CS5540 Project Final Report

Chickpea Leaf Classification

Yoonki Hong

1. Introduction

The goal of this project is to provide additional insights to a study on chickpeas conducted in 2016-2017 [1]. The impetus for this study was to understand adaptations to fertile and non-fertile soil environments to develop more resilient chickpea varieties in the face of climate change. Thousands of data points were collected from the 216 specimens used in the study, including hundreds of photographs of roots and leaves. One of their objectives was to investigate the relationship between leaf shapes and soil nitrogen content but it was too data-intensive and they gave up. This project is an opportunity to resurrect this idea and to analyze leaf shapes using machine learning. Convolutional Neural Network (CNN) will be used to classify the soil nitrogen contents of given leaf shapes.

2. Problem Definition and Algorithm

2.1 Task Definition

Our goal is to classify the soil nitrogen contents of a given image of a chickpea leaf. There are four nitrogen treatment levels: 1, 10, 50, and 100. Treatment 1 means that 1 ppm of N was contained in the soil (or 2.362 mg N source/L planting media).

Input is an image of a chickpea leaf and output is a 4 x 1 vector, Softmax activation applied. An image is resized before being fed into our CNN model. An element of the output corresponds to the probability of one of the nitrogen treatments. We predict that the corresponding treatment of the row which has the greatest value was contained for the given input image.

This project potentially enables analysis on how much nitrogen was applied to a chickpea only by observing an image of a life and can extend to the prediction of a plant given environment.

Figure 1. Example of an input image

$$\begin{bmatrix} 0.0 \\ 0.0 \\ 1.0 \\ 0.0 \end{bmatrix} \begin{array}{l} \text{– Probability of treatment 1 was used} \\ \text{– Probability of treatment 2 was used} \\ \text{– Probability of treatment 3 was used} \\ \text{– Probability of treatment 4 was used} \end{array}$$

Figure 2. Output vector

2.2 Algorithm Definition

(1) Convolutional Neural Networks (CNNs)

Multiple CNNs with different architectures are trained and tested in this project. CNNs are a type of neural network architecture that utilize convolutional layers to extract features from images through convolutional filters. They are widely used in the field of Computer Vision. In this project, we use images of chickpea leaves to predict the soil nitrogen content associated with the leaves. Therefore, CNNs are considered one of the most appropriate approaches to tackle this problem. We created two CNN models from scratch to train and test on our dataset.

(2) Transfer Learning

Transfer learning [2] is a learning technique where a model trained on other dataset is reused or modified with fine-tuning for a different task. In this project, we adopted this technique to utilize already trained model with huge datasets: Inception V3 [3] and Swin Transformer V2 [4].

In summary, we trained and tested four NNs: two custom models and two transfer learning models. Details about each model are described in '3. Experimental Evaluation' section.

3. Experimental Evaluation

3.1 Methodology

(1) Dataset

Our dataset consists of 877 scans of chickpea leaf laminates. Each laminate includes several leaves from the same plant. When it wasn't possible to fit all the leaves in a single laminate, several laminates were created for a single plant.

It is important to note that the quality of the scans and of the laminates is inconsistent. Chickpea leaves are small and made up of tiny delicate leaflets so it can be difficult to isolate the leaves and prepare the laminates. Furthermore, the way the leaves are laid out on each laminate is inconsistent; some display the leaves on horizontal lines, others mimic the patterns of the branches. Also, the light for the scans is inconsistent and the scale is different from one scan to the next.

Each scan is labeled according to the treatment applied: 1, 10, 50, and 100 (treatment refers to the amount of nitrogen added to the soil). The scans are located in 4 folders (1,10,50,100) according to treatment.

