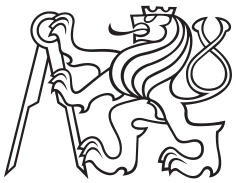
Master Thesis



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Fine-grained Visual Recognition with Side Information

Rail Chamidullin

Supervisor: prof. Ing. Jiří Matas, Ph.D.
Field of study: Computer Vision
January 2022

MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Chamidullin Rail** Personal ID number: **435602**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Open Informatics**

Specialisation: **Computer Vision and Image Processing**

II. Master's thesis details

Master's thesis title in English:

Fine-grained Visual Recognition with Side Information

Master's thesis title in Czech:

Vizuální rozpoznávání do mnoha tříd s vedlejší informací

Guidelines:

The thesis will focus on the problem of fine-grained recognition where side information is available, such as recognition of snake or fungi species for which rich annotated data are available. The side information is e.g. location and the time of observation, or even habitat.
1. Review current state-of-the-art in fine-grained recognition and select a suitable deep net method as a starting point.
2. Consider the problem of the use of side information, and propose a method for exploiting it.
3. Select a suitable dataset and evaluate your method exploiting side information using standard metrics.
4. Discuss the appropriateness of the metrics, and the optimization criteria of the deep net, for the intended applications and whether they align well with the costs of incorrect decision in the applications. If issues are found, propose and test improvement.
5. (Optional) Propose and evaluate both test time and training time techniques such as data augmentation (training), classifier combination (test) to improve overall performance of the system.

Bibliography / sources:

[1] Milan Šulc, "Fine-grained Recognition of Plants and Fungi from Images", PhD thesis, CTU Prague, 2021.
[2] Lukáš Picek, Milan Šulc et al., „Danish Fungi 2020 - Not Just Another Image Recognition Dataset", 2021.
[3] Lukáš Picek, Andrew Durso et al., „Overview of SnakeCLEF 2021: Automatic Snake Species Identification with Country-Level Focus", Working Notes of CLEF 2021, 2021.

Name and workplace of master's thesis supervisor:

prof. Ing. Jiří Matas, Ph.D., Visual Recognition Group, FEE

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **16.09.2021** Deadline for master's thesis submission: **04.01.2022**

Assignment valid until: **19.02.2023**

prof. Ing. Jiří Matas, Ph.D.	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor's signature	Head of department's signature	Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

Acknowledgements

I would like to thank my research supervisor Jiří Matas for the professional guidance and supervision of this thesis. I am grateful to Milan Šulc and Lukáš Picek for support in writing a conference paper. Special thanks to Lukáš Picek for suggestions and valuable feedback on training neural networks.

Equally great thanks belong to my family for their support during my entire studies.

The access to the computational infrastructure of the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics" is also gratefully acknowledged.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodological instructions for observing the ethical principles in the preparation of university theses.

Prague, 04.01.2022 .....

Rail Chamidullin

Abstract

The thesis presents a method for fine-grained visual snake and fungi species recognition with side information. The proposed method is based on state-of-the-art deep neural networks for classification: Convolutional Neural Networks and Vision Transformers. We achieve performance improvements by: (1) adopting loss functions proposed to address the class imbalance; (2) adjusting predictions by prior probabilities of side information like location and time of observation; (3) applying a weakly supervised method to localize snakes and fungi in images and crop the images based on the detected regions to enrich the training data. Finally, we demonstrate the use of the proposed method to decide on medical response to snakebites.

Keywords: Fine-grained visual classification, Snake species identification, Fungi species identification, Side Information inclusion, Machine Learning, Convolutional Neural Networks, Vision Transformers

Supervisor: prof. Ing. Jiří Matas, Ph.D.
Karlovo náměstí 13,
121 35 Praha 2

Abstrakt

Práce se zabývá vizuálním rozpoznávání druhů hadů a hub s vedlejší informací do mnoha tříd. V práci je navržena metoda založená na state-of-the-art hlubokých neuronových sítích pro klasifikaci, tj. konvolučních neuronových sítí a tzv. Vision Transformers. Zlepšení výsledku dosahujeme: (1) zavedením ztrátových funkcí navržených pro situace s nevyváženými třídami; (2) úpravou predikcí podle apriorní pravděpodobnosti vedlejší informace, jako je místo a čas pozorování; (3) použitím metody učení se slabým učitelem k lokalizaci hadů a hub na snímcích a oříznutí snímků na základě detekovaných oblastí pro obohacení trénovacích dat. V závěru demonstrujeme použití navržené metody pro rozhodnutí o postupu lečby hadího uštknutí.

Klíčová slova: Vizuální rozpoznávání do mnoha tříd, Identifikace druhů hadů, Identifikace druhů hub, Zahrnutí vedlejší informace, Strojové učení, Konvoluční neuronové sítě, Vision Transformers

Překlad názvu: Vizuální rozpoznávání do mnoha tříd s vedlejší informací

Contents

Figures

Tables

Chapter 1

Introduction

The fine-grained visual recognition task focuses on differentiating between hard-to-distinguish classes. A common example is animal or plant species identification, where the goal is to assign a species name to an observed specimen. The task is challenging due to high intra-class variability, small inter-class differences, and a large number of classes [1, 2]. This means the objects from different classes are visually similar and have a large variability within the same class. In species recognition, the high intra-class and low inter-class variation in appearance may depend on age, sex, or geographical location [1], see examples of snake species in Figure 1.1.

Traditional species identification relies on guidebooks with identification key [1]. The decision procedure is based on identification rules defined by domain experts and requires users to answer questions about discriminative features of a specimen that lead to a candidate species. The identification can be lengthy and difficult even for the experts. Modern approaches simplify the recognition process for the users using computer vision algorithms. The automated systems require large volumes of annotated data to learn discriminative features on the species appearance and classify species on images without annotations.

Citizen science projects provide large volumes of annotated data. Users, using specialized mobile and web apps, gather image observations of the specimen, and other users like citizen-scientists and domain experts provide

(a) : A snake species (Coluber constrictor) with high intra-class variability.

(b) : Two snake species (Crotalus oreganus and Crotalus scutulatus) with a small inter-class difference.

Figure 1.1: Images demonstrating the high intra-class and low inter-class variation in snake species appearance.

annotations. The image observations often include side information about location and the time of observation, or even habitat.

It has been reported that biodiversity data collection has a taxonomic bias [3] – the fact that some taxa are more investigated while others are ignored. Troudet et al. [3] showed in their study that the taxonomic bias strongly correlates with societal preferences of charismatic species and concluded that: birds, mammals, flowering and seed plants are over-represented in the biodiversity data; the class of reptiles, which include turtles, crocodiles, snakes, and lizards is relatively well represented; while fungi or insects are under-represented. In recent years, the citizen science projects like iNaturalist[1], HerpMapper[2], and Atlas of Danish Fungi[3] [4] helped to increase annotated data of under-represented taxa, including snake and fungi species.

Side information can improve image-based species recognition. For example, having a geographical location of observation narrows down the possible correct identifications of snake species [5]. The fungi species presence depends on a substrate on which fungi live, seasonality, or even (but rarely) the time in a day [6].

This thesis aims to study the problem and develop a computer vision method for fine-grained species recognition based on images and evaluate the effects of the use of side information. We focus on the identification of snake and fungi species where the annotated data with side information are available. Both tasks are non-trivial due to the high diversity: snakes are classified into more than 3 900 species [7], and fungi more than 2 000 species [8]. We base our study on SnakeCLEF 2021 [5] and Danish Fungi 2020 [6] datasets. During our work, we published part of the results of the thesis in a working note paper [9] for SnakeCLEF 2021 challenge.

1.1 Motivation

Snakebites are a global health problem that affects roughly 5.4 million people a year in developing countries, disabling 400 000 and killing 81 000–138 000 people by snakebite envenoming [10]. Taxonomic knowledge about snakes is crucial in diagnosis and medical response to snakebites. Accurate identification of the snake species helps in decision making whether to administer antivenom and which type of antivenom when treating the snakebite victims. Moreover, antivenoms are effective against a limited number of venomous snakes and should not be used to treat bites from non-venomous snakes because of side effects such as allergic reactions [10].

