# Incorporation of object detection models and location data into snake species classification

Regő Borsodi[1], Dávid Papp[1]

[1]*Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Budapest, Hungary*

**Abstract**
Photo-based automatic snake species identification could assist in clinical management of snakebites. LifeCLEF announced the SnakeCLEF 2021 challenge, which aimed attention on this task and provided location metadata for most snake images. This paper describes the participation of the BME-TMIT team in this challenge. In order to reduce clutter and drop the unnecessary background, we employed the state-of-the-art EfficientDet object detector, which was fine-tuned on manually annotated images. Detected snakes were then classified by EfficientNet weighted with the likelihood of location information. Based on the official evaluation of SnakeCLEF 2021, our solution achieved an $F_1$ country score of 0.903, which placed our team at rank 1 position in the challenge.

**Keywords**
classification, object detection, convolutional neural networks, snake species identification, SnakeCLEF

## 1. Introduction

Snakebite envenoming is a potentially life-threatening disease caused by toxins in the bite of a venomous snake. Envenoming can also be caused by having venom sprayed into the eyes by certain species of snakes that have the ability to spit venom as a defense measure [1]. Most of these occur in Africa, Asia, and Latin America. In Asia, up to 2 million people are envenomed by snakes each year, while in Africa, there are an estimated 435 000 to 580 000 snakebites annually that need treatment. Identification of the snake is essential in planning treatment in certain areas of the world, but it is a difficult task and not always possible [2]. Ideally the snake would be brought in with the person, but attempting to catch or kill the offending snake also puts one at risk for re-envenomation or creating a second person bitten. On the other hand, taking a photo of the snake is much more feasible, less dangerous, and generally is recommended. The three types of venomous snakes that cause the majority of injuries are vipers, kraits, and cobras. Knowledge of what species are present locally can be crucial in searching for adequate medicine [3].

Developing a robust system for identifying species of snakes from photographs and geographical information could significantly improve snakebite eco-epidemiological data and correct antivenom administration [2, 4]. In 2021, the fifth round of SnakeCLEF [5] challenge

*CLEF 2021 – Conference and Labs of the Evaluation Forum, September 21–24, 2021, Bucharest, Romania*
rego.borsodi@edu.bme.hu (R. Borsodi); pappd@tmit.bme.hu (D. Papp)
0000-0003-1513-3204 (R. Borsodi); 0000-0002-8814-2745 (D. Papp)

was announced by the LifeCLEF [6] campaign, where the goal was to create a system that is capable of automatically categorizing snakes on the species level. In this paper, we describe the solution of our team (BME-TMIT) in detail, which achieved rank 1 in the challenge. We first perform object detection on the images to separate the snake(s) from the background; then, the next step is to classify each detected snake. Finally, classification data is fused with location information using a likelihood weighting method. These steps are specified in Sections 4, 5, and 6, respectively. Section 2 briefly outlines the corresponding literature in the field; lastly, Section 7 summarizes our conclusion.

## 2. Related work

Common tasks in computer vision, and therefore in image-based snake species identification, include object detection and object classification. The former aims to locate the region of an image where the object of interest may appear [7], while the latter involves categorizing an input image into several predefined classes [8]. In the past decade, Convolutional Neural Networks (CNN) became the most popular approach for visual recognition due to their superior performance. Architectures of CNN that are often used for classification in practice are among others VGGNet [9], Inception [10], Residual Network (ResNet) [11], EfficientNet [12] and MobileNet [13].

Regarding object detection, numerous deep learning frameworks have been proposed in the literature, and they can be organized into two categories: (i) two-stage detectors and (ii) one-stage detectors. Two-stage detection models first generate region proposals, and then they are forwarded to a specific network for further classification. The most prominent two-stage detectors are Region-based CNN (R-CNN) [14] and its' successors: Fast R-CNN [15] , Faster R-CNN [16], Mask R-CNN [17]. Differently, there is no separate procedure for region proposal in one-stage detection models, as they are designed to predict the bounding boxes and the corresponding class probabilities at once [18]. Popular one-stage detectors include You Only Look Once (YOLO) [19, 20, 21, 22], Single Shot Detector (SSD) [23], RetinaNet [24], and EfficientDet [25].

