Plant Trait Classification: Integrating Image and Tabular Data Processing

Andres Espinosa
Industrial and Systems Engineering
University of Florida
andresespinosa@ufl.edu

Kaylie Coatney
Industrial and Systems Engineering
University of Florida
kayliecoatney@ufl.edu

Lexi Bobb
Industrial and Systems Engineering
University of Florida
lexibobb@ufl.edu

***Abstract*—This paper presents a comprehensive approach to predicting plant traits by integrating image and tabular data, leveraging the rich dataset from the PlantTraits2024 Kaggle competition. Our methodology combines deep learning models for image processing with machine learning techniques for tabular data to forecast six essential plant traits. These traits include stem specific density, leaf area per dry mass, plant height, seed dry mass, leaf nitrogen content, and total leaf area, which are crucial for understanding plant adaptation to their environment and broader ecological dynamics. By utilizing a mixed data model, we aim to harness the predictive power of visual cues from images along with contextual environmental data provided in tabular format. The challenge is compounded by the diverse and non-standardized data collection methods inherent in crowd-sourced datasets such as those from iNaturalist and the TRY database, which introduces significant variability. Our results highlight the potential of multimodal learning frameworks in ecological modeling and underscore the need for robust data preprocessing to mitigate issues stemming from data heterogeneity. The insights gained from this study provide a thorough attempt at plant trait prediction, but question the effectiveness of integrating disparate data sources.**

I. Introduction

The competition our group tackled was the PlantTraits2024 Kaggle competition [2]. The goal of the competition is to use a combination of plant image data and world climate data to predict six traits of the plant:

- X4: Stem Specific Density or Wood Density
- X11: Leaf Area per Leaf Dry Mass (Specific Leaf Area)
- X18: Plant Height
- X26: Seed Dry Mass
- X50: Leaf Nitrogen per Leaf Area
- X3112: Leaf area

An important piece of information to note about this competition is that around late March, an issue was detected in which the sample submission could have been used to gain an artificial boost in test error. As a result, comparisons with older submissions are skewed negatively. For example, the baseline performance of the competition provided Kaggle Notebook, prior to the test data change was -3.16509. Running this notebook after the test data change results in a significantly worse performance of around -7.94468. Therefore, we will be using this performance as the baseline for the comparison to our improvements.

A. Dataset Background

Plant traits are an important step in understanding how plants reacting to their environment, and in turn how plants are adapting to growing change in the biosphere. The goal of this competition is to use crowd-sourced plant images and some ancillary data to predict 6 different plant traits which could lead to a better understanding of the global patterns of biodiversity [2]. This competition provided two sources of data to use for training our model. The first source was a collection of images gathered from the iNaturalist database, which includes citizen science plant photographs. iNaturalist is a species identification app that uses AI algorithms and includes its prediction, the photograph, and the geolocation in the database [2]. From the geolocations, six plant trait labels were matched from a secondary data source [2]: World climate data gathered from the TRY database. This database contains species-specific standard deviation and mean of the plant traits [2].

1) Plant Images: The data that was obtained from the iNaturalist collection includes a collection of images of plants, their geolocations and the associated species name that was predicted. The images that are in the dataset could be used to analyze sizes, edges, and shapes of features of the plants, which could help the model predict the 6 traits.

2) Tabular Data: The tabular data contains a combination of the ancillary predictors based on the geolocations from the iNaturalist database and matched to the TRY database. The geolocations were used to find globally available raster data (WORLDCLIM, SOIL, VOD, MODIS) of climate data and specific information that could prove beneficial to the model [2]. The WORLDCLIM data is mainly focused around the temperatures and precipitation levels at the geolocation provided from the combination of the TRY database and the geolocated iNaturalist database [2]. The SOIL columns provide information regarding the various soil properties at the geolocation reference, such as sand content and pH[2]. The VOD columns contains information about the water content and biomass of the plants [2]. Finally, the MODIS columns use data pulled from satellites measuring optical reflectance of sun light [2]. A heatmap of a few features and all six labels can be seen in Figure 1. Unfortunately to the task at hand, the lack of correlations present in this figure persist throughout

the majority of the feature/label combinations.

