Plant and Animal Species Prediction at Certain Locations with Convolutional Neural Networks

Juntao Jiang[1]

[1]Zhejiang University, 38 Zheda Rd., Hangzhou, Zhejiang, 310027, People Republic of China

Abstract

Prediction of plant and animal species at a given location can give help to biodiversity management and conservation. GeoLifeCLEF 2022 competition at LifeCLEF lab aims to predict the species list with the top 30 frequencies' occurrence at a certain localization. This paper is a working note for this competition. We trained EfficientNet-b3 models on the remote sensing imagery with 256x256 RGB patches centered in France and US separately. We achieved 0.75686 top-30 error rate on private leaderboard and ranked at 7th place.

Keywords

Localization, Prediction, Species, GeoLifeCLEF 2022, Remote sensing imagery, EfficientNet-b3

1. Introduction

As one of the most crucial topics in 21^{st} century, biodiversity management and conservation are related to the sustainable development of the economy, the protection of the environment and even the continuation of human civilization. Different species always appear at different frequencies in various regions, thus understanding the relationship between species and regions is a significant step in biodiversity conservation.

GeoLifeCLEF 2022 competition[1] at LifeCLEF lab[2] aims to predict the list of species most likely to be observed at a given location automatically with machine learning methods that can be used to improve species identification tools, develop location-based recommendation services and educational applications. Participants should predict the species list with top 30 frequencies' occurrence at a certain localization from remote sensing imagery, land cover data, altitude data, bioclimatic data as well as pedologic data. The remote sensing imagery is from NAIP for US and IGN for France respectively.

As a working note of this competition, this paper will give a simple solution by using convolutional neural networks (CNNs) with the top 30 frequencies' occurrence at a certain localization from remote sensing ima. Details of models design, data augmentation and implementation methods will be given. The results obtained by our method ranked at 7^{th} place finally. We will also give a discussion of potential methods for further improvement.

The main points of this work are listed below:

- EfficientNet-b3 for RGB image classification;

CLEF 2022: Conference and Labs of the Evaluation Forum, September 5–8, 2022, Bologna, Italy

jj2910@nyu.edu (J. Jiang)

(a) (b)

Figure 1: Samples of 256x256 RGB patches centered at a) France and b) the United States

- Focal loss to solve sample imbalance problems
- Data augmentation to generalize the model;

Due to the time limitation of the challenge, we failed to provide ablation studies for each part of our method, which will be done in further study. We hope this work can provide some hints for researchers to build stronger and more efficient baselines.

2. Methods

2.1. Data Selection and Analysis

Because of the limitation of time and computational power, we are only able to use parts of datasets. We focused on the remote sensing imagery with 256mx256m RGB patches centered at each observation. These images are in JPEG formats and with a resolution of 1 meter per pixel. The sources of these images are NAIP for US and IGN for France.

- Observations in France:
  There are 4858 species and 671244 observations in the dataset from France. That is, this is a classification problem for 4858 classes. There are 671244 images for training and validation.

- Observations in the United States:
  There are 14135 species and 956231 observations in the dataset from the United States. That is, this is a classification problem for 14135 classes. There are 956231 images for training and validation totally.

2.2. Data Augmentation

We applied several different data augmentation methods to make the model more generalizable. We implemented all augmentation methods based on Albumentations, an open-source library[3]

Table 1
Data Augmentation Methods used for Prediction

Composed Methods	Probability	Submethods' Probability
RandomResizedCrop	0.5	/
Flip	0.5	/
RandomRotate90	0.5	/
ShiftScaleRotate	0.5	/
HueSaturationValue	0.5	/
One of (RandomBrightnessContrast, RandomGamma)	0.5	0.5, 0.5
One of (Blur, GaussianBlur, MotionBlur)	0.1	0.1, 0.1, 0.1
One of (GaussNoise, ISONoise, GridDropout, CoarseDropout)	0.1	0.1, 0.1, 0.2, 0.2

Table 2
Specific Details of Some Data Augmentation Methods

Method	Details
RandomResizedCrop	height=256, width=256, scale=(0.08, 1.0), ratio=(0.75, 1.3333333333333333)
ShiftScaleRotate	shift_limit=0.0625, scale_limit=0.1, rotate_limit=45
HueSaturationValue	hue_shift_limit=20, sat_shift_limit=30, val_shift_limit=20
RandomBrightness	limit=0.2
RandomGamma	gamma_limit=(80, 120)
Blur	blur_limit=7
GaussianBlur	blur_limit=(3, 7), sigma_limit=0
GaussNoise	var_limit=(10.0, 50.0)
ISONoise	color_shift=(0.01, 0.05), intensity=(0.1, 0.5)
GridDropout	holes_number_x=image_width//10, holes_number_y=image_height//10,shift_x=0,shift_y=0
CoarseDropout	max_holes=16, max_height=16, max_width=16, min_holes=8, min_height=8, min_width=8

for data augmentation. All methods we used are listed in Table 1. The specific details of these methods are listed in Table 2.

