Cassava Plant Leaf Disease Detection

Yogeshwar Shendye

Department of Master in Computer Applications, Finolex Academy of Management and Technology, Ratnagiri, Maharashtra, India
yushendye[at]gmail.com

Abstract: ***There are various machine learning algorithms being implemented across the agricultural domain as well as other computer vision domains for the image classification problems as well as object detection problems. These algorithms work on feature extraction from the images. One of the most used algorithms is Convolutional Neural Network (CNN), which helps in feature extraction. Another method which is currently ruling the realm of machine learning is transfer learning, where the knowledge gained by machine while learning to solve one problem is applied for solving another problem. This paper demonstrates how various CNN architectures and transfer learning techniques can be applied for the disease detection in cassava plant.***

Keywords: Image Classification, Cassava, Agriculture, Convolutional neural networks, transfer learning

1. Introduction

Manually gathering of the agriculture based data such as images of infected crops and healthy crops is a challenge in 21st century. The reason being it is very time consuming as well it is not much cost effective either. An automated system for this type of task seems a viable option. It will increase the speed of the process as well as make accurate predictions.

This system takes into account the features of the leaf like color, shape of the leaf, texture etc are used to differentiate the infected and healthy plant. The main idea of this differentiation is, no two infections leave the same mark on the leaf as another. The infection in the cassava plant does have other effects as well, but leaves of the plant are the most effective way of disease detection. The effect of the disease remains for long time on the plant leaf.

Not only cassava, but also most of the crop disease detection systems heavily rely on the leaves of respective plants for identifying the type of infection, and so far the results are also promising. However disease detection using this method also has its own challenges.

The very source of producing image data, is the camera. In the industry, there are various cameras available ranging from 2 mega pixel as the lowest resolution to very high resolution cameras, having resolution of more than 256 mega pixels. Even if we decide on using a single resolution for all images, the problems still arise like the color of the image, brightness, time when the image was captured, background noise and so on.

The images used for the training are collected by Makerere University, Africa. Even in this dataset, the images are captured from various sources and it still has the problems like background noise.

Main objective of the system is to develop a application which recognizes the disease and alerts the farmer accordingly. As a next step, I have also added the functionality to communicate the data regarding identification of crop back to the developers, so that we can have more and more data to work upon, making the application more precise.

2. Literature Review

1) Detection of plant leaf diseases using image segmentation and soft computing techniques:

Agricultural productivity is of the utmost importance in countries where it is the primary source of both food and the income. That is why; there is a lot of research going on applying various computer vision methodologies to achieve an accurate prediction result regarding the health of the plant. Using automatic detection systems is extremely beneficial for the farmers as these systems use detection of early symptoms that appear on the plant in form of unusual texture on the leafs. This paper used a segmentation technique that focuses on classification and automatic detection of plant disease [1]. It also emphasizes on various disease detection techniques which are proved efficient till date.

2) Analysis of previous image classification models:

There are various models created in the past using the convolutional neural networks approach [3]. Some of the most successful models are VGG-16, VGG-19, mobilenet and inception model.

These models have achieved a very high accuracy in most of the computer vision tasks. VGG16 was considered as the deepest network, which scored 92.7% accuracy on the imagenet dataset, which consisted 14 million images of 1000 different classes. It replaced large kernel sized filters 11 from 1st convolutional layer and 5 from second convolutional layer with multiple 3*3 filters along with the pooling layers. Successor of VGG-16 is VGG-19. VGG-19 is also trained for classification of 1000 different classes. VGG-19 consists of total 19 layers. It has 16 convolutional layers with 5 maxpooling layers, 3 fully connected layers, also known as dense layers followed by a single maxpooling layer.

Another convolutional model is ResNet model abbreviation of residual network model. ResNet50, which is one of the versions built under ResNet, has achieved more accuracy than both VGG-16 and VGG-19. ResNet50 is considered deeper network, and hence it sometimes fluctuates the accuracy (accuracy first goes up and then comes down). That is why, ResNet is not preferred much.

3) Deep learning for image classification

Deep learning is another methodology used in machine learning in which the neurons are used for learning various features of the data. Deep learning based network architecture such as CNN, ANN and RNN come handy while dealing with image data. CNN (Convolutional neural networks). It consists of a input layer, one or more hidden layers that perform convolution operation and finally an output layer. The input layer converts the image into a grayscale, where computer assigns some value to the pixel based on the pixel value. The hidden layers each comprises of multiple neurons and each have their own activation function. This convolutional layer takes in the input image and output various features. This process is completed with the help of convolutional strides and pooling layers.

