Predicting the Presence of Seedlings and the Health of Crops using Image Classification

by Yarala Reddy

Submission date: 13-Apr-2020 07:57PM (UTC+0530)
Submission ID: 1296464863
File name: eedlings_and_the_Health_of_Crops_using_Image_Classification.docx (2.28M)
Word count: 2982
Character count: 15682

Predicting the Presence of Seedlings and the Health of Crops using Image Classification

A Project Report

Submitted by

YARALA HRUTHIK REDDY – 1700289C203
KURUBA KIRAN KUMAR – 1700228C203

Batch - 2017 CSE 1

Course
MACHINE LEARNING and DATA MINING - CSE2702

BML MUNJAL UNIVERSITY™
FROM HERE TO THE WORLD

SCHOOL OF ENGINEERING & TECHNOLOGY
BML MUNJAL UNIVERSITY

April 2020

Abstract *– An efficient food production system is one major requirements in the present world. Since, majority of the food crops grow from seeds. Classic farming involves a vast amount of labor to seed and later transplant the seedlings to a low-density tray from a high-density tray for further growth. This project proposes an image classification model that would detect the presence of seedlings and also identify the health of the crops. So, we have tried to develop a Convolutional Neural Network(CNN) which would be trained on the image samples so that it could learn patterns in order to identify the seedling's presence or absence and if present then we would like to know whether it's healthy or not. We have used 3 Conv2d layers, 3 MaxPooling layers, 3 Activation Function layers, 1 flatten and 2 dense layers. In order to improve the processing speed, we have tried to normalize the data. We have used 750 samples of Health and 850 samples of Seedlings dataset to evaluate our model. We improved the performance of the model by applying data augmentation. We got our optimum model with highest accuracy of 88.32% (in case of Health dataset) and 94.94% (in case of Seedlings dataset) when learning rate was 0.001 and batch size was 16 and 50 respectively. So, this CNN model is quite efficient in detecting the presence or absence of the seedlings and in identifying whether it is healthy or not.*

Keywords *– Learning Rate, Image Classification, Neural Networks, Overfitting, Seedlings.*

I. INTRODUCTION

Creating an efficient food production system in the most sustainable way is one of the major problems in the present world. Majority of the food crops grown in the world start from their respective seeds. The general way of growing these plants is by bulk seeding trays of soil and then later transplanting the seedlings from the high-density trays to low-density trays for further growth while avoiding weeds. This process involves a vast amount of labor and the guarantee of quality seedlings is subjective to the labor working. With this project we aim to provide an image classification model that would detect the presence of seedling. The idea behind the image classification model is that it would detect the presence of seedlings in the seed trays by contrasting the colors of the soil and the seedlings.

Figure 1:- A sample of images from the dataset used in the prediction of seedlings.

Left image shows Presence of a Seedling while Right image shows No Presence of Seedlings.

While the health of the crop is identified by the color or the condition of the leaves [1]. Farmers have long done the same while deciding upon the pesticides or insecticides to use by examining the state of the crop leaves. Again, a tedious process and in a larger area of land there could be multiple patches of various diseases. The same image classification model can be used to identify healthy and unhealthy crop at various levels such as at seedling stage and pre harvest stage.

Figure 2:- A sample of images from the dataset to be used in the identification of healthy and unhealthy crops.

Left image shows a Healthy Leaf and Right image shows an Unhealthy Leaf.

The model can be implemented on any suitable datasets with similar accuracy metrics. Such a model with a good amount of accuracy could be implemented in automated agriculture systems for better handling and growth of the crops.

II. LITERATURE REVIEW

Studies predict that the world population could surpass ten billion people by 2050. There would also be a resultant increase in the requirements of global food, tripling compared to the present [2]. China is known as the largest food producer and the consumer country in the world. The Chinese agriculture sector is facing substantial environmental challenges than ever, the country is now gearing towards more sustainable agriculture production systems. In fact, any improvements in such a significant food producing and consuming country could result in benefits in the global environment [3]. It is hence essential for countries such as China to incorporate modern, user friendly and automated technologies in its shift towards sustainable agricultural development.

