Efficient Model Integration for Snake Classification

Jun Yu[1], Hao Chang[1], Zhongpeng Cai[1], Guochen Xie[1], Liwen Zhang[1], Keda Lu[1,2], Shenshen Du[1], Zhihong Wei[1,2], Zepeng Liu[1,2], Fang Gao[3] and Feng Shuang[3]

[1]*University of Science and Technology of China, Hefei, Anhui, China*
[2]*Ping An Technology Co., Ltd, Shenzhen, Guangdong, China*
[3]*Guangxi University, Nanning, Guangxi, China*

Abstract

An accurate AI-driven system for automating snake species is of great value, allowing doctors to quickly diagnose the condition of injured people and thus effectively reducing deaths due to snake bites. SnakeCLEF 2022 challenge provides a dataset of 1,572 snake species, with information on their habitats. Because the dataset has a long-tailed distribution, it is difficult to perform accurate identification. We train the models by using the methods of AutoAugment, RandAugment and Focal Loss. Finally, we use the model integration method to effectively improve the recognition accuracy and finally achieve macro f_1 score of 71.82% on the private leaderboard. Our code can be found at https://github.com/CZP-1/An-efficient-model-integration-based-snake-classification-algorithm.

Keywords

Snake identification, EfficientNets, Swin Transformer, BEiT, model integration

1. Introduction

Establishing a snake identification system is important for biodiversity, conservation and global health. But snakes identify visually high intra-class differences and low inter-class differences due to geographic location, color morphology, gender, or age. In contrast to general image classification, the goal of fine-grained image classification is to correctly classify subclasses that belong to the same superclass. Therefore, snake identification is a challenging fine-grained recognition task. Publicly available datasets and benchmarks accelerate machine learning research and allow quantitative comparisons of new approaches. In the fields of deep learning and computer vision, rapid progress over the past decade has been largely driven by the publication of large-scale image datasets. The same is true for the problem of FGVC, where datasets, such as iNaturalist[1], have helped develop and evaluate new methods for fine-grained domain adaptation. But there has been a lack of research on snakes. This paper describes a method for image-based snake species identification submitted by the USTC-IAT-United team to the SnakeCLEF 2022 challenge[2]– a part of LifeCLEF 2022 workshop[3, 4].

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

changhaoustc@mail.ustc.edu.cn (H. Chang); czp_2402242823@mail.ustc.edu.cn (Z. Cai)
0000-0002-3197-8103 (J. Yu); 0000-0001-8123-683X (H. Chang); 0000-0002-2540-3215 (Z. Cai); 0000-0001-6494-6362 (G. Xie); 0000-0002-4757-9473 (L. Zhang); 0000-0002-6328-2653 (K. Lu); 0000-0002-2683-3313 (S. Du); 0000-0002-4100-5700 (Z. Wei); 0000-0003-3052-1328 (Z. Liu); 0000-0003-1816-5420 (F. Gao); 0000-0002-4733-4732 (F. Shuang)

CEUR Workshop Proceedings (CEUR-WS.org)

Generally, the main methods of FGVC mainly focus on how to make the network focus on the most discriminative regions, such as part-based models and attention-based models. Inspired by human observational behavior, these methods introduce localization-induced biases to neural networks with complex structures. In addition, some data augmentation methods and loss functions can also make the model pay more attention to fine-grained feature regions. When some species are visually indistinguishable, some extra-visual information can assist fine-grained recognition, such as spatio-temporal priors and textual descriptions. For example, it is very common that most or all images of a particular snake species may come from a few countries, or even a single country, which inspires us to combine geographic information with fine-grained classification. However, there are currently few studies on snake-related datasets. This motivates us to explore the combination of each module of the deep neural network and model fusion effect for snake fine-grained recognition method.

To solve the problem of snake fine-grained recognition, we use an exploratory data analysis of the SnakeCLEF 2022 dataset and process the characteristics of the dataset such as imbalance and fine-grained accordingly. We apply state-of-the-art (SOTA) data augment methods to expand the dataset, and use Convolutional Neural Network (CNN) and Transformer[5] models with a large number of parameters as the backbone network to extract image features. In addition, we use various loss functions and attention mechanisms to deal with indistinguishable features and data imbalance, respectively. We found a weak improvement in the macro F_1 score of snake recognition results using Focal Loss and Seesaw Loss, although this takes more time.