Fungi have positive effects on global health. Humans have been interested in fungi across different cultures for thousands of years because of their characteristics. Fungi are popular foods because they provide essential nutrients like proteins, vitamins, and fibre and are low in calories, carbohydrates, fat, and sodium. Moreover, fungi have beneficial medical effects: enhancing the

[1] http://www.inaturalist.org/
[2] https://www.herpmapper.org/
[3] https://svampe.databasen.org/en/

immune system, lowering cholesterol, and having antibacterial and antitumoral properties; they are used for the prevention or treatment of diseases like Parkinson, Alzheimer, hypertension, and high risk of stroke. Out of 2 000 fungi species, only a few are commercially cultivated, while wild mushrooms are becoming more important for their pharmacological and therapeutical uses [8]. Recognition of fungi species "in the wild" is vital for biodiversity, conservation, and increasing knowledge about the distribution and ecology of the fungi.

From the machine learning standpoint, fine-grained visual recognition presents an interesting challenge. Designing a recognition system requires collecting and annotating large amounts of annotated data, choosing appropriate evaluation metrics, and selecting a classification algorithm. Early classification approaches used two independent steps: feature extraction and classification [2]. Current state-of-the-art methods rely on end-to-end deep neural networks; the main architecture approaches for visual recognition are Convolutional Neural Networks (CNN) and attention-based Visual Transformers. The networks are optimized using back-propagation; the training procedure can include different techniques like data augmentation and model regularization. Classifying based on images may not be sufficient due to small inter-class differences, and side information inclusion can improve the performance. Different approaches are applied to adjust predictions depending on side information. The deep neural networks are often overconfident [1] and require classifier calibration [11] to correctly adjust predictions.

1.2 Problem Description

We formulate the visual species recognition task as a single-label classification on a closed set of C classes where each class denotes one species. Each recognition task is based on a single taxonomic group, e.g., snakes or fungi. Each observation belongs exactly to one class, and we do not consider out-of-class images depicting different objects or taxa. Technically, the goal is to train a classifier given training samples $\{(o_1, c_1), (o_2, c_2), ..., (o_N, c_N)\}$ of observations o_i labeled with corresponding class $c_i \in \{1, 2, ..., C\}$ to predict the unknown class c_j for new observations o_j. Observations o_i consist of image and side information like location and time. We evaluate the performance of classifiers using standard metrics, Top-1 and Top-3 accuracy, and a metric suitable for datasets with an imbalanced class distribution, macro averaged F_1 score.

1.3 Related Work

Automatic snake species recognition. Large-scale snake species identification has a short history because of the limited availability of annotated image datasets. First introduced in 2020, the Automatic Snake Species Identification Challenge at Conference of Labs of the Evaluation Forum (CLEF),

called SnakeCLEF, was designed to provide an evaluation platform that can help track the performance of end-to-end AI-driven snake species recognition systems [5, 12]. The SnakeCLEF 2021 challenge [5] provided a training dataset with 386 006 images of 772 snake species taken in 188 countries and a competition test set with 23 673 images without publicly available labels. The dataset additionally includes side information about the geographical location of observation and a country-species presence mapping. The majority of participating teams proposed recognition methods based on deep neural networks.

Bloch et al. utilized EfficientNet [13] and Vision Transformer [14] networks. The authors increased performance by adjusting the classifier predictions by prior probabilities of geographical location estimated as relative frequencies on training set. To clean the training dataset from noisy samples, the authors utilized an ImageNet-1k [15] pre-trained ResNet-50 [16] network and discarded images not classified as snake and reptile classes.

Chamidullin et al. [9] used deep residual CNNs: ResNet [16], ResNeXt [17], and ResNeSt [18]. The authors experimented with different loss functions, including standard cross-entropy, weighted cross-entropy and soft F1 loss. The performance improvements were achieved by: utilizing country-species presence mapping to drop predictions of the species not occurring in the country of the given image; and ensembling classifiers by majority voting strategy.

Borsodi et al., first detected regions where snakes occur using an EfficientDet [19] object detector, fine-tuned on manually annotated images, then classified the snake species in the regions using EfficientNet [13] and adjusted the predictions using prior probabilities of geographical location.

Teams [20, 21] that experimented with relatively old CNNs architectures (GoogLeNet [22], VGG16 [23], and ResNet-18 [16]) or different classifiers like gradient boosting classifier [24] achieved inferior results.

Automatic fungi species recognition. Before the existence of large-scale image datasets for fungi species classification, Ottom et al. [25] experimented with different machine learning algorithms to classify images with fungi into poisonous and non-poisonous classes. The authors collected a dataset of 380 images from Mushroom World[4] database, then extracted handcrafted features from images and trained classifiers on them: feedforward neural network, support vector machine, k-nearest neighbours, and decision tree.

In 2018, the FGVCx Fungi Classification Challenge provided a training dataset with 89 760 images of 1 394 fungi species and a competition test set with 9 758 images without publicly available labels. The winning submission, proposed by Šulc et al. [26], was based on an ensemble of Inception-v4 and Inception-ResNet-v2 [27] networks. The authors achieved performance improvements by: adjusting predictions by class priors to address a substantial change of class priors between the training and validation sets; applying image augmentation like central and corner crop at test time, and combining

[4] `https://www.mushroom.world/`

predictions from all augmentations by averaging or mode of the predicted classes. The authors provide an image-based fungi identification service for the Atlas of Danish Fungi [4]. The back-end server accepts an image query and responds with the predicted species probabilities created by CNN. Users upload images and obtain results through mobile and web apps. The observations and the proposed species identifications are uploaded into the Danish Fungal Atlas database, the community then verifies the proposed identifications.

1.4 Structure of the Thesis

The rest of the work is structured as follows. Chapter 2 introduces the datasets and evaluation methodology. Chapter 3 describes the proposed method for recognition of snake and fungi species, including adopted deep net architectures, the optimization procedure, loss functions, side information inclusion, and informed augmentation. Chapter 4 covers all experiments on the proposed method. Chapter 5 demonstrate an example of using the proposed method on the real-world application to decide on snakebite response. Finally, the results are summarized and concluded in Chapter 6.

Chapter 2

Datasets and Evaluation Methodology

The chapter introduces the fine-grained datasets used in this thesis. For snake species recognition, we use a dataset[1] from the SnakeCLEF 2021 challenge [5] — a part of LifeCLEF 2021 workshop [28]; and Danish Fungi 2020 [6] dataset[2] for fungi species recognition. Both datasets are fine-grained with an imbalanced long-tailed class distribution, illustrated in Figure 2.1, and contain image samples with side information like location of observation. The chapter also describes the evaluation methodology suitable for class unbalanced datasets.

2.1 SnakeCLEF 2021 Dataset

The SnakeCLEF training dataset covers 772 snake species and contains 386 006 annotated images, see examples in Figure 2.2. The data come from

[1] Available at: `https://www.aicrowd.com/challenges/snakeclef2021-snake-species-identification-challenge`; visited on 3.1.2022

[2] Available at: `https://sites.google.com/view/danish-fungi-dataset`; visited on 3.1.2022

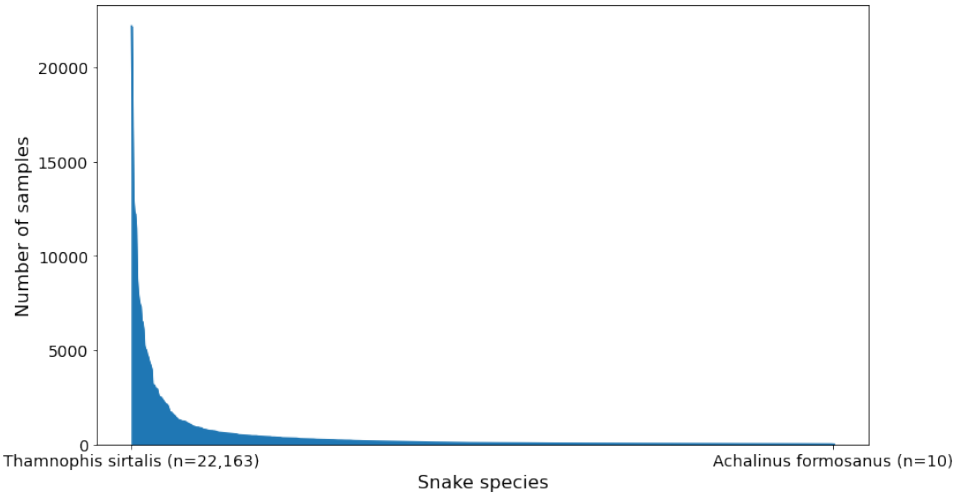


Figure 2.1: Class distribution of the SnakeCLEF dataset.