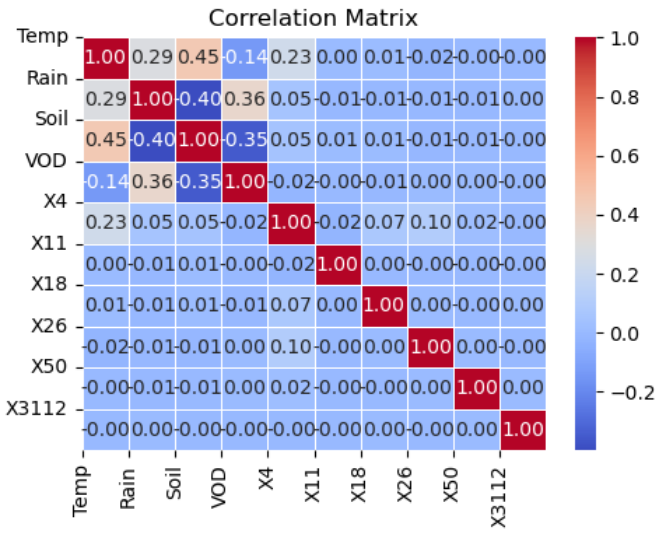

Fig. 1. Correlation matrix of the six labels and four important features

B. Code Organizational Structure and Breakdown

The Kaggle notebook that we chose to use as a starting point for our project was the PlantTraits2024: Keras CV Starter Notebook [2]. This notebook uses the EfficientNetV2 backbone from Keras CV to create a multi-input and multi-output Deep Learning model. We chose this notebook for several important reasons. This notebook had a few excellent features that we could use to launch our efforts into the project. One benefit to this notebook was its use of both the tabular data and the plant image data in order to train its model. This would help point us in the right direction for dealing with both types of data. Additionally, while it wasn't the highest scoring of the example models, it was very well commented and was easier for us to comprehend, which was essential in being able to improve upon it. It was also highly supported by Kaggle members showing that it made for a good starter notebook to build upon. The dataset starts with importing and installing the necessary libraries. Then, it configures the notebook by setting values for the random seed, verbosity, number of classes, number of folds, among others. This is a very important step for the effectiveness of the model later on but also left some room for changes in order to improve upon the model. After configuration, the notebook loads the training and test data in order to start building the model. The starter notebook then starts the next step, which is called "DataLoader". In this step, both the tabular data and images are loaded in as inputs and several augmentations are applies to the data such as flip, rotation and brightness. These augmentations allow the model to have more images to work with, creating more training data for the model to learn on, increasing its accuracy. The starter notebook then takes that data and splits it into 5 folds. Then, it builds the training and validation datasets to be used in the model and looks at samples and their associated labels. The notebook then creates a custom implementation of a coefficient of determination evaluation metric for this competition.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2} \quad (1)$$

This formula quantifies the amount of variation in the outcome that can be explained by the independent variables in the model. The notebook then creates a custom model that combines a pretrained EfficientNetV2 [6] model with a feedforward model to predict two tasks: The mean of the six plant traits and the standard deviation. The model then outputs the results of the main task and evaluates it using equation 1.

C. Peer Submission Summary

For this study, our peers utilized multiple other methods to improve the modeling score to better predict plant traits. One of the first methods present in other kaggle submissions is using different architectures such as ResNetV2 and other versions of ResNet to train image and tabular data. The original starter notebook provided by the competition uses EfficientNetV2 architecture. EfficientNet is a newer family of convolutional network architectures that contain faster training speed and better parameter efficiency than older models, while ResNet is an extremely deep architecture that shows compelling accuracy and nice convergence behaviors [6]. Other competitors found success in building similar but custom architectures with PyTorch in place of tensorflow and received a score of approximately -1.63967 [9]. One student even found significant improvement when utilizing an easy deep learning model specific to the tabular data only and received a score of 0.14953 [10]. This method involved filtering training values to ensure that the values used for training were higher than the lower quartile of 0.0005 of the data and less than 0.985 to get the X4 mean column. The computation gain from excluding complicated CNN architecture proved a significant gain in performance as the model could be trained for longer. One peer took two different approaches, both beginning with similar paths but using different techniques towards the end of the modeling process in order to compare and contrast differences in the modeling techniques. Both processes resulted in a score of approximately 0.23841 [14] and 0.22161 [11]. The Multi-Regression notebook augments EfficientNetB0 with additional layers to incorporate the tabular data with the general goal to use a Convolutional Neural Network (CNN) on this project. He utilizes a multi-target regression approach versus a multiple models approach and then normalizes the tabular input data [11]. His other approach, the XGBoost approach starts out with using EfficientNetB3 to load the images to ImageNet, then use EfficientNet to extract image features into 1280 columns of tabular data. One final project to mention attained a score of 0.38494 and was publicly available which only used images to create the model [13]. The steps that the supplementary notebook took involved V1 demonstrating the training process, while V2 utilizes precomputed DataFrames and pretrained models for inference. V3 involves additional plotting and data filtering based on sample submission minimum and

