Analysis of Different CNN Models for Plant Disease Detection and Identification

Yash Dhage[1], Piyush Dharmani[2], Umesh Jain[3], Ms. Sanober Sultana Shaikh[4]

[1,2,3,4]*Information Technology Department, Thadomal Shahani Engineering College, Mumbai, India*

Abstract - *India is the 2nd largest producer of agricultural product with agricultural activity of $375.61 billion. India's contribution to total agricultural product is 7.39%. Agricultural activities contribute 15.4% to India's GDP. In a world driven by smart cell phones, and processors that can assist computer vision and deep learning, it is possible to help farmers increase their annual yield of crops. Our aim is to work towards making plants' diseases detection easier for farmers and study how convolutional neural networks classify images of different plants as healthy and unhealthy. We have used PlantVillage dataset has 54, 303 healthy and unhealthy leaf images clicked in controlled environments, divided into 38 categories by species and diseases. The average accuracy after training various models over this dataset comes out to be 97.35%.*

Keywords: **Plant disease detection, Machine learning, Convolutional Neural Networks, VGG16, ResNet152, InceptionV3, Xception, MobileNet, statistical models, transfer learning.**

1. INTRODUCTION

The development in agriculture related technologies has boosted the annual yield of crops worldwide [1]. Now the challenge before us is to prevent the loss of crops due to diseases and pests. More than 50% [2] of the yield is lost due to pests and diseases. This issue has a negative impact on the overall food supply of our country as well the financial earning of the farmers. About 80% of the total yield comes from farmers who have small scale farms. Many families are dependent on their crops, not just as a source of primary income but also for the food on their plates. It is found out that a major portion of the people lying at the bottom of the hunger index belong to such farmers' household.

Various measures have been implemented in order to reduce the crop loss. Spraying of fertilizers and fertilizers is the most effective and widely used method used to prevent crops being damaged. It is a pro-active approach and results are notably satisfactory but the insufficient knowledge on the amount of usage and the dangers on the consumers' body after eating such chemically sprayed crops are negative. The nutrition content is compromised or the chemicals lead to diseases in the human body. Apart from this method, it is crucial to identify the disease of the plant much before it begins to spread throughout the yield. Various farmers helpline numbers and farmers education centers were launched in order to educate the farmers about the threats to their yields and the remedies available. The fast-growing reach of mobile phones has boosted this initiative as it is easier for farmers to contact the authorities from anywhere at any time. The addition of fast internet services, HD cameras and remarkable computing resources in mobile phones has made it easier for remote authorities to detect diseases in a plant. Independent of these authorities, the whole disease identification process can be automated. There are applications such as Plantix which help the farmer to detect the plants disease by clicking a picture of the affected plant leaf and then get the suitable remedy's information right on their fingertips. These applications make use of computer vision to classify the leaves based on the plant's type and the diseases it has caught.

In our research, we have used the PlantVillage [3] dataset which has 38 classes of diseases of various plants along with healthy leaves images as well. The photos are taken under controlled and sophisticated environment hence the dataset didn't require any cleaning. Images are available in coloured, grayscale and segmented formats. The results obtained from the PlantVillage dataset won't necessarily match the results obtained by processing images taken in uncontrolled environments or images found on the internet. Instead, this research helps in building an understanding regarding how the convolutional neural networks classify images based on distinctive features and patterns of those images.

Detection of plants diseases on a mobile phone can be very effective in a country like ours because its free, easy to use and quick [4]. The machine learning takes time but once the model is trained, it can be implemented on any platform and the results are obtained instantly. The images are processed in the model which looks for similar patterns across various images and classifies images based on those patterns. In the training phase, the model identifies spots and patches on the leaves and matches the observations with other leaves. It creates an array with probabilities for every class it identifies. When the model is fed a new image, it studies the patterns on the leaf and then tells us which disease the leaf might be having. The model considers the three formats of same image: Coloured, Grayscale, Segmented.

2. STUDY OF LITERATURE

Many methods are already used in order to detect plants diseases such as visible light imaging, hyper spectral imaging, chlorophyll florescence imaging and thermal imaging. But all these methods can be affected by external factors such as environmental and biological complexities. In such a scenario, CNN provides a highly effective and efficient option. The model used to identify images can be trained using multiple techniques such as VGG16, ResNet152, InceptionV3, Xception, MobileNet. The results of each of these techniques are compared later in this paper.

