International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 10, No.1 (Jan-2021)

http://dx.doi.org/10.12785/ijcds/100106

A Study of Different Disease Detection and Classification Techniques using Deep Learning for Cannabis Plant

Kanaad Pathak[1], Arti Arya[2], Prakash Hatti[3], Vidyadhar Handragal [4] and Kristopher Lee[5]

[1] Research Associate, PES University, Bangalore, India
[2]Prof. & HOD, MCA department, PES University, Bangalore, India
[3]Managing Director, Goalsr India Pvt Ltd, Bangalore, India
[4] CEO, Goalsr Inc, San Jose, USA
[5] Program Manager, Goalsr Inc., San Jose, USA

Received 6 Mar. 2020, Revised 14 Jul. 2020, Accepted 10 Sep. 2020, Published 1 Jan. 2021

Abstract: In this paper, different models for disease detection and classification are studied for cannabis plants. Cannabis plants are used for medical and recreational purposes with its recent legalization in some places. Cannabis farmers face problems in cultivation of the crop since it's susceptible to multiple disorders. With early detection of the disease in the crop it is possible to prevent large waste of yield in the crop. A real dataset is considered for disease detection and classification purposes which is a combination of text and image data and that has been collected over a period of one and a half years (Feb 2018-August 2019). The models used in this study are Fast Region Convolutional Neural Network(F-RCNN), MobileNet Single Shot Multibox Detector(MobileNet-SSD), You Only Look Once(YOLO) and Residual Network-50 Layers (ResNet50). It is found that the MobileNet-SSD provided the best accuracy amongst all the object detection models that are studied and has a lesser training time as well. ResNet 50 is used for identifying the number of images required for a good fit without having to label first and then studied for the object detection models.

Keywords: Cannabis Plant, F-RCNN, MobileNet-SSD, YOLO, ResNet50, RCNN, Disease Detection, Classification

1. INTRODUCTION

Cannabis is one of the oldest plants that is still cultivated[1] for multiple purposes. There are multiple uses of this plant like it is grown for fibre, hemp oil, for medicinal or recreational purposes etc. [1,2,4]. It is used as a pain reliever for neurogenic pain caused due to nerve damage, muscular cramps etc.[2]. Its cultivation and use is permitted legally for restricted medical purposes in some states of the US, European Countries, Canada, Israel etc. because it is also used as narcotics[3].

For the study stated in this paper, the real image and text data was collected with respect to Cannabis plants from Feb 2018 to August 2019 in Oregon, USA. The data so collected was preprocessed and object detection methods were applied to the images of Cannabis plants, so as to further analyze the detected object for disease identification and classification.

Object detection is a method in computer vision and image processing [5,7] through which one can semantically detect and classify different classes of objects. Object detection has been around for a long time, however with recent advances in deep learning techniques it is possible to detect and classify a huge variety of objects into multiple classes in real time.We therefore leverage this ability of deep neural networks to detect and classify in real-time to be able to classify different diseases occurring in the cannabis plant.This paper discusses the different deep neural classifiers available for detection and classification such as F-RCNN, ResNet 50, SSD and YOLO and how the models are evaluated on a mixture of data that is similar to the real dataset to determine the best possible model for use when the full dataset is collected. The model is then tested on part of the actual dataset to determine if the sampling method provided reliable results. The results indicate that the model performs well and provides good performance on several examples based on real datasets. The main difference between the proposed algorithm and the existing ones is that the model considers some temporal structure, i.e. it can learn the progression of the disease and the extent of damage over time. The rest of the paper is organised as follows: Section II reviews the available literature of different deep learning models for

E-mail: kanaadpathak@gmail.com, artiarya@pes.edu, prakash.hatti@goalsr.com, vhandragal@goalsr.com, kris.le@goalsr.com

object detection and disease detection and classification. Section III provides the details of the sample and real dataset. Section IV describes the implementation details and explanation of results. Section V briefly summarizes the gist of the paper as Conclusion and also talks about Future scope.

2. Related Work

Four models are applied on the above stated datasets. Three of these being object detection and classification models and one being a simple image classifier namely F-RCNN (Fast Region Convolutional Neural Network), MobileNet-SSD (MobileNet Single Shot Multibox Detector), YOLO (You Only Look Once), ResNet50 (Residual Network - 50 Layers).

A. Fast Region Convolutional Neural Network (F-RCNN)

In order to introduce F-RCNN, first Convolutional Neural Network (CNN)and Region based Convolutional Neural Network (R-CNN) are discussed. A CNN [6] greatly reduces the number of parameters in a neural network for image classification. This allows for development of larger models which was not possible in traditional Artificial Neural Networks. CNN's are great at solving problems in which features are spatially independent. For example, in case of classifying disease on a leaf, the positioning of leaf and the position of disease on the leaf is not a matter of concern. The only concern is to detect the disease regardless of its position in the given image [18]. Region based CNN(R-CNN) is a variant of CNN, used for object detection, which proposes a set of boxes on the image and checks if any of the boxes contain objects [8, 9]. Regions are extracted from the image using selective search algorithms, the boxes within these regions are generated to see if they contain any objects [7,8,9]. After the regions of interest are extracted from the image, a CNN is used for each region to extract specific features defining the region. These features are then used to detect objects within the image. There seems to be some disadvantages of R-CNN for disease detection and classification of Cannabis plants. Some of the problems associated with R-CNN are [7]:

Its training is complicated and more time consuming: A CNN is first fine-tuned on an object location (or object proposal) to a log loss function, then linear Support Vector Machines (SVMs) are fitted to the generated CNN feature maps. The SVMs then behave as object detectors, which essentially replaces the SoftMax function in a classifier. Slow speed of detection: It is slow because the computations for each object location in the forward pass are not shared in a CNN. Since the dataset contains over 10k image samples, training time and speed is an important factor, and because of the above stated drawbacks of R-CNN, Fast-Region Convolutional Neural Networks(F-RCNN) are next explored. F-RCNN aims to fix some of the problems that are seen in R-CNN. F-RCNN also has an upper hand due to its improved accuracy and improved training time. The F-RCNN method has some advantages [7] such as

- High detection quality
- Single stage training
- All layers can be updated during training
- No requirement of disk storage for feature caching

The implementation for Fast R-CNN is freely available at [20].

The network takes the entire image as input along with a set of object locations (which are in the form of a bounding box). The next step is to produce a convolutional feature map. Then, a region of interest pooling layer extracts a fixed length feature vector for each object of interest in the picture. This sequence vector is then fed into a fully connected layer that branches out into two output layers. One produces a SoftMax probability estimate over K object classes and a "background class" [7]. The other layer produces four real valued numbers, which are the coordinates for where to draw the bounding box in the output image. The two output layers use SoftMax and regression loss. The loss function is denoted by the variable ρ(x, y) [13]

$$\rho(x,y) = \frac{1}{n}\sum_i Z_i \qquad (1)$$

where Z_i is given by

Case 1: $0.5\,(x_i - y_i)^2$, if $|x_i - y_i| < 1$ (2)

Case 2: $z_i = |x_i - y_i| - 0.5$, otherwise

The detailed explanation of this loss can be found in [13, 16]. End-to-end learning is performed by back propagating the gradients through all the layers [9].

B. Single Shot Detection & MobileNet-SSD

Single Shot Detection (SSD) is basically a process of detecting objects in real time. In SSD, firstly feature maps are identified and then convolutional filters are applied to images to detect the objects. The SSD uses a feed-forward CNN to produce bounding boxes [11,12,22]. Also, the network in SSD generates a score corresponding to each object in the bounding box and also computes adjustments required for the bounding box to fit the object most appropriately [22]. Since this can be slow for processing images, a faster solution is required so that the model can be deployed on a mobile phone with average graphics processing abilities and MobileNet-SSD overcomes this shortcoming of SSD. MobileNet-SSD comprises depth wise and point wise convolutional architecture. The implementation in TensorFlow comprises the first layer being fully

convolutional and the rest of the architecture based on depth-wise separable convolutions [11, 21]. The last fully connected layer is non-linear and connects into a SoftMax layer for the final classification. The remaining layers are followed by batch normalization and ReLU [11]. The aim here is to minimize the loss function, which is essentially a combination of classification loss and regression loss. So, mathematically

$$L(x, c, l, g) = \frac{1}{n}[L_{conf}(x, c) + \alpha * L_{loc}(x, l, g)] \quad (3)$$

where 'n' is the total number of matches of default boxes. Lloc is the localization loss between the predicted box(l) and the ground truth box(g). Lconf is the softmax loss over multiple confidences(c) and α is the weight term which is usually 1. The regression loss used here is the same as that in the F-RCNN. The implementation for the MobileNet-SSD model is freely available in the TensorFlow model zoo, which can be found at [14].

C. You Only Look Once (YOLO)

YOLO is an object detection framework targeted at doing real-time object detection and is faster than SSD and MobileNet but SSD detects more accurately. The main advantage of using such a network is its speed. This type of model is implemented as a convolutional neural network as shown in Fig. 1[10]. The YOLO network accepts training data in the Pattern Analysis, Statistical Modelling and Computational Learning Visual Object Classes(PASCAL VOC) [23,24] or Common Objects in Context(COCO)[23,25] dataset format. In PASCAL VOC there is a file created corresponding to each image in a dataset but in COCO, there is one single file for the complete dataset. The bounding boxes in both the dataset formats are different. In YOLO, the network consists of twenty-four convolutional and then two fully connected layers.

