Competition Project: Phase 1

Machine learning approaches towards classifying visual water-stress found in soybean images

Abhishek Ranjan Singh
Electrical and Computer Engineering
North Carolina State University
Raleigh, NC, USA
arsingh3@ncsu.edu

Adam Christopher Watts
Wilson College of Textiles
North Carolina State University
Raleigh, NC, USA
acwatts4@ncsu.esu

Sankalp Singh Gaharwar
Computer Science
North Carolina State University
Raleigh, NC, USA
ssgaharw@ncsu.edu

I. Data

A. Data

A training set consisting of 1025 agricultural soybean photos each with a resolution of 640 X 480 pixels was provided. The training set consisting of 5 classes is listed in Table 1. Visual inspection of the photos revealed a wide range of exposure levels with some photos being underexposed and rather dark while others are overexposed and rather bright.

TABLE 1. SOYBEAN PLANT CLASSIFICATIONS AND DESCRIPTIONS.

Categorical Number	Classification Description
0	No Wilting
1	Leaflets folding inward at secondary pulvinus, no turgor loss in leaflets or petioles
2	Slight leaflet or petiole turgor loss in the upper canopy
3	Moderate turgor loss in the upper canopy
4	Severe turgor loss throughout the canopy

B. Augmentation

Image augmentation was performed by using Keras ImageDataGenerator class to synthetically extend the training set to 3075. The first 1025 images were the original training set while the remaining 2050 consisted of 2 complete rounds of augmentation using the parameters found in Table 2. The parameters and values were chosen to emulate the variation found in the provided data set.

TABLE 2. KERAS IMAGEDATAGENERATOR PARAMETERS

Parameter	Value / Boolean
Rescale	1.0/255
Shear Range	0.1
Zoom Range	0.25
Horizontal Flip	True
Rotation Range	25
Fill Mode	‘Wrap’

C. Image Pre-Processing

To identify an object in a particular category all one needs to know is the shape of the object, often the information containing the color is redundant. In these cases, one can drop the color channel information and still obtain fairly good classification results. Doing so reduces both the time and space complexity of any given model. Having a classification task that is independent of color would have been ideal, but the visual inspection of the data concurs that one does indeed need the information containing the color of images. Training of the network on the grayscale input brought a stable accuracy of 78%, less than with color.

Visual inspection of the photos revealed a wide range of exposure levels with some photos being underexposed and rather dark while others were overexposed and rather bright. This issue can cause the network to learn some undesirable features. To address this problem, a histogram equalization on the input image space was performed. Histogram equalization is a type of transformation that preserves the correlation of pixels in an image while resolving the overexposure and under-exposure.

II. Machine Learning Models

A. Mixture of Gaussian (MOG)

One of the first approaches to the problem was using a Gaussian Mixture Model (GMM). The GMM gave an accuracy of 65% on the validation set. The idea behind using GMM was that it could learn the likelihood of the model and then utilize the prior which might help the model do better on the test set which can have a different distribution from the training set. But since the GMM could not reach anything above 65%, it was decided to try an alternative approach.

B. Convolution Neural Networks (CNN)

The first CNN was constructed based on the LeNet-5 architecture using Tensorflow with the Keras API [1], [2]. The input dimensions in the original LeNet-5 were only 32 X 32 greyscale images compared to the 640 X 480 RGB images used here. Activation functions for this project were modified to Relu instead of the original Tanh. The input layer was connected to three, 2-Dimensional convolution layers. The layer was then flattened down to two fully dense multilayer perception (MLP) of 16 and 8 neurons respectively for a total of 5 layers. The stride size for all the convolution layers was set to 1 with the Kernels decreasing from 11, 5, and finally 3 for the 3 convolution layers. The number of filters increased for each subsequently deeper convolutional layer with 64, 128, and 256 filters respectively. A maximum pooling layer with a pool size of 2 was placed in between each convolutional layer. A 50% dropout was performed between each dense MLP.

Stochastic gradient descent (SGD) with used as the optimization algorithm with a batch size of 16. The training

was performed on the training images without any pre-processing or augmentation. A validation split of 15% was utilized with a batch size of 16 without shuffling. The total number of trainable parameters was 1.7 million.

Poor stability was noticed with the training leading to exploding gradients. The optimization algorithm was altered from SGD to ADAM which leads to stable learning [3]. Although stability was achieved, the training and validation accuracy failed to improve with each epoch. Training accuracy and validation accuracy were stagnant at 46.84% and 51.95% respectively. The CNN architecture was modified to as shown in Table 3 to improve accuracy.

