ViTab: Addressing Skew in Multimodal Learning

Justin Kwan
School of Computer Science
University of Waterloo
`j37kwan@uwaterloo.ca`

Abstract

ViTab, a multi-modal learning architecture predicts continuous regression values related to plant family classification by using the RGB images and ancillary numerical data from the iNaturalist and Try plant trait datasets. ViTab addresses challenges of learning joint latent representations from both images and tabular data, in predicting labels with heavy skew long-tail distributions.

1 Introduction

We introduce ViTab, a multi-modal learning architecture designed to predict continuous regression values about the family each sample plant belongs to, using RGB plant images and ancillary numerical data from the iNaturalist [7] image and Try plant trait datasets. ViTab addresses the challenges of learning joint latent representations across images and tabular data for regression, while predicting labels that follow a heavy long-tail distributions.

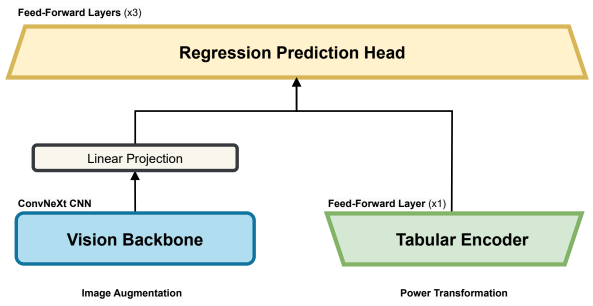

Figure 1: An overview of the proposed ViTab multi-modal architecture includes three main components: (1) a vision encoder backbone leveraging pretrained CNN for image feature extraction, (2) a feed-forward network for tabular feature extraction, and (3) a prediction head to regress six mean plant species traits.

This report presents the empirical accuracy and performance of ViTab on the multimodal iNaturalist and Try plant dataset. It includes ablation studies that compare various ViTab configurations with alternative models using Vision and classic Transformer architectures for multi-modal feature extraction and fusion.

1.1 Notation

We define each plant image height and width as $H = W = 224$ after augmentation, ancillary feature dimension as $d_{\text{feature}} = 163$, and target dimension $d_{\text{target}} = 6$. Denote the training dataset as $\mathcal{D}_{\text{train}} =$

$\{(\mathbf{x}_i^{\text{img}}, \mathbf{x}_i^{\text{tab}}, \mathbf{y}_i) \mid \forall i = 1, \ldots, n\}$ where each data point $\mathbf{x}_i$ contains image $\mathbf{x}_i^{\text{img}} \in \mathbb{R}^{3 \times H \times W}$ and ancillary tabular features $\mathbf{x}_i^{\text{tab}} \in \mathbb{R}^{d_{\text{feature}}}$ describing the plant. Each corresponding target $\mathbf{y}_i \in \mathbb{R}^{d_{\text{target}}}$ contains the output traits describing the species which the plant is a part of.

1.2 Preprocessing

Dataset. The training dataset, consisting of plant images and associated ancillary feature rows, was randomly split into a 1:4 validation to training ratio to ensure enough data to validate model performance while still having enough training dataset diversity to still learn complex plant patterns.

Image Augmentation. Images in each training batch were augmented to improve model generalization and prediction accuracy over future unseen plant images. These augmentations expose ViTab to a higher diversity of plant images in training by simulating plausible variation in future unseen plant images, improving its ability to generalize over test datasets. Images were randomly cropped and resized to 224x224 pixels using `RandomResizedCropAndInterpolation` to introduce different aspect ratios and zoom scales. `RandomHorizontalFlip` of 0.5 probability was applied and brightness, contrast, saturation, and hue were adjusted within specified ranges using `ColorJitter` to better generalize in different outdoor lighting conditions. Pixel values were normalized to a common scale using predefined mean and standard deviation values, to improve training stability in varying lighting environments. The first three augmentations were applied only during training to introduce variation, while normalization was applied in both training and testing.

Tabular Transformation. Ancillary input features and most importantly, target outputs in the training dataset that exhibited long-tail distributions were transformed into normal bell-curve distributions using power transformations. With more normally distributed target labels, the model can learn a wider variance in each target's output predictions, preventing it from simply predicting the most frequently occurring or mean value of a highly skewed distribution.