The YOLO framework functions in the following way. The image is broken down into a n × n grid. Then image classification and localization are applied on each grid. YOLO predicts the bounding boxes and their corresponding class probabilities. In order to train the model labeled data is required. Every cell in the grid contains a y value, which is a one n dimensional vector. If the image is broken in to a 3 x 3 grid, and there are 3 classes that are required to be classified, the y value is a 8 dimensional vector that consists of the following

$$y = [p_c, b_x, b_y, b_h, b_w, c_1, c_2, c_3] \quad (4)$$

Where pc is a value that is either 0 or 1 and describes if there is a presence of an object in that grid cell. b(x, y, h, w) describe the coordinates of the bounding box and c1, c2, c3 describe the presence of a particular class in that grid cell, these values are also binary in nature. The bounding box prediction is decided by a formula called as the Intersection over Union (IoU). IoU calculates the intersection over union of the actual bounding box and the predicted bounding box

$$IoU = \frac{Area\ of\ intersection}{Area\ of\ union} \quad (5)$$

The IoU value must be as low as possible for a good prediction. The loss value is calculated by taking the sum squared error between the predictions and the ground truth[10]. The total loss function comprises of the following

- Classification loss
- Localization loss
- Confidence loss

The total loss is calculated as a sum of these three losses [10]. YOLO has great speed (almost real-time depending on GPU) however suffers from limitations that make its use inadmissible in this application. YOLO fails at detecting smaller sized objects with a reasonable accuracy. It is especially incapable at detecting several small sized objects that are clustered together, this is not a helpful trait for diseases that are present on leaves as small spots.

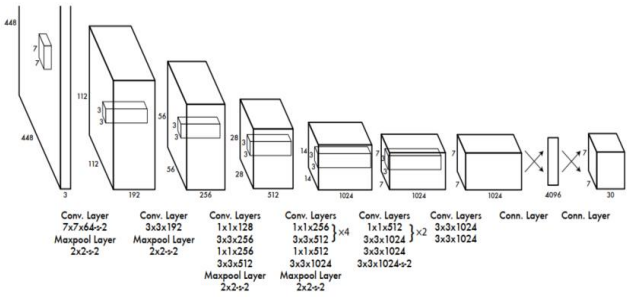

Figure. 1. YOLO architecture as described in [10]

D. Residual Network (ResNet 50)

Residual Network(ResNet 50) attempts to tackle the saturation and degradation of accuracy in deeper neural networks [17]. Instead of learning a direct mapping from say x → y for a function H (x), a residual function F (x) = H (x) − x is defined, where F (x) represents the stacked linear layers and x represents the identity function (also known as identity shortcuts), thereby forming shortcut connections [18]. The network consists of mostly 3*3 filters. CNN layers are down sampled with a stride of 2. The average pooling layer is global and is a fully connected layer with SoftMax at the end. Residual connections are encapsulated in a residual block [18]. For residual learning, when input(x) and output(y) has the

same number of dimensions then the following equation is used

$$y = F(X, \{w_i\}) + x \quad (6)$$

where F(x, {wi}), the residual mapping function, which is to be learnt. Here x and F must have the same dimensions. But if input and output have different dimensions then a linear projection Ws is used to match the dimensions as in (7) .

$$y = F(x, \{w_i\}) + W_s x \quad (7)$$

Therefore, the authors Kaiming He et.al [17] concluded that:

• ResNet converges faster than its counterpart (VGG) [17]

• Only identity shortcuts are used most of the time with projection shortcuts being used when the input dimension of the input changes.

• ResNet 50 has a top-5 error of 5.25 and a top-1 error of 20.74 [17]

Taking these points into consideration, ResNet 50 has many trainable layers and is a fast converging network compared to its counterparts.

File Name	Day of Growth	Disease Present	Name of Disease	Stage of Disease	Cure Recommendation	Light Intensity	CO2	Soil
*.JPG	NUMERIC	1/0		1/2/3	TEXT BASED RECS	SENSOR VALS	----------	
.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	

Figure 2. Format for the structured component of the dataset

3. DATASET

For this study, two datasets are considered, one is a real dataset and one a sample dataset. The structure of real and sample dataset and purpose of sample dataset is described below.

A. Real Dataset

The real dataset is collected over a period of one and a half years(Feb 2018-August 2019) in Oregon, USA. The real dataset is collected in two formats:

- Structured Data
- Image Data

The structured component of the dataset is shown in Fig. 2 The first attribute contains the file name, the file can be in any format (PNG, JPG etc.). The second attribute has the number of days that have passed since the start of the growth of the plant whose picture has been mentioned in the first column. The third attribute is a simple binary indicator whether a disease is present or not, 1 if a disease is present and 0 in the absence of any disease. The fourth attribute is the name of the disease present (in all lower-case letters); and if there are multiple diseases, it is separated by a comma. The fifth attribute describes the stage of the disease whether it is 1st or 2nd or 3rd stage. The sixth attribute is the cure notes or 'remedies' with respect to the plant disease(s) mentioned by the fourth and fifth attribute. The remaining four attributes are light intensity, carbon-di-oxide, soil moisture and soil PH that have been collected from the plant health monitoring sensor (PHMS) that is placed in the soil of the plant pot. A sample of a leaf image from the dataset is illustrated in Fig. 3.

Figure 3. Sample image of cannabis leaf

The unstructured part of the dataset just contains images that are unlabelled, all in one folder. The following six diseases and issues were tracked through the growth cycle of the plant.