maximum values, whereas V5 excludes samples beyond the 0.1 to 99.9 range of the training samples. An article published by professors at the University of Toronto and previously referenced in this section was very beneficial in understanding the use of machine learning methods, particularly CNNs, for object recognition tasks. This paper detailed the importance of reducing overfitting through proper data augmentation and dropout [8]. Overall, there are multiple approaches to creating a model for this project, the overarching question involves what we chose for our implementation, which combines multiple techniques and approaches in order to produce great and trustworthy results and predictions.

II. Implementation

The multimodal nature of this competition would require a complex neural network architecture capable of reasoning through important patterns that interact between the tabular data and image data. Unfortunately, the results of the models explored during this competition proved less than effective at classifying the traits of each plant. Therefore this implementation section will dive into the theory behind our multiple attempts at iteratively improving upon the baseline performance of the model. The results section will outline the potential reasons behind their unexpected lack of performance.

A. Data Processing Improvements

Due to the heterogeneity among the photographed plants, including flowers, trees, bushes, and moss, the six plant traits targeted in the competition exhibit diverse distributions. As a result, data processing must be handled carefully in order to accurately preserve and enhance information for predictions.

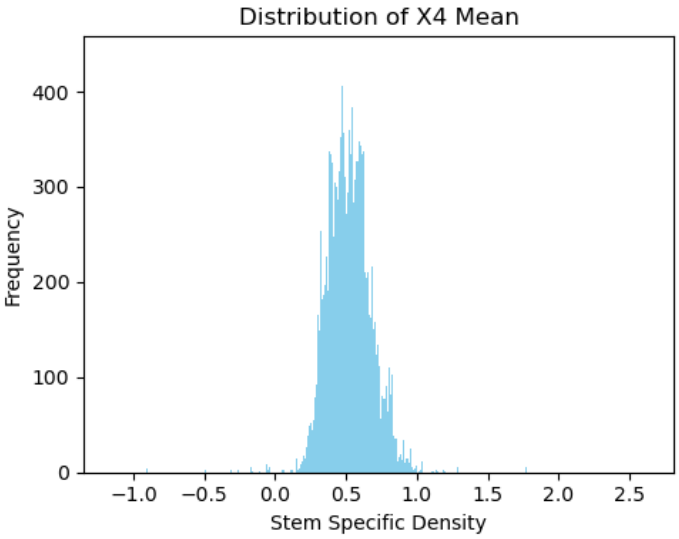

Fig. 2. Plot of X4: Normally distributed

For the majority of attempts, the data processing utilized was a standard scaling fitted on the train set and applied to the train and validation set. For a few attempts, an alternative data processing approach was used through normalizing the

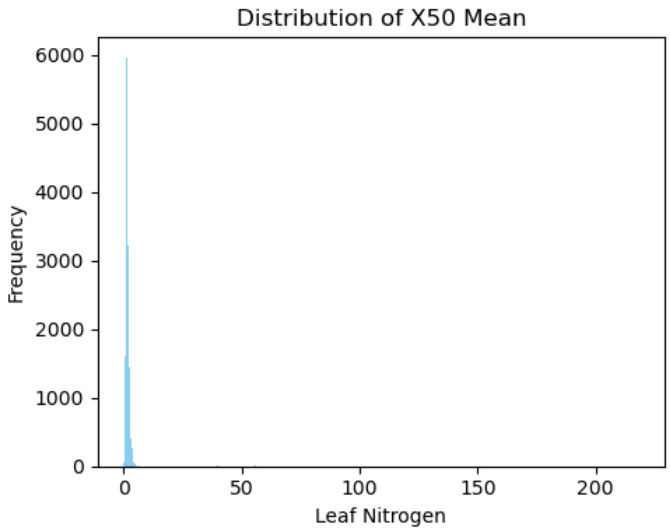

Fig. 3. Plot of X50: Demonstrates heavy right skew of data

X4 column and through applying a logarithmic transformation for the remaining 5 before normalizing. Through some early visualizations such as those in Figures 2 and 3, it is evident that there is a significantly different distribution between X4, X50 and the rest of the label columns.

