A CatBoost regression model to predict plant traits based on the image and numerical metadata

Ding Li
Computational Mathematics
University of Waterloo
Waterloo, ON, N2L 3G1
d376li@uwaterloo.ca
report due: August 12

Abstract

The project proposes a tree-based ensemble model to regress the plant's 6 important traits based on its image and corresponding tabular numerical metadata. The model uses a pretrained Dino ViT model to extract features from the plant's image, and combine the first 1000 degree-2 polynomial features of the tabular metadata. The model is trained and tuned using a 10-fold cross-validation. The optimal model setup is selected to train on the whole dataset for the best generalization on the test set. The model has an R-squared of around 47% as its performance score on our kaggle public test dataset, which is above the 43% baseline. The model is much cheaper to train than other pure deep learning methods, even for the fine-tuned ones.

1 Introduction

This project is to train an ensemble machine learning model for a 6-target plant trait regression task based on 163 numeric features and a 128 by 128 RGB image of the plant. Due to my lack of computing resources, i.e., a GTX 1070, my motivation is to make the model as cheap as possible in terms of training and predicting, while keeping the model's performance above the baseline. Thanks to the diverse pretrained models, we can easily encode our images into a vector representation containing its geometric and structural features without having to train the feature extractors from the start. Nonetheless, just fine tuning is still beyond my allowance, especially when reconstructing models and running-cross validation will re-train the whole model a lot of times and the abandoned models might not help my prediction.

Thus, I choose to build a series of CatBoost regressors for each of the trait targets based on its Dino ViT encoded image feature and the 163 numeric features. The model overview for one iteration of training is demonstrated in Figure 1. The numeric features are made into 1000 2-degree polynomial features, in order from its degree-2 polynomial standard basis. Each image is rescaled to 224 by 224 to fit the ViT embedding layer. The targets are clipped to only preserve quantile 0.1% and 99% to avoid the impacts of the outliers. A 10-fold cross validation is used for hyperparameter tuning and error estimations. The model performance metric, R-squared, is estimated by averaging the R-squared over different targets on the validation sets. The optimal model setup is finally selected to train on the whole given dataset, making sure that the model doesn't miss any species due to set splitting, thus optimizes its potential performance on the competition test set.

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

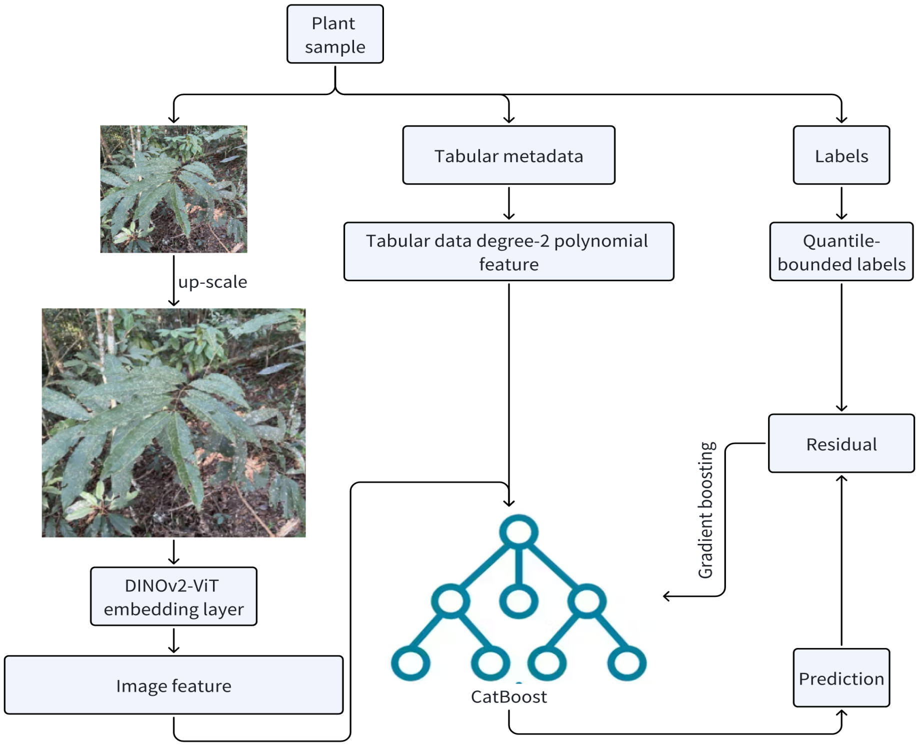

Figure 1: Model overview. We clip the lowest 0.1% and highest 1% quantiles of the label. The plant image is resized to 224 × 224 to fit the DINOv2 ViT embedding layer. The tabular metadata is taken the first 1000 of its degree-2 polynomial standard basis. The image and polynomial features are concatenated to feed into an assembly of tree models in our CatBoost model for label prediction. The prediction residuals with the quantile-bounded labels yield gradient boosting to construct and optimize the next tree model.

