CS 480 Final Project

Diana Brebeanu
School of Computer Science
University of Waterloo
Waterloo, ON, N2L 3G1
dbrebean@uwaterloo.ca
report due: August 12

Abstract

This project explores the use of machine learning, specifically Convolutional Neural Networks (CNNs), to predict six plant traits based on images and corresponding ancillary data. Inspired by previous studies, we implemented a model with two branches: a CNN for image processing and a feedforward network for ancillary data. Various architectural and data preprocessing strategies were evaluated, including fine-tuning a ResNet50-based model. The best configuration achieved an R^2 score of 0.232 on the test set, indicating the potential for further refinement, particularly through advanced feature extraction techniques and ensemble methods. Code for reproducing the results is available on GitHub.

1 Introduction

Well-functioning ecosystems are a key part of the well-being of humans and other organisms and yet they can be greatly affected by environmental issues caused by humans. A way of assessing the functioning of ecosystems can be done through analyzing plant traits such as leaf area, growth height, leaf nitrogen concentration and stem specific density among others. In order to optimize these assessments, it is in the interest of scientists to develop tools that can efficiently measure these plant traits which can then be used to assess the health of the surrounding environment. One such method that we will investigate is doing so through the use of machine learning, more specifically, Convolutional Neural Networks to analyze of images of plants paired with the analysis of ancillary information about the local climate, soil and location.

We focus on constructing a model outlined in Figure 1 to predict the 6 trait values which was inspired by a 2021 study (Schiller et al. 2021). As mentioned in Schiller et al. 2021, CNNs are useful for learning image features and can be applied to learning specific plant traits that are correlated to visible plant features detected by the model. Our model architecture consists of two parallel branches to deal with the two formats of data: a Convolutional Neural Network (CNN) to extract information from the images and a feedforward network that extracts information from the corresponding numerical ancillary data. Afterwards, a regressor combines the outputs of each branch to form final predictions on the trait values. The model is then to be trained on a set of 43363 images paired with their ancillary data and used to predict the 6 traits on unseen test data numbering 6391 samples.

2 Related Works

In 2021, the study *Deep learning and citizen science enable automated plant trait predictions from photographs* (Schiller et al. 2021) conducted predictions for the plant traits on a similar dataset consisting of images of plants collected from the iNaturalist database matched with ancillary data from the TRY database and bioclimatic data from the Wordclim database. In the study, training set images were squared and down-sampled to a size of 512x512 pixels. Moreover, a $\log_{10}$ transformation

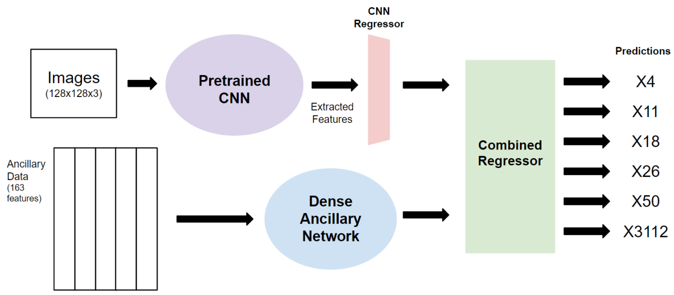

Figure 1: Overall Model Architecture inspired by Schiller et al. 2021.

was applied to the target values and the features and target values were normalized according to the min-max normalization scheme: $target_{norm} = \frac{target - target_{min}}{target_{max} - target_{min}}$. One of the outlined model architectures consisted of two parallel branches: a CNN that worked on the images and a dense feed-forward network that worked on the ancillary (Bioclim) data. The CNN utilized the pre-trained weights from the Inception-Resnet-v2 model. Both branches were then concatenated using a dense-layer regressor. The model was trained using batch sizes of 20 samples using an RMSprop optimiser. This study was limited to its smaller dataset which contained less features than the current dataset we are working on, making generalization of the solution a problem as demonstrated in Section 3. Moreover, the study used these models to predict on each trait separately which resulted in higher R^2 scores for certain trait predictions for the ensemble CNN model such as Growth Height (0.56) and Leaf Area (0.5) and lower R^2 scores for Stem Specific Density (0.2).