Automated image-based snake species identification is challenging due to small inter-class variance, high intra-class variance, and a large number of categories (species) [26]. Furthermore, labeled snake image collections usually contain only several hundred images. 38 different taxonomically relevant features were manually identified to categorize six snake species in [27], while Amir et al. [28] used automated feature extraction to get texture features and distinguish 22 different snake species. Other researchers applied deep learning techniques. For example, Faster R-CNN and ResNet were used to classify nine different snake species occurring on the Galápagos Islands of Ecuador [29]; authors of [30] experimented with three different-sized CNN architectures; recently, Yang and Sinnott classified Australian snakes using deep networks [31]. In cases when only a small amount of labeled snake images are available, it could be beneficial to use a few-shot learning approach, such as the Siamese network [32], or the recently proposed Double-View Matching Network (DVMN) [33], although, the latter was tested on X-ray images. Abeysinghe et al. [34] performed single-shot learning by applying the Siamese network to categorize 84 snake species, where only 3 to 16 training images were available per species. On the other hand, Putra et al. [35] aimed to build a system to recognize the existing bite points on the snake bite image and then classified to venomous snake bite or non-venomous using Chain Code and K Nearest Neighbor (KNN) classification. Their bite mark recognition method achieved $65\%$ accuracy while distinguishing venomous from non-venomous bites was possible with $80\%$ accuracy.

For the SnakeCLEF 2020 challenge, 287 632 photographs were prepared that belong to 783 snake species and were taken in 145 countries [26]. The qualified teams used object detection on the images as a preprocessing step. Gokula Krishnan used an object detection model with ResNet-50 backbone [36], and team FHDO-BCSG utilized a Mask R-CNN model reaching 39.0 mAP (mean average precision) [37] on the COCO (Common Objects in Context) dataset[1].

The dataset was expanded to a size of 414 424 images in SnakeCLEF 2021, which is approximately a $44\%$ increase compared to the previous challange, while slightly lowering the number of snake species to 772; however, the photographs were taken in 188 countries. The full dataset was split into a training subset with 347 405 images, a validation subset with 38 601 images, and a test subset with 28 418 images. Each subset has the same class distribution, while the minimum number of training, validation, and test images per class is nine, one, and one, respectively. Furthermore, the test subset contains all 772 classes and observations from almost all the countries presented in training and validation sets. Similarly to the teams of SnakeCLEF 2020, we applied object detection and then object classification on the images. The main contributions of our work are (i) training the EfficientDet-D1 for accurate snake detection, (ii) using double thresholding to categorize the predicted bounding boxes, and (iii) the fusion of class membership vectors with the likelihood of location data.

## 3. Evaluation metrics

Various metrics were used to measure the performance of classification. The $F_1$ score is computed for the $i$-th species as:

$$F_1^i = \frac{2 \cdot \text{precision}_i \cdot \text{recall}_i}{\text{precision}_i + \text{recall}_i} \tag{1}$$

The macro $F_1$ is calculated by averaging the $F_1$ values over all the species ($N$ is the number of species):

$$\text{macro } F_1 = \sum_{i=1}^{N} \frac{F_1^i}{N} \tag{2}$$

Let $H_i$ be a set containing the indices of species that might appear in the $i$-th country (these data were available, provided by the organizers). Then the country-averaged $F_c$ for the given country is defined as:

$$F_c^i = \sum_{j \in H_i} \frac{F_1^j}{|H_i|} \tag{3}$$

[1]Datasets are available at: https://cocodataset.org/#download

**Table 1**
Comparison of object detection models

| Model | COCO | SnakeCLEF | | |
|---|---|---|---|---|
| | mAP | mAP | AP@0.5 | AP@0.75 |
| Mask R-CNN | 39.0 | N/A | N/A | N/A |
| SSD MobileNet v2 | 20.2 | 51.2 | 90.0 | 53.4 |
| EfficientDet-D1 | 38.4 | 56.9 | 91.0 | 64.4 |

The $F_1$ country score is the average of the $F_c$ values for all countries (let their number be $M$):

$$F_1 \text{ country} = \sum_{i=1}^{M} \frac{F_c^i}{M} \tag{4}$$

The primary metric in the challenge is the $F_1$ country score, and the secondary is the macro $F_1$ value. For evaluating the models, we also use classification accuracy defined as:

$$\text{Accuracy} = \frac{\text{\#correct predictions}}{\text{\#samples}} \tag{5}$$

Categorical cross entropy is used both as an evaluation metric and as the function to be optimized during training the network (loss). If the output of the network for a sample is $\mathbf{y}$ and the ground truth is $\hat{\mathbf{y}}$, categorical cross entropy is calculated as:

$$\Gamma(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_{i=1}^{N} \hat{\mathbf{y}}_\mathbf{i} \cdot \log \mathbf{y}_\mathbf{i} \tag{6}$$

When evaluating multiple samples, their losses are averaged.

