Plant Traits Prediction Using ViT

Malvika Patel
School of Computer Science
University of Waterloo
Waterloo, ON, N2L 3G1
malvika.patel@uwaterloo.ca

Abstract

This report details the procedure and result analysis of using Convolutional Neural Networks and Vision Transformers to predict plant properties (also known as plant traits) from citizen science plant photographs, and their corresponding ancillary information. The results are evaluated based on the R^2 value of 6 target traits. Github Link: Predicting Plant Traits

1 Introduction

The 21st century brings exponential change to ecosystem functioning and processes leading to a loss in biodiversity. To measure ecosystem functioning, plant functional traits such as plant size, tissue constituents or stem specific density are used as they represent a "trait composition of the species that compose a plant community" (Schiller et al., 2021). However, given the necessary data, it is difficult to measure these traits in an efficient and accurate manner due the large volume of data available.

This experiment uses citizen science plant photographs, and their corresponding ancillary information from the TRY database paired with sample images from the iNaturalist database with plant trait data that scientists have been curating for decades for various species. Each image is associated with 164 columns of ancillary information related to the condition of its local climate, soil and some information gathered from satellites. The objective of this experiment is to train 6 trait values, namely: X4, X11, X18, X26, X50, X3112. The accuracy of the predictions are measured using the coefficient of determination, R^2 value. This regression problem is solved using a Vision Transformer, which is a transformer based model adapted for image processing.

2 Related Works

In an effort to facilitate rapid ecosystem monitoring to understand the impacts of environmental changes on biodiversity, there have been several efforts to use machine learning in using plant photograph databases and related numerical data to predict plant species identification specifically through plant traits. This experiment initially approached the problem using a CNN to train the images and ancillary data which has been explored by various researchers. One such research is conducted by Schiller, C., Schmidtlein, S., Boonman, C. et al. *Deep learning and citizen science enable automated plant trait predictions from photographs*. They used Convolution Neural Networks (CNNs) in four different approaches to predict the plant traits. During these approaches, a baseline model of Inception-Resnet-V2 was used with plasticity (target augmentation) by taking the standard deviations from the TRY database, altered within a Gaussian distribution and clipped between 0 and 1 after normalization (Schiller et al., 2021). For the training process, they used a batch size of 20 and an RMSprop optimiser with a learning rate of 0.001 and a learning rate decay of 0.0001. The plasticity approach was not explored by the ResNet50 model used in this experiment and instead a batch size of 32 and learning rate of 1e-5 with Adam optimizer was used.

Submitted to CS480/680, 2024 Spring. Do not distribute.

In another experiment on predicting plant traits conducted by Adil Sheerazi, they experimented on three different models to predict the plant traits: 1) ResNet50, 2) Improved ResNet50 and 3) A Three Way Ensemble. In the first approach using ResNet50, they trained images sized 224x224. In this experiment, the image size was 128x128 when training ResNet50. Both experiments used standard scalar normalization, trained over 10 epochs with batch size 32 and used Adam optimizer. For their experiment however, a log10 scaling was applied in the training to the output variables which was not applied during this experiment. Their experiment yielded a R^2 value of -34.27399 whereas this experiment using ResNet50 yielded 0.19855. Their second approach was optimizing the ResNet50 model to use data augmentation and a learning rate scheduler. Both these approaches were considered in this experiment as well, again yielding 0.19855 compared to their score of -17.71411. Their final model was a combination of different kinds of models namely, Vision Model, Weather Model and Soil Model. This approach yielded a R^2 value of -0.02982. Although this experiment's R^2 value out performs their R^2, combining different models as in their third approach could yield a better score for the CNN trained in this experiment. However, due to the limitation of compute power on the training environment, all possible models were not explored in this experiment.