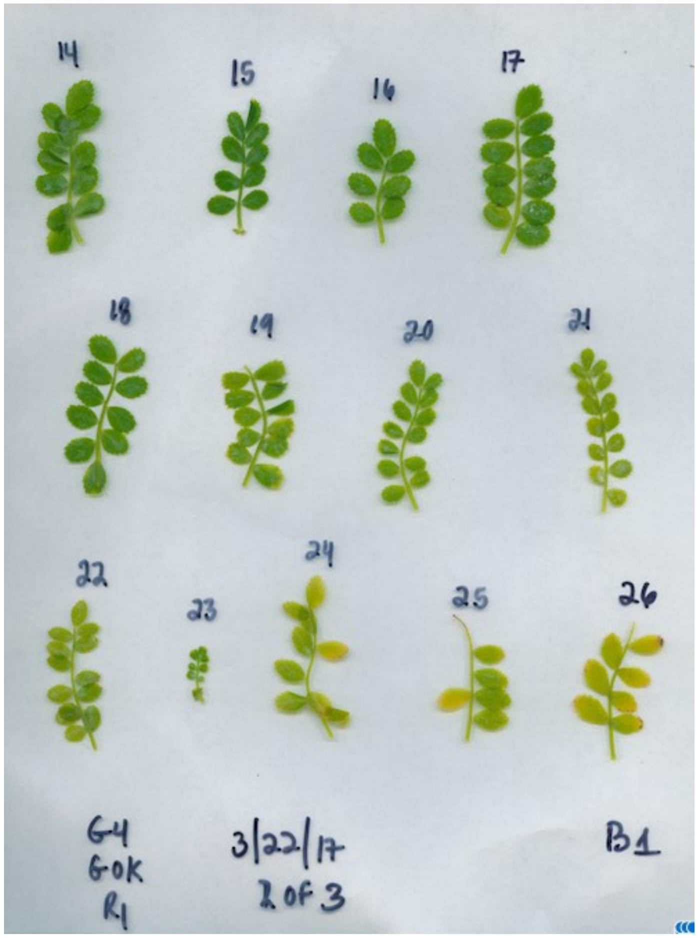

Figure 3. Example of a scanned laminate

We used ImgLab to create bounding boxes for each leaf and crop [5]. Metadata for the bounding boxes is saved with the images and in an XML file. An example cropped image is shown in Figure 1.

After cropping images, 1,542, 1,261, 1,490, and 2,263 images for each class were obtained. There was an imbalance in numbers of data for each class

(2) Data Balancing and Augmentation

To sort out the imbalance, we randomly picked 1,200 images from each class. This number was determined by the fact that the class with the least data number contains 1,261 images. As a result, total 4,800 images (1,200 per class) were utilized for training and validating the NN models.

As a next step, data augmentation was performed on the balanced dataset [6]. Using image modification algorithms such as rotating, shearing, flipping, and cropping. 10 additional images were created for each image. After augmentation, total 52,800 images were obtained. These augmented images were only used for training, and not for validation.

(3) Metric

Accuracy and Loss are used to evaluate our model. We compute the accuracies and losses for training and test data. We plot accuracies for training and validation data in the same plot, and losses for training and validation data in another plot to observe any strange behavior such as overfitting.

(4) Loss Function

Since the goal of this project is to classify among four categories, cross entropy loss is used.

$$CrossEntropyLoss = -\sum_{i=1}^{4} y_i \log(\hat{y}_i)$$

Where:

y_i : $The\ true\ label\ for\ class\ i\ (1\ if\ it\ is\ the\ correct\ class, 0\ otherwise)$.

$\hat{y}_i$: $The\ predicted\ probability\ for\ class\ i$

$\hat{y}_i$ is the output passed through the Softmax function to convert raw scores into probabilities.

Softmax function for class i:

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^{4} e^{z_j}}$$

(5) CNN models Used in the project

a. Base model

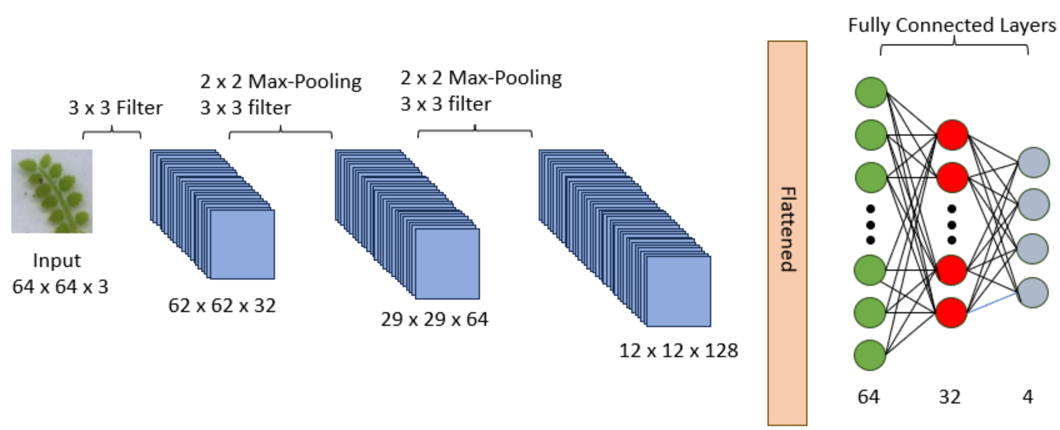

Figure 4. Architecture of the Base model

Base model consists of three convolutional layers and three fully connected (FC) layers and input shape is 64x64x3.

b. Base+ model

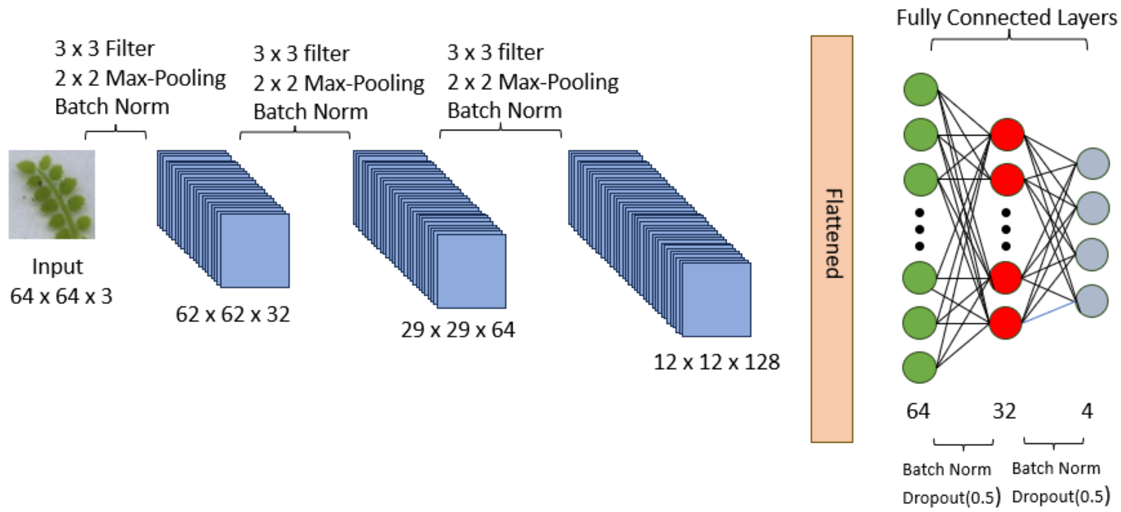

Figure 5. Architecture of the Base+ model

Base+ model consists of the same convolutional and fully connected layers. In addition, batch normalization layers are added between all layers, and dropout layers between FC layers.