- Powder mildew
- Calcium deficiency
- Heat stress
- Water issue
- Humidity
- Potassium deficiency

Each of the diseases and issues can broadly be categorized into three stages according to the farming experts so consulted. Hence in totality there are eighteen categories into which these images can be classified. However, when this study started, till that time the data so collected did not have ample number of images representing Calcium deficiency on the cannabis plant. So, while this study was conducted, Calcium deficiency was not considered and only five categories with three stages each were contemplated

Figure 4. Example of one image from each of the 6 classes

Fig. 4 shows one sample from each stage of disease or issue. The image portion of the dataset along with the annotations was converted into Pascal VOC format and COCO dataset format. Image annotation is basically labeling the images like in our case it is labeled by bounding boxes. And these labels are defined by system designers or engineers. The annotations are saved in XML files in Pascal VOC, where there is one XML file per image. In the generated XML file, the path to the file is stored in the < path > element and the bounding boxes are stored in individual < object > tags. Each bounding box is defined by the upper left and bottom right corner coordinates of the image. The only difference with the COCO format is that it stores this information in a JSON structure. Thus, one can store bounding box coordinates in array-like structures. JSON is a more popular format to pass information through the internet, hence it is beneficial to convert to the COCO format as per the model requirements.

B. Sample Dataset

Initially all the models were tested on a sample dataset that was similar to the real dataset.

Figure 5. Sample dataset

This dataset consisted of 6 categories of flowers and 6 categories of playing cards. Playing cards were purposefully included in the sample dataset so that the cards could be captured with small and large image section overlaps (as seen in last image of Fig. 5) whereas flower pictures available online did not have much of overlap. Also, there were no images of different categories of flowers that overlapped in the same picture. However, the cards dataset did exhibit this characteristic. The playing cards data was taken from images captured from Edje Electronics. Introducing playing cards images into the sample dataset is essential because the difference between different playing cards is very small, just like how the structure of the cannabis leaf will be the same but the diseases will be different on the leaf. Flowers are visually similar in nature to a cannabis leaf due to it being from a category of flora and having no fixed edge shapes. An example of this sample dataset is demonstrated in Fig. 5. As observed in the sample dataset visualization, there are twelve categories in this dataset, which should translate to the real dataset performing in a similar way on the best observed model. The sample dataset consists of the following 12 classes, which are

- Cannabis
- Daisy
- Dandelion
- Rose
- Sunflower
- Tulip
- Diamond Card
- Heart Card
- King Card
- Ace Card
- Jack Card
- Queen Card

Each of the classes in the sample dataset consist of 400 images. The average dimension of the image in the dataset is 840 x 840. So each image in the dataset has been reduced to that dimension using a python script and has no impact on features of the image and loss of significant information.

4. IMPLEMENTATION AND RESULT

The experiments were carried out on an Intel core i7 8750 H @ 2.2Ghz machine. The machine had 16GB RAM, 8 GB VRAM and Nvidia GTX 1070 GPU. The softwares so used are LabelImg, Tensorflow, Anaconda and Python. Before labelling the data, it is split into appropriate folders using a simple python script. Each image belonging to one class is put into one folder. Therefore, all the images containing sunflower will be in a single folder. In the next step the images are labelled according to their respective classes using a tool called LabelImg [20] as shown in Fig. 6.

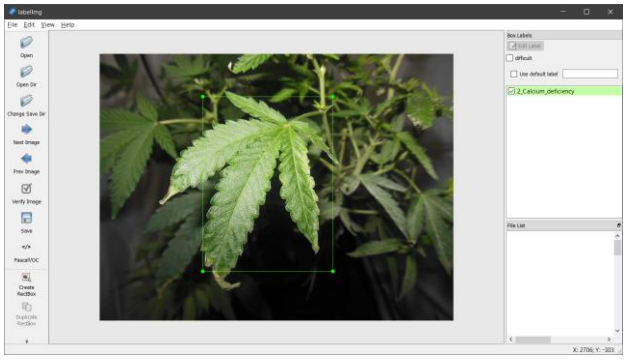

Figure 6. GUI of labeling software for drawing bounding boxes

The user must draw a rectangle around the 'area of interest' and the tool saves the coordinates of the bounding boxes in a ".xml" file format. The coordinates it saves are [xmin, ymin, xmax, ymax]. Later a script is used to extract only the coordinates of the bounding boxes along with the image file name and the class into a CSV file format. The LabelImg tool also saves the coordinates of the area of the bounding box directly in the YOLO format. The real dataset contains around 4000 images so far and the size of the dataset is still increasing as new images of the cannabis plant keep pouring in from time to time. With each image being around 6 MB, the total size of the dataset is around 15.6 GB. This is too large to process all at once, since the GPU available at hand only has limited memory. Thus, the data set is compressed to at least one third of the size without compromising on the image quality. This was done by stripping the EXIF (Exchangeable image file format) data from each image. This reduced the total size of the available dataset to about 6 GB.

A. MobileNet-SSD Results

The model was trained with the default parameters which took a time of approximately 2 hours and 20k epochs. From Fig. 7, the loss value is 0.06, which is the acceptable loss value quoted by [11,12], the authors to achieve a good training accuracy of the model. The light graph line in Fig.7 is more spiked and a smooth loss line is an approximate representation of the same. As the number of epochs are increasing the loss of the model is decreasing. Loss is a number that basically indicates how bad a model is in predictions. The loss graph demonstrated in Fig. 7 is the total loss for the model and not for a single instance. The inference graph for the model was then frozen so that it could be used with a simple NodeJS front end for testing purposes. The frozen inference graph is a graph that cannot be trained any further. This model achieved an accuracy of 85%.