The contribution of this paper are summarized as follows:

- We test the performance of different backbone networks based on CNN (EfficientNets[6, 7]) and transformer (Swin-L[8], BEiT-L[9]) on snake classification tasks.
- We use different data augment methods and loss functions on the snake classification task and find a effective combination.
- We give appropriate weights to different models for model integration and achieve macro F_1-score of 78.27% on the public leaderboard and 71.82% on the private leaderboard of the SnakeCLEF 2022.

2. Related work

2.1. Fine-grained classification

Existing fine-grained classification methods can be divided into visual-only classification methods and multimodal classification methods. The former relies entirely on visual information to solve the problem of fine-grained classification, while the latter tries to use multimodal data to build a joint representation that merges multimodal information to facilitate fine-grained recognition. Fine-grained classification methods that rely solely on vision can be broadly classified into two categories: localization methods [10] and feature encoding methods [11].

Early work [12] used partial annotations as supervision to make the network notice subtle differences between certain species and suffer from their expensive annotations. RA-CNN [13] was proposed to amplify subtle regions to recursively learn to distinguish region attention and region-based feature representations at multiple scales in a mutually reinforcing manner. NTSNet [14] proposed a self-supervised mechanism to efficiently localize information regions.

Feature encoding methods are dedicated to enriching feature representation capabilities to improve the performance of fine-grained classification. CAP [15] designed context-aware attention pools to capture subtle changes in images. TransFG [16] proposed a part selection module that applies a visual transformer to select discriminative image patches.MetaFormer[17] suggests pooling layer is an alternate to self-attention. Compared with localization methods, feature encoding methods are difficult to clearly distinguish the distinguishing regions between different species.

To distinguish these challenging visual categories, additional information, i.e., geographic location, attributes, and textual descriptions, can be helpfully utilized. Geo-Aware [18] introduced geographic information prior to fine-grained classification and systematically examined various previous approaches using geographic information, including post-processing, whitelisting, and feature modulation. Presence-only [19] also introduced a spatio-temporal prior to the network, which was shown to be effective in improving the final classification performance. CVL [20] proposed a two-branch network in which one branch learns visual features and the other branch learns textual features, and finally combines these two parts to obtain the final latent semantic representation. All the above methods were designed for specific prior information and cannot be flexibly adapted to different auxiliary information.

2.2. Snake identification

The SnakeCLEF dataset is a well-known snake identification dataset, and many teams continue to explore further based on this dataset. BME-TMIT[21] use a two-stage approach for subject detection and image classification, and augmented the classification's macro F_1 score with location information. CMP[22] used residual convolutional neural networks of different sizes, and combined the original and improved cross-entropy to improve the classification performance. In addition, the use of voting for fusion further improved the accuracy on the Snake dataset. FHDO-BCSG[23] combined the current SOTA CNNs and Transformers, and performed model fusion to obtain excellent score on the SnakeCLEF 2021 challenge.

3. Methodology

As shown in Figure 1, we use various data augment methods on the data, and then train four models : Swin-L[8], EfficientNet-B7[6], EfficientNet-L2[7], and BEiT-L[9]. Finally, we fuse the features extracted from each model to obtain the final classification results.

3.1. Dataset

For SnakeCLEF 2022 Chenllage, the dataset is based on 187,129 snake observations with 318,532 photographs belonging to 1,572 snake species and observed in 208 countries. The photographs originated from the online biodiversity platform – iNaturalist. In fact, for the trainset, there are a total of 270,251 images from 158,698 snake observations belonging to 1,572 snake species and observed in 207 countries.

