Leaf Wilting Detection in Soybean

Michael Basch
mbasch@ncsu.edu

Bhavana Vijay Lalwani
blalwan@ncsu.edu

Mallika Sinha
msinha2@ncsu.edu

I. Methodology

The training images and labels were loaded in python and size of the images was checked. All the images had the same size (640x480x3) and were used in analysis with the same dimensions.

The images were first converted to grayscale to simplify the model. Further the images were converted to numpy [1] arrays and normalized to reduce distortions. The model used for classification is a simple sequential neural network with 2 hidden layers:

- Input Layer - 307200 nodes
- Hidden Layer 1 - 562 nodes (relu activation)
- Hidden Layer 2 - 256 nodes (relu activation)
- Output Layer - 5 nodes (softmax activation)

Listed in Table I below is a summary of the toolboxes and the major classes and functions utilized. Pandas [2] was used to open and parse the provided training data file "TrainAnnotations.csv" and for preparation and storage of the predictions of the results of the test data set. Pillow [3] and Numpy were used for image processing, filtering and conversion of images in preparation for interfacing with the model. It was also used for conversion of results to "one-hot" encoded outputs. The tensorflow toolbox, specifically Keras [4] was used for the core of the model. This included functions for describing, compiling, training and evaluating the model. Finally, pyplot [5] was used to display sample images and plot the summary of model training.

TABLE I
Toolboxes & Functions

Toolbox	*Classes, Functions*	*Application*
Pandas	read_csv, DataFrame, Series	CSV data file parsing, preparation for storage, display of distribution of test-set predictions
Numpy	array, argmax	image data, model label categorization preparation
PIL (Pillow)	Image	Image processing, filtering
matplotlib	pyplot	plot results
tensorflow	Keras: {Model, Sequential,Activation, Dense, SGD}	Build sequential model, activation functions, optimization

II. Model Training

There was a total 1025 images in the training set. These were split into 90 % for training and 10 % for the validation set. Mini-batch gradient descent was used here to improve the accuracy. Specifically, for model compilation the keras stochastic gradient descent optimizer was used with a learning rate of 0.001 and a loss function of categorical crossentropy. This loss function was used due to the fact that it is applicable for the type of target classification desired for the specific leaf wilting status as annotated from 0 to 4 at an interval of 1. Hyper parameters were set to following values:

- Batch Size = 25
- Number of Epochs = 100

Below, Fig. 1 is a diagram describing the layers of the model used along with the parameters at the inputs and outputs of each layer.

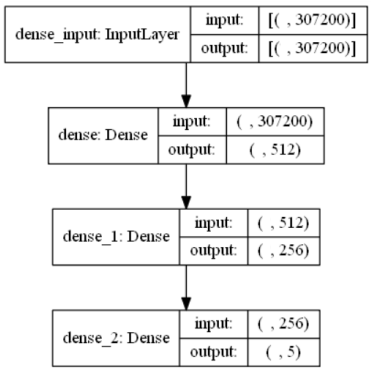

Fig. 1. Model Diagram

Fig. 2, below is a summary which includes details on the network layers, neurons and associated trainable parameters.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_3 (Dense)              (None, 512)               157286912
_________________________________________________________________
dense_4 (Dense)              (None, 256)               131328
_________________________________________________________________
dense_5 (Dense)              (None, 5)                 1285
=================================================================
Total params: 157,419,525
Trainable params: 157,419,525
Non-trainable params: 0
```

Fig. 2. Model Parameter Details

III. Evaluation

The metric used for evaluating the model was accuracy, which is defined as follows-

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \quad (1)$$

where,
TP = True Positives, TN = True Negatives, FP = False Positives, FN = False Negatives [6]

Cross-Entropy loss function for multi class was used to evaluate training and validation. The formula for the loss function is as follows-

$$-\sum_{c=1}^{M} y_{o,c} log(p_{o,c}) \quad (2)$$

where,
M=Number of classes
log=the natural log
y=binary indicator(0 or 1) if the class label is correct classification for observation o
p=predicted probability observation o is of class c [7]

The approached methodology produced a training accuracy of 98.24% and training loss of 0.0714. The validation accuracy and validation loss were 82.52% 0.55, respectively.

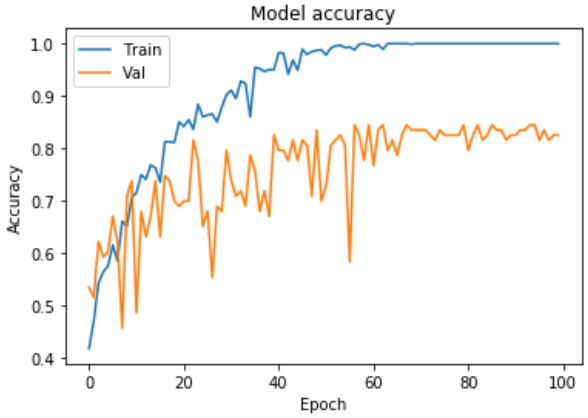

Fig. 3. Model Accuracy

In preparation for generating test data predictions, we used the Keras utility function 'to_categorical' to format the output of the model in 'one-hot' encoding format. From there, Pandas 'DataFrame' module and '.to_csv' function was used to export and store the results in comma-separated-value format.

IV. Results

The model was further used to predict the labels of test images from 'Test Data' folder. There were 300 test images in the folder. The test predictions for these images are as below:

- Images with label 0: 276
- Images with label 1: 12
- Images with label 2: 7
- Images with label 3: 5

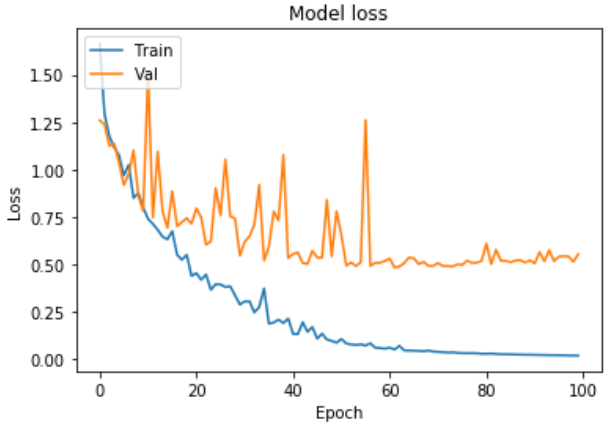

Fig. 4. Model Loss

From the results, we can infer that label 0 is highly cluttered while label 4 has not been recognised by the current model. The test predictions are highly unbalanced. This may be because of the imbalance in training data or poor model fit. Use of balanced training set or improvised deep learning models will help us achieve better classification results.

References

[1] T. Oliphant, "NumPy: A guide to NumPy," USA: Trelgol Publishing, 2006–. [Online]. Available: http://www.numpy.org/

[2] W. McKinney *et al.*, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445. Austin, TX, 2010, pp. 51–56.

[3] F. Lundh, "Python Imaging Library," Tech. Rep., 2009–. [Online]. Available: http://www.pythonware.com/products/pil/

[4] F. Chollet *et al.*, "Keras," Tech. Rep., 2015. [Online]. Available: https://keras.io

[5] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in science & engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[6] "Evaluation metrics for machine learning - accuracy, precision, recall, and f1 defined." [Online]. Available: https://pathmind.com/wiki/accuracy-precision-recall-f1

[7] "Loss functions." [Online]. Available: https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html