A separate study published in 2023, *Effects of Different Pretrained Deep Learning Algorithms as Feature Extractor in Tomato Plant Health Classification* (Chong et al. 2023), compared five different pretrained deep learning CNNs on classifying tomato plant health. The study focused on ResNet50, AlexNet, GoogleNet, VGG16 and VGG19. It ran images of these plants through each of these networks to extract specific features and then used these features to train a Support Vector Machine (SVM) to classify the images. The results of the study indicate that the SVM coupled with ResNet50 gave the best training and testing accuracies of 98.26% and 93.33% respectively. This study differs from our current problem as it worked on classifying the health of the tomato plants, a simpler problem than predicting the 6 continuous trait values. However, the accuracy achieved using ResNet on plant images specifically is worth noting.

3 Main Results

We are given a set of plant images along with corresponding ancillary data, matched through their ids. Our goal is to predict the continuous values for the 6 plant traits based on the data given to us. As we are predicting these continuous values, our aim is to minimize the mean squared errors: $MSE = \frac{1}{6}\sum_{i=1}^{6}(y_i - \hat{y}_i)^2$ where $\hat{y}_i$ is the prediction and y_i is the actual value for trait i.

3.1 Recreating the Schiller et al. model

As the problem aligned closely with the Schiller et al. 2021 study, the first step taken was recreating the model architecture and data preprocessing steps outlined in the study to see how well it could generalize to a larger dataset. The $\log_{10}$ values were taken from target features in the training set. Then, outliers that were not within 3 standard deviations of the mean were removed from the training dataset, reducing the size of the dataset from 43363 samples to 39556. Finally, all features and targets of the training and test sets underwent min-max normalization. The training data was then split into training and validation sets with a 4:1 ratio. The images of the training underwent augmentation through randomized horizontal/vertical flips, brightness, contrast, saturation changes between 0.9 and 1.1, allowing for regularization.

The model architecture consisted of a CNN using pretrained weights from ResNet50 which worked on the images. ResNet50 was chosen as a base starting point for the CNN for its accessibility via PyTorch and accuracy as mentioned in the Chong et al. 2023 study. The CNN regressor consisted of an average-pooling layer with two dense layers of 512 and 4 output units. The ancillary data was then fed into a dense feed-forward network with 3 dense layers of 64, 32 and 4 output units. The final regressor also contained 4 dense layers with 8, 8, 6, 6 output units for the 6 traits, modifying the study's configuration to output 6 units instead of 1. RMSprop was set as the optimizer. This configuration resulted in an R^2 score of 0.06 on the test dataset showing that the architecture and pre-processing steps performed in the study were not transferable to a larger dataset.

3.2 Data Preprocessing

The first step in improving the results was modifying the data pre-processing. After removing the $\log_{10}$-transformation of the targets and replacing min-max normalization of the features and targets with z-score normalization, the R^2 score on the test set largely improved to 0.187. Z-score normalization was chosen as an alternative as it is an industry-standard technique and decreases the effect that outliers can have on the predictions (Turing 2024).

3.3 Ablation

The RMSprop optimizer was then swapped with Adam in favour of its robust hyperparameters as well as use of momentum to accelerate convergence (Kingma and Ba 2014) which boosted the R^2 score on the dataset to 0.191.

In order to determine how accurate the predictions were on each trait, 6 separate models with the same updated configurations were run which gave the R^2 scores in Figure 2. Predictions on some of the traits improve over each epoch as the R^2 are seen to steadily increase except for the the traits X11, X26 and X3112. The highest R^2 score is only 0.3 which is achieved on the training set for the third trait X18 while the validation score jumps down. This could mean that the model was under-fitting to the data and its architecture did not allow it to extract enough information from the images as well as capture patterns in the data.