Accuracy didn't improve until dropout was completely removed leading to overfitting. Although removing dropout led to overfitting of the training data, it demonstrated that high dropout can severely impede model learning even when the architecture was drastically changed to almost 80 million parameters. Dropout was restored prior to the Softmax output layer with a 0.20 value, resulting in a training and validation accuracy of 96.10% and 73.38% respectively at 15 epochs.

TABLE 3. CNN ARCHITECTURE FOR THE SECOND ATTEMPT. C, S, F, D, O STAND FOR CONVOLUTION, SUBSAMPLING, FLATTEN, DENSE MLP, AND DROPOUT LAYER RESPECTIVELY.

Layer	**Output Size**	**Filters**	**Kernel / Pool**	**Padding**
1	640 X 480	-	-	-
C1	640 X 480	64	11 X 11	SAME
S2	320 X 240	-	2 X 2	-
C3	320 X 240	128	3 X 3	SAME
C4	320 X 240	128	3 X 3	SAME
S5	160 X 120	-	2 X 2	-
C6	160 X 120	256	3 X 3	SAME
Layer	**Output Size**	**Neurons**	**Dropout**	**Activation**
F7	1,228,800	-	-	-
D8	64	64	-	Relu
O9	64	-	0.50	-
D10	32	32	-	Relu
O11	32	-	0.50	-
D12	5	5	-	Softmax

Augmented images with their histograms equalized were then added to the training and data. It was hypothesized that the larger data set would improve accuracy as the CNN would have a larger data set to learn from over the same number of epochs. However, both training and validation accuracy deteriorated and could not improve above 50-60%. Batch normalization was added between each dense layer which sped up convergence but didn't improve accuracy [4]. Furthermore, the shuffling of the dataset was also performed to minimize the chances that the CNN would "memorize" the data. The augmented images were then removed from the training set to further diagnose the poor accuracy results.

To further optimize the validation accuracy of the CNN, a max-pooling of 4 was added to the architecture prior to reaching the dense MLP layers for the CNN. The dense layers themselves were fortified by incorporating 4000 units within the first dense MLP layer and 2000 units within the second dense MLP layer. Additionally, the training of the CNN with more computational power and greater GPU memory allowed the batch size to increase during training to 21 when prior it was only 16. Other aspects of the CNN such as batch normalization between dense layers and the dropout were retained as well. By running this modified CNN architecture over 21 epochs, promising results were discovered as the validation accuracy for the network shot up and started approaching the 80% mark while still ensuring that the model was not overfitting on training data.

The final task was to optimize the CNN parameters to enhance the accuracy of our model to the maximum extent. As a part of optimizing the network, iteratively altering the dropout rate between the dense layers was used to gauge the optimal value. Starting from a dropout value of around 25%, it was scaled it up in a stepwise manner and progressively saw the accuracy of the model improve. A 50% dropout rate between each dense layer was chosen as the final dropout value.

Additionally, it was noticed that the CNN architecture generally trended towards overfitting after the 18th epoch and hence the number of epochs was limited to 18. Therefore, the final CNN model with histogram equalization on the input images gave performance metrics of 98% for training accuracy and 85.7% for validation accuracy.

III. RESULTS

The CNN model tends to be stable as it improves its performance with each progressive epoch. We notice that the validation accuracy tends to improve along with the training accuracy, as the model learns the data set better with each passing epoch. Apart from some abrupt drops entailed by random sampling, the CNN model never drops below 50% validation accuracy and tends to achieve a respectable validation accuracy in the range of 80%-85 % as the training accuracy approaches 90%. An important trend to note is that the model can sustain this validation accuracy over 8-9 epochs which validates the good performance.