Let $\mathbf{x}_{ij}$ denote the jth ancillary feature value for the ith image data point. The minimum value across all data points in the jth feature dimension is $\min_i \mathbf{x}_{ij}$. Each fixed jth feature dimension is normalized based on the severity of skewness in its distribution:

$$\text{Normalize}(\mathbf{x}_{ij}) = \begin{cases} \mathbf{x}_{ij} - \min_i \mathbf{x}_{ij} & \text{if Kurtosis}(\mathbf{x}_{ij}) < 0.5, \quad \forall i = 1, \ldots, n \\ \sqrt{\mathbf{x}_{ij} - \min_i \mathbf{x}_{ij}} & \text{if } 0.5 \leq \text{Kurtosis}(\mathbf{x}_{ij}) < 1, \quad \forall i = 1, \ldots, n \\ \log_{10}\left(\mathbf{x}_{ij} - \min_i \mathbf{x}_{ij} + 10^{-6}\right) & \text{if Kurtosis}(\mathbf{x}_{ij}) \geq 1, \quad \forall i = 1, \ldots, n \end{cases}$$

Then, each normalized feature and target label dimension (column) was standardized via standard scalar to adjust the range of label values to lie within one standard deviation from the mean. Standardization scales each of the feature and label values to a consistent and uniform scale, which improves the dimensional symmetry of gradient updates during training. This stabilizes training while increasing the likelihood of converging to a global optimum.

Let σ_j be the standard deviation and μ_j be the mean of each jth training dataset feature dimension. Then, the each fixed jth feature and target label dimension is standardized as follows:

$$\text{Standardize}(\tilde{\mathbf{x}}_{ij}) = \frac{\mathbf{x}_{ij} - \mu_j}{\sigma_j}, \quad \forall i = 1, \ldots, n$$

Normalization and standardization were applied to the ancillary tabular training, validation and test datasets, **only** using the parameters obtained from the training dataset. This was done to prevent ViTab from inadvertently memorize the specific statistics, such as means and standard deviations, of the validation and test sets (data-leakage).

1.3 ViTab Architecture

Vision Backbone. A convolutional neural network (CNN) was chosen as the feature extraction backbone to analyze plant images from the iNaturalist dataset. Its spatially inductive bias allows it to effectively process lower-resolution plant images that are geometrically connected. Unlike Vision Transformers (ViT)[3] which require computationally expensive pairwise attention across image patches, CNNs can efficiently extract local receptive features in linear time per pass. Capturing every possible relationship within plant images does not justify higher computational cost of training ViTs, especially with the limited resources of a university student.

Commonly practiced in transfer learning, we take advantage of ConvNeXt-large[9] vision backbone's pre-trained weights by freezing them during training. This preserves its feature extraction and generalization capabilities that it has learned over the large and diverse ImageNet-22K[11] and ImageNet-1K[1] datasets. Additionally, we prevent gradients from unnecessarily flowing through the backbone during training back-propagation to prevent unnecessary gradient update computations, speeding up training time.

Each augmented image $\tilde{\mathbf{x}}_i^{\text{img}} \in \mathbb{R}$ is passed into the ConvNeXt-large backbone. The series of convolution operations measure the spatial similarity and **extract** relevant features, condensing the image into a latent 2D feature map $\in \mathbb{R}^{C\times H\times W}$. This Feature map is then flattened into a dense latent vector representation $\mathbf{z}_i^{\text{img}} \in \mathbb{R}^{1536}$ that encodes and captures only the most important features, patterns and spatial relationships within the plant image. This dense extracted feature representation is later used by the final feed-forward prediction head to learn a joint latent representation between the most important features within the image and corresponding ancillary tabular metadata. A subsequent learned linear layer defined by weights $\mathbf{W} \in \mathbb{R}^{768\times 1563}$ and biases $\mathbf{b} \in \mathbb{R}^{768}$, down-scales $\mathbf{z}_i^{\text{img}}$ to a further condensed $\mathbf{h}_i^{\text{img}} = \mathbf{W}\mathbf{z}_i^{\text{img}} + \mathbf{b} \in \mathbb{R}^{768}$ while also fine-tuning the representation for the plant specific feature extraction task.