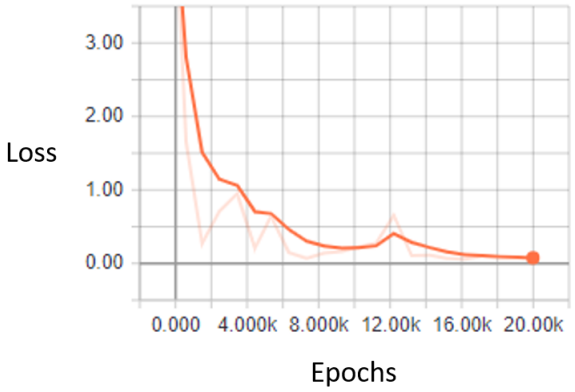

Figure 7. Epochs vs. Loss graph for F-RCNN

The architecture of the end-to-end system is demonstrated in Fig. 8 and the frozen inference graphs were swapped out in the python script component for testing.

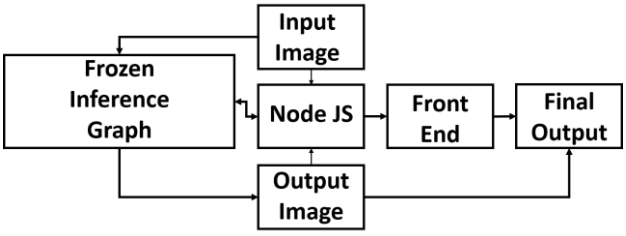

Figure 8. NodeJS architecture

B. F-RCNN Results

The F-RCNN model was found faster in training and reached a similar loss value in lesser epochs and timesteps but suffered in accuracy when multiple objects belonging to different classes were present in the same frame. Fig. 9 shows the loss value of approximately 2 at 190 epochs which when extrapolated to 350 epochs, it achieved a loss of 0.05 per epoch. Here, the number of epochs are extrapolated to 350 because for 350 epochs the loss value is 0.05 which is an acceptable value for loss for the F-RCNN model.

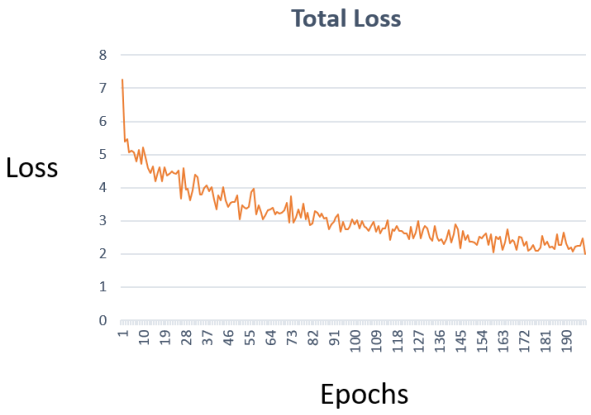

Figure 9. Epochs vs. Loss graph for F-RCN

C. YOLO

When the data set was tested with YOLO, it did not converge and when investigated, it was found that it failed to converge due to a limited number of images. As the size of dataset is increasing because new images are still getting added to the dataset collected so far, so in the future when more images are available for training, the dataset will be appropriately tested on YOLO again.

D. ResNet 50

Since ResNet 50 does not require the dataset to be labelled with bounding boxes, the dataset was split in three ways

- Five diseases with all stages at once that is with 15 categories.
- Only five diseases with no stage information that is with 5 categories.
- Two diseases with three stages each that is with 6 categories related to two diseases only.

Table I shows the accuracies of all the different splits of the data sets. Fig. 10 shows the confusion matrix for five diseases with 3 stages each. Fig. 10 clearly shows that two diseases, Powder Mildew Stage 1 and Water Issue Stage III have the best accuracy.

TABLE I
ACCURACY METRICS FOR THE DATASET

Set Type	Train Loss	Valid Loss	Error Rate	Accuracy
5 diseases all stages	0.859280	0.90504	0.371528	0.628472
5 diseases only	0.482605	0.476542	0.277778	0.7222
2 diseases 3 stages each	0.210241	0.227485	0.116384	0.883636

Fig.11 shows the loss graph when the model was trained on all five diseases with all three stages. The accuracy of the model is 62%. which is quite less than desirable. One reason identified for this is the number of images for each class and then each stage of disease is not enough.

