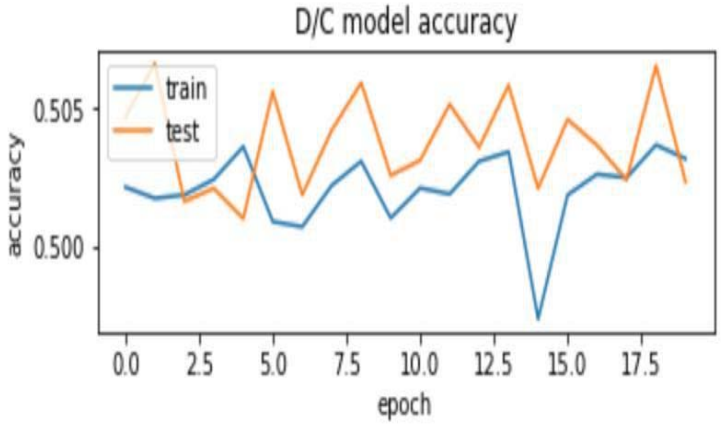


Fig. 5. ACCURACY OF SIMPLE CLASSIFIER

C. *Using VGGNet Structure*

Due to the poor accuracy seen using the classifier mentioned earlier, it was decided to adopt a proven network structure, VGG16 in this case. The model was trained from scratch after modifying the output layer to satisfy the classification requirements. It was observed that the model had an accuracy of 50.26% from the graph in Fig. 6 and converges very slowly. Due to this fact, it was decided to use transfer learning to have faster training times and better accuracy. These advantages are because the model has learned to recognize most of the features and must only undergo a few epochs of training in order to improve accuracy and fine-tune prior knowledge to suit the application.

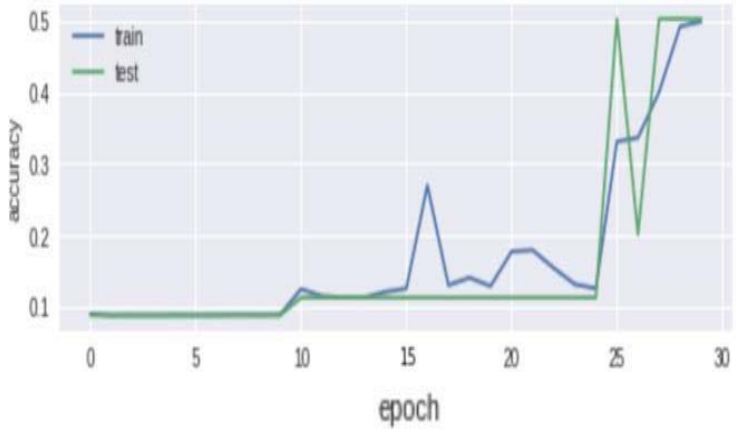


Fig. 6. ACCURACY OF VGG16 CLASSIFIER

D. Transfer Learning with ResNet

The weights of ResNet18 trained on the ImageNet dataset were used. The network was then fine-tuned to obtain a better accuracy on the given problem. All the layers except the last 3 ResNet blocks were frozen during training. The initial blocks are to be frozen as these layers learn more abstract features which apply to a large variety of classification problems. The weights of the last few layers, however, are to be modified as these are responsible for learning more task-specific features. The accuracy of the system after 25 epochs drastically increased to 78% as in Fig. 7. The higher accuracy is because ResNet's are different in its approach to the forward propagation step. Each ResNet block sends the error (Residue) to the following block which acts as a self-correcting mechanism. This lead to the decision of using this model for both stages II and III of the system.

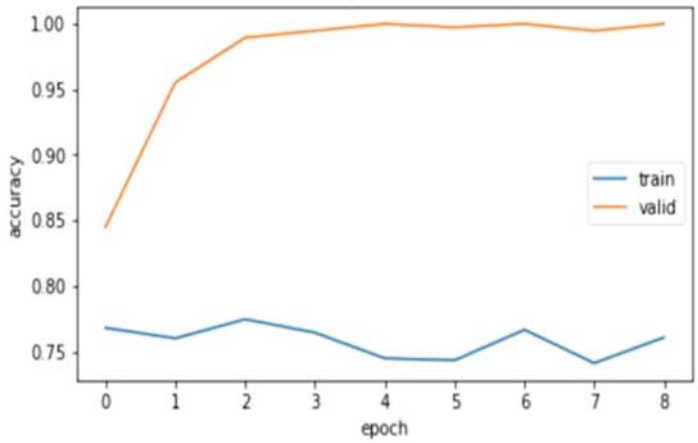


Fig. 7. ACCURACY OF RESNET18 CLASSIFIER

E. Further Improvisation

During the classification of a leaf based on plant, it was observed that in spite of augmenting the data over a wide range of rotations the model performed better when images were closer to upright positions than when rotated over a large angle. To overcome this challenge and achieve a better accuracy, the input cropped image is rotated around in increments of 15° which gives 24 different rotations. Each of these rotations is fed into the classifier to obtain predictions. The majority of these predictions are then taken as the final output of the model. This helped increase the accuracy of the system to 96%. The reason behind low increments in the rotation is to obtain higher rotated samples. As the model performed better on most rotations, more rotations would lead to lesser error. This solution was vital to the overall accuracy of the model as a wrongly classified plant could drive the system in the wrong direction.