Figure 2.2: Image examples from the SnakeCLEF dataset (resized to 224×224).

three different sources. The majority of images are from iNaturalist[3] (277 025; 71.8%) and HerpMapper[4] (58 351; 15.1%); their labels are confirmed by human annotators. The subset from Flickr[5] is the smallest, with 50 630 (13.1%) web-scraped images that contain noisy data. The dataset is unbalanced: more than 22 000 images represent the most frequent species, while the least frequent species have only 10 images, see Figure 2.1.

The challenge organizers suggested a reduced training set (SnakeCLEF-Reduced) made of 70 208 samples from INaturalist and HerpMapper that cover 768 species. After the challenge, the organizers provided annotated test set with 23 673 samples.

2.1.1 Side Information

In addition to the images, the SnakeCLEF dataset contains side information about the country origin of the observation. The training dataset includes images from 188 countries. Images from North America (258 732; 67.0%) dominate the dataset. For 51 061 (13.2%) images, information about the geographical location is missing. Table 2.1 shows the distribution of images in geographical regions. The least represented species are often found in regions like Central and South America, South Africa, and Australia.

The dataset comes with additional information about each snake species: country-species presence mapping – a list of countries where the specific species occurs; and medical importance of a species – information whether the specific species is venomous or non-venomous.

[3] `http://www.inaturalist.org/`
[4] `https://www.herpmapper.org/`
[5] `https://www.flickr.com/`

Region	Number of images	Percentage of images
North America	258 732	67.0%
Europe	18 689	4.8%
Central America	17 403	4.5%
Asia	16 518	4.3%
South America	12 735	3.3%
Africa	6 017	1.6%
Australia	4 313	1.1%
Oceania	538	0.1%
unknown	51 061	13.2%

Table 2.1: Geographical distribution of SnakeCLEF 2021 images.

2.1.2 Data Preparation

We made several adjustments, described in this section, to the SnakeCLEF training, SnakeCLEF-Reduced training, and test datasets. The updated files are available online[6].

Dealing with Noisy Data

The Flickr subset contains noisy data – out-of-class images that capture different animal species or objects instead of snakes. We employ ImageNet-1k [15] pre-trained ResNet-101 [16] to detect the noisy data as follows: (1) the network classifies each image into ImageNet-1k classes; and (2) we discard images that do not belong to reptile classes[7]. In total, 8 778 (17.3%) Flickr images were discarded; and 41 852 (82.7%) were kept.

We statistically verify the number of noisy data in the dataset. We use the Student's t-distribution with $n = 20$ samples and $n - 1$ degrees of freedom. Each sample denotes the percentage of non-relevant images in a set of randomly selected 100 images. We estimate that the percentage of non-relevant images is 10.6 ± 0.1, with a 95% confidence interval. Based on that, we assume approximately 6.7% images, which were discarded, are false positives. During exploration of the relevant images, we did not discover any out-of-class images and assume the number of false negatives is very low.

After removing non-relevant records, the dataset size became 377 228. We discovered that 2 species were not represented in any image; thus, the dataset covers only 770 snake species.

Reduced and Test Sets Preparation

The SnakeCLEF-Reduced training set and the test set do not cover all 770 classes. We added 232 images to the reduced training set to cover all classes with at least 9 images per species, and 2 555 images to the test set with at

[6] `https://github.com/chamidullinr/fine-grained-visual-recognition`

[7] As reptile classes we consider ImageNet-1k classes containing words: snake, cobra, viper, mamba, boa, lizard, and alligator.

Figure 2.3: Image examples from the Danish Fungi dataset (resized to 224×224).

least 1 image per species. The added images were taken from the SnakeCLEF training set. Our new SnakeCLEF-Reduced dataset contains 70 440 images and the test set 26 227 images.

2.2 Danish Fungi 2020 Dataset

The Danish Fungi 2020 (DF20) dataset [6] contains annotated images from the Atlas of Danish Fungi [4, 29], see examples in Figure 2.3. The dataset covers 1 604 species and contains 295 938 images split into training and test sets with 266 344 and 29 594 images. The training set has more than 1 700 images for the most frequent species while only 28 images for the least frequent species.

The observations originate from 30 countries and include samples from all seasons. Majority of images were recorded in Denmark (263 040; 99.2%) and Sweden (1 151; 0.4%). The remaining countries have below 150 observations. Figure 2.4 shows geographical distribution of observations in Europe. The majority of images were taken between 2017 - 2020 (195 114; 73.3%).

For the fast training and reduced hardware requirements, the authors suggest a hand-picked mini subset (DF20M) of 36 393 images covering 182 species from 6 genera. The subset is split into 32 753 training and 3 640 test samples.

2.2.1 Side Information

The dataset contains rich metadata, including information related to the environment, place, and time of observation. In particular, each record comes with the following variables:

- **Habitat** – The environment where the specimen was observed containing 32 values like deciduous, coniferous, or mixed woodland. Appendix A.1 contains the full list of habitat values.

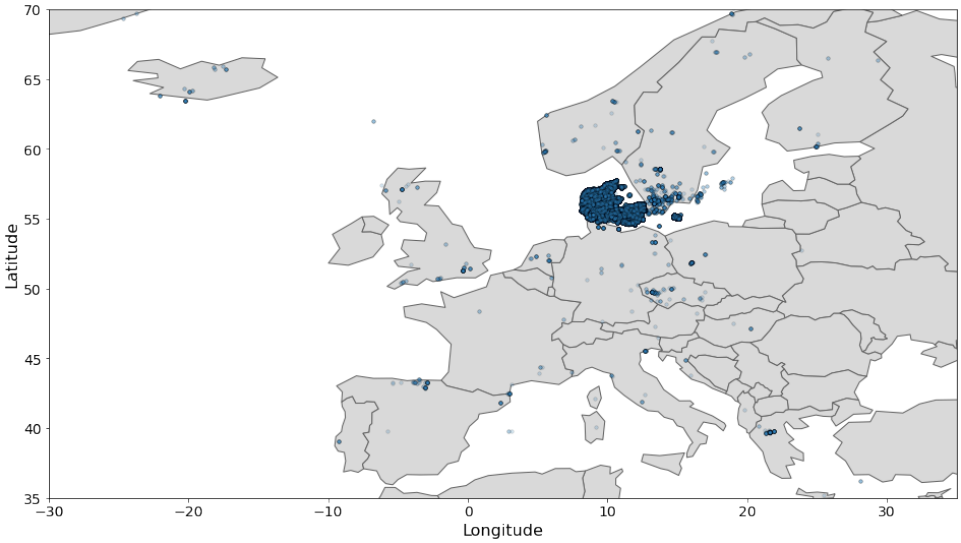


Figure 2.4: Geographical distribution of observations in DF20 focused on Europe.