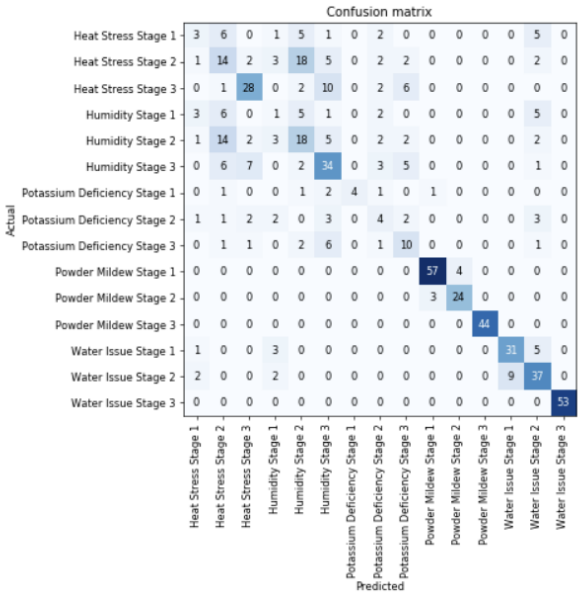

Figure 10. Confusion matrix for all diseases

Fig.11 shows us that the model is a good fit after seven epochs.

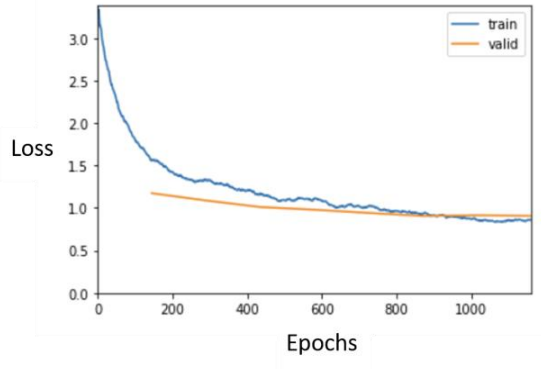

Figure 11. Epochs vs Loss graph for ResNet 50 for all the diseases and all stages

The confusion matrix for the five diseases without considering the stages (that is all three stages are considered as one stage only for each of the five diseases) is demonstrated in Fig. 12.

This confusion matrix clearly indicates that Powder Mildew and Water Issue are two diseases having the best accuracy. When the five diseases are considered without segregating three different stages then the accuracy of the model has improved from 62% to 72% because the number of images per category are increased, so the overall accuracy drastically shot up.

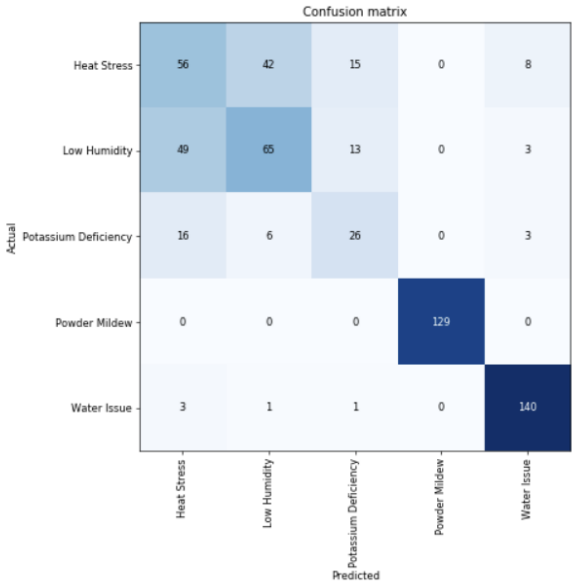

Figure 12. Confusion matrix for all diseases without stages

The training and loss graph as demonstrated in Fig. 13 shows that the model is a good fit and the validation loss is significantly better when more images are added to the dataset.

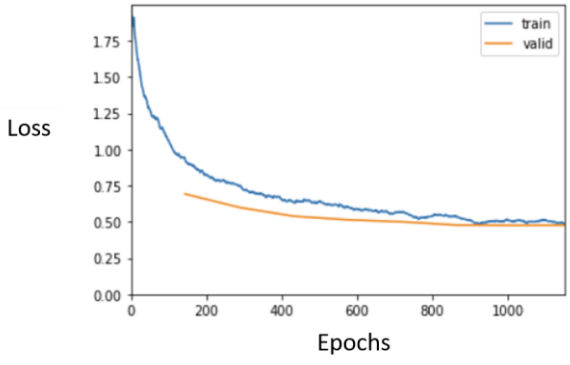

Figure 13. Epochs vs Loss graph for diseases without considering stages

As observed from table I and Fig. 14 the model performed the best in classifying two diseases with three stages each with an accuracy of 88%. This is mainly attributed to having more image data in those two classes of diseases. This proves that the model can be scaled up to identify more diseases along with their stages provided we give it more training data. The accuracy achieved is impressive considering that there are very minuscule differences between the different stages of the diseases. This model was also trained for seven epochs and hence the results are comparable to the previous outcomes of ResNet 50.