3 Main Results

Predicting plant traits based on the iNaturalist and TRY databases requires a regression problem to be solved through the use of machine learning to process and train the image data augmented with numerical data. In this experiment, a pre-trained Vision Transformer from the Hugging Face Library was used to train the ancillary features and images due to its ability to perform better on large datasets such as `ImageNet` compared to CNNs. ViT is pre-trained on the "ImageNet-21k dataset" and uses a patch size of 16x16 pixels and input resolution of 224x224 pixels (n.d). A pre-trained model was preferred as it is trained on large, diverse datasets resulting in substantial fine-tuning and better performance for smaller, specific datasets compared to a model created and trained from scratch. In a high-level context, ViT operates by dividing an image into fixed-size patches. These patches are then linearly embedded and fed into a transformer model. Self-attention is used to capture dependencies and relationships between different parts of the image (Boesch, 2024). Below is detailed diagram on the ViT transformer architecture (Figure 1): Upon training the ViT model on

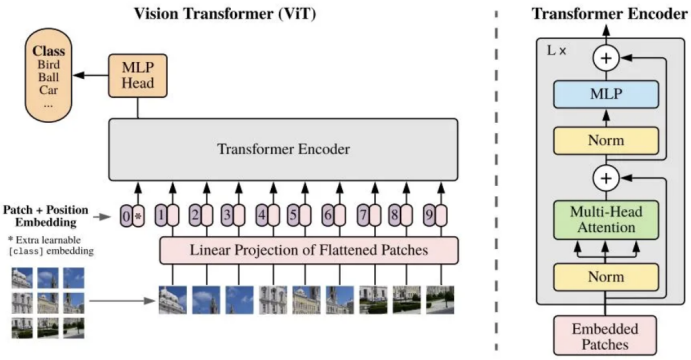

Figure 1: The image is split into fixed-size patches and linearly embedded. Position embeddings is then added to the image which outputs a sequence of vectors that is fed into a standard transformer encoder (Fahim Rustamy, 2024).

10 epochs of batch size 8, the associated R^2 value yielded a score of 0.26869 on 55% of the test data. These results are formulated from applying a variety of pre-processing and optimization techniques on the data set and the ViT model. Below is a step by step methodology of the pre-processing and optimizations applied to the regression problem:

1. The features (columns 1-164) and targets (remaining 6 columns) are extracted and separated into X_ancillary (ancillary features) and y (targets).
2. Using z-scores ($Z = \frac{x-\mu}{\sigma}$), outliers are removed from the training and test sets using a standard deviation of $\sigma = \pm 3$. This step is done since using a standard scaler in step 3

cannot guarantee balanced feature scales in the presence of outliers. The outliers skew the data when taking the mean and subtracting each values from the standard deviation.

3. Both the X_ancillary data and test data are normalized using a StandardScaler (each value is normalized by subtracting the mean and dividing by the standard deviation).
4. The training and test sets are down sampled by randomly selecting a smaller fraction of the data to reduce their computational load and prevent over fitting. This experiment randomly samples 30% of the rows in the training set.
5. Image transformations are applied to each image. Each image is resized to 224x224 pixels as required by the ViT standard. Then, a random horizontal flip and a random rotation of 10 degrees is applied. The pixel values of each image are then scaled using `transformers.ToTensor()` from the range $[0, 255]$ to $[0, 1]$. Finally, the image tensors are normalized using a specified mean and standard deviation for each color channel to ensure consistency in input distribution. The standard (mean=$[0.485, 0.456, 0.406]$, std=$[0.229, 0.224, 0.225]$) for the ViT pre-trained model was used for this experiment.
6. Dataloader is used to load the data in mini-batches, shuffle the data and use multi-threading to load the data. Due to memory constraints on the training environment (Google Colab with Macbook Pro M2 Chip 24 GB RAM), a batch size of 8 samples was used. The training set is shuffled in the beginning of each epoch to avoid order-specific training biases. Based on the training environment, 8 CPU cores were used to load the data in parallel for faster data loading. In addition, a pre-fetching factor of 2 was chosen to load 2 batches of data in advance to optimize CPU utilization.
7. AdamW optimizer, a variation of Adam optimizer, decoupled with weight decay (regularization) to prevent overfitting is used to train the model.
8. A gradient scaler for mixed precision training is used to avoid numerical instability issues when using 16-bit floating-point arithmetic. It scales the loss before back propagating to prevent underflow. This is mainly used to speed up training and reduce RAM usage on a low-computationally powered environment.
9. RandomForestRegressor and XGBRegressor (gradient boosting algorithm) were used to improve accuracy and reduce overfitting. The RandomForestRegressor fits a number of decision tree regressors on various sub-samples of the dataset and uses the best split strategy averaging to improve the predictive accuracy and control over-fitting (n.d). After performing randomized search, the best RandomForest and XGBoost models are selected to be retrained on the entire training set. The predictions from these two regressors are combined with predictions of the transformer model to evaluate the final predictions for the plant traits. For this experiment, the traits are assigned as y_test_predictions = (0.5 * transformer_predictions + 0.25 * randomforest_predictions + 0.25 * xgb_predictions) with the transformer predictions given more weight than the regressor predictions.
10. RandomizedSearchCV was used for hyper parameter tuning. It uses k-fold cross-validation to optimize classifiers used for predictions.