Whether in gardens or farms, seedling transplantation is one of the fundamental aspects while growing plants. Many prefer growing plants from a seedling to growing them from a seed. But growing these seedlings must start from the seed itself, which is a tedious process. The seeds must be bulk sown in seed trays as not every seed guarantees a seedling. The then grown seedlings must be transferred into a lower density growing trays or pots for further growth while also discarding inferior seedlings. While bulk sowing is must, there is a high chance of weeds also growing together with the intended crop seedling. The whole transplantation process from high density trays to low density tray is labor extensive.

So, in order to remove dependency on labor, there is a strong need of automation. It would help the agricultural industry boom by helping the farmers reduce costs and achieve maximum accuracy. Therefore, one such technology which is popular in this domain is Machine Vision Technology with Convolutional Neural Networks. MVT will capture the images using video cameras and send them as inputs to our CNN model which would look for patterns in the input sent and try to predict several things which would help farmers in increasing accuracy and thus help them achieve optimum efficiency.

Many automatic seedling transplanters use machine vision technology to automate this process [3]. These machines greatly reduce the amount of labor required and efficiently do it on a large-scale transplantation. Most of the machine vision technology involves three processes to do any vision-based analysis: image acquisition, processing, and recognition.

Zhang Xiao et al, used an industrial camera of 5 million pixels as an image acquisition device to collect 3400 pepper tray grids to build a recognition method to identify seedling grids without any seedlings based on deep convolutional neural network [4]. The processing was done by selecting a batch size of 48 images to further segment them by the grid and normalize to suitable pixel size. Sometimes researchers use data augmentation to handle any insufficiencies in the dataset that might cause overfitting in the model. Catherine R. Alimboyong et al used a variety of image transformations such as flip, rotate, scale, flip scale, and histogram equalization to ultimately obtain accuracies of 99.74% and 99.69% in their classification model to classify plant seedlings [5]. The recognition accuracy generally depends upon the data quality and the features used to recognize the images.

Health of the crop is to be monitored frequently otherwise there is a risk of losing most of the harvest or might result in a low yield. Again, machine vision and image processing help in this case to detect changes of color or shape in the leaves. J. W. Lee determined leaf color and the health state of lettuce using machine vision [1]. The evaluation of health was done by the color distribution of lettuce images. The study found that the images chosen to analyze must be consistently illuminated. It also stated the need of calibration procedures to maintain the consistency.

III. METHODOLOGY

A. Datasets

The seedling dataset consists of various species of crop such as Chickweed, Cleavers and Mayweed as this was the only available public datasets of seedlings. The leaf dataset consists of tomato crop with different kinds of diseases such as Bacterial Spot, Target Spot, Septoria Leaf Spot and Blight.

B. Image Preprocessing

In order to process images, we have used Binary Image Classification. We have developed a Convolutional Neural Network using a user-friendly library called Keras. A CNN is one of the types of neural network which is used in Image Classification and Natural Language Processing. A CNN is made up of various layers:

- Convolution Layer
- Max Pooling Layer
- Activation Function Layer
- Fully Connected Layer (Dense)

Convolution layer is the most important layer as it convolves the image by using filters. Our model uses 3 convolution layers of 32,32 and 64 nodes respectively and a filter of (2,2) matrix in each layer. Convolution is necessary as images are complex and we need to simplify it for faster processing. So, a convolution layer works by placing filters over an array of image pixels and finally creates a convolved feature map, which is of reduced dimension as compared to its original image. It's a bit like looking an image through a window which allows you to see specific features of image which otherwise you would not have able to see. Next, we have **MaxPooling Layer** which downsizes the sample so that further non-important features get eliminated and so our model can focus only on important features. As a result, we get a Pooled feature map. In our case, we are using 3 MaxPooling Layers with filters of (2,2) dimension. There are 2 ways to get a pooled feature map:

- **Max Pooling**: it takes maximum input of particular convolved feature map.
- **Average Pooling:** it takes avg. of all inputs of particular convolved feature map.