B. Image Classification Model

Narrowing the focus of implementation on classifying the images, there are a few approaches to best utilizing the images to classify the plant traits. An approach that was used was using an ImageNet pre-trained model, which could be further split into deciding whether or not to freeze layers of the pre-trained model. Freezing layers of the pre-trained model can significantly reduce the computation time while still maintaining important information from the task. However, freezing layers forces the model to utilize the pre-trained task of classifying images into one of 1K ImageNet classes [8].

In order to take best advantage of this computation-specificity tradeoff, another approach was used that combined all pre-trained models for an ensemble model aimed at taking advantage of the differences between them. The models used include:

- EfficientNetV2 (B2, S, and M) [6]
- MobileNetV3 (Small and Large) [15]
- Resnet50 [16]
- YoloV8 [17]
- cspDarkNet [18]

One of the architectures that took advantage of this ensemble model is visualized in Figure 4

C. Tabular Data Processing

CNN architectures are models that benefit from ability to slide kernels over 2D images. However, for the 152 ansillary variables present in the dataset, other model types would likely perform better for the task. XGBoost [19] is a popular model

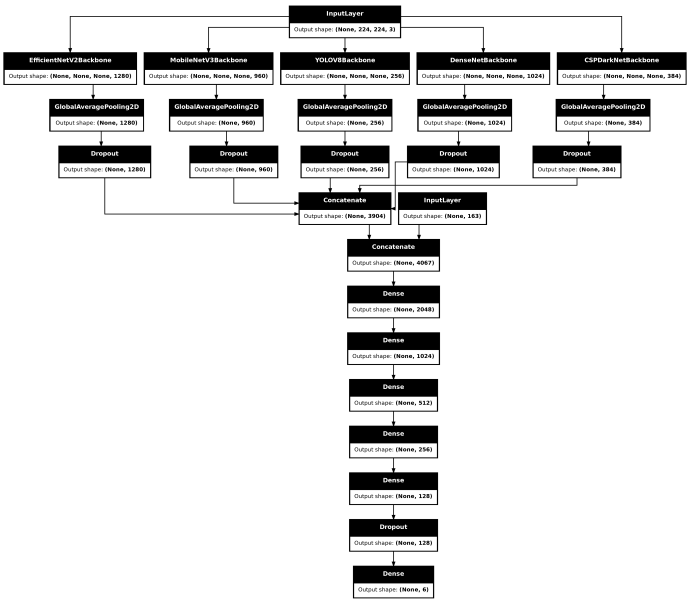

Fig. 4. Model visualization of five CNN pre-trained models ending in a six layer FNN

that is especially effective at drawing complex relationships in tabular data. Due to the incredibly low correlation between features and labels for this dataset, XGBoost may perform well in capturing these relationships.

Another architecture that was used for the tabular data was a feedforward neural network, the "standard" type of neural network used to model non-linear data. As depicted in the previous figure 4, a dense and deep neural network was used with six layers of nodes, starting with 2048 and decreasing by half at each step.

A simpler architecture that demonstrated relatively good results was a multiple linear regression. After some preprocessing of the data and image feature extraction, multiple linear regression was performed on the dataset to attempt to model the differing correlations between features and labels.

D. General Model Architecture

The tabular data processing and image classification models were combined in a few different ways to attempt to gather an encompassing view on their performances.

One such combination involved using the image classification pre-trained models to extract important features from the images, which were then fed into the tabular data processing model to expand on the 152 provided features. The idea behind this approach was to extract general information from the model such as the perceived size of the plant, the general area the plant resided in, the visible quality of soil.

Another combination of image and tabular data involved keeping the two models separate but then incorporating them both into one concatenated layer. This involved processing the image through a pretrained model and the tabular data through a feedforward neural network, which was then fed into one output layer that would predict the six labels.