Before extensively training on ViT, ResNet50 was explored. ResNet50 is a convolutional neural network with 50 neural network layers used for image classification. CNNs take an image as input and process the image through convolutional layers using filters to extract features. It uses residual connections to mitigate the vanishing gradient problem (gradients used to update the network become extremely small or converge to 0 as they are backpropogated from the output layers to the initial layers), allowing the model to learn effectively (n.d). The ResNet-50 architecture can be broken down into 6 parts: 1) Input Pre-processing, 2) Cfg[0] blocks, 3) Cfg[1] blocks, 4) Cfg[2] blocks, 5) Cfg[3] blocks, and 6) a fully-connected layer. Input is passed through preprocessing and sent to Cfg[0] blocks where batch normalization, ReLu and Max Pooling occurs. The output of Cfg[0] blocks is sent to Cfg[1] blocks and propagated down to the Cfg[3] blocks. After applying AveragePooling, a Dropout layer and fattening the feature map, it is pushed into a Fully Connected Layer which is then used for producing predictions (Mukherjee, 2022).

However, upon comparing the accuracy of both models, ViT outperformed ResNet50 due its ability to perform well on large, complex datasets. CNNs are renowned to perform well on smaller datasets since convolutions are used to learn local features creating a strong inductive bias. Below is a figure (Figure 2) showcasing the mean-squared error (MSE) loss on ResNet50 and ViT. Based on the graphs, we see that ViT results in a lower MSE loss over higher epochs compared to ResNet50.

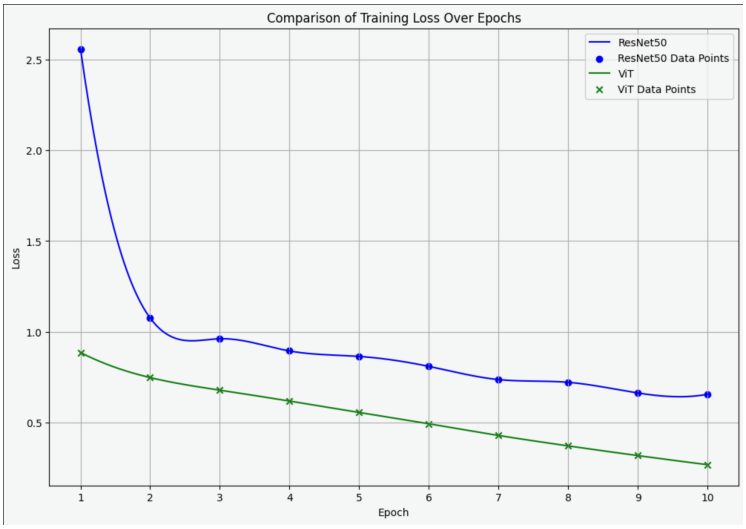

Figure 2: Comparison of training loss over epochs for ResNet50 and ViT.