Now, these steps involve in feature extraction & network builds up a picture of image data according to its mathematical rules. Those mathematical rules are nothing but the activation functions along with the loss functions that are used to optimize the model. So, we have used a combination of ReLU and Sigmoid functions. We have used **binary_crossentropy** as our loss function and **Adam** as our optimizer. In order to perform classification, we need to make use of fully connected layers. So before using it we need to flatten the data because a neural network which has a complex set of connections can only process linear data.

Binary Cross Entropy is considered to be the first choice as a loss function for any binary classification problems. It is so because the target set has values either 0 or 1. Cross-entropy will calculate a score that summarizes the average difference between the actual and predicted probability distributions for predicting class 1 [6]. **Adam** uses individual separate learning rates for its parameters. Learning rate keeps on changing with the training steps, but every learning rate must vary between 0 and λ (lambda).

C. Data Augmentation

It is a strategy by which Data Scientists try to increase data samples in a dataset so that our model can train on more data and become better and better with time. So, this can be done using

ImageDataGenerator class [7]. We have used **rotation_range**, **width_shift_range, height_shift_range, shear_range, zoom_range, horizontal_flip** as the arguments.

Figure 3:- Augmented Tomato Leaf Images.

Figure 4:- Augmented Seedling Images

D. Training and Classification

The model was trained on the two datasets Seedlings and Health with slight changes made in batch sizes. The Health dataset had a train set of 2263 samples and a test set of 750 samples. While the Seedlings had a train set of 1810 samples and a test set of 850 samples. The model was trained on Google Colab, a cloud service equipped with RAM of 12GB and accelerated image processing with NVIDEA TESLA V100 of 16GB. In order to improve the generalization ability of the model we applied zooming, shearing, horizontal flipping, rotation width shift and height shift transformations of ranges 0.2, 0.2, True, 0.2, 0.2 respectively on the train samples of both the datasets. The model was trained on batch-sizes equal to 16,50 and 65 respectively with epochs equal to 10 in each case.

IV. PERFORMANCE

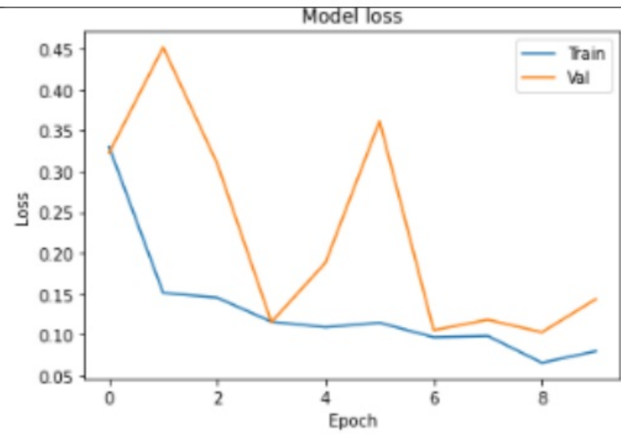
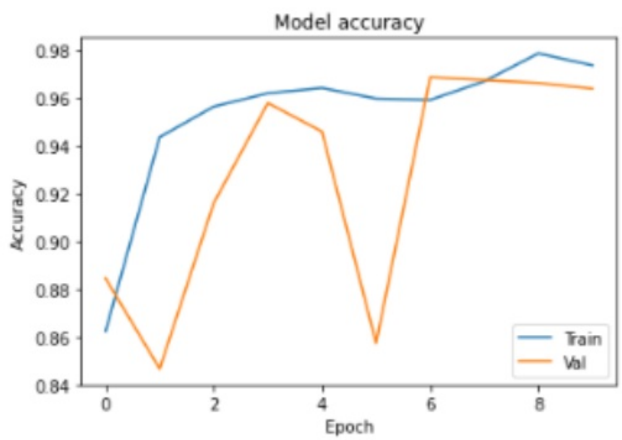
At the start while we were training the model, we observed that there is a lot of **Overfitting**. **Overfitting** is the case when model performs well in case of train data but gives a degraded performance with test data. It was observed on both the datasets.