Table 1: Performance of Different Vision Backbones on iNaturalist Dataset

Pretrained Vision Backbone	ImageNet	Normalization	MSE Train	R^2 Validation	R^2 Test
DeiT3-large[12]	22K + 1K	Log	0.5144	0.4306	0.3359
DeiT3-base	22K + 1K	Log	0.5514	0.3105	0.3043
ConvNeXtV2-large[13]	22K + 1K	Log	0.4996	0.4437	0.3449
ConvNext-large[9]	22K + 1K	Log	0.4576	0.4444	0.3351
ConvNeXt-large	1K	Log	0.5522	0.3428	0.2556
ConvNeXt-large	1K	Sqrt	0.6798	0.2637	0.1841

Tabular Encoder. A single feed-forward linear layer and smooth non-linear GELU[6] activation, inspired by Hager et al. [5], was chosen as the tabular feature extractor to capture and represent the latent patterns encoded in the ancillary input metadata describing each plant image. This layer, defined by weights $\mathbf{W} \in \mathbb{R}^{256\times 163}$, biases $\mathbf{b} \in \mathbb{R}^{256}$, and GELU activation $\sigma(\mathbf{x}) = \mathbf{x}\Phi(\mathbf{x})$, transforms each input vector $\mathbf{x}_i^{\text{tab}} \in \mathbb{R}^{d_{\text{feature}}}$ into a higher-dimension latent representation $\mathbf{h}_i^{\text{tab}} = \sigma(\mathbf{W}\mathbf{x}_i^{\text{tab}} + \mathbf{b}) \in \mathbb{R}^{256}$.

We initially implemented with a transformer-based tabular encoder to enrich tabular features through self-attention, capturing relationships like their correlations as seen in [2]. Despite experimenting with various transformer layer configurations, shallower and wider and deeper but narrower, this approach yielded slightly lower validation and test predictive R^2 scores, while increasing computational training overhead. The simpler, more effective feed-forward tabular feature extractor was empirically validated, demonstrating a practical application of Occam's Razor.

Prediction Head The final prediction head that produces six regressive plant trait values, uses a feed-forward network of three linear down-scaling projection layers to learn a learn a joint latent representation between the most important features within the image and corresponding ancillary tabular metadata. The prediction head is passed the concatenated $\mathbf{h}_i = \mathbf{h}_i^{img} + \mathbf{h}_i^{tab} \in \mathbb{R}^{1024}$ latent feature representations extracted by the image backbone and tabular encoder. The feed forward network, defined by $\text{Regression}(\mathbf{x}) = \mathbf{W}_3\left(\text{DO}_{0.1}\left(\sigma\left(\text{LN}\left(\sigma\left(\text{LN}\left(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1\right)\right)\mathbf{W}_2 + \mathbf{b}_2\right) + \left(\sigma\left(\text{LN}\left(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1\right)\right)\right)\right)\right)\right) + \mathbf{b}_3$, where weights $\mathbf{W}_1 \in \mathbb{R}^{1024\times 256}$, $\mathbf{W}_2 \in \mathbb{R}^{256\times 256}$, $\mathbf{W}_3 \in \mathbb{R}^{256\times 6}$ and biases $\mathbf{b}_1, \mathbf{b}_2 \in \mathbb{R}^{256}$ and $\mathbf{b}_3 \in \mathbb{R}^6$. The final prediction is obtained as $\hat{\mathbf{y}}_i = \text{Regression}(\mathbf{h}_i) \in \mathbb{R}^{d_{\text{target}}}$.

The three prediction head layers gradually reduce dimensionality to six final regressed outputs, preserving important multi-modal latent features and relationships between the image and ancillary data. It also the residual connection to be applied between the two $\mathbb{R}^{256}$ dimensional layers, which maintains gradient flow during backpropagation through potentially inactive linear layers that are slow to learn due to the sufficient inferencing ability of the pretrained vision backbone. A dropout layer is applied after the penultimate layer to regularize and improve ViTab's robustness by preventing any layer collusion that could cause overfitting. Empirically, the residual connections and dropout improved test R^2 scores by approximately 6%.

1.4 Training

The ViTab model was trained over 20 epochs using the AdamW[10] optimizer which introduces L2 weight decay to increase training stability and imrpove the rate of global minimum convergence. A faster learning rate of 1×10^{-3} was used in combination with a weight decay of 1×10^{-4} and a relatively smaller batch size of 64.

Applying the trick of **gradient clipping** with a max norm of 1.0 significantly improved the training stability of ViTab, resulting in a smoother validation loss curve. By capping the magnitude of gradients during back-propagation to prevent them from becoming too large causing unstable updates, the model's gradient weight updates were more symmetric and balanced over all the feature dimensions (uniform learning) which prevents any one feature from significantly influencing the training behavior. Empirically, gradient clipping improved the test R^2 scores by about 7%.