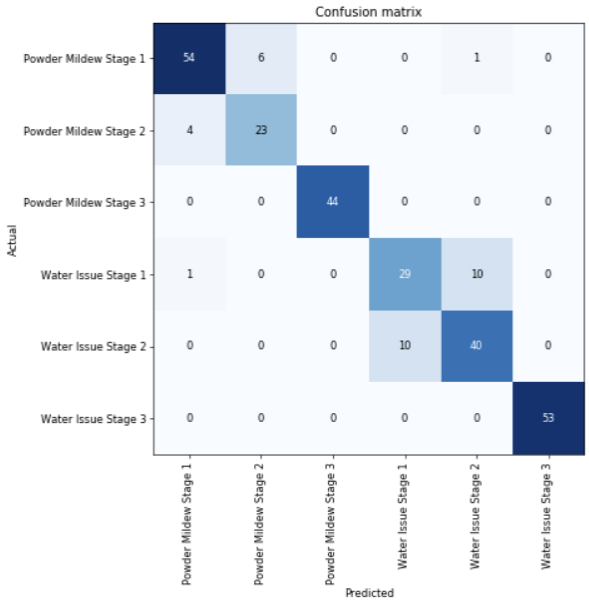

Figure 14. Confusion matrix for two disease with three stages each

It can be observed from Fig. 15 that the validation loss is significantly lower than the previous two experiments and that the model is a good fit.

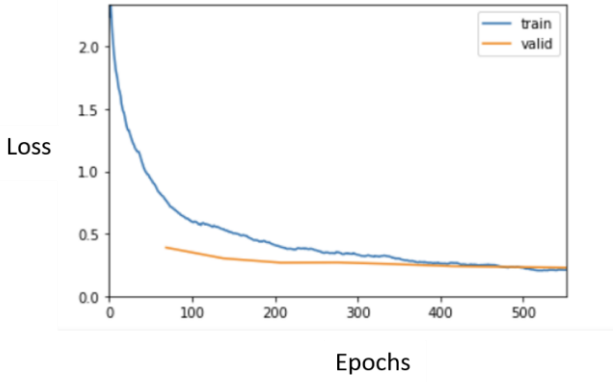

Figure 15. Two diseases with three stages each

5. Conclusion And Future Work

From the above experiments and results it can be concluded that MobileNet-SSD has achieved 85% of accuracy for all five categories of the diseases with three stages each. ResNet 50 shows an accuracy of 88% for detecting and classifying two different diseases with three stages each and for all five classes with three stages each, the accuracy is approximately 62%. The accuracy of F-RCNN was lesser than MobileNetSSD. The real dataset is yet to be fully labelled to be tested on the object detection models, however the results from the

experiments instill a confidence that a detection and localization of the diseases can be performed using the parameters set in the object detection models. In the future it is planned to incorporate the structured component of the dataset to augment the detection and classification process.

ACKNOWLEDGMENT

The authors would like to thank the farmers at Phatt Ranch, Oregon, USA for taking the time from their schedule to individually take pictures of the leaves and label the dataset collected thus far.

REFERENCES

[1] Gould, Julie. ”The cannabis crop.” Nature 525, no. 7570 (2015): S2-S2.

[2] Bifulco, Maurizio, and Simona Pisanti. ”Medicinal use of cannabis in Europe.” EMBO reports 16, no. 2 (2015): 130-132.

[3] Madras, Bertha K. ”Update of cannabis and its medical use.” Geneva: World Health Organization (2015).

[4] Volkow, Nora D., Ruben D. Baler, Wilson M. Compton, and Susan RB Weiss. ”Adverse health effects of marijuana use.” New England Journal of Medicine 370, no. 23 (2014): 2219-2227.

[5] Borji, Ali, Ming-Ming Cheng, Qibin Hou, Huaizu Jiang, and Jia Li. ”Salient object detection: A survey.” Computational Visual Media (2014): 1-34

[6] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. ”Imagenet classification with deep convolutional neural networks.” In Advances in neural information processing systems, pp. 1097-1105. 2012

[7] Girshick, Ross. ”Fast r-cnn.” In Proceedings of the IEEE international conference on computer vision, pp. 1440-1448. 2015.

[8] Tang, Tianyu, Shilin Zhou, Zhipeng Deng, Huanxin Zou, and Lin Lei. ”Vehicle detection in aerial images based on region convolutional neural networks and hard negative example mining.” Sensors 17, no. 2 (2017): 336.

[9] Wang, Xiaolong, Abhinav Shrivastava, and Abhinav Gupta. ”A-fast-rcnn: Hard positive generation via adversary for object detection.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.

[10] Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. ”You only look once: Unified, real-time object detection.” In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779- 788. 2016.

[11] Li, Yiting, Haisong Huang, Qingsheng Xie, Liguo Yao, and Qipeng Chen. ”Research on a surface defect detection algorithm based on MobileNet-SSD.” Applied Sciences 8, no. 9 (2018): 1678.

[12] Biswas, Debojit, Hongbo Su, Chengyi Wang, Aleksandar Stevanovic, and Weimin Wang. ”An automatic traffic density estimation using Single Shot Detection (SSD) and MobileNet-SSD.” Physics and Chemistry of the Earth, Parts A/B/C 110 (2019): 176-184

[13] Girshick, Ross. “Fast R-CNN.” Github, 23 Jan. 2018, github.com/rbgirshick/ fast-rcnn.

[14] TensorFlow. “Model Zoo TensorFlow.” GitHub, 18 July 2019, github.com/TensorFlow/models.

[15] “Loss Function PyTorch.” Torch.nn - PyTorch Master Documentation, PyTorch, pytorch.org/docs/master/nn.html#torch.nn.SmoothL1Loss.