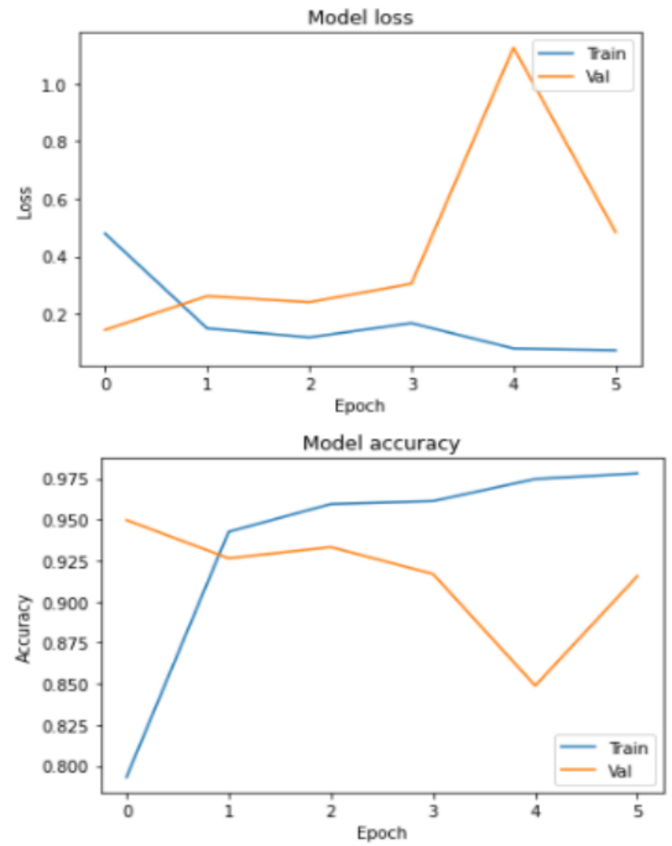
Table 1:- Epoch Training Result

Dataset	Performance
Seedlings	Found 1810 images belonging to 2 classes. Found 850 images belonging to 2 classes. Epoch 1/10 113/113 [==============================] - 157s 1s/step - loss: 0.3305 - acc: 0.8634 - val_loss: 0.3225 - val_acc: 0.8844 Epoch 2/10 113/113 [==============================] - 158s 1s/step - loss: 0.1502 - acc: 0.9441 - val_loss: 0.4517 - val_acc: 0.8465 Epoch 3/10 113/113 [==============================] - 157s 1s/step - loss: 0.1443 - acc: 0.9569 - val_loss: 0.3092 - val_acc: 0.9161 Epoch 4/10 113/113 [==============================] - 158s 1s/step - loss: 0.1151 - acc: 0.9624 - val_loss: 0.1155 - val_acc: 0.9580 Epoch 5/10 113/113 [==============================] - 160s 1s/step - loss: 0.1084 - acc: 0.9646 - val_loss: 0.1881 - val_acc: 0.9460 Epoch 6/10 113/113 [==============================] - 160s 1s/step - loss: 0.1139 - acc: 0.9602 - val_loss: 0.3613 - val_acc: 0.8573 Epoch 7/10 113/113 [==============================] - 158s 1s/step - loss: 0.0962 - acc: 0.9596 - val_loss: 0.1055 - val_acc: 0.9688 Epoch 8/10 113/113 [==============================] - 152s 1s/step - loss: 0.0978 - acc: 0.9674 - val_loss: 0.1184 - val_acc: 0.9676 Epoch 9/10 113/113 [==============================] - 150s 1s/step - loss: 0.0760 - acc: 0.9751 - val_loss: 0.1030 - val_acc: 0.9664 Epoch 10/10 113/113 [==============================] - 150s 1s/step - loss: 0.0795 - acc: 0.9740 - val_loss: 0.1437 - val_acc: 0.9640