Over four training runs of 20 epochs each, ViTab achieved a mean minimum MSE training loss of 0.4820 ± 0.0381 mean maximum validation R^2 score of 0.4445 ± 0.0018, and mean maximum combined test R^2 score of 0.3470 ± 0.0017.

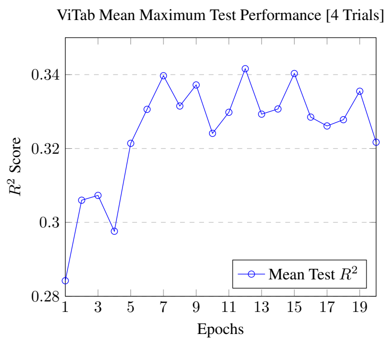

Figure 2: Test R^2 scores over epochs, from four separate trials.

2 Conclusion

Through this project of training and testing ViTab over the iNaturalist and Try datasets, we've learned the incredible importance of preprocesing and feature engineering, where the model can only perform well if the dataset is of high quality. Techniques such as data normalization and standardization are very important as each feature must contribute equally to achieve more stable and dimensionally symmetric training weight updates. This experience has shown that a model can only perform as well as the quality of the dataset it's trained on, and we cannot simply improve performance by using more powerful or larger models without thinking about the data.

In the future, we hope to explore adding of classification components to the model, that is, performing auxiliary tasks[8], such as predicting the plant species class since we can infer each class based on each unique set of six species trait labels. The loss attained from auxiliary learning could guide the gradient updates in the correct general direction to improve ViTab's ability to generalize and accurately predict the same six regression values.

It would also be interesting to analyze which component of the ViTab could be causing premature over-fitting, whether it's the vision backbone or the tabular encoder. We could freeze the over-fitting component while allowing the other to continue training to further improve the model's predictive accuracy.

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li. "ImageNet: a Large-Scale Hierarchical Image Database". In: June 2009, pp. 248–255.

[2] J. Deng, Z. Yang, T. Chen, W. Zhou, and H. Li. "TransVG: End-to-End Visual Grounding with Transformers". 2022. arXiv: 2104.08541 [cs.CV].

[3] A. Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". 2021. arXiv: 2010.11929 [cs.CV].

[4] M. Goldblum et al. "Battle of the Backbones: A Large-Scale Comparison of Pretrained Models across Computer Vision Tasks". 2023. arXiv: 2310.19909 [cs.CV].

[5] P. Hager, M. J. Menten, and D. Rueckert. "Best of Both Worlds: Multimodal Contrastive Learning with Tabular and Imaging Data". 2023. arXiv: 2303.14080 [cs.CV].

[6] D. Hendrycks and K. Gimpel. "Bridging Nonlinearities and Stochastic Regularizers with Gaussian Error Linear Units". *CoRR*, vol. abs/1606.08415 (2016). arXiv: 1606.08415.

[7] G. V. Horn, O. M. Aodha, Y. Song, Y. Cui, C. Sun, A. Shepard, H. Adam, P. Perona, and S. Belongie. "The iNaturalist Species Classification and Detection Dataset". 2018. arXiv: 1707.06642 [cs.CV].

[8] S. Liu, A. J. Davison, and E. Johns. "Self-Supervised Generalisation with Meta Auxiliary Learning". 2019. arXiv: 1901.08933 [cs.LG].

[9] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. "A convnet for the 2020s". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 11976–11986.

[10] I. Loshchilov and F. Hutter. "Decoupled Weight Decay Regularization". 2019. arXiv: 1711.05101 [cs.LG].

[11] T. Ridnik, E. Ben-Baruch, A. Noy, and L. Zelnik-Manor. "ImageNet-21K Pretraining for the Masses". 2021. arXiv: 2104.10972 [cs.CV].

[12] H. Touvron, M. Cord, and H. Jégou. "DeiT III: Revenge of the ViT". 2022. arXiv: 2204.07118 [cs.CV].

[13] S. Woo, S. Debnath, R. Hu, X. Chen, Z. Liu, I. S. Kweon, and S. Xie. "ConvNeXt V2: Co-designing and Scaling ConvNets with Masked Autoencoders". 2023. arXiv: 2301.00808 [cs.CV].