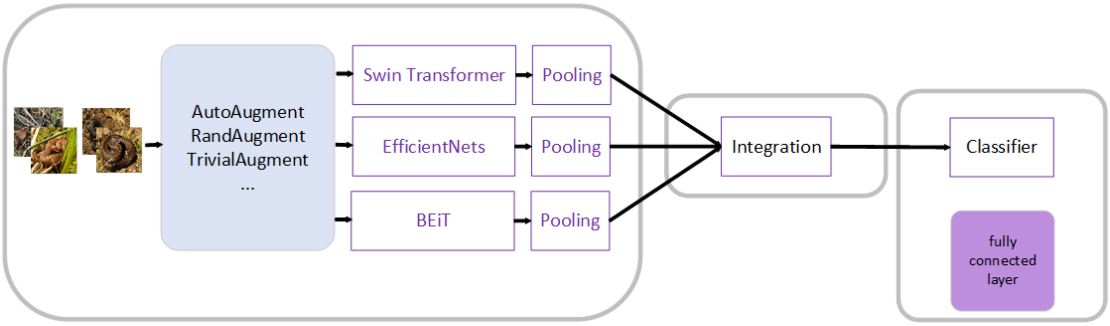

Figure 1: Our method for snake classification: firstly, we use data augment methods to pre-process the data, then the processed data is used to train different backbone networks, and finally the features extracted from each model are fused to obtain the final classification results.

In addition to the image data information, the challenge also provides metadata, including information on whether snake species are endemic and where snakes are observed. However, despite some exploration and attempts, we still have not found an effective method to improve the accuracy of snake identification using metadata.

3.2. Data Augment

We do not experiment with subtle data augment combinations on the SnakeCLEF 2022 dataset, but use NAS-based (Neural Architecture Search) or improved data augment methods, namely AutoAugment[24], RandAugment[25], and TrivialAugment[26].

AutoAugment. AutoAugment is a simple search-based data augment method. The main idea of which is to create a search space of data augment strategies and evaluate the quality of a particular strategy directly on some datasets. In the experiments of AutoAugment, the authors design a search space where each strategy consists of many substrategies, and in each batch a substrategy is chosen randomly for each image. The substrategies contain two operations, each of which is an image processing method, such as translation, rotation or clipping, and for each operation there is a set of probabilities and magnitudes to characterize the nature of the use of this operation. Using a search algorithm, search for the best policy that enables the neural network to achieve the highest validation set accuracy on the target dataset. AutoAugment achieves the same performance as the semi-supervised approach without using any unlabeled data. Finally, the strategies learned from one dataset can be directly transferred to other similar datasets and perform equally well. For example, the strategy learned in ImageNet can achieve state-of-the-art accuracy on the fine-grained visual classification dataset Stanford Cars without fine-tuning the pre-trained model with new data.

RandAugment. RandAugment investigates a data augment strategy based on NAS method search, which provides a significant improvement in search cost compared to earlier NAS search data augment strategies. NAS-method-based data augment strategies (e.g., AA and other methods) suffer from two major drawbacks. First, they use separate search phases, thus increasing the complexity of training and greatly increasing the consumption of computational resources. In addition, since the search phases are separate, these methods cannot adjust the regularization strength according to the model or dataset size. That is, we usually train small models by training them on small datasets and then apply them to train large models. The proposal of RandAugment (later referred to as RA) solves these two problems by significantly reducing the search space and allowing training directly on the target task without a proxy task, and by tailoring the regularization strength of data augment to different datasets and models.

TrivialAugment. The TrivialAugment data augment strategy originates from the NAS approach and outperforms NAS, implementing SOTA's data augment strategy in a simpler way. Although the approach of data augment using NAS method for automatic search is effective, the limitation lies in the need to trade-off the search efficiency and the performance of data augmentation. To solve this problem, TrivialAugment data augment strategy (later referred to as TA) is proposed. Compared with previous data augment strategies, TA is parameter-free and uses only one data augment method per image, so its search cost is almost free compared to AA and even RA, and it achieves SOTA results. TA uses the same data augment method as RandAugment. Specifically, data augment is defined as consisting of a data augment function a and the corresponding intensity value m (some data augment functions do not use intensity values). For each image, TA samples a data augmentation function and an intensity value uniformly, and then returns the augmented image. In addition, while previous methods tend to overlay multiple data augment methods, TA only uses a single data augment method for each image. Using such an approach, the TA-augmented dataset can be viewed as a single image augmented separately using all data augment methods, and then uniformly sampled from it.

3.3. Image feature extraction backbones

Backbone is crucial for feature extraction of SnakeCLEF 2022 dataset. Different backbones have different learning ability for features and different focus on the dataset, and the choice of different models can bring us richer data features. In this challenge, we use the following models.