Health	Found 2263 images belonging to 2 classes. Found 750 images belonging to 2 classes. Epoch 1/10 141/141 [==============================] - 458s 3s/step - loss: 0.4782 - acc: 0.7939 - val_loss: 0.1447 - val_acc: 0.9497 Epoch 2/10 141/141 [==============================] - 152s 1s/step - loss: 0.1519 - acc: 0.9417 - val_loss: 0.2624 - val_acc: 0.9264 Epoch 3/10 141/141 [==============================] - 152s 1s/step - loss: 0.1186 - acc: 0.9597 - val_loss: 0.2410 - val_acc: 0.9332 Epoch 4/10 141/141 [==============================] - 153s 1s/step - loss: 0.1680 - acc: 0.9614 - val_loss: 0.3052 - val_acc: 0.9169 Epoch 5/10 141/141 [==============================] - 152s 1s/step - loss: 0.0803 - acc: 0.9747 - val_loss: 1.1255 - val_acc: 0.8488 Epoch 6/10 141/141 [==============================] - 153s 1s/step - loss: 0.0750 - acc: 0.9777 - val_loss: 0.4845 - val_acc: 0.9155

Table 2:- Visualization of the Model Loss and Accuracy

Dataset	Model Loss and Accuracy
Seedling	 

Health	

So, here you can see that model's loss on train data is decreasing with each epoch and its accuracy is increasing with each epoch. While model's behavior with the test data is just opposite which is not a good behavior. This shows that our Neural Network is biased towards the train data and so it's not able to perform well on the test data. Therefore, it is a pure case of **Overfitting**.

i. Handling Overfitting

Later we removed overfitting using the following things:

- **L2 Regularization**: We have got the following loss function:

$$\boldsymbol{Loss\ Function(J) = -1/N(\textstyle\sum_{i=1}^{N} yi\,.log\big(p(yi)\big) + 1 - yi.log(1 - p(yi))}$$

So, in order to reduce the loss, we add regularizer parameter $\boldsymbol{\lambda}$ which try to reduce the loss by penalizing the larger weights and effect of regularization is less over smaller weights. In the addition of the Frobenius norm, which the subscript *F* denotes. It is in fact equal to the squared norm of a matrix [8].

$$J\big(w^{[1]}, b^{[1]}, \dots, w^{[L]}, b^{[L]}\big) = \frac{1}{m}\sum_{i=1}^{m} L\big(\hat{y}^{(i)}, y^{(i)}\big) + \frac{\lambda}{2\mathrm{m}}\sum_{l=1}^{L}\left\|w^{[L]}\right\|_F^2$$

By adding the squared norm of the weight matrix and multiplying it by the regularization parameters, large weights will be driven down in order to minimize the cost function.

Therefore, we used $\boldsymbol{kernel\_regularizer = regularizers.l2(0.7)}$ where 'λ' is 0.7.

- **Dropout**: We used **dropout(0.3)** and **dropout(0.5)** in some cases. So, this means that neurons of the previous layer have the probability of 30% and 50% to be dropped out during training.