[16] Gao, Hao. “Understand Single Shot MultiBox Detector (SSD) and Implement It in Pytorch.” Medium, Medium, 26 June 2018, medium.com/@smallfishbigsea/understand-ssd-and-implement-yourown-caa3232cd6ad.

[17] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. ”Deep residual learning for image recognition.” In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

[18] Jay, Prakash. ”Understanding And Implementing Architectures Of Resnet And Resnext For State-Of-The-Art Image Classification: From Microsoft To Facebook [Part 1]”. Medium, 2018, https://medium.com/@14prakash/understanding-and-implementingarchitectures-of-resnet-and-resnext-for-state-of-the-art-imagecf51669e1624. Accessed 29 Aug 2019.

[19] Albawi, Saad, Tareq Abed Mohammed, and Saad Al-Zawi. ”Understanding of a convolutional neural network.” In 2017 International Conference on Engineering and Technology (ICET), pp. 1-6. IEEE, 2017.

[20] GitHub. 2015. Tzutalin/Labelimg. [online] Available at: ¡https://github.com/tzutalin/labelImg¿ [Accessed 4 October 2019].

[21] W. Budiharto, A. A. S. Gunawan, J. S. Suroso, A. Chowanda, A. Patrik and G. Utama, ”Fast Object Detection for Quadcopter Drone Using Deep Learning,” 2018 3rd International Conference on Computer and Communication Systems (ICCCS), Nagoya, 2018, pp. 192-195, doi: 10.1109/CCOMS.2018.8463284.

[22] iu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. ”Ssd: Single shot multibox detector.” In European conference on computer vision, pp. 21-37. Springer, Cham, 2016.

[23] Zhou, Xinyi, Wei Gong, WenLong Fu, and Fengtong Du. ”Application of deep learning in object detection.” In 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), pp. 631-634. IEEE, 2017.

[24] Everingham, Mark, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. ”The pascal visual object classes (voc) challenge.” International journal of computer vision 88, no. 2 (2010): 303-338

[25] Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and C. Lawrence Zitnick. ”Microsoft ´ coco: Common objects in context.” In European conference on computer vision, pp. 740-755. Springer, Cham, 2014.

Kanaad Pathak Has completed his bachelor's of engineering from PESIT, Bangalore South Campus and an MSc from the University of Strathclyde, Scotland. Currently he is pursuing his PhD in the field of Information Retrieval in the capacity of Marie Curie Early-Stage Researcher at the University of Strathclyde. Prior to his master's degree he worked as a research assistant at PES University for one year in collaboration with Goalsr India Pvt Ltd and made several key contributions to Goalsr's IP which include the filing of two patent disclosures, both in the field of AI.

Arti Arya Arti Arya has completed BSc (Mathematics Hons) in 1994 and MSc (Mathematics) in 1996 from Delhi University. She has completed her Doctorate of Philosophy in Computer Science Engineering from Faculty of Technology and Engineering from Maharishi Dayanand University, Rohtak, Haryana in 2009. She is working as Professor and Head of MCA dept in PESIT, Bangalore South Campus. She has 19 yrs of experience in academics, of which 11yrs is of research. Her areas of interest include spatial data mining, knowledge based systems, text mining, unstructured data management, knowledge based systems, machine learning, artificial intelligence, applied numerical methods and biostatistics. She is a life member of CSI and senior member IEEE. She is on the reviewer board of many reputed International Journals.

Prakash Hatti He is Entrepreneurial Engineering Leader with 20+ years in Technology Management & Product Engineering with specialty in both start-ups and scaled growth stages. He has held a variety of leadership roles at General Electric (GE), SAMSUNG & AIRBUS. He is an alumnus of Central Institute of Plastics Engineering & Technology, Mysore-India. A certified Six Sigma Black Belt & Reliability Practitioner, he has played a key role in the construction of value engineered medical systems, various material systems, consumer electronics products & large appliances.

Vidyadhar Handragal Founder & CEO of Goalsr Inc., and a serial entrepreneur, Vidyadhar has a track record of building and managing high-performance teams & companies ranging from IT, ITES that cater to varied industry segments such as Aerospace, FMCG, Retail, Medical & Agriculture. His earlier experience includes working on Product Engineering, Release management & Test Automation with Global organizations such as Dell, Spirent Communications and HCL-CISCO in addition to his rich experience being part of successful start-ups such as Fanfare Group, Elementum and Hivepoint. Vidyadhar is an alumnus of BITS, Pilani with a Master's in Software Systems and Bachelor's Degree in Computer Science from REC, Balki.

Kristopher Lee. He is the Program Manager with Goalsr Inc., Kris is also the Cofounder of Zuna Inc., a company that makes use of innovative agriculture technology and IT data platforms for real-time analysis, risk management & providing expert managed knowledge. Kris has served with a California Technology Startup iCANN as Head of Marketing and Sales, where he was responsible for developing and executing competitive strategy, sales channel management, marketing analytics, marketing communications, and product marketing. Kris earned his BS degree in Marketing from San Francisco State University. In his spare time, he loves anything to do with cars and other automobiles.