## 4. Object detection

Object detection can improve snake classification accuracy as results from earlier rounds of the SnakeCLEF competition show. As mentioned in Section 2, top-scoring teams applied ResNet-50 and Mask R-CNN in round 4 (SnakeCLEF 2020). We conducted experiments with a lightweight SSD MobileNet v2 and a state-of-the-art EfficientDet-D1 model. More complex models (D2 to D7) might improve the results in the presence of sufficient training data; however, they are prone to overfitting. Table 1 shows the comparison of the described models. Both SSD and EfficientDet-D1 are one-stage networks, which generally run faster than the two-stage Mask R-CNN.

A subset of the train and val datasets consisting of $4700$ images was annotated with the help of the labelImg utility [38]. These samples were split in an $85\% - 15\%$ ratio to construct a training and a validation input for the object detection models. The models were initialized with weights pre-trained on the COCO 2017 dataset and then fine-tuned on the annotated part of the SnakeCLEF dataset using the Tensorflow Object Detection API. Both models were trained to

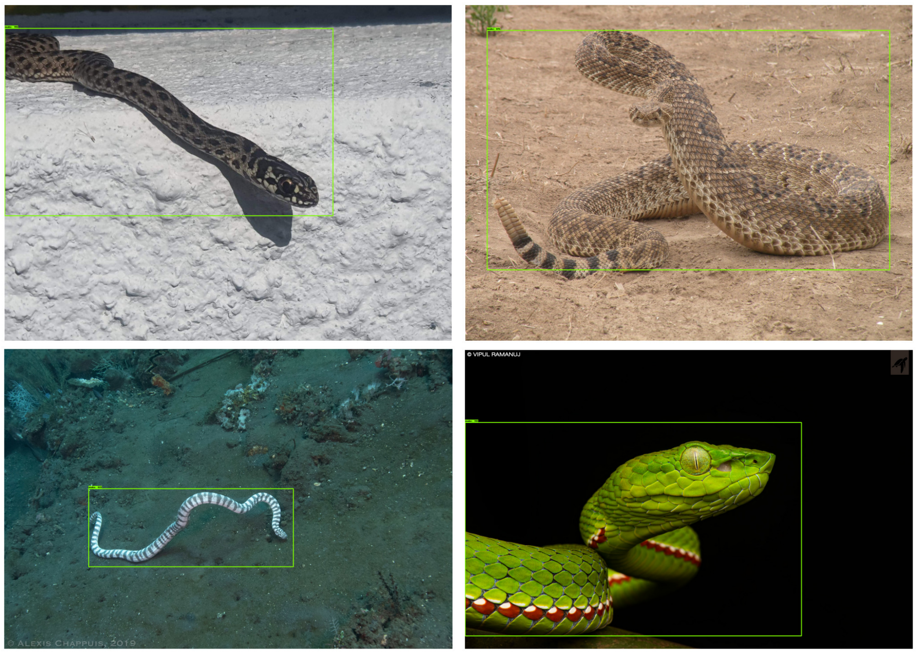

**Figure 1:** Snakes detected by the EfficientDet model. ©Lili, iNaturalist, CC-BY-NC; ©Gerson Herrera, iNaturalist, CC-BY-NC; ©oranglautan, iNaturalist, CC-BY-NC; ©Vipul Ramanuj, iNaturalist, CC-BY-NC

detect two classes (snake and background) and used with mostly the default settings. For data augmentation, only vertical flipping was used. The validation mAP's plateaued before reaching 30000 steps in both cases. The best results are shown in Table 1. The EfficientDet recorded higher mAP; however, the difference is not as big as in the case of the COCO dataset, possibly because there are only two classes. Some examples of detected snakes are shown in Figure 1.

As the EfficientDet-D1 performed better, only this model was put to use in image classification. After running the inference on a picture, the model's output consists of the coordinates of bounding boxes and the $p_0, p_1$ probabilities of the box containing background or snake, respectively. One image could contain multiple snakes (most likely of the same species) or none at all; therefore, our method of using the boxes (ordered descending by $p_1$) is the following:

- If for the first box $p_1 \geq p_{high}$, then we take this, and the next boxes having $p_1 \geq p_{high}$ (up to a maximum of three boxes, although there was not a single image in the dataset for which at least 3 boxes met this constraint for $p_{high} = 0.75$)[2].
- If for the first box $p_{low} \leq p_1 < p_{high}$ we take only the first box.
- If $p_1 < p_{low}$ for the first box, then we take the whole image. During training, we experimented with dropping such images, but it is not possible in the case of validation or testing.

[2] The reason for this is that one-phase object detection networks use *non-max suppression* to prevent overlapping boxes containing the same class in the result.