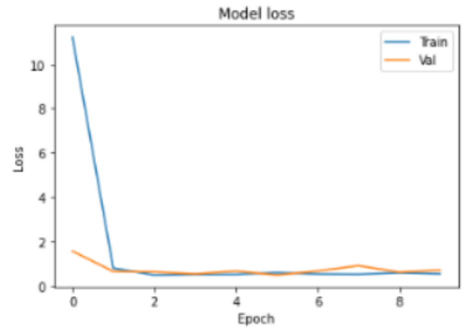
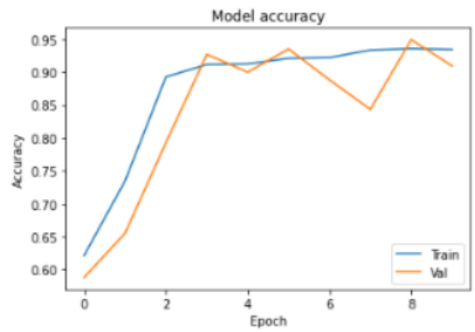
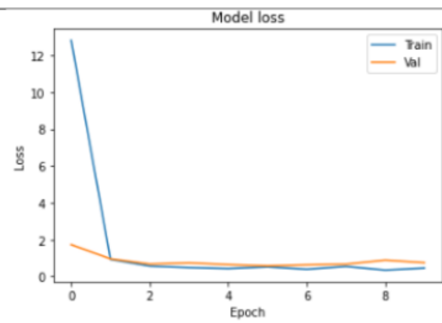
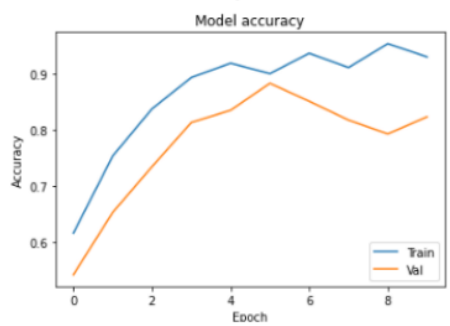
- **Early stopping Monitor**: This stops the model from further training as soon as model stops improving after '**n**' number of epochs.

These parameters helped the model in significantly reduce the overfitting.

Table 3:- Improved Epoch Training Result

Dataset	Performance
Seedlings	Found 1810 images belonging to 2 classes. Found 850 images belonging to 2 classes. Epoch 1/10 36/36 [==============================] - 148s 4s/step - loss: 11.1277 - acc: 0.6211 - val_loss: 1.5554 - val_acc: 0.588 Epoch 2/10 36/36 [==============================] - 145s 4s/step - loss: 0.7977 - acc: 0.7317 - val_loss: 0.6368 - val_acc: 0.6553 Epoch 3/10 36/36 [==============================] - 147s 4s/step - loss: 0.4767 - acc: 0.8928 - val_loss: 0.6163 - val_acc: 0.7929 Epoch 4/10 36/36 [==============================] - 142s 4s/step - loss: 0.4975 - acc: 0.9133 - val_loss: 0.5260 - val_acc: 0.9271 Epoch 5/10 36/36 [==============================] - 144s 4s/step - loss: 0.5028 - acc: 0.9144 - val_loss: 0.6538 - val_acc: 0.9000 Epoch 6/10 36/36 [==============================] - 143s 4s/step - loss: 0.5848 - acc: 0.9184 - val_loss: 0.4764 - val_acc: 0.9353 Epoch 7/10 36/36 [==============================] - 146s 4s/step - loss: 0.5226 - acc: 0.9217 - val_loss: 0.6531 - val_acc: 0.8882 Epoch 8/10 36/36 [==============================] - 146s 4s/step - loss: 0.5146 - acc: 0.9285 - val_loss: 0.8965 - val_acc: 0.8435 Epoch 9/10 36/36 [==============================] - 145s 4s/step - loss: 0.5766 - acc: 0.9372 - val_loss: 0.6030 - val_acc: 0.9494 Epoch 10/10 36/36 [==============================] - 147s 4s/step - loss: 0.5322 - acc: 0.9344 - val_loss: 0.6953 - val_acc: 0.9094
Health	Found 2263 images belonging to 2 classes. Found 750 images belonging to 2 classes. Epoch 1/10 34/34 [==============================] - 128s 4s/step - loss: 12.7834 - acc: 0.6148 - val_loss: 1.7240 - val_acc: 0.5413 Epoch 2/10 34/34 [==============================] - 126s 4s/step - loss: 0.9295 - acc: 0.7525 - val_loss: 0.9559 - val_acc: 0.6526 Epoch 3/10 34/34 [==============================] - 128s 4s/step - loss: 0.5603 - acc: 0.8377 - val_loss: 0.6734 - val_acc: 0.7343 Epoch 4/10 34/34 [==============================] - 127s 4s/step - loss: 0.4796 - acc: 0.8934 - val_loss: 0.7369 - val_acc: 0.8131 Epoch 5/10 34/34 [==============================] - 127s 4s/step - loss: 0.4278 - acc: 0.9189 - val_loss: 0.6400 - val_acc: 0.8350 Epoch 6/10 34/34 [==============================] - 127s 4s/step - loss: 0.5171 - acc: 0.9005 - val_loss: 0.5740 - val_acc: 0.8832 Epoch 7/10 34/34 [==============================] - 126s 4s/step - loss: 0.3905 - acc: 0.9366 - val_loss: 0.6287 - val_acc: 0.8511 Epoch 8/10 34/34 [==============================] - 128s 4s/step - loss: 0.5414 - acc: 0.9112 - val_loss: 0.6660 - val_acc: 0.8175 Epoch 9/10 34/34 [==============================] - 127s 4s/step - loss: 0.3471 - acc: 0.9540 - val_loss: 0.8854 - val_acc: 0.7927 Epoch 10/10 34/34 [==============================] - 128s 4s/step - loss: 0.4529 - acc: 0.9306 - val_loss: 0.7471 - val_acc: 0.8234