- **Substrate** – The natural substance on which the specimen lives containing 32 values like as wood, soil, or stone. Appendix A.2 contains the full list of substrate values.
- Geographical location categorized into: 30 **countries**; 112 **country regions**; 311 **district regions**; 8 862 small **locality** regions like part of a city or a specific forest; and **GPS coordinates**.
- Timestamp with **year**, **month**, and **day**.
- Full taxonomy labels categorized into: 3 **kingdoms**; 5 **phyla**; 23 **classes**; 66 **orders**; 190 **families**; 566 **genera**; 1 578 **species**; and 1 604 **scientific names** of species.

Small portion of the records have missing metadata values: Habitat (975; 0.37%); Substrate (1 368; 0.51%); Locality (71, 0.03%); and time attributes (16, 0.01%).

2.2.2 Side Information Analysis

As initial experiment, we analyze relationships between species (scientific name) and side information like habitat, substrate, geographical location (country, country region, district region, locality), and time (month).

We first apply the non-parametric Chi-square (χ^2) test to test categorical variables' independence: the null hypothesis states variables are independent; the alternate hypothesis states the variables depend on each other. We set significance level $\alpha = 0.01$ and reject null-hypothesis if p-value is lower than α. Table 2.2 lists side information variables with estimated p-values. All p-values are lower than α therefore, we reject null-hypothesis and assume each metadata variable affects the species probability.

Next, we employ a decision tree as a weak classifier to estimate the effect of each side information variable on the species. The species decision tree

Variable	chi^2 p-value	DF20 Top-1	DF20 Top-3	DF20M Top-1	DF20M Top-3
Dummy Baseline	×	0.6	1.7	3.5	9.0
Month	<0.001	+0.5	+1.5	+2.2	+5.3
Habitat	<0.001	+0.9	+2.4	+2.7	+5.1
Substrate	<0.001	+2.0	+4.3	+3.7	+7.0
Country	<0.001	+0.0	+0.0	+0.0	+0.1
Country region	<0.001	+0.1	+0.2	+0.0	+0.2
Disctict region	<0.001	+0.6	+1.3	+1.1	+2.2
Locality	<0.001	+1.4	+2.8	+3.2	+4.4

Table 2.2: Relationship analysis between the side information and the species.

classifier is based on a single one-hot encoded independent variable. We evaluate the decision tree using stratified k-fold cross-validation[8] with $k = 5$ and calculate average classification scores: Top-1 and Top-3 accuracy, described in Section 2.3. We compare the classification results with a "dummy" classifier that always predicts the most frequent class and estimates posteriors as the class priors. The dummy classifier achieves 0.6% Top-1 and 1.8% Top-3 accuracy on DF20 and 3.5% Top-1 and 9.0% Top-3 accuracy on DF20M. Table 2.2 shows the change in accuracy of the decision tree when compared to the scores of the dummy classifier. In DF20 and DF20M, the highest effect on the species has a substrate followed by locality, habitat, and month. Country region and district region have a lower effect than locality; the country has no effect given the majority of the images were recorded in Denmark.

Note that some Danish Fungi images belong to the same observation. Meaning, multiple images may capture a single mushroom from different angles. Such samples have the same metadata, and therefore, for this experiment, we remove duplicates from the dataset to avoid over-fitting.

2.3 Evaluation Methodology

The commonly used classification metrics are Top-1 and Top-3 accuracy. In general, Top-k accuracy is computed as a fraction of correctly classified records where a classification is considered correct if the ground-truth label appears in the top k predictions. The accuracy is biased by class frequencies, meaning the majority classes get higher importance over the minority. It is a disadvantage in the fine-grained classification problems with long-tailed class distributions where the training procedure can ignore the least frequent classes. The F_1 score measure is introduced to overcome this.

The F_{1c} score for each class $c \in \{1, 2, ..., C\}$ is computed as a harmonic

[8]Stratified k-fold cross-validation splits the dataset into k consecutive folds. Each fold is used once as a test set, and the remaining folds form the training set. The stratified folds are made by preserving the percentage of samples for each class [30, 31].

mean of precision (p_c) and recall (r_c):

$$p_c = \frac{\mathrm{tp}_c}{\mathrm{tp}_c + \mathrm{fp}_c}, \qquad r_c = \frac{\mathrm{tp}_c}{\mathrm{tp}_c + \mathrm{fn}_c}, \qquad F_{1c} = \frac{2p_c r_c}{p_c + r_c}. \tag{2.1}$$

The true positives (tp_c) represent a number of correctly classified predictions of the class c; the false positives (fp_c) a number of predictions incorrectly classified instead of the class c; the false negatives (fn_c) a number of predictions incorrectly classified as the class c. The macro averaged F_1 score is computed to obtain a single scalar value by averaging F_{1c} scores of all classes:

$$F_1 = \frac{1}{C} \sum_{c=1}^{C} F_{1c}. \tag{2.2}$$

The weighting of each class can be added to accuracy and F_1 score. For example, in species recognition, the medical importance of species can be reflected by class weighting. Additionally, the usage of F_1 score allows setting different cost values for the false positive and the false negative errors. For example, it can be used in snake recognition, where misclassifying a venomous species for the non-venomous is a more significant issue than the opposite.

Chapter 3

Proposed Method

The chapter describes the proposed method for fine-grained species recognition with side information. We use image classifiers, namely deep neural networks, described in Section 3.1. The classifiers are pre-trained on a large corpus of labelled images called ImageNet-1k [15], and we fine-tune them on our downstream tasks: snake and fungi species recognition. For training the classifiers, we consider different objective functions from Section 3.2. To improve classification performance, we include side information, covered in Section 3.3: (1) the classifier produces image-based predictions; (2) we represent side information as a prior probability; (3) we calibrate image-based predictions and combine them with the side information priors. We employ a weakly supervised localization technique to detect and crop regions containing snakes or fungi in our images, and we train the classifiers on the cropped images to improve the classification scores, see Sections 3.4. The experiments on the proposed method are covered in the following chapter, Chapter 4.

3.1 Deep Neural Networks

We utilize state-of-the-art deep neural networks for classification. In particular, we apply two different architecture approaches: Convolutional Neural Networks (CNN) and Vision Transformers.

3.1.1 Convolutional Neural Networks

EfficientNet. The CNNs have been the main deep net design for image interpretation tasks like classification, object detection, and semantic segmentation. One of the most popular architecture families is EfficientNet [13], optimized for parameter efficiency. Tan et al. identified that balancing network depth, width, and resolution leads to a better performance and applied neural architecture search to design a baseline network: EfficientNet-B0. The baseline is then scaled up to obtain larger networks: EfficientNet-B1-B7. The main building blocks are based on the Inverted Bottleneck Residual Blocks (MBConv) of MobileNetV2 [32] and Squeeze-and-excitation Blocks [33].

Noisy Student. Xie et al. improved the performance of EfficientNet by introducing a semi-supervised teacher-student learning approach, called

Noisy Student Training [34], which includes additional unlabeled training data. The approach has three main steps: (1) a teacher network is trained on the labelled dataset; (2) the teacher then generates pseudo-labels on the unlabeled dataset; and (3) a student network, which is equal or larger than the teacher, is trained on the combination of labelled and pseudo-labelled datasets. The algorithm is iterated a few times by treating the student as a teacher and training a new student. Xie et al. found that three iterations worked the best in their experiments. The training of the student adds noise using regularization methods like RandAugment [35] data augmentation, dropout [36], and stochastic depth [37]. Each EfficientNet variant has an equivalent Noisy Student pre-trained variant marked as NoisyStudent-B0-B7.

EfficientNetV2. Tan et al. later proposed EfficientNetV2 [38] which improves training speed and the parameter efficiency. In the early MBConv layers, the new architecture replaces depthwise convolution – a type of convolution that applies a single convolutional filter for each input – with a regular convolution applied over multiple input channels. The network is trained using progressive learning with adaptive regularization: the training starts with small image size and a weak regularization and then gradually increases the image size and adds a stronger regularization. Regularization methods include dropout [36], data augmentation RandAugment [35], and Mixup [39]. The authors propose a baseline variant, EfficientNetV2-S and scale it up to obtain EfficientNetV2-M/L similarly to EfficientNet.