4 Conclusion

The currently trained ViT model yields an average R^2 value of 0.26869. This regression problem helped explore the specific use cases and tradeoffs of CNNs and Vision Transformers. In addition, it allowed for exploration of different optimization techniques that could be applied during pre-processing and training on the training and test sets. Both the CNN and transformer models were experimented with different epochs of sizes 5, 10, 15, and 20. Due to the computational limitation of the training environment, the model did not fully train at 20 epochs. However, it was seen that training at 10 epochs was optimal (R^2 value of 0.19855 for ResNet50) since training with 15 epochs gave a lower R^2 value (0.17944 for ResNet50) possibly due to overfitting of the model on the training set. A major limitation that was faced during this experiment was the limited resources related to the training environment such as limited RAM and storage. The Google Colab environment only supported 12 GB RAM which caused memory constraints due to Macbook limitations, not allowing this experiment to explore higher batch sizes, model depth, and the number of experiments that could run concurrently. When attempting to train on better pre-trained transformer models, the RAM consistently ran out causing the program to crash. Other limitations include target traits being imbalanced where some traits are more common than others. This could result in the model being biased towards more frequent traits, leading to poor performance on underrepresented traits. Using ViT, RandomForest and XGBoost could result in overfitting of the training data due to their complexity. Although train-validation splits were used in the experiment, using k-fold cross-validation for the ViT model on the entire training set would possibly yield more accurate predictions. Finally, if the dataset contained biases towards the ancillary features such as locations, seasons or plant species, the model could be biased towards those conditions, causing the model to make inaccurate predictions on the test set.

For the future, it is worth exploring better pre-trained models for both CNNs and transformers to see whether it changes the accuracy of predictions for the 6 plant traits. For CNNs, models that should be experimented with given the needed computational power and memory support are Xception, EfficientNet-B7 and VGG16 since they perform better than ResNet50 (Mohanty, 2020). In addition, other transformer models such CLIP and DeiT could be experimented on to see if the prediction accuracy is better.

To summarize, the key findings of this experiment showcased that Vision Transformer (ViT) yields a higher accuracy in plant trait predictions compared to using Convolution Neural Networks (ResNet50). While there are trade-offs to using both models such as a high computational capacity required by transformer models, in terms of accuracy, ViT outperforms ResNet50.

Acknowledgement

Thank you to Saber Malekmohammadi for providing constructive feedback on my model and suggesting optimization techniques to mitigate crashes and training issues related limited RAM.

References

AI, V. (2024). "Vision Transformer (ViT)". Accessed: 2024-08-13.

Chatterjee, C. C. (2019). "Basics of the classic CNN".

"CLIP" (n.d.).

Developers, S.-l. (2024). "RandomForestRegressor". Accessed: 2024-08-13.

Face, H. (2024). "google/vit-base-patch16-224". Accessed: 2024-08-13.

Kamal, K. and E.-Z. Hamid (2023). "A comparison between the VGG16, VGG19 and ResNet50 architecture frameworks for classification of normal and CLAHE processed medical images".

Mohanty, A. (2020). "EfficientNet: The State of the art in ImageNet". Published on Analytics Vidhya.

Mukherjee, S. (2022). "The annotated ResNet-50".

"Pre-trained model definition" (n.d.).

Rustamy, F. (2024). "Vision Transformers vs. Convolutional Neural Networks".

Schiller, C., S. Schmidtlein, C. Boonman, et al. (2021). "Deep learning and citizen science enable automated plant trait predictions from photographs". *Scientific Reports*, vol. 11, p. 15334.

Sheerazi, A. (2024). "PLANTTRAITS2024-A Kaggle competition".

"Vanishing gradient problem: Everything you need to know" (n.d.).

"What is a convolutional neural network?: 3 things you need to know" (2024).