Table 4:- Improvised Model's Loss and Accuracy

Dataset	**Model Loss and Accuracy**
Seedling	
Health	

So, here the patterns observed in case of model loss and model accuracy on both the training and testing dataset is very similar. Thus, we have successfully reduced the biasness of our neural network on the train dataset and so the **overfitting** has reduced significantly.

V. RESULT

13 One observation with the Adam optimizer was that, with the increase in the learning rate, the performance of the Convolutional Neural Network on both the datasets (Seedlings and Health) was decreasing and vice-versa.

Table 5:- Test Result of Deep Convolutional Neural Network on the Seedlings Dataset

Network	Optimizer	Learning Rate	Max. Test Accuracy Achieved (%)
Deep CNN Using Keras	Adam	0.1	58.82
Deep CNN Using Keras	Adam	0.05	94.00
Deep CNN Using Keras	Adam	0.01	58.82
Deep CNN Using Keras	Adam	0.001	94.94

Table 6:- Test Result of Deep Convolutional Neural Network on the Health Dataset

Network	Optimizer	Learning Rate	Max. Test Accuracy Achieved (%)
Deep CNN Using Keras	Adam	0.1	53.3
Deep CNN Using Keras	Adam	0.05	54.60
Deep CNN Using Keras	Adam	0.01	54.89
Deep CNN Using Keras	Adam	0.001	88.32
Deep CNN Using Keras	SGD	0.01	83.36

VI. CONCLUSION AND FUTURE SCOPE

We tried to improve the seedling transplantation method by building a Deep Convolutional Neural Network. It took the images of seedlings directly as inputs and tried to recognize certain patterns when it is trained on train set and it tried to predict the images. The model was trained on 2263 samples of Health dataset and 1810 samples of Seedling dataset. And, it was evaluated on 750 samples of the Health dataset and 850 samples of the Seedlings dataset. We observed that when the learning rate was 0.1, model's accuracy was 53.3% and when learning rate was 0.001, the validation accuracy got increased to 88.32%. In the beginning we observed overfitting and later this overfitting was reduced significantly by applying regularization, dropouts and early stopping monitor**.** The model improved when fine tunings were done with batch sizes and data augmentation techniques were used**.**

In the future, this model could be generalized for all kinds of seedlings with proper lighted images and for all kinds of distinguishable leaf diseases in crops. The model with object detection functionality could be used with a pick and place robot to handle the seedling transplantation with greater accuracy while creating less strain on the seedling. The next objective is to work on the object detection and implementing the model through a pick and place robot or in a simulation.