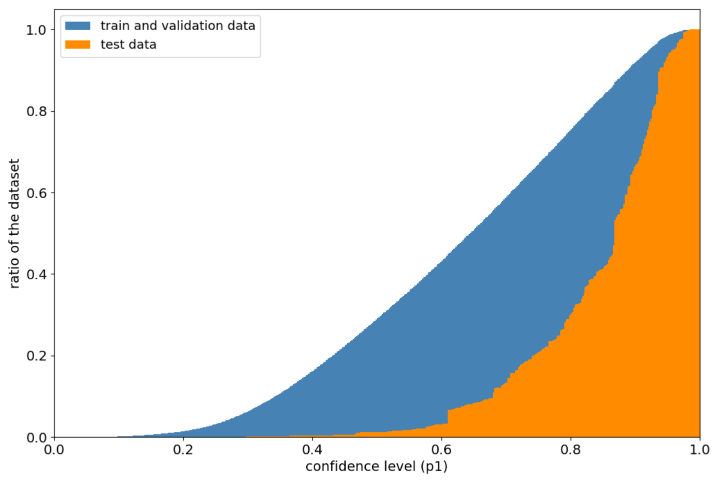

**Figure 2:** Cumulative distribution functions of the $p_1$ probabilities of the first boxes.

If multiple boxes are cropped from training or validation images, all of them are taken to the dataset with the ground truth label. In the case of testing, the classifier is run for all boxes, and the results are combined by first summing the prediction vectors and then normalizing it to a unit vector. More accurately, if the prediction vectors for the $n$ different boxes are: $\mathbf{y_1}$, $\mathbf{y_2}$, ..., $\mathbf{y_n}$, then the combined prediction is $\mathbf{y}$ (where $||\cdot||$ means the Euclidean norm):

$$\mathbf{y} = \frac{\sum_{i=1}^{n} \mathbf{y_i}}{||\sum_{i=1}^{n} \mathbf{y_i}||} \tag{7}$$

The EfficientDet detector was run on the whole dataset after training. Figure 2 shows the cumulative distribution functions (CDF) of the $p_1$ probabilities of the **first** bounding boxes on the test data and the combined training and validation datasets (trainval). The medians are $0.87$ and $0.629$, respectively. For the **second** bounding boxes, the medians are $0.068$ and $0.146$. Comparing these numbers with the assumption that the vast majority of the images contain a single snake, the network was more accurate on the test dataset.

Another conclusion is that the straightforward $p_{low} = 0.5$ choice (i.e., only keeping boxes for which the network predicts a higher probability of containing a snake than background) is not necessarily the best. For this value, $30\%$ of the training images would be preserved without cropping, as the CDF shows. In most cases, these images contained a snake that is harder to recognize or has a bounding box with an unusual aspect ratio (as the network was fitting boxes with aspect ratios of $0.5$, $1.0$, and $2.0$).

## 5. Classification

During classification, intermediate results were evaluated on the validation dataset, while scores on the test set are only available for those attempts, which were submitted for the challenge

**Table 2**
Parameters (with $\eta$ meaning the learning rate) and properties of the trained EfficientNet-B0 models

| | $\eta_{start}$ | $\eta_{end}$ | batch size | checkpoint | frozen layers | epochs |
|---|---|---|---|---|---|---|
| Baseline | $0.01$ | $1 \times 10^{-5}$ | 64 | ImageNet-1k | - | 15 |
| Min-train | $0.01$ | $1 \times 10^{-4}$ | 64 | ImageNet-1k | 5 epochs | 15 |
| Full-train 1 | $1 \times 10^{-4}$ | $8 \times 10^{-5}$ | 64 | Min-train | 4 epochs | 12 |
| Full-train 2 | $0.001$ | $1 \times 10^{-4}$ | 64 | Min-train | only BatchNorm | 15 |
| Full-train 3 | $0.003$ | $1 \times 10^{-4}$ | 128 | Min-train | only BatchNorm | 10 |
| Full-train 4 | $0.01$ | $3.5 \times 10^{-5}$ | 128 | Min-train | only BatchNorm | 14 |

and evaluated by the organizers. Accuracy, $F_1$ country, and macro $F_1$ scores were calculated in both cases (as defined in Section 3), while loss values are only available on the validation data.

## 5.1. Preprocessing

The first preprocessing step in the case of Full-train networks (see Tables 2 and 3) was running the EfficientDet object detector on the trainval dataset as described in Section 4, and saving the results in the TFRecord format. For validation and test datasets, $p_{low} = 0.2$ and $p_{high} = 0.75$ were used, while for training data, $p_{low} = 0.5$ was the default.