Although this model did not end up getting tested, another approach that was considered involved training the models completely separately, and then using a weighted-average of the results to attempt to skew the final results closer to the output that each model generated well. The reasoning behind this comes from the lack of correlation between the labels. An example discovered from the conducted exploratory data analysis, the stem specific density correctly reveals very little information about the leaf area and yet the image model will likely reveal more information about the leaf area than the tabular data. This could lead to the models becoming specialized toward each label and therefore providing a corresponding weight to each prediction could benefit the final performance.

III. Results

As mentioned previously, despite extensive efforts taken in the theoretical and strategic aspect of the competition, the empirical results delivered from the models performed much worse than expected. The majority of models returned a coefficient of determination lower than desired. For brevity, this section will include the performance increase or decrease in parentheses from the baseline (-7.94468).

A. Improvements

As discussed in the implementation section, there were many different types of architectures and models that were used in the testing of the competition. The first model that performed better was switching out the baseline notebook's pre-trained image model to a more powerful CNN architecture (+0.1388) or to freeze the layers of the pretrained model(+1.1893). Despite the intuitive thought that allowing the model to train would improve performance, what likely occurs is the stability of a fully trained model gives better results than a model that is having its weights become unstable during early training. A model was also ran without any training at all, keeping only the pre-trained weights and the randomized weights of the feedforward (+7.4615). The surprise performance of a completely randomized model led our group to doubt the validity of our previous models and to some extent, the task itself.

The best performing model, a simple multiple linear regression with a simple MobileNetV2 image preprocessing step achieved a total test score of 0.22378, placing our team 79th out of 237 competitors. To compare to figures 2 and 3, the distribution of the predictions from the best model are presented in figures 5 and 6. The success of arguably the simplest model our team decided to use was very surprising and something that is difficult to reason about. However, the fact that the highest score achieved is 0.22378, a score that seemed to be rather consistent with the highest correlations found between features and labels, meant that the simplicity of the linear model allowed it to capture the linear correlations extremely well, but nothing else.

B. Data Collection Limitations

The success of the simplest model led our team to inquire about what could have lead to the difficulty of complex models

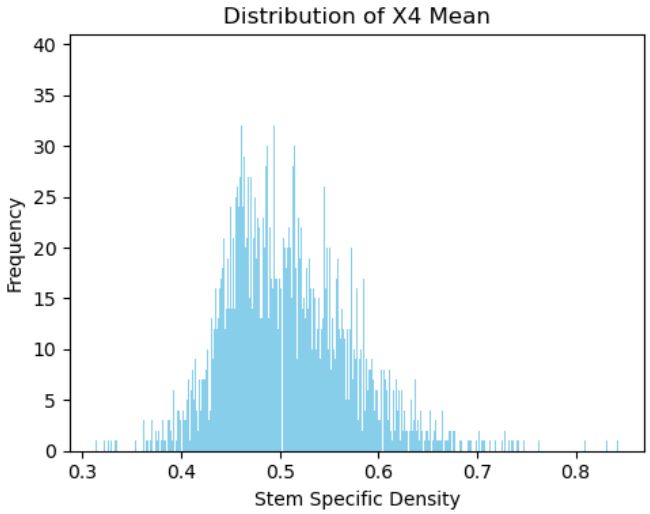

Fig. 5. Stem Specific Density distribution of the predictions of the best performing model

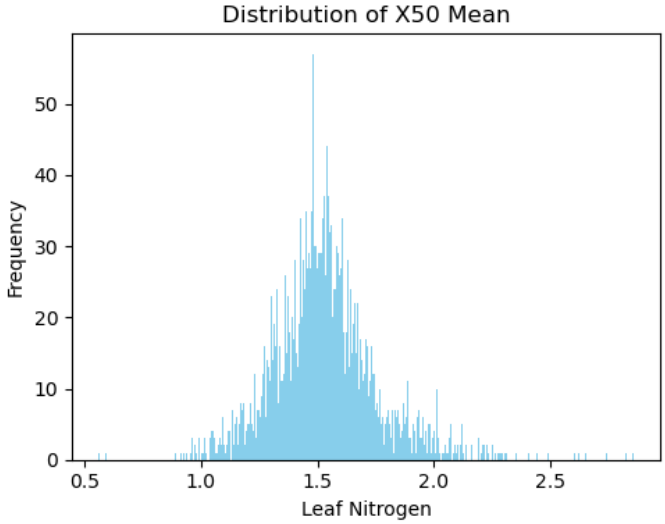

Fig. 6. Leaf Nitrogen distribution of the predictions of the best performing model

to capture any meaningful information about the data. A rather unique aspect of the competition involves the the data collection methodology employed by the competition. The images for this task are taken through a citizen science plant observation app that attempts to use artificial intelligence to classify the plant and to record some information about the ecology at the time the photo was taken. This data was then used to match against a different database in which the plants were matched to their traits from other algorithms. Therefore, this poses an inherent difficulty for any algorithm to not only model the underlying distribution of plant traits, but also to model the algorithms that were used to create the underlying distributions, which have been constantly changing throughout the duration of the databases.