2.2.1. Noising Methods

- Gaussian Noising
  Gaussian noise refers to a class of noise whose probability density function follows a Gaussian distribution.

- ISO Noising
  We also applied Poisson noise to image to simulate camera sensor noise. The noise's probability density function follows a Poisson distribution.

2.2.2. Dropout Methods

- Coarse Dropout:
  Coarse Dropout[4] is a data augmentation method proposed in 2017, also called Cutout, which can achieve conversion by losing information on a rectangular area with selectable sizes and random locations. The loss of information produces black rectangular blocks in all channels, which also produces color noise in some channels.
- Grid Dropout:
  Grid Dropout[5] is a data augmentation method proposed in 2020, which drops out rectangular regions of an image in a grid fashion which is defined as a mask M. Details of representation of M and choice of parameters can be found in [5].

2.2.3. Blurring Methods

- Blurring
  The Blur operation in Albumentations just takes the average of all pixels under the kernel area and replaces the center element.
- Motion Blurring
  The Motion Blurring operation is to simulate the blur of an object caused by movement.
- Gaussian Blurring
  The Gaussian blurring operation calculates the smoothing weight of each pixel through a two-dimensional Gaussian function, and can obtain a smoother blurred result. The 2D Gaussian function is shown as below:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\left(x^2+y^2\right)/2\sigma^2}$$

2.3. Models

2.3.1. EfficientNet

The common operation to improve performance in deep learning tasks includes increasing the network width, depth, or input image resolutions. EfficientNet[6] used a compound scaling method, simultaneously scaled the depth, width and input resolution of the network to achieve a trade-off between accuracy and computational complexity.

We implemented EfficientNet-b3 model by using PyTorch-Image-Models library[7]. The pre-trained weights on Imagenet[8] were also used. The final classification layer was customized, which was made of a linear layer, an activation layer, a dropout step, and the final linear layer.

The detailed architectures of the model are shown in Figure 2 and 3.

2.3.2. Focal Loss

We applied Focal Loss[9] in our solution, which was proposed in 2017 to solve the problem of extreme sample imbalance. It can increase the weight of target categories with a small number of samples during training. The loss function can be represented as:

$$\mathrm{FL}\left(p_{\mathrm{t}}\right) = -\alpha_{\mathrm{t}}\left(1-p_{\mathrm{t}}\right)^{\gamma}\log\left(p_{\mathrm{t}}\right)$$