3. Proposed Methodology

1) Data Collection

The data was openly available on kaggle competition's page. The data was kept in directory named "training_images" and one CSV file was also provided. This file had stored the data in format

Image_id gives us the filename and label gives us the predicted disease. One JSON file was also provided the mapping between label and the disease name structured as follows:

```
{
"0": "Cassava Bacterial Blight (CBB)",
"1": "Cassava Brown Streak Disease (CBSD)",
"2": "Cassava Green Mottle (CGM)",
"3": "Cassava Mosaic Disease (CMD)",
"4": "Healthy"
}
```

This data is collected by Makerere AI lab. This is a dataset of 21,367 labeled images collected during a regular survey in Uganda. Most images were crowd sourced from farmers taking photos of their gardens, and annotated by experts at the National Crops Resources Research Institute (NaCRRI) in collaboration with the AI lab at Makerere University, Kampala. This is in a format that most realistically represents what farmers would need to diagnose in real life.

2) Data Visualization

Figure 1: CBB

Figure 2: CBSD

Figure 3: CGM

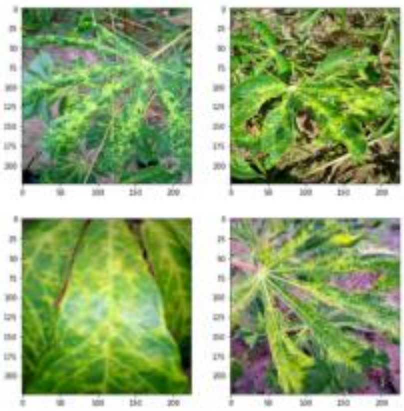


Figure 4: CMD

Figure 5: Healthy

Training and validation splitting:
70% and 30% splitting criteria was used for splitting up data. This gave us 70% of data as a training set and remaining 30% of the dataset was decided to be used for the validation purpose.

Configurations used

Table 1: Various configurations used for model training and callbacks

S.No	Parameter	Value
1	learning_rate	1e-04
2	min_delta (early stopping)	0.001
3	patience(early stopping)	5
4	Factor by which learning rate will drop	0.3
5	Patience(for learning rate scheduling)	5
6	Optimizer	Adam
7	Loss	Categorical Cross Entropy

Model creation and training:
Models like VGG-16[4] and VGG-19[4], mobilenet [6] are very deep and therefore able to extract more and more features from the image. Instead of using their complete architecture, only the neck was decided to be used. For training parameters from table 1 were used at initial stage for every architecture.

Experiment with custom model:
CNN model was developed for feature extraction and disease prediction. It has the architecture is as follows:

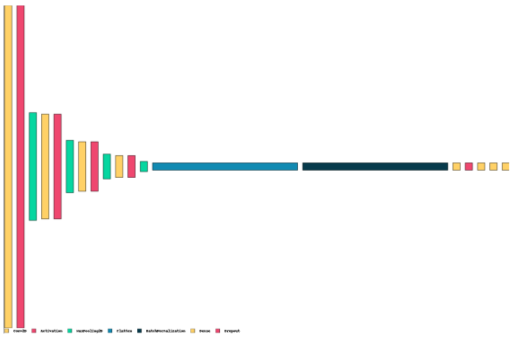


Figure 6: Architecture of Custom model

Using this custom model I got the training accuracy of 68% with training loss 0.82 and validation accuracy 67% with validation loss of 0.88

Experiment with VGG-16 architecture:
Architecture:

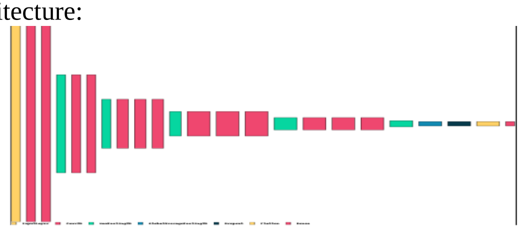


Figure 7: Architecture of VGG 16

This model uses the neck of VGG-16[4] for feature extraction and then uses layers like Global average pooling followed by a dropout layer followed by a flattening layer followed by two fully connected dense layers.