3.3.1. Swin Transformer

There are two main challenges in the application of Transformer to the image field. Visual entities are highly variable, and the visual Transformer performance may not be very good in different scenes. With many pixel points, Transformer's global self-attention-based computation leads to a large computational effort. To address these two problems, Swin Transformer proposes a Transformer with a hierarchical design that includes a sliding window operation, which consists of a non-overlapping local window and an overlapping cross-window. Restricting the attention computation to a single window can introduce the localization of CNNs convolution operations on the one hand and save computation on the other. The overall architecture of Swin Transformer is hierarchical, with four stages, each of which reduces the resolution of the input feature map and expands the perceptual field layer by layer like CNNs. There are several places where it is handled differently than ViT[27]. ViT will position-encode the embedding on the input, while Swin-T is here as an option. Swin-T does a relative position encoding when calculating Attention. ViT will add a learnable parameter separately as a token for classification, while Swin-T directly averages and outputs classification, which is somewhat similar to the final global average of CNN. pooling layer. On the ImageNet22K[28] dataset, the accuracy rate of Swin Transformer can reach an astonishing 86.4%, which is one of the current SOTA models.

3.3.2. EfficientNet

The traditional practice of model scaling is to arbitrarily increase the depth or width of the CNNs, or to use a larger input image resolution for training and evaluation. While these approaches do improve accuracy, they typically require long periods of manual tuning and still often yield sub-optimal performance. A new approach to model scaling is to use a simple and efficient composite coefficient to scale CNNs in a more structured way. Unlike traditional methods that arbitrarily scale network dimensions such as width, depth, and resolution, EfficientNets uniformly scales network dimensions with a fixed set of scale scaling factors. By using this novel scaling method and AutoML (Auto Machine Learning) techniques, the authors call this model EfficientNets, which is up to 10 times more efficient (smaller and faster). To understand the effect of network scaling, the authors systematically studied the effect of scaling different dimensions on the model. While scaling individual dimensions can improve model performance, the authors observe that balancing all dimensions of the network based on available resources can maximize overall performance. In addition, the effectiveness of model scaling relies heavily on the baseline network. To further improve performance, the authors also develop a new baseline network that optimizes accuracy and efficiency by performing neural structure search using the AutoML MNAS framework. The final architecture uses moving inverse bottleneck convolution (MBConv).

3.3.3. Bidirectional Encoder representation from Image Transformer

Following the development of BERT[29] in the field of natural language processing. proposed a masked image modeling task to pretrain visual Transformers. Specifically, in our pre-training, each image has two views, namely image patches (e.g. 16×16 pixels) and visual tokens (i.e. discrete tokens). They first "tokenize" the original image into visual tokens. Then randomly mask some image patches and feed them into the backbone Transformer. The goal of pretraining is to recover original visual tokens from corrupted image patches. After pretraining BEiT, they directly fine-tune model parameters on downstream tasks by appending task layers on the pretrained encoder. Experimental results on image classification and semantic segmentation show that the model achieves better results than previous pre-training methods. For example, base-size BEiT achieves 83.2% top-1 accuracy on ImageNet-1K[28], significantly outperforming DeiT[30] (81.8%) trained from scratch under the same settings. Furthermore, large-size BEiT achieves 86.3% accuracy using only ImageNet-1K, even better than ViT-L[27] (85.2%) with supervised pretraining on ImageNet-22K.

3.4. Loss function

For the task of snake classification, the Loss function we use is a cross-entropy loss function like the work for training. In addition, we use Focal Loss[31] and Seesaw Loss[32] to mitigate the problem of long-tailed distribution of the dataset. Where Focal Loss uses a modulating factor to the cross-entropy loss to reduce the loss contribution from easy examples and elevate the importance of hard examples, Seesaw Loss achieves a relative balance of positive and negative sample gradients by dynamically reducing the weight of the excessive negative sample gradients imposed by the head category on the tail category.

3.5. Fine-grained classification strategy to improve F_1 score

We use the fine-grained classification method of PIM[33] to help us with the task of snake classification. PIM is an excellent method for fine-grained classification that automatically finds the most discriminative regions and uses local features to provide features that are more helpful for classification. PIM is a plug-in module, so it can be integrated into very many common CNN-based or Transformer-based network backbone, such as Swin Transformer, EfficientNet, etc. The plugin module can output pixel-level feature maps and fuse filtered features to enhance fine-grained visual classification. It can be briefly explained by selecting appropriate output feature maps from the backbone's blocks to input to a weakly supervised selector to filter out regions with strong discriminative power or regions with little relevance to classification, and finally fusing the features from the selector's output with a combiner to obtain prediction results.