3.1.2 Vision Transformers

ViT. The Vision Transformers are based on the Transformer architecture, introduced by Vaswani et al. [40] for natural language processing (NLP) tasks. For applying Transformers on images, Dosovitskiy et al. introduced Vision Transformer (ViT) [14] architecture, which processes an image as a sequence of non-overlapping patches. Instead of using convolutional operations on a full image, ViT applies the Multi-head Self-attention mechanism [40] on image patches. Similar to BERT [41] in NLP, ViT uses for classification a class token, a trainable vector passed as an input together with patch tokens; the hidden output of the class token is used as an input to the classification head. ViT requires a large dataset for pre-training to generalize well and is fine-tuned on smaller task-specific datasets. ViT comes in several variants that differ in model size: Base/Large/Huge; input image size: 224×224 / 384×384; and patch size: 16×16 / 32×32.

DeiT. Touvron et al. introduced an optimized variant of ViT – Data Efficient Image Transformer (DeiT) [42], which is trained using a distillation strategy. Knowledge distillation, proposed by Hinton et al. [43], is a teacher-student learning approach, where a student learns to generalize in the same way as a teacher. Compared to Noisy Student Training, which uses equal or larger student than the teacher, distillation uses a larger teacher to transfer knowledge to a smaller student. Additionally, DeiT is trained with regularization methods like stochastic depth [37] and data augmentation techniques like

RandAugment [35], Random Erasing [44], Mixup [39], and Cutmix [45]. DeiT has smaller model size variants than ViT: Tiny/Small/Base; the architecture design of ViT-Base and DeiT-Base is identical. Touvron et al. pre-trained DeiT checkpoints with distillation using CNN network RegNetY-16GF [46] as a teacher and checkpoints without distillation.

BEiT. Another ViT variant is Bidirectional Encoder representation from Image Transformer (BEiT) [47] pre-trained using a self-supervised masked image modelling (MIM) task, which is similar to the masked language modelling (MLM) task used in BERT [41]. MIM task randomly masks some image patches, and the objective for the network is to predict them. The pre-trained BEiT is then fine-tuned on ImageNet-1k in a supervised manner. Different from ViT, which uses a class token for classification, BEiT applies mean-pooling on the hidden outputs of the patch tokens.

3.1.3 Implementation Details

The proposed method was developed using the PyTorch [48] machine learning framework. The code is available online[1]. The pre-trained CNNs were used from PyTorch Image Models [49] library, and the pre-trained Vision Transformers were used from Hugging Face Trasformers [50] library. All networks were fine-tuned on one NVIDIA Tesla A100 with 40GB graphic memory.

3.1.4 Optimization Procedure

We use the Stochastic Gradient Descent with Momentum as an optimization algorithm for training all networks. The momentum is set to 0.9, and the initial learning rate is 0.01. During the training, a scheduler reduces the learning rate by a factor of 0.9 if the F_1 score on the validation set does not improve for two epochs. We fine-tune the networks from ImageNet-1k pre-trained checkpoints for 30 epochs, and we keep fine-tuned checkpoint with the highest F_1 score on the validation set.

Training large architectures requires a lot of graphics processing unit (GPU) memory, and a relatively large mini-batch size (32, 64) does not fit into the memory. To have the same mini-batch size of 64 for all architectures, we consider common training approaches: gradient accumulation and mixed precision training [51], described below.

Gradient Accumulation

The gradient accumulation method applies multiple backward passes before updating the parameters during the optimisation procedure instead of performing an update after every single mini-batch. The user-defined hyper-parameter, called accumulation steps, represents a number of backward passes applied before the update. The effective mini-batch size (mini-batch size $*$

[1] `https://github.com/chamidullinr/fine-grained-visual-recognition`

accumulation steps) is a number of training examples consumed in one training step. This approach enables using the same effective mini-batch size for all networks.

Mixed Precision Training

Mixed precision training [51] is a technique that combines single-precision (32-bit floats, "FP32") and half-precision (16-bit floats, "FP16") float numbers and enables to fit the network with a larger mini-batch size into GPU memory. In order to lower the memory requirements, the forward and backward passes use a half-precision version of the model. Then, the gradient descent is applied to the single-precision version of the model. In every training step, the following procedure is applied:

1. Apply the forward pass, compute the loss and apply backward pass on a model in FP16.
2. Convert the gradients from FP16 to FP32.
3. Apply the update on the primary model in FP32.
4. Create a copy of the primary model in FP16.

3.1.5 Data Augmentation

During the training, we apply only the standard data augmentation techniques: random resized crop with scale of $0.8 - 1.0$; random horizontal flip with probability 0.5; random vertical flip with probability 0.5; and random brightness and contrast with probability 0.2.

3.2 Loss Functions

The baseline loss function for training the classifiers is the standard cross entropy loss (L_{CE}):

$$L_{\mathrm{CE}}(p, t) = -\log p_t, \tag{3.1}$$

where p is the classifier predicted probability distribution over C classes and t is the ground truth target. The probability distribution is calculated with softmax function over all classes as:

$$p_t = \frac{exp(z_t)}{\sum_j exp(z_j)}, \tag{3.2}$$

where z_t is the predicted output from the network.

The SankeCLEF and Danish Fungi datasets are imbalanced with long-tailed class distribution, meaning they have a disproportionate ratio of observations in each class, and the training procedure may prioritize the majority classes and ignore the minority. The following subsections describe the loss functions proposed to address the class imbalance during training.

3.2.1 Focal Loss

The focal loss (L_{FL}) [52] extends the standard cross entropy by adding a factor $(1-p_t)^\gamma$ in order to focus learning on hard, misclassified examples and to reduce the relative loss for easy, well-classified examples. The focal loss is defined as:

$$L_{\mathrm{FL}}(p,t) = -(1-p_t)^\gamma \log p_t, \tag{3.3}$$

where $\gamma \geq 0$ is the user-defined hyperparameter. L_{FL} is equivalent to L_{CE} when $\gamma = 0$. The effect of the factor is increased as γ increases. Lin et al. found that $\gamma = 2$ worked the best in their experiments.

3.2.2 Weighted Loss

The L_{CE} and L_{FL} functions can be extended by assigning weight to each class. The weighted cross entropy loss (L_{WCE}) is a variant of CE that adds a weighting hyperparameter w_t as follows:

$$L_{\mathrm{WCE}}(p,t) = -w_t \log p_t. \tag{3.4}$$

Similarly, the weighted focal loss (L_{WFL}) adds weighting hyperparameter w_t to L_{FL}:

$$L_{\mathrm{WFL}}(p,t) = -w_t(1-p_t)^\gamma \log p_t. \tag{3.5}$$

In practice, various weighting strategies are adopted for setting w_t for each class $t \in \{1, 2, ..., C\}$. Note that when $w_t = 1$ L_{WCE} is equivalent to L_{CE}, and L_{WFL} is equivalent to L_{FL}.

Inverse class frequency (ICF). ICF, defined as $w_t = 1/f_t$ where f_t is a frequency of class t, assigns higher misclassification costs to the minority classes and lower costs to the majority.

Linear class (LC) and modified linear class (MLC). In SnakeCLEF 2020 and 2021 challenges [53, 54], Bloch et al. experimented with LC weight function $w_t = \max(f_t)/f_t$ and a modified variant, which oversamples less very low frequencies, defined as:

$$w_t = 1 - \frac{1}{\frac{\max(f_t)}{f_t} + 0.5}. \tag{3.6}$$

Class-balanced (CB). Recently proposed method [55] is a class-balanced weighting based on the effective number of samples defined as:

$$w_t = \frac{1-\beta}{1-\beta^{f_t}}. \tag{3.7}$$

The hyperparameter $\beta \in [0; 1)$ allows to adjust the weighting between no weighting ($\beta = 0$) and weighting by inverse class frequency ($\beta \approx 1$). The weights w_t are normalized $\sum_{t=1}^{C} w_t = C$ to make the total loss roughly in the same scale when applying w_t. Cui et al. experiment with variants $\beta \in \{0.9, 0.99, 0.999, 0.9999\}$, and for focal loss $\gamma \in \{0.5, 1.0, 2.0\}$.