c. Modified Inception V3 model – Transfer Learning

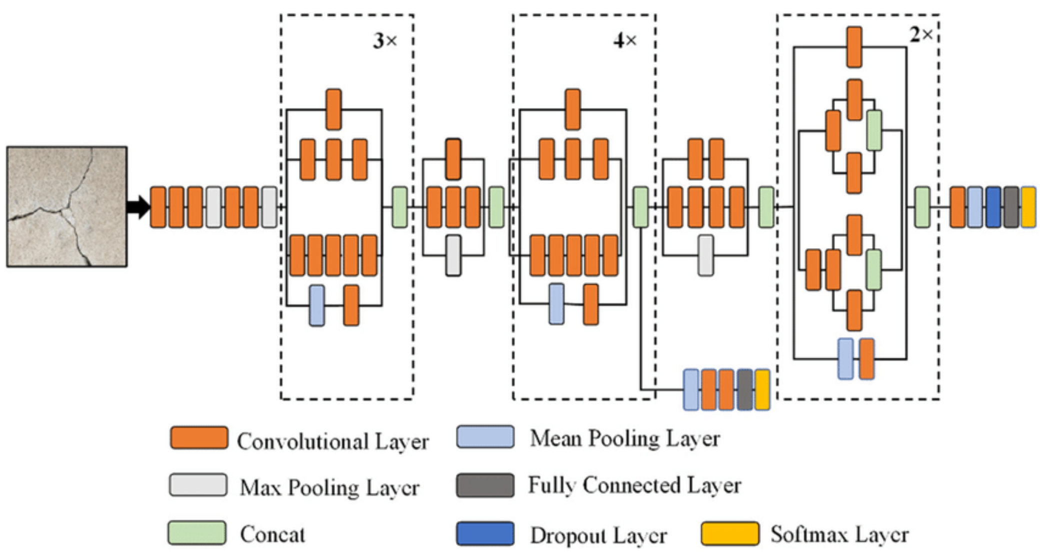

Figure 6. Inception V3 model architecture

Figure 6 shows the original architecture of the Inception V3 model [3]. We loaded the architecture and the pre-trained weights of the Inception V3 model, and modified the output layer (FC layer) from (2048 x 1000) to (2048 x 256 x 4). We set the output layer as (4x1) since we have four classes. When training, we froze all the layers except the output FC layers and the convolutional layer closest to the output FC layers.

d. Swin Transformer V2 model – Transfer Learning

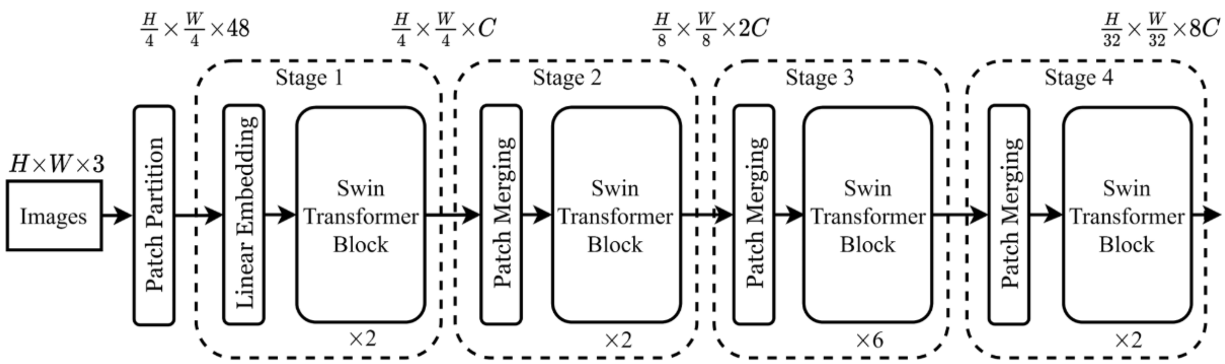

Figure 7. Swin Transformer V2 model architecture

We loaded the architecture and the pre-trained weights of the Swin Transformer V2 model [4], and modified the output layer from (1024 x 1000) to (1024 x 4). When training, we froze all the layers except the output FC layer.