Before running the network on a batch of images, some preprocessing steps were needed. The following steps were executed[3] each time an image was read into memory (the images in the dataset were not modified/overwritten):

1. grayscale to RGB conversion, if needed
2. Rescaling to $224 * 224 * 3$ (the standard input format for EfficientNet-B0)

†3. Vertical and horizontal flip with probabilities of $p = 0.5$

†4. Brightness and contrast modification with $p = 0.2$

5. Float conversion. (scaling from $[0, 255]$ to $[0.0, 1.0]$ and normalization were not executed, as these are part of the EfficientNet model in `Keras` [39])

## 5.2. Models

The parameters used to train the different models are shown in Table 2 (note that EfficientNet-B0 networks were used in each case). The Baseline model, trained on a dataset of a reduced size, was provided by the organizers. Min-train was trained with the `Tensorflow` library using mostly the same parameters and the same dataset as the Baseline, without using object detection. The Full-train networks were trained on the complete training dataset with object detection included. The loss function was categorical cross entropy in all cases.

Adam optimizer was used for training all the models with a learning rate decay from $\eta_{start}$ to $\eta_{end}$. The $\beta_1 = 0.9$ and $\beta_2 = 0.999$ values were used as the exponential decay rate for the first and second moment estimates, respectively. The parameter *epochs* shows the number of

[3]Items marked with a † symbol were only executed in the phase of training but not during validation or testing.

**Table 3**
Evaluation of the EfficientNet-B0 models on the validation and test datasets

| | validation | | | | test | | |
|---|---|---|---|---|---|---|---|
| | accuracy | macro $F_1$ | $F_1$ country | loss | accuracy | macro $F_1$ | $F_1$ country |
| Baseline | 44.7% | 0.327 | - | 2.340 | - | - | - |
| Min-train | 42.8% | 0.294 | 0.276 | 3.132 | - | - | - |
| Full-train 1 | 69.5% | 0.512 | 0.513 | 1.577 | - | - | - |
| Full-train 2 | 70.0% | 0.512 | 0.513 | 1.530 | 85.86% | 0.738 | 0.773 |
| Full-train 3 | 70.4% | 0.515 | 0.496 | 1.253 | 92.13% | 0.786 | 0.789 |
| Full-train 4 | 68.3% | 0.498 | 0.488 | 1.344 | 85.55% | 0.737 | 0.752 |

epochs before the results on the validation data plateaued, and the training was terminated. Some models started the training process with frozen layers, leaving only the classification head trainable to prevent the big initial gradients from destroying the pre-trained weights. However, all Batch Normalization layers were kept frozen in all the models even after unfreezing the others. This is a common technique suggested by `Keras` developers, in order to prevent the pre-trained weights from being destroyed [39, 40], which is not preferable with a dataset of this size, due to the risk of overfitting.

As Table 3 shows, the Full-train networks performed quite similarly. Full-train 1 converged to the worst optimum, possibly due to the low initial learning rate and the frozen layers in the beginning. Interestingly, the $F_1$ scores were not inferior to the other networks on the validation data. This might be caused by some noise in the validation data, or another option is that the loss does not correlate well with the $F_1$ scores. However, the networks with lower loss generally predict higher probability for the correct species[4], thus, they are a better candidate for reweighting using location data, as described in Section 6.

**Submission 1** The first submission included the Full-train 2 network. The learning rate was $0.001$ in the first 6 epochs, then multiplied by $0.5$ or $0.75$ in alteration. During object detection, the parameters were $p_{low} = 0.5$ and $p_{high} = 0.75$, and images having a best bounding box under $p_{low}$ were preserved without cropping. Despite recording the highest $F_1$ country score on the validation data, the loss remained higher than in the next attempts.

**Submission 3** The Full-train 3 model was trained with a similar approach as in the case of Submission 1. A notable difference was that the batch size was increased to 128 and the initial learning rate to $0.003$. On the validation data, this resulted in a major difference in the loss values, and a top $92.13\%$ accuracy on the test data.

**Submission 4** The fourth submission was the model Full-train 4. In this attempt, $p_{low}$ was modified to $0.2$ on the training data, and images falling below this value during object detection (5008 altogether) were dropped. The learning rate schedule followed the same pattern as in

[4]Considering categorical cross entropy loss, the $\hat{\mathbf{y}}$ vector contains a single nonzero element referring to the correct class, where the value is 1. Substituting into the formula, the loss is $-\log \mathbf{y_i}$, if the ground truth class is $i$. As $\mathbf{y_i} \leq 1$, the $-\log \mathbf{y_i}$ loss value decreases, as the $\mathbf{y_i}$ prediction increases.