3.2.3 F1 Loss with Soft Assignments

The F_1 score from Equation 2.1 is not differentiable and thus cannot be utilized as a loss function for back-propagation. The soft F_1 loss is an approximation of the F_1 score, which uses soft assignments that make the function differentiable:

- The true positives for class t are estimated using the matrix of softmax predictions $P^{N\times C}$ and the one-hot encoded target matrix $T^{N\times C}$ for N samples and C classes as follows: $\widehat{\mathrm{tp}}_t = \sum_i P_{it} T_{it}$.
- The false positives are estimated as: $\widehat{\mathrm{fp}}_t = \sum_i P_{it}(1 - T_{it})$.
- The false negatives are estimated as: $\widehat{\mathrm{fn}}_t = \sum_i (1 - P_{it}) T_{it}$.

Notice, that $\widehat{\mathrm{tp}}$, $\widehat{\mathrm{fp}}$, and $\widehat{\mathrm{fn}}$ are now real valued. The soft F_1 score ($\widehat{F_{1t}}$) for class t is obtained by computing the harmonic mean of precision $\widehat{p}_t$ and recall $\widehat{r}_t$:

$$\widehat{p}_s = \frac{\widehat{\mathrm{tp}}_t}{\widehat{\mathrm{tp}}_t + \widehat{\mathrm{fp}}_t}, \qquad \widehat{r}_s = \frac{\widehat{\mathrm{tp}}_t}{\widehat{\mathrm{tp}}_t + \widehat{\mathrm{fn}}_t}, \qquad \widehat{F_{1s}} = \frac{2\widehat{p}_t \widehat{r}_t}{\widehat{p}_t + \widehat{r}_t}. \tag{3.8}$$

The macro averaged soft F_1 score ($\widehat{F_1}$) is obtained by averaging $\widehat{F_{1t}}$ over all classes:

$$\widehat{F_1} = \frac{1}{C} \sum_{t=1}^{C} \widehat{F_{1t}}. \tag{3.9}$$

The final loss function is $L_{F_1} = 1 - \widehat{F_1}$, so that it ranges from 0 (perfect) to 1 (worst).

3.3 Side Information-based Classification

We include side information in visual snake and fungi species recognition. Each task has different types of metadata for which we apply different approaches.

3.3.1 Country-specific Removal of Predictions

The SnakeCLEF dataset contains side information about the geographical location (country) of observation and a country-species presence mapping – a list of countries where the specific species occurs. We utilize this information to adjust the network predictions at test time as follows: the predictions p_c are set to 0 for all species c that do not occur in the country of the given image; then all predictions are normalized to $\sum_{c=1}^{C} p_c = 1$.

3.3.2 Adjusting Predictions using Side Information Priors

The Danish Fungi metadata include information about habitat, substrate, geographical location (latitude, longitude), and time (month) of observation. Differently from the approach in SnakeCLEF, we estimate side information priors and combine them with the network predictions.

Picek et al. [6] proposed a method for using of the side information to improve categorization performance in the Danish Fungi dataset. For a given image I and a metadata value x which are conditionally independent given the class c, the posterior probability of c can be obtained as:

$$P(c|I,x) = P(c|I)\frac{P(c|x)}{P(c)}, \tag{3.10}$$

where $P(c)$ is the class prior, $P(c|x)$ is the prior probability of c given x, and $P(c|I)$ is the calibrated classifier confidence of c given I. Equation 3.10 can be extended for multiple metadata values x_1, x_2, ..., x_n which are conditionally independent given c:

$$P(c|I,x_1,x_2,...,x_n) = P(c|I)\frac{P(c|x_1)}{P(c)}\frac{P(c|x_2)}{P(c)}...\frac{P(c|x_n)}{P(c)}. \tag{3.11}$$

Estimating Side Information Priors

The probability distributions $P(c)$ and $P(c|x)$ are estimated as relative frequencies in the training set. We employ a Bayesian approach to avoid zero values using m-estimates of probability:

$$P(c) = \frac{n_c + m}{\sum_{i=1}^{C} n_i + m}, \qquad P(c|x) = \frac{n_{cx} + m}{\sum_{i=1}^{C} n_{ix} + m}, \tag{3.12}$$

where n_c is a number of occurrences of class $c \in \{1,2,...,C\}$, n_{cx} is a number of occurrences of c given x, and m is a user-defined parameter denoting a pseudo-count. We estimate the discrete probability for the variables: habitat, substrate, and month.

For the continuous variables latitude and longitude, we approximate the probability density using kernel density estimation (KDE) [56, 57], also known as Parzen's window. For each estimation point $x_t = (lat, lon)$ in the training set T, KDE weights data points using a smooth function K, called kernel function:

$$f(x) = \frac{1}{|T|h^d}\sum_{x_t \in T} K\left(\frac{x - x_t}{h}\right), \tag{3.13}$$

where $f(x)$ is the estimated density at point x, $h > 0$ is a user-defined parameter, called bandwidth, that controls the amount of smoothing, and $d = 2$ is dimension of the space. Figure 3.1 shows the common examples of $K(x)$:

- Gaussian – $K(x) = e^{-||x||^2/2}$,
- Linear – $K(x) = 1 - ||x||$,
- Cosine – $K(x) = \cos(\frac{\pi}{2}||x||)$,
- Exponential – $K(x) = e^{-||x||}$.

The probability densities correspond to $f(x) \approx P(x)$ and $f_c(x) \approx P(x|c)$ where $f_c(x)$ is estimated over training set T_c with samples belonging to the class c. Bayes' rule is applied to obtain $P(c|x)$:

$$P(c|x) = \frac{f_c(x)P(c)}{f(x)}. \tag{3.14}$$

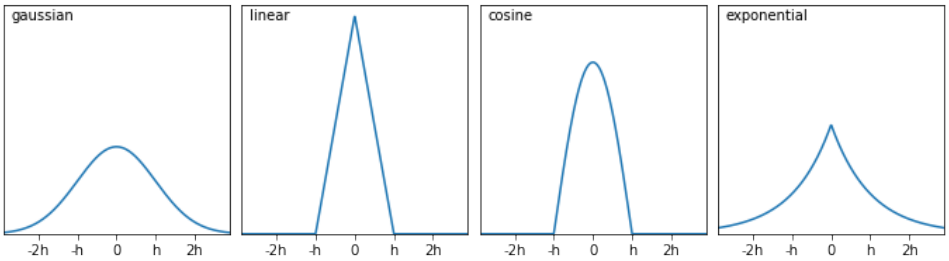


Figure 3.1: Examples of kernel functions.

Calibrating Classifier Confidence

The classifier confidence – a probability distribution on all classes created by a probabilistic classier – represents uncertainty about predictions. The confidence is called calibrated when the probability distribution is consistent with the empirical frequencies observed from realized outcomes [11, 58]. The calibrated confidence can be incorporated into other probabilistic models, for example, combining classifier outputs with prior information.

The modern neural networks are often miscalibrated; due to over-fitting, the network predictions are overconfident [1], meaning that the predicted probabilities are skewed to either 1 or 0 and do not reflect expected correctness. Guo et al. [11] observed that increasing network capacity (depth and width), adding batch normalization, or decreasing weight decay (L_2 regularization) negatively affect model calibration in neural networks.