3.2 Results

(1) Base model with imbalanced and not-augmented data set

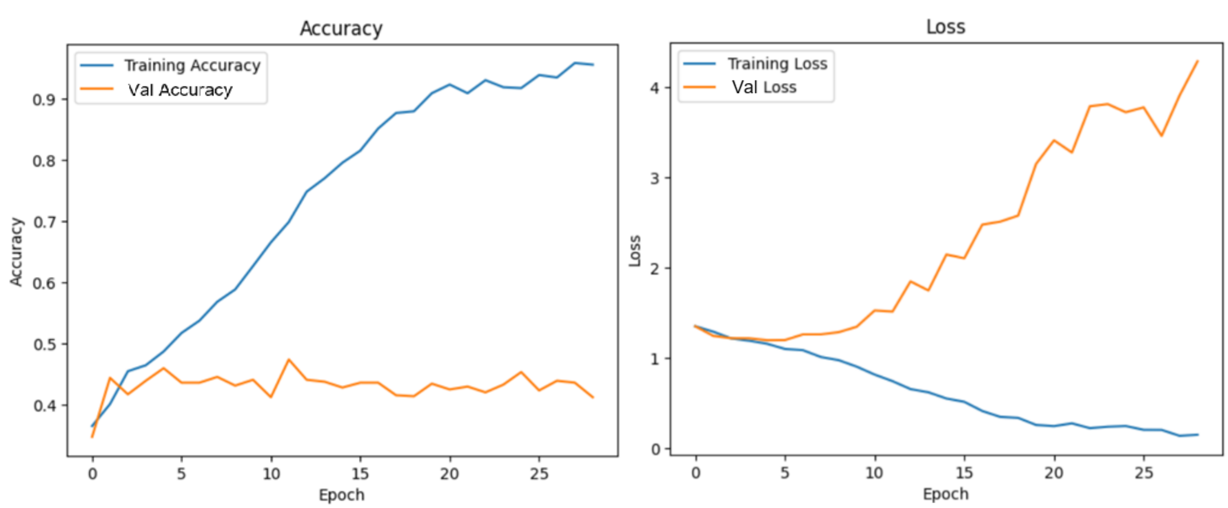

Figure 8 Results of Base model with imbalanced dataset

5,724 training and 633 validation data were used.

While training accuracy and loss consistently improved over epochs, validation accuracy and loss don't show similar improve. Validation loss rather diverges. There is a clear sign of an overfitting.

(2) Base model with balanced and augmented data set

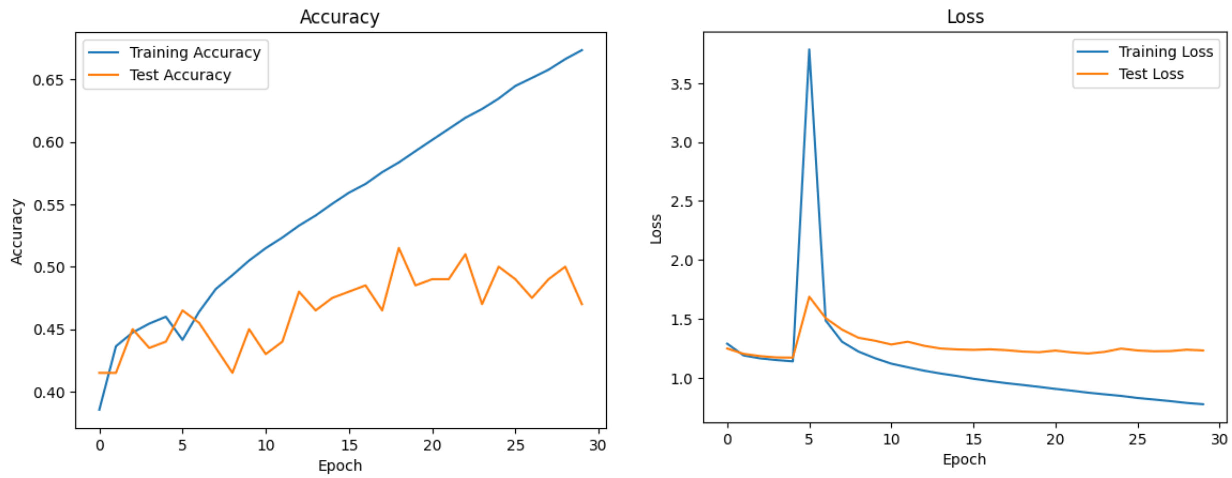

Figure 8 Results of Base model with balanced dataset

27600 training and 200 validation data were used.

Validation accuracy and loss are slightly better than the one yielded by the model trained with imbalanced data. Validation loss doesn't diverge anymore. However, the accuracy is merely a few percentages better and there is still an overfitting.