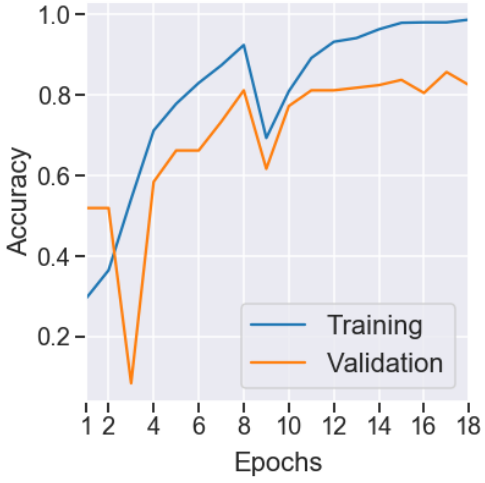

Fig.1. Training and Validation accuracy vs Epochs

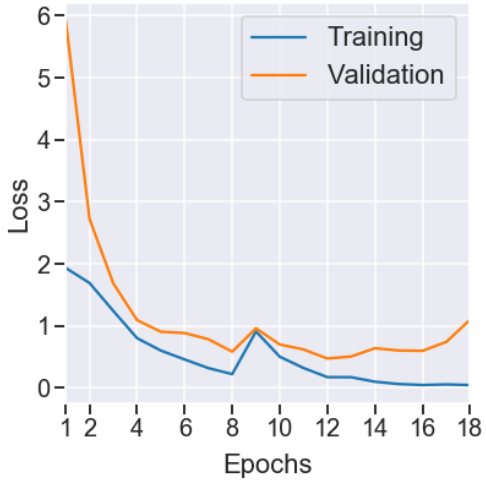

Fig.2. Training and Validation loss vs Epochs

Similar trends are seen for the validation loss and training loss metrics which tend to decrease with each progressive epoch. One is able to ensure that both the loss metrics are maintained at values less than 1 over the later epochs of the CNN model.

TABLE 4. FINAL CNN ARCHITECTURE.

Layer	Output Size	Filters	Kernel / Pool	Padding
1	640 X 480	-	-	-
C1	640 X 480	64	11 X 11	SAME
S2	320 X 240	-	2 X 2	-
C3	320 X 240	128	3 X 3	SAME
C4	320 X 240	128	3 X 3	SAME
S5	160 X 120	-	2 X 2	-
C6	160 X 120	256	3 X 3	SAME
S7	80 X 60	-	2 X 2	-
C8	80 X 60	256	2 X 2	SAME
S9	20 X 15	-	4 X 4	=
Layer	**Output Size**	**Neurons**	**Dropout**	**Activation**
F10	76,8000	76,8000	-	-
D11	4,000	4,000	-	Relu
O9	4,000	4,000	0.50	-
D10	2,000	2,000	-	Relu
O11	2,000	2,000	0.50	-
D12	5	5	-	Softmax

The initial training data set was heavily skewed towards the class-0 label from the data set with very minimal representation for classes 1,2,3 and 4. This trend continues in the predicted labels assigned to the testing data where one sees the maximum amount of images being labeled as class-0. An interesting trend is that a small ratio of testing data images is still labeled as classes 1,2 or 3, no test data image is assigned the label for class 4. The prognosis for this trend is that it may be due to an inherent inequality in the distribution for test data images that mirrors the training data set. Another possibility is that the model exhibits selection bias creeping in from the training data set which is evidently highly skewed towards the class-0 and gives minimum representation to class-4.

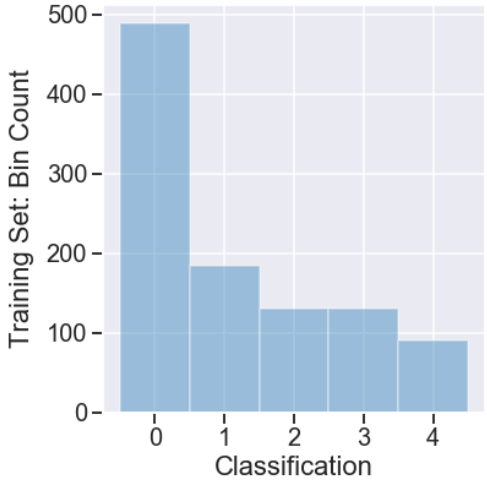

Fig.3. Distribution of training data set vs Class labels

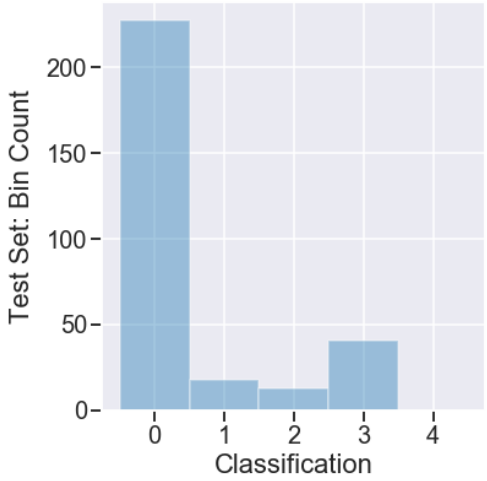

Fig.4. Distribution of test data set vs Predicted class labels

IV. CONCLUSION

Training a CNN network from scratch requires large computational resources especially when the input images are fairly large. Transfer learning is another approach that may be implemented in the future. The testing results also suggest the importance of having a balanced training data set in regard to the classification distribution. Highly skewed training sets appear to lead to skewed test results.

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Ha, "Gradient-Based Learning Applied to Document Recognition," p. 46, 1998.

[2] F. Chollet, *Keras*. 2015.

[3] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *ArXiv14126980 Cs*, Jan. 2017.

[4] A. Géron, *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Incorporated, 2019.