Confidence calibration is applied as a post-processing step to calibrate neural networks. Guo et al. [11] compared calibration methods and concluded that temperature scaling is the simplest, fastest and yet the most effective method. The method uses a single scalar parameter $T > 0$ for all classes, called temperature, to rescale the network outputs z_c in the softmax function:

$$p_c^{\mathrm{TS}} = \frac{exp(z_c/T)}{\sum_j exp(z_j/T)}. \tag{3.15}$$

The temperature scaling does not affect classification scores because T does not change the relative order of softmax outputs. The temperature T is optimized with respect to the cross entropy loss on the validation set. In our work, we optimize T on the combined probability distribution $P(c|I, x)$, instead of just $P(c|I)$.

3.4 Informed Augmentation

We observed a common property of images in the SnakeCLEF dataset: often, only a small part of the image contains a snake, and the rest is background. We assume the reason is that the snakes are usually found on the ground, and people take photos standing from a safe distance capturing a wide scene. We consider a simple localization technique using saliency map [59] to detect

a single region containing snake in each image, described in Section 3.4.1. Then, we utilize "saliency regions" to crop images and drop the unnecessary background to improve the classification performance, see Section 3.4.2. We also test the method on the Danish Fungi dataset to confirm the technique generalizes well.

3.4.1 Saliency-based Object Localization

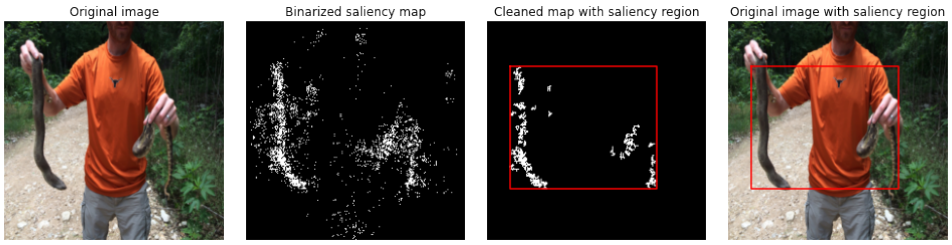
The saliency-based object localization is a weakly supervised method – the network is trained on the image labels without additional annotations like bounding boxes or segmentation masks. The method employs fine-tuned a CNN to obtain a saliency map and then applies post-processing to obtain the saliency region in the form of a rectangle bounding box.

Saliency map generation. The saliency map is based on computing the gradient of the network output z_c of the ground-truth class c with respect to the input image. Given an RGB image $I^{h \times w \times 3}$ with height h, width w, 3 channels, the method computes image derivative $D^{h \times w \times 3} = \frac{\partial z_c}{\partial I}$; the saliency map $S^{h \times w}$ is then obtained by selecting the highest value for each pixel (i, j) across color channels ch: $S_{ij} = \max_{ch} D_{cij}$. The procedure is similar to the standard back-propagation pass used to optimize network weights, except the method back-propagates the network output z_c instead of the classification loss and the gradient is computed with respect to the input image instead of network weights. The interpretation of the method is that the image derivative indicates which pixels need to be changed the least to affect the class score the most [59].

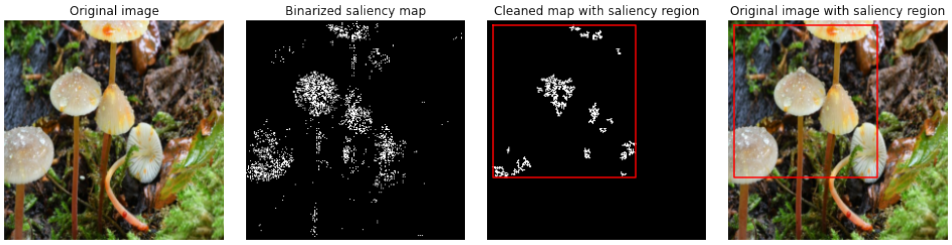
Saliency map post-processing. In the post-processing step, the method binarizes the saliency values by thresholding: the saliency values higher than a threshold are marked as foreground and the rest as background. The threshold is set to the 95% percentile of the value distribution in the saliency map. The method then finds connected components[2] of foreground pixels and discards those that belong to a component with less than 10 pixels by marking them as background. Finally, the saliency region is created as the smallest rectangular bounding box that contains all foreground pixels in the binarized and cleaned saliency map.

Figure 3.2 illustrates steps to create saliency regions in two images from SnakeCLEF and Danish Fungi datasets. The saliency map is generated for each image using EfficientNet-B0 fine-tuned using cross entropy loss and optimization procedure from Section 3.1.4. The first example captures a man holding two parts of a dead snake; interestingly, the saliency-based localization detects both parts, and the saliency region contains almost all pixels belonging to the snake. On the other hand, in the second example with five mushrooms of the same species, the method focuses mainly on mushroom caps, ignoring stems.

[2] A connected component in an image is a set of adjacent pixels with the same non-zero value. Two pixels are considered adjacent if they are neighbours: above, below, to the left or right in the 2-dimensional space.



(a) : Example on snake species – Heterodon platirhinos



(b) : Example on fungi species – Mycena crocata

Figure 3.2: Illustration of steps to create saliency regions.

3.4.2 Training on Cropped Images

We use saliency regions to crop images focusing on the object of interest. We consider several approaches to apply cropped images for training or evaluation to improve classification results.

Evaluation on the cropped images. The network, fine-tuned on the full-size images, classifies cropped images.

Training on the cropped images. The network is fine-tuned on cropped images or a mix of full-size and cropped images. We consider two options for using the mixed dataset: (1) training on 15 epochs with full-size images and 15 epochs with cropped images in alternating order; or (2) for each sample, randomly select whether to use a full-size or cropped image with probability 0.5. The evaluation is done on either a full-size or cropped dataset.

Training and evaluation on 6-channel images. The approach concatenates full-size and cropped images into a single 6-channel image. The network is adjusted to accept 6-dimensional input.

Training using a mixup-based technique. Mixup [39] is an augmentation technique that combines random image pairs and their labels. Given two feature-target vectors (x_i, y_i) and (x_j, y_j), drawn at random from the training dataset, the method produces virtual feature-target vector $(\widehat{x}, \widehat{y})$:

$$\begin{aligned} \widehat{x} &= \lambda x_i + (1-\lambda)x_j, \\ \widehat{y} &= \lambda y_i + (1-\lambda)y_j, \end{aligned} \tag{3.16}$$

where $\lambda \sim \mathrm{Beta}(\alpha, \alpha)$, for a user-defined hyperparameter $\alpha \in (0, \infty)$ that controls the strength of interpolation between feature-target pairs.

We propose a mixup variant that combines full-size (x_{full}) and cropped

(x_{crop}) images of a single observation:

$$\widehat{x} = \alpha x_{\text{full}} + (1 - \alpha) x_{\text{crop}}, \tag{3.17}$$

where $\alpha \in (0, 1)$ is a user-defined hyperparameter that controls the strength of interpolation between full-size and cropped images. The adjustment is applied for each observation with a user-defined probability p. Notice, the mixup variant does not affect labels.

Chapter 4

Experiments

The chapter covers experiments of the proposed method for snake and fungi species recognition. First, Section 4.1 evaluates the results of fine-tuning deep neural networks without any adjustments. Then, Section 4.2 compares training with different loss functions to adopt loss with the highest F_1 score for each dataset and network architecture type. The results of classification with side information are covered in Section 4.3, and experiments on informed augmentation are applied in Section 4.4. Finally, Section 4.5 summarizes the experiment results.

We employ CNN and Vision Transformer architectures from Section 3.1. We select EfficientNet-B0 and ViT-Base-224 as baselines for evaluating our methods. Both networks are fast to train and take similar time to run. We assume that if a tested method improves the performance of small baselines, it would also improve larger networks. We utilize large CNNs: EfficientNet-B4, NoisyStudent-B4, and EfficientNetV2-S; and Vision Transformers: ViT-Base-384, DeiT-Base-384, BEiT-Base-384, and ViT-Large-384. All selected Vision Transformers have 16×16 patch size. We use DeiT-Base-384 pre-trained checkpoint without distillation which, in our preliminary experiments, performed better on SnakeCLEF and Danish Fungi datasets than the checkpoint with distillation. The architectures are listed in Table 4.1.