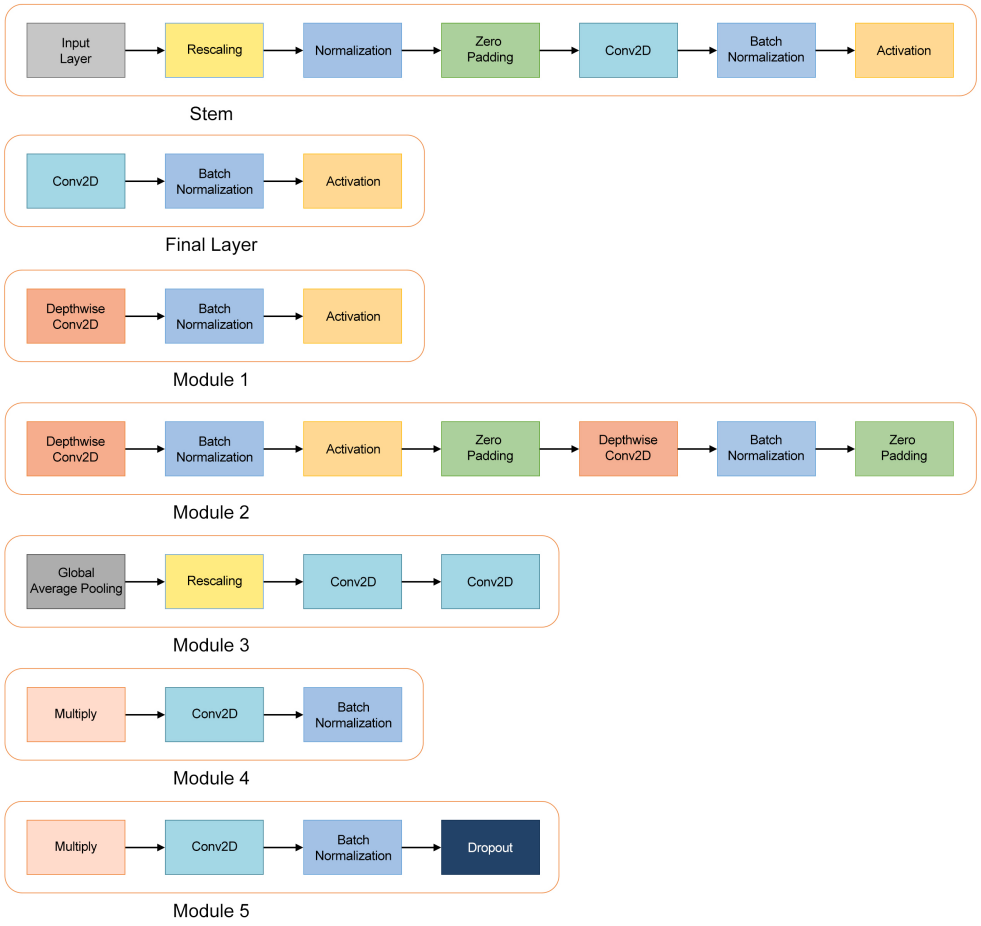

Figure 2: Architectures of Modules in EfficientNet

where

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise} \end{cases}$$

2.4. Training Process

2.4.1. Implementation Details

We trained, validated, and tested images obtained from US and France separately. We used StratifiedKFold scikit-learn[10] library to split the dataset as a training set and a validation set with the ratio of 3:1. For France's observation, the training set contains 503433 images and the validation set contains 167811 images. For US' observation, the training set contains 717174 images and the validation set contains 239057 images.

For US's observation, we used the best model to test while for France's observation we used the last model to test. The deep learning framework used is PyTorch[11], and the PyTorch-

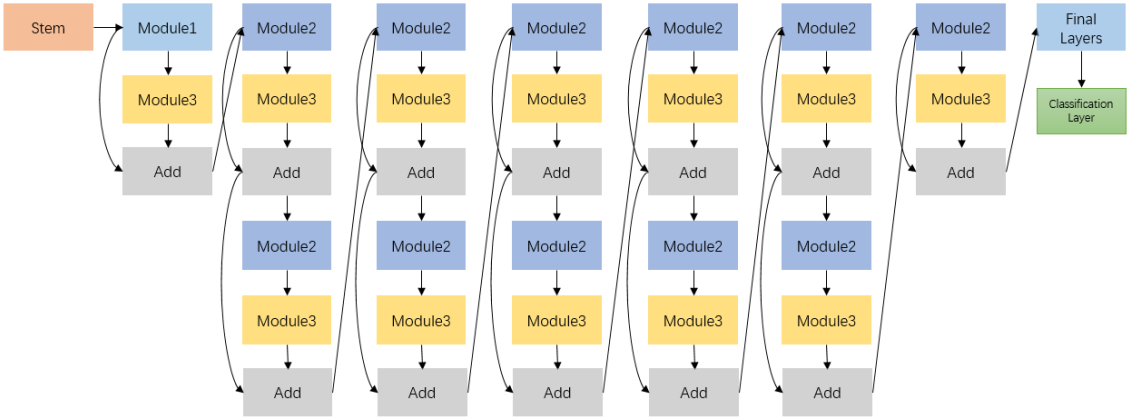

Figure 3: The Architecture of EfficientNet-b3 Model

Table 3
Versions of Some Packages Used in the Experiments

Package	version
PyTorch-Image-Models	0.5.4
Albumentations	1.2.0
PyTorch Lighting	1.6.4
PyTorch	1.11.0
scikit-learn	1.1.0
Numpy	1.22.3

lightning framework is also used. We trained models on Titan RTX with 24G RAM. We used Adam optimizer[12] and used OnecycleLR[13] as scheduler. We used the valid loss as evaluation metric in the validation stage.

2.4.2. Package Versions

The versions of some packages used in the experiments are shown in the Table 3.

2.4.3. Other Parameters' settings

The global seed is 42, the input image sizes are 256, the max learning rate is 1e-3 and the batch size is 32. The normalizing parameters are: mean=[0.485, 0.456, 0.406] and std=[0.229,0.224,0.225] respectively, for training, validation and test process.

3. Other Methods Tried in the Competition

3.1. Data Selection

We also tried to use latitude and longitude data to predict the observations. The data from the United States and France were used together, which are 1627475 observations totally. We split the dataset as training set and validation set, which are of 1587395 and 40080 observations respectively.

3.2. Methods and Implementation

We implemented the K-Nearest Neighbor(KNN) method by using scikit-learn library. The general process is as below:

1. Given an unknown object in the test set, compute its distance from each sample in the training set.
2. Find the k training samples that are closest to the unknown object.
3. Sort the frequency of each category in the k nearest neighbors and return the 30 categories with the highest frequency.