(3) Base model with balanced and augmented data set

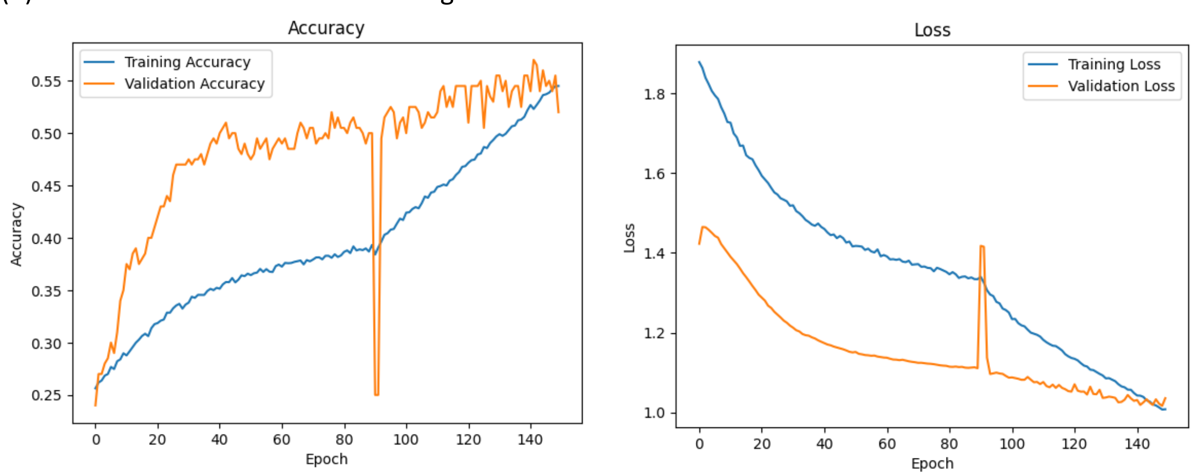

Figure 9. Results of Base+ model with balanced data set

50,600 training and 200 validation data were used.

Best accuracy is slightly below 55%. Learning seems relatively stable. Strange behavior around epoch 90 appeared after recompiling the model with a new learning rate. However, it has disappeared within a short period of epochs.

Normalization and dropout layers seem to work as regularization and resolve the overfitting at a certain level.

(4) Inception V3 model (Transfer Learning) with balanced data set

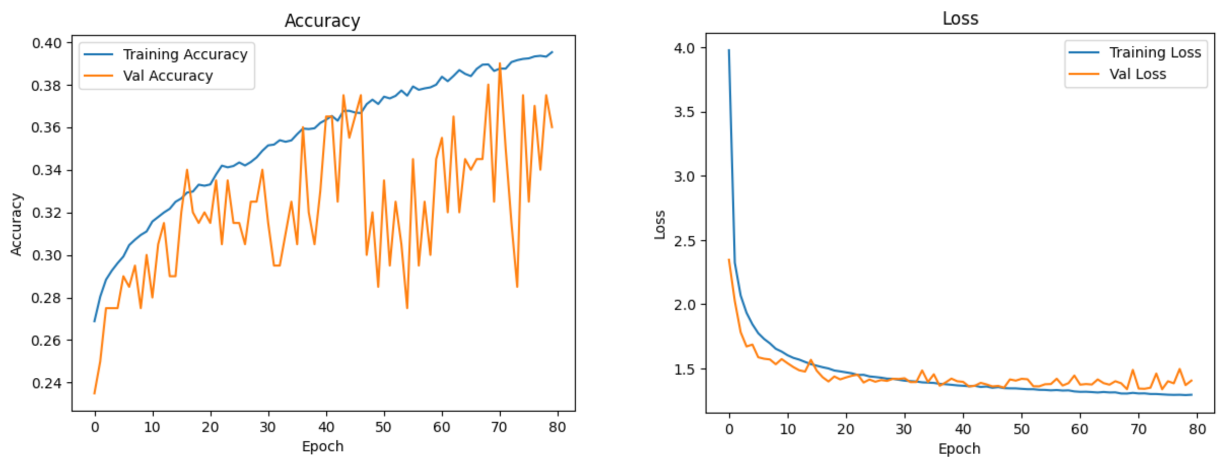

Figure 10. Results of Inception V3 (Transfer Learning) model

25200 training and 200 validation data were used.

Learning was unstable. The accuracy and Loss were inferior to those of Base model. It seems that the replaced layer was not appropriately shaped. In addition, deeper fine-tuning might be needed to produce a better performance.