**Table 4**
Comparison of the submissions not using location data, evaluated on the validation and test datasets

| | validation | | | | test | | |
|---|---|---|---|---|---|---|---|
| | accuracy | macro $F_1$ | $F_1$ country | loss | accuracy | macro $F_1$ | $F_1$ country |
| Submission 1 | 70.0% | 0.512 | 0.513 | 1.530 | 85.86% | 0.738 | 0.773 |
| Submission 3 | 70.4% | 0.515 | 0.496 | 1.253 | 92.13% | 0.786 | 0.789 |
| Submission 4 | 68.3% | 0.498 | 0.488 | 1.344 | 85.55% | 0.737 | 0.752 |
| Submission 5 | 60.6% | 0.425 | 0.416 | 1.662 | 87.78% | 0.705 | 0.727 |

Submission 1, but started from a higher value of 0.01. The results were lower on the validation and the test set than those of the previous submission.

**Submission 5** This submission included the Full-train 3 network (as Submission 3), but during the evaluation of the test set, object detection was not used. This attempt performed the worst in all metrics (see Table 4), except accuracy on the test set. The difference from other submissions is higher on the validation set, which can be explained by the lower quality of the images, where the object detection might greatly improve the visibility of the snake.

## 6. Use of location data

The models described in Section 5.2 did not utilize the location data provided with the images. This information can be used to predict the class label independently, using frequentist statistics. Let $Y$ be the random variable of the image label (class) and $C$ of the country, then the probability that a sample belongs to class $i$ can be approximated from $C$ as:

$$P(Y = i \mid C = c) \approx \frac{\#\text{images in the training set from class } i \text{ and country } c}{\#\text{images in the training set from country } c} \tag{8}$$

If the prediction of the neural network for the $i$-th class is $y_i$ and the image is from country $c$, a new prediction can be created by multiplication with probabilities from the location data:

$$y_i^* = y_i \cdot P(Y = i \mid C = c) \tag{9}$$

Normalizing $y^*$ to unit length gives the new prediction vector. However, one could apply Bayes' Theorem to split the probabilities in the following way:

$$P(Y = i \mid C = c) = \frac{P(C = c \mid Y = i) \cdot P(Y = i)}{P(C = c)} \tag{10}$$

As $P(C = c)$ is only a normalizing constant (the same for all $i$), and the prediction vectors are normalized to unit length anyway, it can be omitted. The $P(Y = i)$ factor is the prior probability of the $i$-th class. It puts more weight on more frequent classes and less on the least frequent ones, which might not harm classification accuracy, but is clearly not preferable for macro $F_1$

**Table 5**
Evaluation of the predictions with location data for different $\varepsilon$ values

| | validation | | | | test | | |
|---|---|---|---|---|---|---|---|
| | accuracy | macro $F_1$ | $F_1$ country | loss | accuracy | macro $F_1$ | $F_1$ country |
| Full-train 3 | $70.40\%$ | $0.515$ | $0.496$ | $1.253$ | $92.13\%$ | $0.786$ | $0.789$ |
| $\varepsilon = 0$ | $75.90\%$ | $0.652$ | $0.680$ | $0.951$ | $94.39\%$ | $0.840$ | $0.878$ |
| $\varepsilon = 7 \times 10^{-6}$ | $75.94\%$ | $0.654$ | $0.685$ | $0.930$ | $94.82\%$ | $0.855$ | $0.903$ |
| $\varepsilon = 7 \times 10^{-4}$ | $75.66\%$ | $0.644$ | $0.669$ | $0.941$ | $94.94\%$ | $0.864$ | $0.901$ |

and country $F_1$ scores, which put equal weights on classes. Therefore, the $P(C = c \mid Y = i)$ *likelihood* remains, which is approximated as:

$$P(C = c \mid Y = i) \approx \frac{\#\text{images in the training set from class } i \text{ and country } c}{\#\text{images in the training set from class } i} \tag{11}$$

The more training images are present for a class, the more accurate the approximation is. However, multiplication by 0 is not preferred, as (i) a snake of a particular species might appear anywhere with a small probability (e.g., a captive snake), (ii) for classes with only a few training samples, there is a high chance that we do not have samples from each country where the species is native. Therefore, a small $\varepsilon$ constant is used:

$$y_i^* = y_i \cdot \max\{P(C = c \mid Y = i), \varepsilon\} \tag{12}$$

The method was tested for the prediction of the Full-train 3 network, as this model had the lowest loss on the validation dataset. The predictions were evaluated for three different $\varepsilon$ values: $\varepsilon = 0$ (submission 7), $\varepsilon = 7 \times 10^{-6}$ (submission 2) and $\varepsilon = 7 \times 10^{-4}$ (submission 6). The results are shown in Table 5. Adding location data to the model increased all the metrics to a great extent, affecting $F_1$ country the most, which saw an increase of $0.089$ on the test data.