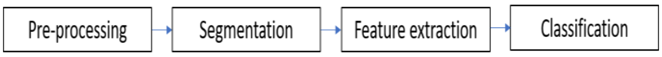

Fig -1: Steps in CNN processing

3. METHODOLOGY

The biggest challenge before us was to find a suitable dataset with good quality and classification of images. [5] The Plantix dataset isn't open hence we had to use the PlantVillage dataset which has 54,303 images of leaves, healthy as well as infected. The images are stored in different folders. The first level of classification is on the basis of the type of image viz. Coloured, Grayscale, Segmented. Further, each of these folders contain folders with each type of disease class. Total 38 classes are available. Each folder follows a naming convention [Plant name] _ [Disease name] and the healthy leaf images are stored as [Plant name] _healthy. We used TensorFlow [5] and Keras [6] Python libraries for pre-processing and processing of images by applying different models. Tensor flow is used for machine learning and it is open source. It has state-of-the-art features, tools that are comprehensive, flexible according to the developers' needs and community resources that allows us to easily develop ML powered applications. Keras is a high-level neural networks API, coded in Python and can run on top of TensorFlow, CNTK or Theano. It was developed to achieve fast experimentation.

Fig -2: Example image of Tomato leaf from dataset

The images are split into 80% training images and 20% testing images. The model will learn from the 80% images and then apply its learnt knowledge to the 20% images and report the accuracy for training and testing sets. This reading can be validated by sample inputs later. The split ratio can be 60-40, 50-50, 40-60 or 20-80 and each analysis gives a different insight into how every model works. We can give our own inputs to validate the outputs. We have considered the best and most feasible outcome in our conclusion. The images are converted into 256*256 matrices based on their pixels and each block of the matrix is given a rating on the basis of its colour density. Example: 0 rating indicates that the pixel is black and 255 rating indicates that the pixel is white. Then these images are passed for training through model of our choice. The model is loaded from the Keras library.

Fig -3: Steps taken in methodology at the client end of our demo application

To understand the working of convolutional neural networks in this application [7], we need to understand what is the meaning of feature maps and strides in a CNN model. Imagine looking at an image through a smaller window and move that window to the right and down. That way you can find features in that window, for example a particular shape or a line or a curve or a cluster of pixels etc. The network decided which feature is important while learning. You report wherever you find those features on the image to the features map. A certain pattern observed in features in a specific area can signify that a major, more significant feature exists there. For example, the first layer can be designated to look for curves specially. The next feature map could look at a combination of curves that build a larger shape or pattern. The next feature map could detect a solid shape from lines and circle features, such as a kite. Another important term to understand is 'stride'. Stride is a major component of a CNN or any neural network used to compress image or video information. Stride is a component of the neural network's filter that modifies the amount of image over the image or video. For example, if a neural network's stride is set to 1, the filter (window) will move one pixel, or 1 set unit, at a time. The size of the window effects the output generated, so stride is most often set to a whole integer.

We will consider VGG16 model in our research to understand how the CNN processes images from PlantVillage dataset. The VGG16 architecture consists of twelve convolutional layers, some of which are followed by maximum pooling layers and then four fully-connected layers and finally a 1000-way softmax classifier [8].

First and Second Layers:

The input is a 224x224x3 RGB/Grayscale/Segmented image which is passed through first two convolutional layers with 64 feature maps (filters) having window size 3×3 and similar pooling with a stride of 14. The image dimensions are modified to 224x224x64.

Then the VGG16 model applies max. pooling layer with a filter size 3×3 and a stride of two. The image we get as a result, its dimensions are decreased to 112x112x64

Third and Fourth Layer:

There are two convolutional layers with 128 feature maps having size 3×3 and a stride of 1. Then further ahead there is a max. pooling layer with filter window size 3×3 and it will move 2-pixel units after each iteration. This layer is similar to previous pooling layer except it has 128 feature maps hence the output will be reduced to 56x56x128.

Fifth and Sixth Layers:

The next two layers are convolutional layers with filter size 3×3 and a stride of one. Both layers used 256 feature maps.