4. Experiments

Firstly, we train various well-known architectures such as EfficientNet-B7, EfficientNet-L2, Swin-L, and BEiT-L on the CNN and Transformer families respectively. Furthermore, we use data augment methods to improve the performance of the model. Finally, the models of various different architectures are integrated.

4.1. Setup

In this section, we describe the complete training and evaluation procedure, including the training strategy, image augment, and testing procedures. At first, the dataset is split 9:1 between training and validation to select backbones. All architectures are initialized with publicly available pretrained checkpoints and the same policies are trained using the PyTorch framework in a Tesla A100 Tensor Core GPU. All neural networks are optimized using stochastic gradient descent with momentum set to 0.9. The starting learning rate (LR) is set to 0.01 and is further reduced by a specific adaptive learning rate planning strategy and the multi-step adjustment strategy is used when training on the full dataset. In order to speed up the efficiency of the model, the batch size is determined according to the memory consumption, which is between 4 and 128. For training, in addition to the conventional data augment methods, we also adopt a more advanced automatic augment technology. More specifically, we use random horizontal flip with 50% probability, random vertical flip with 50% probability, random adjustment crop with 0.8 - 1.0 scale, random brightness/contrast adjustment with 40% probability. All images are resized to the desired network input size: in the CNN performance experiments, the input size is 600 × 600. For Swin Transformer, the input size is 384 × 384, while on BEiT, it is 512 × 512.

In the process of training the model, we replace various loss functions for evaluation. In addition, data augmentation is introduced to fine-tune the training of the data based on the full amount of very unbalanced dataset used for training in the first phase, hoping to improve the accuracy and robustness of the model. Finally, model ensemble is applied, which is fused separately from the result layer and the feature layer.

4.2. Metrics

According to the requirements of the competition, the evaluation metrics we use is the macro F_1 score, which is not affected by the class frequency and is more suitable for the long-tailed class distribution observed in nature. Interestingly, despite the high performance requirements for overall classification in nature-related applications, most existing benchmarks only use accuracy as a scoring criterion. Given that the dataset is highly imbalanced and has long-tailed distributions, the learning process may ignore the least present species. Furthermore, using F_1^m score allows to easily assign a cost value to each label's two error types and measure more task-related performance. For example, mistaking venomous snake species for non-venomous snake species is a more serious problem when it comes to snake identification. Define F_1^m score as the mean of all F_1 scores:

$$Precision = \frac{TP}{TP+FP}, Recall = \frac{TP}{TP+FN} \tag{1}$$

$$F_1^S = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{2}$$

$$F_1^m = \frac{1}{N}\sum_{S=1}^{N} F_1^S \tag{3}$$

where N represents the number of classes and S is the species index.Final Macro F_1 score is calculated by computing the F_1 score for each species as the harmonic mean of the species Precision and the Recall .

4.3. Results

In this section, we compare the performance of well-known CNN-based models and Transformer-based models on the macro F_1 score. Additionally, we calculate the highest macro F_1 score of the single model on the dataset. Finally, the best results are obtained using model integration.

The impact of different data augments. In order to reduce the impact of long-tail distribution and fine-grained on identification, we use different data augmentation methods on the Swin-L, such as AutoAugment, RandAugmet, TrivialAugment. The results can be seen in Table 1. The results show that the combination of RandAugment and Swin-L works best.

Fine-grained classification strategy. The attention mechanism PIM is introduced to solve part of the problem of Fine-grained classification. As shown in Table 2, we can see a significant improvement over the original baseline. It is worth noting that we do not use the full data set, but divide the training set and the validation set according to 9:1 for training, and conduct online evaluation on the test set of the challenge. However, it takes more than twice as long

Table 1
The effect of different data augments (on public leaderboard)

Augment method	F_1^m score	Score fluctuation
Swin-L	68.14%	-
Swin-L + AutoAugment	68.29%	+0.15%
Swin-L + RandAugment	68.44%	+0.3%
Swin-L + TrivialAugment	68.33%	+0.19%

Table 2
The effect of PIM module (training on 90% trainset, on public leaderboard)

method	F_1^m score	Score fluctuation
Swin-L	62.38%	-
Swin-L + PIM	63.8%	+1.42%

Table 3
The effect of different loss function (training on 90% trainset, on public leaderboard)

loss function	F_1^m score	Score fluctuation
EfficientNet-B7 + CE Loss	68.14%	-
EfficientNet-B7 + Seesaw Loss	68.64%	+0.5%
EfficientNet-B7 + Focal Loss	68.89%	+0.75%

to train a model with PIM module than to train only backbone. Due to our limited computing resources, there is no way to train all the backbones with PIM module. We believe that if we train all the models with PIM and then do model integration, we can improve our performance greatly.