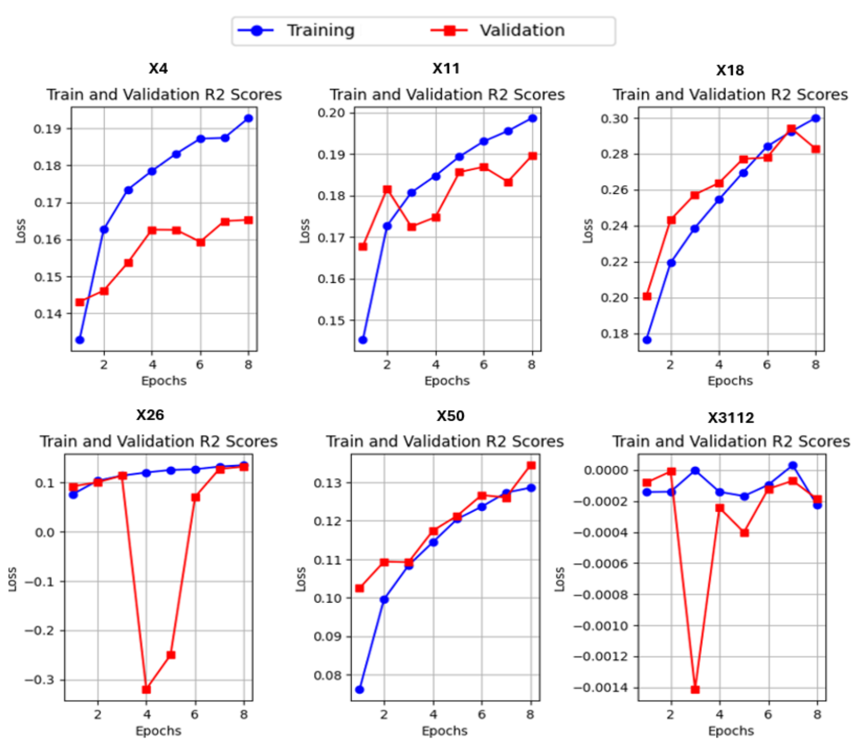

Figure 2: The training and validation losses and R^2 scores on Resnet50 with Adam optimizer over 8 epochs ordered as: X4, X11, X18, X26, X50, X3112.

Table 1: Comparison of Base CNN models and additional changes

Pre-trained CNN Model	**Changes**	R^2 **Score**
ResNet50	Replaced RMSProp with Adam	0.191
ResNet50	Self-Attention on CNN outputs and final regressor	0.180
ResNet50	AdamW with weight decay 0.1	0.156
ResNet50	AdamW with weight decay 0.01	0.198
ResNet50	Deeper Networks	0.176
EfficientNet	Replaced RMSProp with Adam	0.176
EfficientNet	Self-Attention on CNN outputs and final regressor	0.175
EfficientNet	Deeper Networks	0.176

Further improvement came with introducing regularization through the use of the AdamW optimizer in PyTorch which allows for the specification of the weight decay (PyTorch 2024). This optimizer implements decoupled weight decay regularization outlined in the Loshchilov and Hutter 2017 paper. The R^2 score improved only slightly from 0.191 to 0.198.

Several changes were also attempted including changing the base CNN model to EfficientNet to allow for faster computations due to time limitations while still keeping a similar accuracy. A 2019 study (Tan and Le 2019) demonstrated how EfficientNet maintained a high accuracy on ImageNet while being more computationally efficient. The other changes made included tuning the AdamW weight decay hyperparameter and deepening the regressor and ancillary networks to increase model complexity. These changes are summarized in Table 1. To prevent over-fitting on the data, 5-fold cross validation was also implemented on the configuration with the highest R^2 score at the time (ResNet50 base model that used the AdamW optimizer) which resulted in a lower R^2 score of 0.173 most likely due to the reduced training data.

3.4 Fine-tuning

Focusing on the base ResNet50 model's ability to extract features, the model's parameters were fine-tuned during training after 4 epochs over a total of 8 epochs of training. This strategy, informed by the findings of Yosinski et al. 2014, allowed the first 4 epochs to focus on adapting the regressor layers and ancillary network to the data while the ResNet50 layers remained frozen. Subsequently, the ResNet50 layers were also trained, enabling the entire model to better adapt to the task. This approach resulted in an improved R^2 score of 0.232 and the reduced losses compared to previous configurations.