(5) Swin Transformer V2 model (Transfer Learning) with imbalanced data set

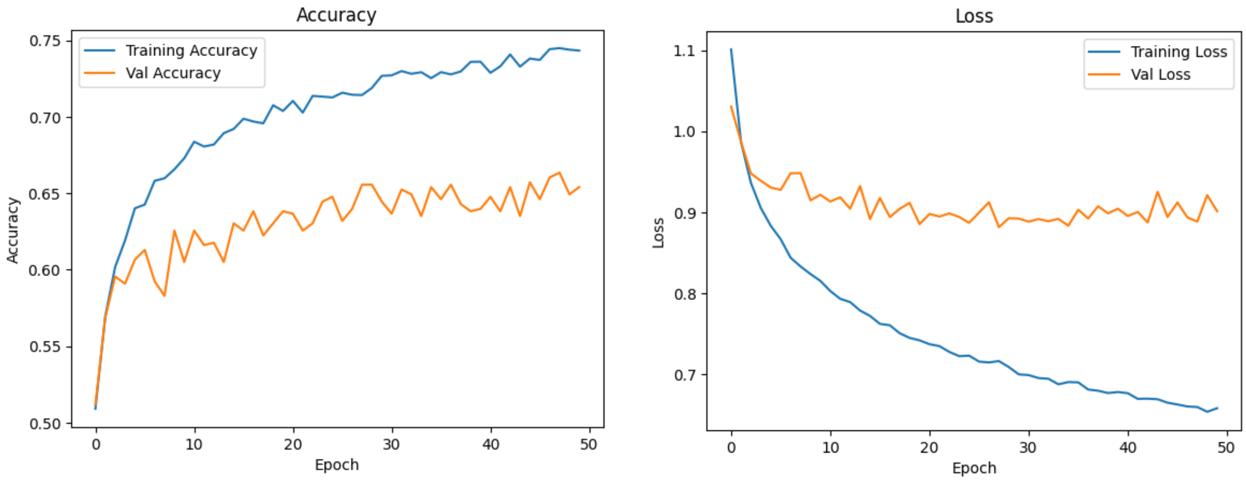

Figure 11. Results of Swin Transformer V2 (Transfer Learning) model

5,724 training and 633 validation data were used.

The best accuracy and loss, around 65% and 0.9 respectively, were achieved by Swin Transformer V2 model. This model shows fairly stable learning.

3.3 Discussion

The Base model showed moderate improvements in performance with data balancing and augmentation, though overfitting remained an issue. The Swin Transformer V2 model achieved the highest accuracy (~65%) and demonstrated stable learning, while the Inception V3 model underperformed, likely due to issues with layer modifications and insufficient fine-tuning.

Transfer learning models, particularly Swin Transformer V2, performed well, showing the benefits of leveraging pre-trained architectures. Data preprocessing steps like balancing and augmentation improved validation performance and reduced overfitting, but challenges with the Inception V3 model highlighted the importance of proper model adjustments and deeper fine-tuning.

4. Related Work

Marques et al. used the WinRHIZO image analysis system (version *Arabidopsis*) on the leaf scans to determine leaf areas. A nested linear mixed model (LMM) was used in R (*lmertest* package) to analyze the correlation between leaf area (one of many measurements), treatment (high or low soil nitrogen content), chickpea variety (there were 27), and domestication history (i.e., domesticated or wild variety) [1].

Haider et al. computes a leaf's color similarity with the reference colors to estimate nitrogen content. In particular, a green color value index was used as a nitrogen indicator [7]. Othman et al. adopted a CNN to classify images of coriander and parsley leaves [8].

Keivani et al. used leaf geometric features, color, texture, and vein patterns for plant identification. They tested several ML techniques and concluded that decision tree was the best technique [9].

Afonso et al. reviewed the use of transfer learning in computer vision to address data limitations by reusing models trained on one domain for related tasks. They highlighted recent advancements in transfer learning techniques and their impact on solving diverse computer vision problems with reduced training data requirements [2].

In this project we combine techniques from previous works in an effort to be more efficient. A CNN will be used to classify soil nitrogen content of Chickpeas with an image of a Chickpea's leaf. Various CNN architectures will be tried to find the best model.

5. Code and Dataset

(1) First, we used 'sort_by_treatment.py' file to split the raw scan files into folders which are named as nitrogen treat level: 1, 10, 50, and 100.

(2) Then, we used Imglab [2] application to store the coordinates of leaves in each scan. This information is stored in xml files.

(3) We used functions in 'read_xml.py' to crop leaves to each separate image file.

(4) We used 'pick_test.py' file to randomly pick the data.