2 Related Works

The competition was first proposed by a research group in ecosystems in 2021, who used the ensemble of 3 simple CNN models (Lins et al. 2021). Despite their insufficient model powers, they indeed show some useful practices: 1) log-transform and normalize the labels to smooth outliers and eliminate the huge dimensionality difference between labels, and 2) assemble more models that are good at different things to boost the prediction. Those practices have been the verified consensus for most of the following works on the task, and will be taken in this project. Then, we can see some SOTA models for the task in the PlantTraits2024 - FGVC11 Kaggle competition (Awsaf et al. 2024b). The rank-one solution is called "PlantHydra", which proves that the embedding from DINOv2-ViT is powerful to extract plant image features, which outperforms from the original CNN and even our set baseline (i.e., 43%) (Awsaf et al. 2024a). The optimal pretrained Dino ViT model open to us should be the pretrained models on the flora of the south western Europe based on a subset of Pl@ntNet collaborative images and a ViT base patch 14 dinoV2 (Goëau et al. 2024). However, I verify that the encoder part is too large and takes around 100 minutes to run one epoch, which is too time-consuming, so I use "dinov2_vitg14_reg" from Facebook as an alternative ((FAIR) 2023). The rest of the model paradigm using cheaper ensemble methods comes from 2 other solutions in this competition (hdjojo 2024, Korotas 2024). The solutions show the optimality of CatBoost out of other ensemble paradigms, and proposed extracting effective features using 1000 degree-2 polynomial of the tabular data through their EDA and experiments. They also suggest quantile-cutting of labels for removing outliers, which outperforms 10-log transform plus standardization in tree models through my experimental validation. Other hyperparameters they use are borrowed as my model's initial hyperparameters.

Table 1: Model validation R-squared under different configurations. **Bolded: Tuned and highlights**.

Description	Validation R-squared	N fold	Minimum label quantile	Maximum label quantile	First N polynomial features	Iterations
Baseline	42.5%	5	0.001	0.98	1000	1500
Add N-fold	**43.7%**	**10**	0.001	0.98	1000	1500
Tune the number -of polynomial features	43.1%/42.3%	10	0.001	0.98	**1500/500**	1500
Tune the clipped -label quantiles	**46.3%/46.5%**	10	**0.005/0.0005**	**0.99**	1000	1500
Optimal	**46.9%**	10	0.001	0.99	1000	1500

3 Main Results

At the very early stage of my experiment, I tried a modified VGG11 model plus 3 extra convolution layers, which yields a 15% R-squared. Then, by replacing the image feature extractor from self-trained convolution layers to other pretrained encoders, the R-squared goes to around 29% with the CLIP encoder as the optimal one. By assembling all the predictions from the fair models I had with an R-squared above 25% using a trained GAM under 10-folder cross-validation, the R-squared goes to around 30% on the public test set. Each model converges in around 6 epoches, which takes about one hour on average on my GPU. The early-stage experiments show that 1) pretrained encoders outperform much from the self-trained feature extractors, 2) adjusting and fine-tuning models with pretrained encoder layers can be too time-consuming for my available GPU resources, and 3) assembling weaker (but cheaper) models can be achievable with my GPU resources and acceptable in R-squared.

Using the finalized CatBoost model setup, the model yields above-40% validation R-squared accuracy under different hyperparameters. The model configuration and R-squared is shown as the first row of Table 1, which is used as our new baseline. The validation R-squared under different model hyperparameters is shown in Table 1. The most important hyperparameters are verified to be the maximum label quantile in the outlier clipping and the number of polynomial features of the tabular data. Changing the number of polynomial features by 500 drops down the validation R-squared by about 1%. Adding the maximum label quantile from 0.98 to 0.99 significantly increases the validation R-squared from 43% to around 47% , which goes above the kaggle baseline. Further increasing the label will keep dropping the validation R-squared, until going below 30% when the number goes to 1, i.e., no right-tail clipping. The phenomenon is because there is a balance trade-off between long-tail species accuracy against outlier downsides. According to our exploratory data analysis, clipping a too long right tail will worsen the model's performance on the species with a long-tail target. Unfortunately, the number of long-tail species in the dataset is unignorable. On the other hand, if we preserve the whole right tail, then the model is impacted by the outliers. By running several trials, the optimal hyperparameter is decided at the end of Tab.1. The maximum label quantile is 0.99, and we keep the first 1000 polynomial features. Other hyperparameters are not considered very important to the model accuracy, so I keep them as the initial values.