Seventh to Twelfth Layer:

Here firstly we have the two sets of 3 CNN layers followed by a maximum pooling layer. All these layers have 512 filters of size 3×3 and the window moves over the image one pixel per iteration. The final size of the output gets reduced to 7x7x512.

Thirteenth Layer:

The convolutional layer's result is flattened through a layer that is fully connected which has 25088 feature maps and every one of its size is 1x1.

Fourteenth and Fifteenth Layers:

Next again, like the previous layer, we have two layers that are fully connected with 4096 units.

Output Layer:

Finally, there is a softmax output layer y with 38 possible values.

Fig -4: Snapshot of our application used to carry out the experiments and note the observations

4. RESULTS

Following are the results from the models we ran on the PlantVillage dataset. We ran every model from VGG16, Xception, InceptionV3, DenseNet201, MobileNet, ResNet152. Each of these models were ran on three sets of images: Coloured, Segmented, Grayscale.

Table -1: Coloured Images dataset

Coloured images dataset			
Model	Training	Validation	Testing
VGG16	75.75%	72.69%	82.14%
Inception V3	84.36%	70.69%	92.31%
MobileNet	91.28%	70.97%	87.89%
ResNet152	98.84%	98.60%	99.53%
DenseNet201	92.08%	59.62%	79.90%
Xception	94.31%	89.68%	93.33%

Table -2: Segmented Images dataset

Segmented images dataset			
Model	Training	Validation	Testing
VGG16	79.12%	74.46%	79.55%
Inception V3	84.75%	82.10%	88.85%
MobileNet	90.12%	83.32%	91.00%
ResNet152	83.67%	76.98%	79.43%
DenseNet201	91.62%	78.61%	95.11%
Xception	93.33%	92.24%	96.48%

Table -3: Grayscale Images dataset

Grayscale images dataset			
Model	Training	Validation	Testing
VGG16	72.90%	67.75%	77.74%
Inception V3	80.13%	72.90%	85.88%
MobileNet	87.01%	82.40%	90.58%
ResNet152	77.90%	43.63%	70.13%
DenseNet201	88.06%	69.07%	91.06%
Xception	90.71%	78.58%	92.70%

5. CONCLUSION

Based on our observations, the Resnet model works best with Coloured image set to give the most accurate values after processing. The biggest challenge here is to make the system ready for images clicked in real life environment. If the system can find out plant diseases from random images from the internet and real time images of plants then it will be a really powerful tool to boost the agricultural sector all around the world.

REFERENCES

[1] Leaf disease detection using image processing. Research by- Sujatha R, Y Sravan Kumar and Garine Uma Akhil. School of Information Technology and Engineering, VIT University, Vellore https://www.researchgate.net/publication/318109025_Leaf_Disease_Detection_using_Image_Processing

[2] Using Deep Learning for Image-Based Plant Disease Detection Sharada P. Mohanty[1], David P. Hughes and Marcel Salathé https://www.frontiersin.org/articles/10.3389/fpls.2016.01419/full

[3] PlantVillage Dataset https://www.kaggle.com/emmarex/plantdisease

[4] Disease Detection on the Plant Leaves by Deep Learning: Selected Papers from the XX International Conference on Neuroinformatics, October 8-12, 2018, Moscow, Russia. P. Goncharov (&), G. Ososkov, A. Nechaevskiy, A. Uzhinskiy 2, and I. Nestsiarenia

[5] Tensorflow docs: https://www.tensorflow.org/datasets/catalog/plant_village

[6] Keras docs- https://keras.io/

[7] Understanding of Convolutional Neural Network (CNN) — Deep Learning – Prabhu: https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148

[8] VGG16 – Implementation Using Keras, Muhammad Rizwan https://engmrk.com/vgg16-implementation-using-keras/

[9] 38 classes of leaves: https://s3.amazonaws.com/salathegroup-static/plantvillage/plantvillage-min.png

[10] DeepAI- https://deepai.org/machine-learning-glossary-and-terms/stride

[11] CNN architectures: https://medium.com/@RaghavPrabhu/cnn-architectures-lenet-alexnet-vgg-googlenet-and-resnet-7c81c017b848