The impact of different loss function. In addition, the influence of different loss functions on the model accuracy is also tested. As shown in Table 3, while training on 90% trainset, Focal Loss performs better in fine-grained classification.

The effect of different backbones. As shown in Table 4, we use different backbone training on the full dataset of this challenge, adopt a high-performing data augment approach, replace the loss function with Focal Loss to obtain the results of the online testset evaluation. It is a pity that we do not use PIM because it requires a lot of computing resources and time. It can be seen that EfficientNet-L2 has the best performance, the performance of BEiT-L is only a little worse than that of EfficientNet-L2. From the comparison of the number of parameters, it seems that the model with more parameters works better for CNN-based architecture and transformer-based architecture. Of course, it needs more experiments to prove this.

Model Integration and Challenge Score. We fuse the softmax layers of each model in turn in different proportions according to the macro F_1 score order of each model, and use the parameters of the last layer to infer the category of that image. The fusion weights of each of these models are carefully tuned by us, and the final macro F_1 score is shown in Table 5. We achieve a result of 71.82% on the public leaderboard (20% of the test data) and a score of 78.22%

Table 4
The effect of different backbones

Backbone	Parameter	F_1^m score (Public)	F_1^m score (Private)
Swin-L	197M	68.14%	61.7%
Efficientnet-B7	66M	71.79%	64.99%
BEiT-L	306M	74.77%	67.61%
EfficientNet-L2	480M	75.77%	69.6%

Table 5
Effect of Model Integration

Backbone	weight	F_1^m score(Public)	F_1^m score(Private)
EfficientNet-L2	1	75.77%	69.6%
EfficientNet-L2 + BEiT-L	6:4	77.58%	71.76%
EfficientNet-L2 + BEiT-L + EfficientNet-B7	6:4:6	78.22%	71.93%
EfficientNet-L2 + BEiT-L + EfficientNet-B7 + Swin-L	6:4:6:0.1	78.27%	71.82%

Table 6
The final score

leaderboard	F_1^m score	rank
Public	78.27%	6
Private	71.82%	7

on the private leaderboard (approximately 80% of the test data). Surprisingly, from the private leaderboard, the result without Swin-L is slightly higher, reaching 71.93%.

In the end, as shown in Table 6, we rank 6th on the public leaderboard and 7th on the private leaderboard.

5. Conclusions

This paper proposes a model integration approach for fine-grained classification of SnakeCLEF 2022 dataset. Since this dataset has the characteristics of fine-grained and long-tailed distribution. We finally train four models, EfficientNet-L2, BEiT-L, EfficientNet-B7 and Swin-L. We find that for CNN-based or transformer-based models, models with a larger number of parameters work better. Then we fuse the features extracted from each model by the model integration method and obtain a significant improvement in performance. Finally, after adjusting the weights, we achieve macro F_1 score of 71.82% on private leaderboard and macro F_1 score of 78.27% on the public leaderboard. A noteworthy conclusion is that model integration is only better when single models perform similarly. If a model's macro F_1 score much lower than other models (like Swin-L), the improvement it can bring is limited and may even reduce our original macro F_1 score. A foreseeable result is that the final macro F_1 score can be improved if we continue to add suitable backbones for model integration.

References

[1] G. Van Horn, O. Mac Aodha, Y. Song, Y. Cui, C. Sun, A. Shepard, H. Adam, P. Perona, S. Belongie, The inaturalist species classification and detection dataset, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8769–8778.

[2] L. Picek, A. M. Durso, M. Hrúz, I. Bolon, Overview of SnakeCLEF 2022: Automated snake species identification on a global scale, in: Working Notes of CLEF 2022 - Conference and Labs of the Evaluation Forum, 2022.