After finalizing the optimal model structure and hyperparameter, I re-train the whole model 3 times under different random seeds on the whole given dataset without train-validation splitting. The final predictions for the kaggle test set are averaged over the 3 models. The reason why I did this is because most species only show up only once or twice in the dataset, which means that a huge portion of species won't be seen by the model if we use a split training set. We will sacrifice model score, especially when the missed species deviates a lot from other seen species, which expands the impact from the distribution gap between the training set and the test set distribution. Thus, to eliminate the risk as we can, we should leverage all the available data. The final model indeed gives the best performance so far, with 47.3% R-squared on the public test set. The learning rate is kept as 0.6, the number of iterations is set to 1500, and the random seed used during the hyperparameter tuning stage is 42. The finally submitted model averages outputs from 3 optimally-configured models trained on the whole dataset under 3 random seeds: 42, 43, and 44.

Algorithm 1: Pseudo-code for model training

```
Input: Dataset $D = \{(I_k, x_k, y_k) \mid k = 1, 2, \ldots, N\}$
Output: $f_1, f_2, f_3, f_4, f_5, f_6$
for $k = 1, 2, \ldots, N$ do
    $I_k \leftarrow \texttt{resize}(I_k, 224)$;
    $Z_k \leftarrow \texttt{dino\_ViT}(I_k)$;
    $P_k \leftarrow \texttt{Poly}(x_k, \texttt{deg} = 2)[: 1001]$;
    $F_k \leftarrow \texttt{Concatenate}(Z_k, P_k)$;
    $y_k \leftarrow \texttt{quantileBound}(y_k, \texttt{lower} = 0.001, \texttt{upper} = 0.99)$;
$M_i = \{(F_k, y_{ki}) \mid k = 1, 2, \ldots, N\}$;
for $i \leftarrow 1$ to $6$ do
    $f_i \leftarrow \texttt{CatBoostTraining}(M_i, \texttt{iter = 1500})$;
```

4 Conclusion

In conclusion, the pretrained image encoders indeed outperform the self-trained feature extractors with limited time and training data. Nonetheless, fine-tuning models with the pretrained image encoders is still time consuming with a GTX 1070 GPU. Ensemble models that combine multiple simple weak models can hugely reduce training and predicting time, while preserving above-baseline R-squared. A CatBoost model trained based on Dino-ViT embedding layer encoded image features and 1000 degree-2 polynomial features of the 163 tabular metadata can reach an R-squared of about 47% on the kaggle test set. Bounding the target labels from quantile 0.1% to 99% effectively reduces outliers while keeping enough right-tailed species, which is the most important hyperparameter in this model. The sparse species distribution in the dataset expands the impact of the distribution gap between the training set and the test set. By training on the whole given dataset with 3 different seeds then predicting the test set with the averaged values, we avoid the impacts of the unseen anomaly species contributing significantly to the prediction score to our best efforts.

In the future, the model should use another Dino-ViT encoder pretrained using the PlantCLEF2024 dataset to better extract features from plant images specifically. The encoder takes a much longer time to extract features from an image and yields a much longer feature vector, but can capture finer plant traits. In addition, a better tabular data feature representation might be found. For example, training a self-attention header to capture the correlations between the tabular metadata features is adapted by the first-rank solution. The solution also indicates the benefit of finding a more appropriate loss function, which can help reduce the impacts of the long-tail species and the sparcity of species in the dataset.

Acknowledgement

Thank one of my friends who is the PhD student at Peiking University for discussing some ideas with me. The referenced public solutions have all been cited in-text and shown in the reference section.

References

(FAIR), F. A. R. (2023). “DINOv2: Learning Visual Features with the Knowledge of Visual Features”. GitHub repository.

Awsaf, A. Sharma, HCL-Jevster, inversion, M. Görner, and T. Kattenborn (2024a). “Discussion on PlantTraits2024 - FGVC11”. Kaggle discussion post.

– (2024b). “PlantTraits2024 - FGVC11”.

Goëau, H., J.-C. Lombardo, A. Affouard, V. Espitalier, P. Bonnet, and A. Joly (Mar. 2024). “PlantCLEF 2024 pretrained models on the flora of the south western Europe based on a subset of Pl@ntNet collaborative images and a ViT base patch 14 dinoV2”.

hdjojo, K. (2024). “Modified PlantTraits2024 EDA and Training Notebook”. Kaggle notebook.

Korotas, K. (2024). “9th Place Solution for PlantTraits2024: DINOv2 + CatBoost”. Kaggle notebook.

Lins, D., B. Meyer, R. Karutz, T. Keil, M. Braun, G. Sudeck, and H. Vogel (2021). “How does medical specialist decision-making differ under stress?” *Scientific Reports*, vol. 11, no. 1, p. 16006.