On the test data, there was another major increase in the $F_1$ country score when the $\varepsilon$ was not set to 0, while the macro $F_1$ and accuracy scores also rose. The results did not differ much between $\varepsilon = 7 \times 10^{-4}$ and $\varepsilon = 7 \times 10^{-6}$; however, $7 \times 10^{-6}$ seems to be better in general.

## 7. Conclusion

We elaborated an automated snake species identification system that first applies object detection to separate the snake(s) from the background then incorporates visual information and location metadata into a classification algorithm to categorize the detected snakes. EfficientDet and EfficientNet were used for object detection and classification, respectively. Based on our experiments, we can conclude that object detection can positively influence snake identification; moreover, the inclusion of geographical data showed further significant improvement. Our best submission achieved an $F_1$ country score of $0.903$ and almost $95\%$ classification accuracy in the official evaluation of SnakeCLEF 2021.

## References

[1] W. H. O. (WHO), Snakebite envenoming - Key Facts 2021, 2021. URL: https://www.who.int/news-room/fact-sheets/detail/snakebite-envenoming.

[2] I. Bolon, A. M. Durso, S. Botero Mesa, N. Ray, G. Alcoba, F. Chappuis, R. Ruiz de Castañeda, Identifying the snake: First scoping review on practices of communities and healthcare providers confronted with snakebite across the world, PLOS ONE 15 (2020) e0229989. doi:https://doi.org/10.1371/journal.pone.0229989.

[3] A. Pathmeswaran, A. Kasturiratne, M. Fonseka, S. Nandasena, D. Lalloo, H. De Silva, Identifying the biting species in snakebite by clinical features: an epidemiological tool for community surveys, Transactions of the Royal Society of Tropical Medicine and Hygiene 100 (2006) 874–878. doi:https://doi.org/10.1016/j.trstmh.2005.10.003.

[4] R. R. de Castañeda, A. M. Durso, N. Ray, J. L. Fernández, D. J. Williams, G. Alcoba, F. Chappuis, M. Salathé, I. Bolon, Snakebite and snake identification: empowering neglected communities and health-care providers with ai, THe Lancet Digital Health 1 (2019) e202–e203. doi:https://doi.org/10.1016/s2589-7500(19)30086-x.

[5] L. Picek, A. M. Durso, R. Ruiz De Castañeda, I. Bolon, Overview of snakeclef 2021: Automatic snake species identification with country-level focus, in: Working Notes of CLEF 2021 - Conference and Labs of the Evaluation Forum, 2021.

[6] A. Joly, H. Goëau, S. Kahl, L. Picek, T. Lorieul, E. Cole, B. Deneu, M. Servajean, R. Ruiz De Castañeda, G. H. Bolon, Isabelle, R. Planqué, W.-P. Vellinga, A. Dorso, P. Bonnet, I. Eggel, H. Müller, Overview of lifeclef 2021: a system-oriented evaluation of automated species identification and species distribution prediction, in: Proceedings of the Twelfth International Conference of the CLEF Association (CLEF 2021), 2021.

[7] P. Druzhkov, V. Kustikova, A survey of deep learning methods and software tools for image classification and object detection, Pattern Recognition and Image Analysis 26 (2016) 9–15. doi:https://doi.org/10.1134/s1054661816010065.

[8] W. Rawat, Z. Wang, Deep convolutional neural networks for image classification: A comprehensive review, Neural computation 29 (2017) 2352–2449. doi:https://doi.org/10.1162/neco_a_00990.

[9] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014).

[10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9. doi:https://doi.org/10.1109/cvpr.2015.7298594.

[11] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778. doi:https://doi.org/10.1109/cvpr.2016.90.

[12] M. Tan, Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 6105–6114.

[13] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861 (2017).

[14] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580–587. doi:https://doi.org/10.1109/cvpr.2014.81.

[15] R. Girshick, Fast r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448. doi:https://doi.org/10.1109/iccv.2015.169.

[16] S. Ren, K. He, R. Girshick, J. Sun, Faster r-cnn: Towards real-time object detection with region proposal networks, arXiv preprint arXiv:1506.01497 (2015). doi:https://doi.org/10.1109/tpami.2016.2577031.

[17] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2961–2969. doi:https://doi.org/10.1109/iccv.2017.322.

[18] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, M. Pietikäinen, Deep learning for generic object detection: A survey, International journal of computer vision 128 (2020) 261–318.