4 Conclusion

After working on the model architecture consisting of a single CNN paired with an ancillary network, it is evident that the single model was too weak to capture enough information from the images and numerical data to form accurate trait predictions. The best performing model consisted of one in which the pre-trained CNN was fine-tuned to adapt more closely to the plant images and the regression task. This shows that focusing more on specific feature extraction from the plant images to be paired with ancillary data merits greater exploration. For example, a study done in 2022 (Amulya et al. 2022) showed how to successfully extract features from plant leaves using the Gabor filter (which has many applications in texture analysis). These features were then passed to a CNN which was able to classify plants as medicinal vs non-medicinal with an accuracy of 97%. Applying this technique to regression problems based on images would be something to explore in the future.

Another promising avenue, also inspired from Schiller et al. 2021, would be to use the ensemble model architecture where several models are combined to form more accurate predictions. Moreover, using attention mechanisms to place higher importance on relevant features and intermediate outputs would allow for the potential reduction in noise and more accurate predictions. This could be applied both to the CNN output as self-attention as well as when combining the CNN and ancillary network outputs through cross-attention. These are some of numerous avenues to explore for accurately predicting plant traits. The expanding resources and data in this field, along with new discoveries, are likely to have valuable applications in other domains.

Acknowledgement

I would like to thank Professor Yaoliang Yu and Saber Malekmohammadi as well as the course TAs for providing timely feedback and accurate resources both during the project timeline as well as the term. The architecture for the model in Figure 1 is inspired by the Schiller et al. 2021 study.

References

Ahmad, M. U., S. Ashiq, G. Badshah, A. H. Khan, and M. Hussain (2022). "Feature Extraction of Plant Leaf Using Deep Learning". *Complexity*, vol. 2022, no. 1, p. 6976112. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1155/2022/6976112`.

Amulya, K., D. K Deepika, and D. P Kamakshi (2022). "An optimized hyper parameter-based CNN approach for predicting medicinal or non- medicinal leaves". *Advances in Engineering Software*, vol. 172, p. 103181.

Chong, H. M., X. Yap, and K. Chia (2023). "Effects of Different Pretrained Deep Learning Algorithms as Feature Extractor in Tomato Plant Health Classification". *Pattern Recognition and Image Analysis*, vol. 33, pp. 39–46.

G, C. (2020). "Fine-Tuning a Model Using PyTorch (Part 2)". `https://medium.com/@caterine/fine-tuning-a-model-using-pytorch-part-2-e86a548ac4fb`. Accessed: 2024-08-11.

Kingma, D. P. and J. Ba (2014). "Adam: A method for stochastic optimization". *arXiv preprint arXiv:1412.6980*.

Loshchilov, I. and F. Hutter (2017). "Fixing Weight Decay Regularization in Adam". *CoRR*, vol. abs/1711.05101. arXiv: `1711.05101`.

PyTorch (2024). "AdamW Optimizer Documentation". `https://pytorch.org/docs/stable/generated/torch.optim.AdamW.html`. Accessed: 2024-08-12.

Schiller, C., S. Schmidtlein, C. Boonman, A. Moreno-Martínez, and T. Kattenborn (2021). "Deep learning and citizen science enable automated plant trait predictions from photographs". *Scientific Reports*, vol. 11, no. 1, p. 16395.

Tan, M. and Q. V. Le (2019). "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". *CoRR*, vol. abs/1905.11946. arXiv: `1905.11946`.

TorchVision (2024). "ResNet50". `https://pytorch.org/vision/main/models/generated/torchvision.models.resnet50.html`. Accessed: 2024-08-12.

Turing (2024). "Effects of Normalization Techniques on Logistic Regression in Data Science". `https://www.turing.com/kb/effects-of-normalization-techniques-on-logistic-regression-in-data-science`. Accessed: 2024-08-12.

Yosinski, J., J. Clune, Y. Bengio, and H. Lipson (2014). "How transferable are features in deep neural networks?" *CoRR*, vol. abs/1411.1792. arXiv: `1411.1792`.