Although it likely is possible for plant traits to be classified

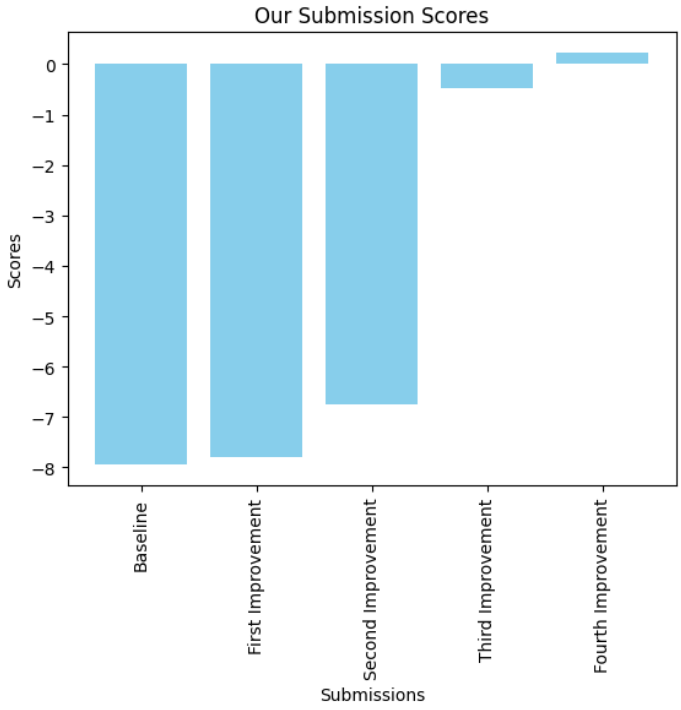

Fig. 7. Depiction of notable improvements on the baseline score

correctly with better algorithms and higher compute, the biggest performance gain would likely come from improvements made in the data collection methodology.

IV. Conclusion and Future Work

The PlantTraits2024 competition presented a unique challenge of integrating diverse data types—namely, high-dimensional image data and complex tabular data—to predict plant traits that are critical for understanding biodiversity and ecological adaptations. Our approach leveraged a combination of deep learning techniques and traditional machine learning methods to tackle this problem, revealing the nuanced interplay between different data modalities.

Throughout the project, we explored various model architectures and data processing techniques to improve prediction accuracy. Our efforts included employing pre-trained convolutional neural networks (CNNs) for image feature extraction and integrating these features with processed tabular data using both ensemble methods and multi-input neural network architectures. Notably, the simpler models, such as multiple linear regression combined with basic image preprocessing, often performed unexpectedly well, suggesting that sometimes simpler models are more robust, especially when data relationships are not exceedingly complex or when the dataset contains inherent noise and weak signal-to-label correlations.

However, the models' performances also highlighted significant challenges, including the difficulty of extracting meaningful predictions from highly heterogeneous and sparsely labeled data. This was compounded by the issues in data collection methodologies and the intrinsic variability of biological data.

The competition's results prompt a reconsideration of how data collection and preprocessing impact machine learning in ecological contexts, where data can be exceptionally varied and influenced by numerous uncontrolled factors.

For future work, there is substantial room for improvement in both the methodological approaches to multimodal data integration and the underlying data collection processes:

- **Enhanced Data Collection:** More rigorous data collection methods, perhaps guided by clearer ecological hypotheses, could reduce variability and improve model training. Integrating domain-specific knowledge more deeply into the feature engineering and model design phases could also yield benefits.
- **Advanced Modeling Techniques:** Exploring newer or less conventional machine learning models that are specifically tailored for ecological data may provide breakthroughs. Techniques such as transfer learning, semi-supervised learning, and generative models could potentially address some of the challenges related to sparse and imbalanced data.
- **Interdisciplinary Collaboration:** Closer collaboration between data scientists, botanists, and ecologists could lead to better-defined problem statements and more targeted data collection, improving the relevance and accuracy of predictive models.
- **Broader Dataset:** Expanding the dataset to include more diverse ecological zones and plant types could help in developing more robust models that generalize better across different environments.