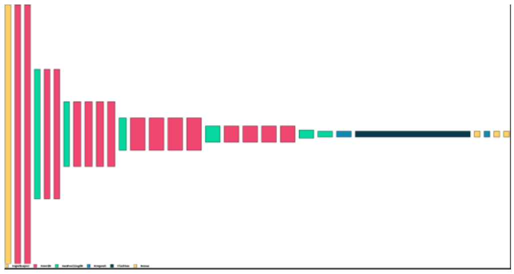
For this model the adam optimizer with learning rate of 10^{-4} was used along with learning day decay. To stop the overfitting problem early stopping was used. While training the learning rate was decreased two times and the training stopped after the 11th epoch.

This model was able to get the training accuracy of 93.46% along with the validation accuracy of 84.68%. The learning loss was 0.1968 and validation loss was 0.5316.

This model was not overfitting but it was not performing very well. Therefore I decided to go with deeper architecture like VGG-19.

Experiment with VGG-19 architecture:

Architecture



This model uses the neck of VGG-19 for feature extraction and then uses layers like max pooling layer followed by a dropout layer, followed by flattening layer followed by two fully connected dense layers having a dropout layer added in between them to avoid overfitting.

In this model, Adam optimizer with learning rate of 10^{-03} was used along with the learning rate decay, an early stop mechanism. Despite having a deeper architecture this model was performing worse than the VGG-16 model. Accuracy of 92.12% percent with validation accuracy of 80.60% was obtained. The training loss was 0.2 whereas validation loss was 0.67.

Using Transfer learning
For transfer learning [7] various models were tested such as ResNet50, NASNet, EfficientNet models but they also scored accuracy near 84%. After a lot of experiments, a cassava model trained by google was found to be the most efficient one. This model scores training accuracy of 89% with 0.2 loss and validation accuracy of 90% with 0.1 loss. There were some models that were very huge in size but they were not giving as precise results as google's cassava model.

4. Results

Results were calculated two times. First without label smoothing and then with label smoothing set to 0.3 The obtained results were as follows:

Table 2: Comparison of performance using various models (without label smoothing)

S. No	Architecture	Training accuracy	Training loss	Validation accuracy	Validation loss
1	Custom model	68	0.82	67	0.88
2	VGG 16	93	0.13	84	0.53
3	VGG 19	92	0.2	80	0.67
3	MobileNet	93	0.96	80	0.98
5	Transfer learning	93	0.17	89.75	0.30

Table 3: Comparison of performance using various models (without label smoothing)

S. No	Architecture	Training accuracy	Training loss	Validation accuracy	Validation loss
1	Custom model	78	0.82	75	0.88
2	VGG 16	93	0.13	84	0.53
3	VGG 19	92	0.5	84	0.9
3	MobileNet	95	04	79	0.5
5	Transfer learning	93	0.17	89.75	0.30

5. Conclusion

Models such as VGG-16, VGG-19, Mobilenet(V1 and V2) were tried. All those models reached the training accuracy upto 84%. The model was performing better when trained with a learning rate scheduler. After introducing the dropout layer the model was able to learn without overfitting.

When trained with a transfer learning model with label smoothing model performance increased by 3%. When retrained the model with cassava hub layer provided by google, the top 89.75% accuracy was achieved. Label smoothing had a positive effect in some architectures whereas in some architectures it had no effect.

References

[1] Vijai Singh, A K Misra "Detection of plant leaf diseases using image segmentation and soft computing techniques", Computer Science Department, IMS Engineering College, Ghaziabad, UP, India

[2] M Manoj Krishna, M Neelima, M Harshali, M Venu Gopala Rao "Image classification using deep learning" Department of ECE, KLEF India

[3] Jiuxiang Gu, Zhenhua Wang,, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, TingLiu, Xingxing Wang, Li Wang, Gang Wang, Jianfei Cai, Tsuhan Chen, "Recent advances in convolutional neural networks"

[4] Karen Simonyan, Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition"

[5] Ernest Mwebaze, Timnit Gebru, Andrea Frome, Solomon Nsumba, Jeremy Tusubira "iCassava 2019 Fine-Grained Visual Categorization Challenge"

[6] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications"

[7] Anjaneya Teja Kalvakolanu, "Plant Disease Detection Using Deep learning"