(5) We used 'image_augment.py' to augment the data

(6) We used 'experiment_tf.ipynb' to create our custom CNN models and load the Inception V3 model for transfer learning. We also trained and plotted the results with this file

(7) We used 'transfer_swinV2_torch.ipynb' to perform experiment with Swin Transformer V2 model.

Github link - https://github.com/u0953009/CS5540_Project

6. Conclusion

6.1 Summary

In this project, we aimed to classify soil nitrogen levels of chickpea leaves chickpea plants using NN models. We explored the effectiveness of CNNs and transfer learning models on our problem. The Swin Transformer V2 model achieved the best performance with an accuracy of 65%, demonstrating stable learning and highlighting its potential for this task. While the Base+ model with regularization techniques showed a certain level of promise, leaving a challenge of overfitting. Transfer learning, as demonstrated with Swin Transformer V2, proved to be more effective than the custom Base models.

A few challenges arose in data sets and exploring complex models. The dataset posed challenges due to inconsistent scan quality, lighting, and layout, which likely impacted model learning. In addition, the extended duration required for training made it infeasible to explore a sufficient number of different architectures of NNs.

6.2 Future Work

Reflecting on the challenges, there are a few areas to explore for finding improved solutions to our problem.

First, obtaining more data is crucial. We have observed that even a relatively simple neural network exhibited overfitting with the dataset before augmentation. It indicates the need for a larger dataset to achieve higher accuracy. Furthermore, if it is possible to get more high-quality data, a significant improvement in performance could be achieved.

In addition, the effectiveness of transfer learning was demonstrated by the Swin Transformer V2 model, which yielded the best results in our experiments. This suggests that further exploration of transfer learning approaches could lead to promising outcomes. As a next step, we propose experimenting with the modified Swin Transformer V2 by enabling one or more additional layers close to the output layer to learn. Allowing these layers to adapt to the project-specific data could enable the model to capture more detailed features, potentially leading to improved performance.

Reference

[1] Marques, E., Krieg, C. P., Dacosta-Calheiros, E., Bueno, E., Sessa, E., Penmetsa, R. V., & Von Wettberg, E. (2020). The impact of domestication on aboveground and belowground trait responses to nitrogen fertilization in wild and cultivated genotypes of chickpea (Cicer sp.). Frontiers in Genetics, 11, 576338.

[2] Panda, A., Panigrahi, D., Mitra, S., Mittal, S., & Rahimi, S. (2024). Transfer Learning Applied to Computer Vision Problems: Survey on Current Progress, Limitations, and Opportunities. *arXiv preprint arXiv:2409.07736*.

[3] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2818–2826. https://doi.org/10.1109/CVPR.2016.308

[4] Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Zhang, J., Cao, Y., Zhang, Z., Dong, L., Wei, F., Lin, S., & Guo, B. (2022). Swin Transformer V2: Scaling Up Capacity and Resolution. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 12009–12019. https://doi.org/10.1109/CVPR52688.2022.01171

[5] Imglab. https://github.com/NaturalIntelligence/imglab

[6] Yang, S., Xiaon W., Zhang M., Guo, s., Zhao, J., Shen, F. (2022) "Image Data Augmentation for Deep Learning: A Survey" arXiv:2204.08610

[7] Haider, T.; Farid, M.S.; Mahmood, R.; Ilyas, A.; Khan, M.H.; Haider, S.T.-A.; Chaudhry, M.H.; Gul, M. A Computer-Vision-Based Approach for Nitrogen Content Estimation in Plant Leaves. Agriculture 2021, 11, 766. doi.org/10.3390/agriculture11080766

[8] N. A. Othman, N. S. Damanhuri, N. M. Ali, B. C. Chiew Meng and A. A. Abd Samat, "Plant Leaf Classification Using Convolutional Neural Network," 2022 8th International Conference on Control, Decision and Information Technologies (CoDIT), Istanbul, Turkey, 2022, pp. 1043-1048, doi: 10.1109/CoDIT55151.2022.9804121.

[9] Keivani, M., Mazloum, J., Sedaghatfar, E., & Tavakoli, M. B. (2020). Automated analysis of leaf shape, texture, and color features for plant classification. TRAITEMENT du Signal, 37(1), 17-28. doi.org/10.18280/ts.370103