In conclusion, while the competition provided valuable insights into the complexities of plant trait prediction using machine learning, it also highlighted the limitations of current methodologies and the importance of thoughtful data collection and preprocessing. The future of plant trait classification lies in the convergence of ecological science and advanced data analytics, where each domain enriches the other, leading to more accurate and ecologically meaningful predictions.

REFERENCES

[1] A. Sheerazi, "PLANTTRAITS2024 – A Kaggle competition," Medium, [Online]. Available: https://medium.com/@adil.she/planttraits2024-a-kaggle-competition-8c6731003356

[2] Awsaf, AyushiSharma, HCL-Jevster, inversion, Martin Görner, Teja Kattenborn. (2024). PlantTraits2024 - FGVC11. Kaggle. https://kaggle.com/competitions/planttraits2024

[3] Wolf, S., Mahecha, M.D., Sabatini, F.M. et al. Citizen science plant observations encode global trait patterns. Nat Ecol Evol 6, 1850–1859 (2022). https://doi.org/10.1038/s41559-022-01904-x

[4] Moles, A.T., Xirocostas, Z.A. Statistical power from the people. Nat Ecol Evol 6, 1802–1803 (2022). https://doi.org/10.1038/s41559-022-01902-z

[5] Schiller, C., Schmidtlein, S., Boonman, C. et al. Deep learning and citizen science enable automated plant trait predictions from photographs. Sci Rep 11, 16395 (2021). https://doi.org/10.1038/s41598-021-95616-0

[6] Tan, M., Le, Q. V. (2021, June 23). EFFICIENTNETV2: Smaller models and faster training. arXiv.org. https://arxiv.org/abs/2104.00298

[7] He, Kaiming, et al. "Identity mappings in deep residual networks." Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands. https://arxiv.org/abs/1603.05027

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Communications of the ACM, vol. 60, no. 6, pp. 84–90, May 2012, doi: https://doi.org/10.1145/3065386.

[9] rui314, "Kaggle Notebook Tabular Data Deep Learning" Kaggle. https://www.kaggle.com/code/rui314/planttraits-resnet18

[10] Michalinahulak, "Tabular Data Deep Learning," Kaggle. https://www.kaggle.com/code/michalinahulak/tabular-data-deep-learning/notebook

[11] Richolson, "Tabular ImageNet Multi-Target Regression," Kaggle. https://www.kaggle.com/code/richolson/tabular-imagenet-multi-target-regression/notebook

[12] Mark Wijkhuizen, "PlantTraits2024 EDA and Training Pub," Kaggle. https://www.kaggle.com/code/markwijkhuizen/planttraits2024-eda-training-pub

[13] Hdjojo, "Modified PlantTraits2024 EDA and Training," Kaggle. https://www.kaggle.com/code/hdjojo/modified-planttraits2024-eda-training/notebook

[14] Richolson, "ImageNet Tabular XGBoost," Kaggle. https://www.kaggle.com/code/richolson/imagenet-tabular-xgboost/notebook

[15] A. Howard et al., "Searching for MobileNetV3," arXiv.org, 2019. https://arxiv.org/abs/1905.02244

[16] [3]K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," 2016. Available: https://openaccess.thecvf.com/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf

[17] D. Reis, J. Kupec, J. Hong, and A. Daoudi, "Real-Time Flying Object Detection with YOLOv8." Accessed: Apr. 29, 2024. [Online]. Available: https://arxiv.org/pdf/2305.09972

[18] C.-Y. Wang, H.-Y. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I-Hau. Yeh, "CSPNet: A New Backbone that can Enhance Learning Capability of CNN." Available: https://openaccess.thecvf.com/content_CVPRW_2020/papers/w28/Wang_CSPNet_A_New_Backbone_That_Can_Enhance_Learning_Capability_of_CVPRW_2020_paper.pdf

[19] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System." Accessed: Apr. 29, 2024. [Online]. Available: https://arxiv.org/pdf/1603.02754