VII. References

[1] J. Lee, "Determination of Leaf Color and Health State of Lettuce using Machine Vision," *Journal of Biosystems Engineering, Korean Society for Agricultural Machinery,* vol. 32, no. 4, pp. 256-262, 2007.

[2] Food and Agriculture Organization of the United Nations, "FAO Expert Papers," [Online]. Available: http://www.fao.org/fileadmin/templates/wsfs/docs/expert_paper/How_to_Feed_the_World_in_2050.pdf.

[3] T. N. Syed, I. A. Lakhiar and F. A. Chandio, "Machine vision technology in agriculture: A review on the automatic seedling transplanters," *International Journal of Multidisciplinary Research and Development,* vol. 6, no. 12, pp. 79-88, 2019.

[4] Z. Xiao, X. Liu, Y. Tan, F. Tian, S. Yang and B. Li, "Recognition Method of No-seedling Grids of Trays based on Deep," in *Proceedings of the 38th Chinese Control Conference*, Guangzhou, China, 2019.

[5] C. R. Alimboyong, A. A. Hernandez and R. P. Medina, "Classification of Plant Seedling Images Using Deep Learning," in *TENCON 2018 IEEE Region 10 Conference*, Jeju, Korea, 2018.

[6] J. Brownlee, "How to Choose Loss Functions When Training Deep Learning Neural Networks," 30 January 2019. [Online]. Available: https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/. [Accessed 11 April 2020].

[7] Keras, "Image Preprocessing," [Online]. Available: https://keras.io/preprocessing/image/. [Accessed 12 April 2020].

[8] M. Peixeiro, "How to Improve a Neural Network With Regularization," 12 March 2019. [Online]. Available: https://towardsdatascience.com/how-to-improve-a-neural-network-with-regularization-8a18ecda9fe3. [Accessed 12 April 2020].

[9] C. R. Alimboyong and A. A. Hernandez, "An Improved Deep Neural Network for," in *IEEE 15th International Colloquium on Signal Processing & its Applications (CSPA 2019)*, Penang, Malaysia, 2019.

[10] Z. Xiao, Y. Tan, X. Liu and S. Yang, "Classification Method of Plug Seedlings Based on Transfer Learning," *Applied Sciences,* 2019.

Predicting the Presence of Seedlings and the Health of Crops using Image Classification

ORIGINALITY REPORT

10%	7%	3%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	hub.packtpub.com Internet Source	2%
2	towardsdatascience.com Internet Source	2%
3	www.mdpi.com Internet Source	1%
4	news.datascience.org.ua Internet Source	1%
5	Submitted to Loughborough University Student Paper	1%
6	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	1%
7	Submitted to Universidade do Porto Student Paper	1%
8	www.pp1pp.com Internet Source	<1%

9	Submitted to National University Philippines Student Paper	<1%
10	onlinelibrary.wiley.com Internet Source	<1%
11	Submitted to The Scientific & Technological Research Council of Turkey (TUBITAK) Student Paper	<1%
12	JONG WHAN LEE. "MACHINE VISION MONITORING SYSTEM OF LETTUCE GROWTH IN A STATE-OF-THE-ART GREENHOUSE", Modern Physics Letters B, 2011 Publication	<1%
13	Submitted to University of New South Wales Student Paper	<1%
14	Junhua Tong, Chuanyu Wu, Huanyu Jiang, Yaxin Yu, Xiuqin Rao. "Optimizing the path of seedling low-density transplanting by using greedy genetic algorithm", Computers and Electronics in Agriculture, 2017 Publication	<1%

Exclude quotes	On	Exclude matches	< 10 words
Exclude bibliography	On		