3.3. Post-Processing

We removed species that are never predicted from training process and redid the K-Nearest Neighbor process. The new predictions would be regarded as the final results.

4. Results

4.1. Evaluation

For test stage, the evaluation metric for this competition is the top-30 error rate. Each observation $\boldsymbol{i}$ is associated with a single ground-truth label $\boldsymbol{y}_i$ corresponding to the observed species. For each observation, the submissions will provide 30 candidate labels $\hat{y}_{i,1}, \hat{y}_{i,2}, \ldots, \hat{y}_{i,30}$. The top-30 error rate is then computed using Top-30 error rate:

$$\text{Top-30 error rate } = \frac{1}{N}\sum_{i=1}^{N} e_i$$

$$\text{where} \quad e_i = \begin{cases} 1 & \text{if } \forall k \in \{1,\ldots,30\}, \hat{y}_{i,k} \neq y_i \\ 0 & \text{otherwise} \end{cases}$$

4.2. Results and Comparison

Our method reached 0.75686 top-30 error rate in private leaderboard during the competition and ranked at 7th place. The comparison with other methods are shown in Table 4.

Table 4
Results on the Private Leadrboard

Method	Top-30 error rate
Top-30 most present species	0.94465
KNN on environmental vectors(the number of neighbors is 1800)	0.79954
KNN on environmental vectors(the number of neighbors is 1500)	0.79918
KNN on environmental vectors(the number of neighbors is 2100)	0.79820
Random Forest (100 trees of max depth 16) on environmental vectors	0.76153
CNN on RGB patches (pretrained ResNet-50, learning rate = 0.01, batch size = 32, early stropping on top-30 error rate)	0.73659
Our Methods	0.75686

5. Conclusion and Discussion

This paper gave a simple baseline of using CNNs to predict plant and animal species at a certain location. Although failed to give a competitive result, we still hope this work can give some hints to researchers who are interested in that.

Here are some potential directions to improve the results: a) Ensemble of different Models; b) Ensemble of results from remote sensing imagery, land cover data, altitude data, bioclimatic data and pedologic data. c) Semi-Supervised learning d) Metric loss

We believe that even with the results achieved in the first place, there is still a lot of potential for improvement, which will be left for continuing research.

Acknowledgments

The author would like to thank iNaturalist, Pl@ntNet citizen and other science platforms in this competition to provide such a large and valuable dataset, as well as their continuing effort of scientific work on biodiversity management and conservation. We also thank LifeCLEF lab at CLEF2022 and FGVC9 at CVPR2022.

References

[1] T. Lorieul, E. Cole, B. Deneu, M. Servajean, A. Joly, Overview of GeoLifeCLEF 2022: Predicting species presence from multi-modal remote sensing, bioclimatic and pedologic data, in: Working Notes of CLEF 2022 - Conference and Labs of the Evaluation Forum, 2022.

[2] A. Joly, H. Goëau, S. Kahl, L. Picek, T. Lorieul, E. Cole, B. Deneu, M. Servajean, A. Durso, H. Glotin, R. Planqué, W.-P. Vellinga, A. Navine, H. Klinck, T. Denton, I. Eggel, P. Bonnet, M. Šulc, M. Hruz, Overview of lifeclef 2022: an evaluation of machine-learning based species identification and species distribution prediction, in: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, 2022.

[3] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, A. A. Kalinin, Albumentations: fast and flexible image augmentations, Information 11 (2020) 125.

[4] T. DeVries, G. W. Taylor, Improved regularization of convolutional neural networks with cutout, arXiv preprint arXiv:1708.04552 (2017).
[5] P. Chen, S. Liu, H. Zhao, J. Jia, Gridmask data augmentation, arXiv preprint arXiv:2001.04086 (2020).
[6] M. Tan, Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: International conference on machine learning, PMLR, 2019, pp. 6105–6114.
[7] R. Wightman, Pytorch image models, https://github.com/rwightman/pytorch-image-models, 2019. doi:10.5281/zenodo.4414861.
[8] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems 25 (2012).
[9] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2980–2988.
[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, the Journal of machine Learning research 12 (2011) 2825–2830.
[11] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., Pytorch: An imperative style, high-performance deep learning library, Advances in neural information processing systems 32 (2019).
[12] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).
[13] L. N. Smith, N. Topin, Super-convergence: Very fast training of neural networks using large learning rates, in: Artificial intelligence and machine learning for multi-domain operations applications, volume 11006, International Society for Optics and Photonics, 2019, p. 1100612.