[3] A. Joly, H. Goëau, S. Kahl, L. Picek, T. Lorieul, E. Cole, B. Deneu, M. Servajean, A. Durso, I. Bolon, et al., Lifeclef 2022 teaser: An evaluation of machine-learning based species identification and species distribution prediction, in: European Conference on Information Retrieval, Springer, 2022, pp. 390–399.

[4] 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.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[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] Q. Xie, M.-T. Luong, E. Hovy, Q. V. Le, Self-training with noisy student improves imagenet classification, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 10687–10698.

[8] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 10012–10022.

[9] H. Bao, L. Dong, F. Wei, Beit: Bert pre-training of image transformers, arXiv preprint arXiv:2106.08254 (2021).

[10] W. Ge, X. Lin, Y. Yu, Weakly supervised complementary parts models for fine-grained image classification from the bottom up, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 3034–3043.

[11] C. Yu, X. Zhao, Q. Zheng, P. Zhang, X. You, Hierarchical bilinear pooling for fine-grained visual recognition, in: Proceedings of the European conference on computer vision (ECCV), 2018, pp. 574–589.

[12] W. Luo, X. Yang, X. Mo, Y. Lu, L. S. Davis, J. Li, J. Yang, S.-N. Lim, Cross-x learning for fine-grained visual categorization, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 8242–8251.

[13] J. Fu, H. Zheng, T. Mei, Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4438–4446.

[14] Z. Yang, T. Luo, D. Wang, Z. Hu, J. Gao, L. Wang, Learning to navigate for fine-grained classification, in: Proceedings of the European Conference on Computer Vision (ECCV),

2018, pp. 420–435.

[15] A. Behera, Z. Wharton, P. Hewage, A. Bera, Context-aware attentional pooling (cap) for fine-grained visual classification, arXiv preprint arXiv:2101.06635 (2021).

[16] J. He, J.-N. Chen, S. Liu, A. Kortylewski, C. Yang, Y. Bai, C. Wang, A. Yuille, Transfg: A transformer architecture for fine-grained recognition, arXiv preprint arXiv:2103.07976 (2021).

[17] W. Yu, M. Luo, P. Zhou, C. Si, Y. Zhou, X. Wang, J. Feng, S. Yan, Metaformer is actually what you need for vision, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 10819–10829.

[18] G. Chu, B. Potetz, W. Wang, A. Howard, Y. Song, F. Brucher, T. Leung, H. Adam, Geo-aware networks for fine-grained recognition, in: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, 2019, pp. 0–0.

[19] O. Mac Aodha, E. Cole, P. Perona, Presence-only geographical priors for fine-grained image classification, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9596–9606.

[20] X. He, Y. Peng, Fine-grained image classification via combining vision and language, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5994–6002.

[21] R. Borsodi, D. Papp, Incorporation of object detection models and location data into snake species classification., in: CLEF (Working Notes), 2021, pp. 1499–1511.

[22] R. Chamidullin, M. Šulc, J. Matas, L. Picek, A deep learning method for visual recognition of snake species (2021).

[23] L. Bloch, C. M. Friedrich, Efficientnets and vision transformers for snake species identification using image and location information., in: CLEF (Working Notes), 2021, pp. 1477–1498.

[24] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, Q. V. Le, Autoaugment: Learning augmentation strategies from data, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 113–123.

[25] E. D. Cubuk, B. Zoph, J. Shlens, Q. V. Le, Randaugment: Practical automated data augmentation with a reduced search space, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, 2020, pp. 702–703.

[26] S. G. Müller, F. Hutter, Trivialaugment: Tuning-free yet state-of-the-art data augmentation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 774–782.

[27] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, arXiv preprint arXiv:2010.11929 (2020).

[28] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE conference on computer vision and pattern recognition, Ieee, 2009, pp. 248–255.

[29] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

[30] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, H. Jégou, Training data-efficient image transformers & distillation through attention, in: International Conference on

Machine Learning, PMLR, 2021, pp. 10347–10357.

[31] 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.

[32] J. Wang, W. Zhang, Y. Zang, Y. Cao, J. Pang, T. Gong, K. Chen, Z. Liu, C. C. Loy, D. Lin, Seesaw loss for long-tailed instance segmentation, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 9695–9704.

[33] P.-Y. Chou, C.-H. Lin, W.-C. Kao, A novel plug-in module for fine-grained visual classification, arXiv preprint arXiv:2202.03822 (2022).