V. FUTURE WORK& CONCLUSIONS

The above described system could be extended to be as a real-time system based on video input that could open a possibility to unattended plant care. Another aspect that could be added to the given system would be have the intelligent system suggest a cure for the identified disease. A plant disease identification system with an accuracy of 96% was developed. Research shows that management of crop diseases can help improve yield by about 50% as seen in Fig. 1. The implementation of more accurate deep learning systems could help in better diagnosis of diseases in crops as these analyze down to the smallest unit of an image, a pixel. This level of detail cannot be analyzed using the naked eye.

REFERENCES

[1] Liu, Bin, et al. "Identification of apple leaf diseases based on deep convolutional neural networks." Symmetry 10.1 (2017): 11..
[2] Jeon, Wang-Su, and Sang-Yong Rhee. "Plant leaf recognition using a convolution neural network." International Journal of Fuzzy Logic and Intelligent Systems 17.1 (2017): 26-34.
[3] Amara, Jihen, Bassem Bouaziz, and Alsayed Algergawy. "A Deep Learning-based Approach for Banana Leaf Diseases Classification." BTW (Workshops). 2017.
[4] Lee, Sue Han, et al. "How deep learning extracts and learns leaf features for plant classification." Pattern Recognition 71 (2017): 1-13.
[5] Sladojevic, Srdjan, et al. "Deep neural networks based recognition of plant diseases by leaf image classification." Computational intelligence and neuroscience 2016 (2016).
[6] Lee, Sue Han, et al. "Plant Identification System based on a Convolutional Neural Network for the LifeClef 2016 Plant Classification Task." CLEF (Working Notes). 2016.
[7] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
[8] Zagoruyko, Sergey, and Nikos Komodakis. "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer." arXiv preprint arXiv:1612.03928 (2016).

[9] Heredia, Ignacio. "Large-scale plant classification with deep neural networks." Proceedings of the Computing Frontiers Conference. ACM, 2017.
[10] Akiba, Takuya, Shuji Suzuki, and Keisuke Fukuda. "Extremely large minibatch SGD: training resnet-50 on imagenet in 15 minutes." arXiv preprint arXiv:1711.04325 (2017).
[11] Jaware, Tushar H., Ravindra D. Badgujar, and Prashant G. Patil. "Crop disease detection using image segmentation." World Journal of Science and Technology 2.4 (2012): 190-194.
[12] Hewitt, Charlie and Marwa Mahmoud. “Shape-only Features for Plant Leaf Identification.” CoRR abs/1811.08398 (2018): n. pag.
[13] Camargo, A., and J. S. Smith. "An image-processing based algorithm to automatically identify plant disease visual symptoms." Biosystems engineering 102.1 (2009): 9-21.
[14] Munisami, Trishen, et al. "Plant leaf recognition using shape features and colour histogram with K-nearest neighbour classifiers." Procedia Computer Science 58 (2015): 740-747.
[15] Al-Hiary, Heba, et al. "Fast and accurate detection and classification of plant diseases." International Journal of Computer Applications 17.1 (2011): 31-38.
[16] http://flavia.sourceforge.net/Naikwadi, Smita, and Niket Amoda. "Advances in image processing for detection of plant diseases." International journal of application or innovation in engineering & management (IJAIEM) 2.11 (2013).
[17] Kadir, Abdul, ehttp://flavia.sourceforge.net/t al. "Leaf classification using shape, color, and texture features." arXiv preprint arXiv:1401.4447 (2013).
[18] SOURCEFORGE.NET. Leaf recognition algorithm for plant classification using probabilistic neural network. Available http://flavia.sourceforge.net/.
[19] Carapeto A., Porto M., Araújo P.V., Clamote F., Lourenço J., Pereira A.J., Almeida J.D., Holyoak D.T., Pereira P., Portela-Pereira E., Silva A., Aguiar C., Henriques T.M., Gomes C.T., Silveira P., Schwarzer U., Caraça R., Chozas S., Canha P., Covelo F., Marabuto E., Farminhão J., Ribeiro S., Cardoso P., Peixoto M., Guiomar N., Rosa-Pinto J.M., Jacinto V., Engels H., Silva V., Clemente A., Silva C. et al. (2016). Flora-On: Interactive Flora of Portugal. Sociedade Portuguesa de Botânica. Online: http://www.flora-on.pt/
[20] Redmon, Joseph, and Ali Farhadi. "Yolov3: An incremental improvement." arXiv preprint arXiv:1804.02767 (2018).
[21] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).
[22] Goel, Surbhi, et al. "Reliably learning the relu in polynomial time." arXiv preprint arXiv:1611.10258 (2016).™
[23] Gao, Bolin & Pavel, Lacra. (2017). On the Properties of the Softmax Function with Application in Game Theory and Reinforcement Learning.