[19] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788. doi:https://doi.org/10.1109/cvpr.2016.91.

[20] J. Redmon, A. Farhadi, Yolo9000: better, faster, stronger, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 7263–7271. doi:https://doi.org/10.1109/cvpr.2017.690.

[21] J. Redmon, A. Farhadi, Yolov3: An incremental improvement, arXiv preprint arXiv:1804.02767 (2018).

[22] A. Bochkovskiy, C.-Y. Wang, H.-Y. M. Liao, Yolov4: Optimal speed and accuracy of object detection, arXiv preprint arXiv:2004.10934 (2020).

[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, Ssd: Single shot multibox detector, in: European conference on computer vision, Springer, 2016, pp. 21–37. doi:https://doi.org/10.1007/978-3-319-46448-0_2.

[24] 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. doi:https://doi.org/10.1109/iccv.2017.324.

[25] M. Tan, R. Pang, Q. V. Le, Efficientdet: Scalable and efficient object detection, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 10781–10790. doi:https://doi.org/10.1109/cvpr42600.2020.01079.

[26] L. Picek, I. Bolon, A. M. Durso, R. R. de Castañeda, Overview of the SnakeCLEF 2020: Automatic Snake Species Identification Challenge, 2020.

[27] A. P. James, B. Mathews, S. Sugathan, D. K. Raveendran, Discriminative histogram taxonomy features for snake species identification, Human-Centric Computing and Information Sciences 4 (2014) 1–11. doi:https://doi.org/10.1186/s13673-014-0003-0.

[28] A. Amir, N. A. H. Zahri, N. Yaakob, R. B. Ahmad, Image classification for snake species using machine learning techniques, in: International Conference on Computational Intelligence in Information System, Springer, 2016, pp. 52–59.

[29] A. Patel, L. Cheung, N. Khatod, I. Matijosaitiene, A. Arteaga, J. W. Gilkey, Revealing the unknown: real-time recognition of galápagos snake species using deep learning, Animals

10 (2020) 806. doi:https://doi.org/10.3390/ani10050806.

[30] I. S. Abdurrazaq, S. Suyanto, D. Q. Utama, Image-based classification of snake species using convolutional neural network, in: 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), IEEE, 2019, pp. 97–102. doi:https://doi.org/10.1109/isriti48646.2019.9034633.

[31] Z. Yang, R. Sinnott, Snake detection and classification using deep learning, in: Proceedings of the 54th Hawaii International Conference on System Sciences, 2021, p. 1212. doi:https://doi.org/10.24251/hicss.2021.148.

[32] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, R. Shah, Signature verification using a" siamese" time delay neural network, Advances in neural information processing systems 6 (1993) 737–744. doi:https://doi.org/10.1142/s0218001493000339.

[33] G. Szűcs, M. Németh, Double-view matching network for few-shot learning to classify covid-19 in x-ray images, INFOCOMMUNICATIONS JOURNAL 13 (2021) 26–34. doi:https://doi.org/10.36244/icj.2021.1.4.

[34] C. Abeysinghe, A. Welivita, I. Perera, Snake image classification using siamese networks, in: Proceedings of the 2019 3rd International Conference on Graphics and Signal Processing, 2019, pp. 8–12. doi:https://doi.org/10.1145/3338472.3338476.

[35] R. M. Putra, D. Q. Utama, et al., Snake bite classification using chain code and k nearest neighbour, in: Journal of Physics: Conference Series, volume 1192, IOP Publishing, 2019, p. 012015. doi:https://doi.org/10.1088/1742-6596/1192/1/012015.

[36] GokulaKrishnan, Diving into deep learning — part 3 — a deep learning practitioner's attempt to build state of the art snake-species image classifier, 2019. URL: https://medium.com/@Stormblessed/diving-into-deep-learning-part-3-a-deep-learning-practitioners-attempt-to-build-state-of-the-2460292bcfb.

[37] L. Bloch, A. Boketta, C. Keibel, E. Mense, A. Michailutschenko, O. Pelka, J. Rückert, L. Willemeit, C. Friedrich, Combination of image and location information for snake species identification using object detection and efficientnets, In: CLEF working notes 2020, CLEF: Conference and Labs of the Evaluation Forum (2020).

[38] Tzutalin, Labelimg, 2015. URL: https://github.com/tzutalin/labelImg.

[39] Keras documentation: Image classification via fine-tuning with EfficientNet, 2020. URL: https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/.

[40] Keras documentation: Transfer learning & fine-tuning, 2020. URL: https://keras.io/guides/transfer_learning/.