Leaf Wilting Detection in Soybean

Sushant Sadangi
Department of ECE
North Carolina State University
Raleigh, NC, USA
ssadang@ncsu.edu

Yash Thakkar
Department of CSC
North Carolina State University
Raleigh, NC, USA
yrthakka@ncsu.edu

Adarsh Puri
Department of ECE
North Carolina State University
Raleigh, NC, USA
apuri3@ncsu.edu

I. Methodology

Loss of humidity and disease mitigation are considered as two major cause of farming failures. However monitoring the leaves from an early stage itself helps prevent further crop failure. To address this aforementioned issue, we have followed the following basic steps:

- Image Acquisition
- Implementing Notebook using Google Colabs
- Build a Deep Convolutional Neural Network using FastAI and Resnet with 50 layers (Resnet-50)
- Change the hyper parameter for tuning.

A. Dataset

[1] For the training, testing and Validation of the results, we have used the NCSU Soybean Datasets that comprised of over 1000 images of Soybean leaves. The images were collected during different times of the day and with different wilting levels. Based on the level of wilting, the Soybeans are classified into five groups and are labeled accordingly.

The training data consists of 80% of the images and the rest 20% of the images are used for validation.

Instead of feeding the model with the same pictures every time, we do small random transformations using data augmentation techniques like vertical flip, lighting, zoom, wrap functionalities.

B. Google Colab

[2] Colaboratory or the Colab, allows easy access to the users for writing Python codes with zero configurations, free access to GPUs and easy sharing mechanisms hence removing the dependency of version control tools wile working in teams. Colabs run on Google servers, leveraging the power of GPUs and TPUs that allows the user to run the codes irrespective the computational power of personal machine.

For this project, we used google Colab with the following configurations:

- RAM of 12.72 GB
- Disk Size of 68.40 GB

C. ResNet-50

[3] ResNet or the Residual Network are a form of deep convolutional networks with the core idea of using shortcut connections to skip the intermediate convolutional layers. This skip connection add up to the output of the previous layers to the output of the stacked layers. The basic blocks (bottleneck) are designed using two design rules:

- for the same output feature map size, the layers have the same number of filters
- if the feature map size is halved, the number of filters is doubled

When the size of the input and the output layers are the same then the identity shortcut is used. When the dimensions increase, the projection shortcut is used to match the dimensions using convolution. A basic block diagram of Resnet architecture has been shown below:

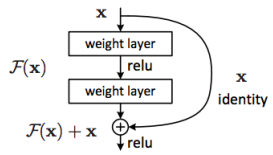

Fig. 1. Resnet-50

In this implementation, we use ResNet-50 as the base model, for object detection task on the given dataset. Resnet-50 has been trained with over a million images from the imagenet database.Hence this was one of the most important reasons for selecting this architecture over any other architecture. In this implementation we exploited the transferred learning mechanism to transfer the first 49 layers of ResNet-50, which are used for the classification task. These layers can be seen as the feature extraction layers. Activation maps generated by those learned features extraction layers are usually called bottleneck features. Using these features of the image, we have trained 5 fully connected softmax and have classified the outputs as 0, 1, 2, 3 or 4.

Category	Description
0	No Wilting
1	Leaflets folding inward at secondary pulvins, no tugor loss in leaflets or petioles
2	Slight leaflet or petiole turgor loss in upper canopy
3	Moderate turgor loss in upper canopy
4	Severe turgor loss throughout canopy

D. One Hot Encoding

[4] One hot encoding is one of the most popular mechanism to convert the categorical data to the numerical data. Using natural ordering between categories of data in integer format may result in poor performance and biased results. Hence rather than labeling categories based on decimal system, we use a binary style of categorizing i.e. One Hot Encoding Mechanism. In our case the one hot encoding uses 5 bits to encode the different categories of plants and the one hot codes are as follows:

Category	One Hot Code
0	00001
1	00010
2	00100
3	01000
4	10000

II. Hyper Parameters and Model Training

The main hyper parameters that were considered for the implementation is the learning rate and mini batch size. The mini batch size was selected to be 16 which was normalized. Then the learning rate was tuned by using the method developed by Leslie Smith. The central idea is to train data for one epoch and increase learning rate for each mini batch. First, we started with a very small learning rate and increased it progressively for each of the successive iterations for each mini batch. We chose the learning rate to be 10 times lesser than the value where the training loss started to increase. Figure 2 shows the training loss for different learning rates. Now for Training we

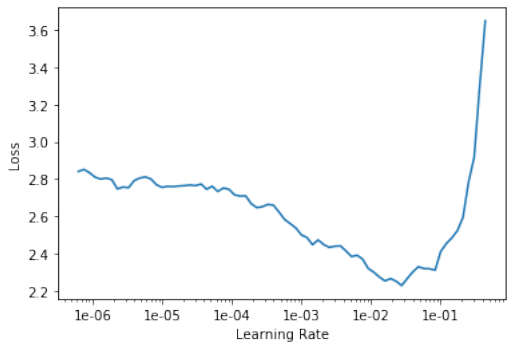

Fig. 2. Loss vs Learning rate Graph

use fit one cycle method which is describe as follows:

- The learning rate has been progressively increased from $lrmax\ /\ divfactor$ to lrmax and at the same time the momentum was progressively decreased from mom max to mom min
- Next the exact opposite was done by progressively decreasing the learning rate from lrmax to $lrmax\ /\ divfactor$ and at the same time progressively increasing the momentum from mom min to mom max.
- There after the learning rate has been further decreased from $lrmax\ /\ divfactor$ to $lrmax\ /\ divfactor \times 100$ by keeping the momentum steady at mom max.

Here lrmax is maximum learning rate and mom max is maximum momentum. The lr max we choose for this implementation is 3e-3 and the number of epochs is 50. The data set was also split into two categories i.e. the training ansd the validation set with 80 percent of the data for the former category and the remaining 20 percent for the latter.

III. Evaluation

The evaluation of the implemented network has been done in terms of training and validation loss. 50 epochs were used to train the Resnet-50 network and the results were recorded and plotted. Figure 3 shows the graph of the training and Validation data.We achieved the validation accuracy of 81.56

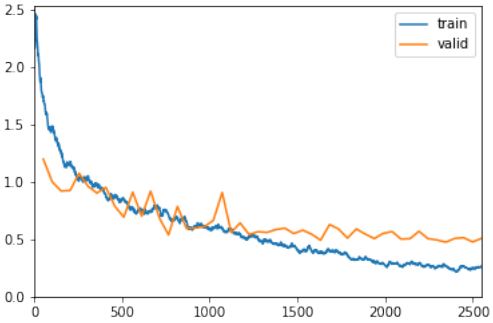

Fig. 3. Training and Validation Loss

Confusion matrix is another efficient means to measure the effectiveness of the prediction. Hence to measure the effectiveness the implemented model, the confusion matrix was also plotted. Figure 4 displays the confusion matrix for our implementation.

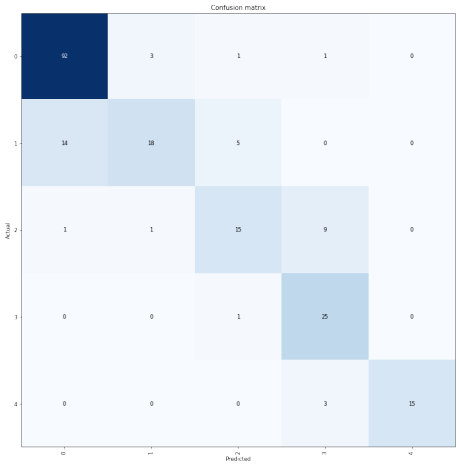

Fig. 4. Confusion Matrix

REFERENCES

[1] Dataset:https://drive.google.com/drive/u/1/folders/1BWLw_GehAQ9LxRW6P-jMnA0_-XFXQnR0

[2] Google Colab:https://colab.research.google.com/

[3] 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.

[4] Cerda, Patricio, Gaël Varoquaux, and Balázs Kégl. "Similarity encoding for learning with dirty categorical variables." Machine Learning 107, no. 8-10 (2018): 1477-1494.