The networks are fine-tuned from ImageNet-1k [15] pre-trained checkpoints for 30 epochs using optimization procedure from Section 3.1.4. We use

Architecture	Image Size	Image Norm	# Params
EfficientNet-B0	224×224	ImageNet	5.3M
ViT-Base-224	224×224	(0.5, 0.5, 0.5)	86.6M
EfficientNet-B4	380×380	ImageNet	19.3M
NoisyStudent-B4	380×380	ImageNet	19.3M
EfficientNetV2-S	384×384	(0.5, 0.5, 0.5)	21.5M
ViT-Base-384	384×384	(0.5, 0.5, 0.5)	86.9M
DeiT-Base-384	384×384	ImageNet	86.9M
BEiT-Base-384	384×384	(0.5, 0.5, 0.5)	86.7M
ViT-Large-384	384×384	(0.5, 0.5, 0.5)	304.7M

Table 4.1: The architectures and training parameters used in experiments.

the same image size and normalization as in the network pre-training, see Table 4.1. Our preliminary experiments showed that increasing image size positively affects classification scores; therefore, we compare networks with the same or similar image size. The input images are normalized across the RGB channels with ImageNet-1k or (0.5, 0.5, 0.5) mean and standard deviation. We use smaller mini-batches to fit the large networks into GPU memory and accumulate gradients to have the same effective mini-batch size of 64 for all networks. We do not train using mixed precision because it runs slower in our computational environment than single precision.

For comparing different methods we use metrics from Section 2.3: Top-1, Top-3 accuracy, and F_1 score. We prioritize F_1 score over accuracy because it is more suitable for imbalanced datasets with long-tailed class distribution while accuracy is biased by class frequencies.

4.1 Fine-tuning Deep Neural Networks

In the first experiment, we trained different networks from Section 3.1 without any adjustments, on SnakeCLEF and Danish Fungi datasets, using cross entropy loss from Section 3.2.

Table 4.2 shows classification scores on SnakeCLEF-Reduced and Snake-

Architecture	Top-1	Top-3	F_1	Epoch	Total
SnakeCLEF-Reduced					
EfficientNet-B0	**69.1**	**80.7**	57.5	06m	2h 49m
ViT-Base-224	66.4	80.1	**63.3**	06m	2h 43m
EfficientNet-B4	73.5	89.2	68.7	12m	5h 59m
NoisyStudent-B4	**80.2**	**91.6**	**71.0**	13m	6h 06m
EfficientNetV2-S	79.4	92.1	70.5	11m	5h 09m
ViT-Base-384	**79.5**	88.6	**73.5**	14m	6h 33m
DeiT-Base-384	77.6	**94.8**	72.7	14m	6h 34m
BEiT-Base-384	77.6	91.8	67.9	15m	7h 19m
ViT-Large-384	**85.1**	**95.6**	**75.5**	35m	17h 23m
SnakeCLEF					
EfficientNet-B0	85.1	94.2	70.6	24m	11h 32m
ViT-Base-224	**89.9**	**95.6**	**76.0**	25m	12h 04m
EfficientNet-B4	91.4	96.9	80.2	57m	28h 23m
NoisyStudent-B4	**91.7**	**97.4**	81.0	57m	28h 23m
EfficientNetV2-S	91.6	96.0	**81.3**	48m	23h 56m
ViT-Base-384	91.0	97.3	81.2	1h 03m	31h 07m
DeiT-Base-384	92.0	96.9	81.3	1h 03m	31h 04m
BEiT-Base-384	**93.5**	**97.7**	**84.0**	1h 10m	34h 46m
ViT-Large-384	91.9	97.7	83.0	2h 47m	83h 13m

Table 4.2: SnakeCLEF dataset - Classification scores on different architectures fine-tuned with cross entropy.

CLEF datasets. SnakeCLEF-Reduced is a subset of training samples from SnakeCLEF; both datasets have the same test set. In both, Vision Transformers achieve the highest F_1 score. On SnakeCLEF-Reduced, ViT-Large-384 trained for about 17.5 hours has the highest scores (Top-1=85.1; F_1=75.5); the second highest scores (Top-1=79.5; F_1=73.5) has ViT-Base-384 which was 3.5× faster to compute. On SnakeCLEF, BEiT-Base-384 has the highest scores (Top-1=93.5; F_1=84.0); interestingly, both DeiT and BEiT achieve higher scores than ViT. When comparing CNNs, NoisyStudent-B4 performs the best on SnakeCLEF-Reduced (Top-1=80.2; F_1=71.0), while on SnakeCLEF NoisyStudent-B4 has the highest Top-1 accuracy (91.7) and EfficientNetV2-S has the highest F_1 score (81.3). Comparing SnakeCLEF-Reduced and SnakeCLEF shows that having more data improves the scores.

Table 4.3 contains the results of Danish Fungi datasets: DF20 and DF20M, where each dataset has a different number of classes and test samples. The results look similar as in Table 4.2. Vision Transformers achieve higher scores than CNNs: ViT-Large-384 has the highest scores on both DF20M (Top-1=75.5; F_1=66.1) and DF20 (Top-1=80.6; F_1=74.5). DeiT and BEiT have lower scores than ViT. When comparing CNNs, NoisyStudent-B4 achieves the highest scores on DF20M (Top-1=70.2; F_1=59.7), while on DF20 EfficientNetV2-S has the highest Top-1 accuracy (76.7) and NoisyStudent-B4 has the highest F_1 score (68.1).

Architecture	Top-1	Top-3	F_1	Epoch	Total
DF20M					
EfficientNet-B0	63.1	81.0	51.8	03m	1h 05m
ViT-Base-224	**68.5**	**85.4**	**58.6**	03m	1h 12m
EfficientNet-B4	68.4	85.1	58.2	06m	2h 34m
NoisyStudent-B4	**70.2**	**86.1**	**59.7**	06m	2h 33m
EfficientNetV2-S	70.0	85.6	59.4	05m	2h 09m
ViT-Base-384	**74.7**	**88.5**	**65.7**	06m	2h 45m
DeiT-Base-384	73.0	87.6	63.4	06m	2h 45m
BEiT-Base-384	73.2	88.1	64.3	07m	3h 07m
ViT-Large-384	**75.5**	**89.3**	**66.1**	15m	7h 26m
DF20					
EfficientNet-B0	67.7	83.3	58.2	19m	9h 28m
ViT-Base-224	**73.4**	**87.0**	**66.0**	19m	9h 26m
EfficientNet-B4	74.8	88.2	66.1	42m	20h 51m
NoisyStudent-B4	76.3	89.0	**68.1**	42m	20h 42m
EfficientNetV2-S	**76.7**	**89.2**	68.0	35m	17h 27m
ViT-Base-384	**78.7**	**90.5**	**71.7**	45m	22h 05m
DeiT-Base-384	78.0	89.9	70.9	45m	22h 04m
BEiT-Base-384	77.1	89.6	69.5	52m	25h 31m
ViT-Large-384	**80.6**	**91.8**	**74.5**	2h 0m	59h 53m

Table 4.3: Danish Fungi dataset - Classification scores on different architectures fine-tuned with cross entropy.

4.2 Evaluation of Loss Functions

After fine-tuning baseline networks with a standard cross entropy loss (L_{CE}) we applied experiments to test loss functions from Section 3.2 and find loss that improves classification scores. We cannot apply exhaustive search to test all loss functions on all networks and datasets because of computational and time requirements, especially on the large datasets. Instead, we use a two-step heuristic method: (1) fine-tune all loss functions on small networks (EfficientNet-B0, ViT-Base-224) and small training sets (SnakeCLEF-Reduced, DF20M); then (2) use the loss function with the highest F_1 score to fine-tune large networks.

Fine-tuning loss functions on small networks. As a baseline, we use L_{CE} and compare it to focal loss (L_{FL}) and weighted cross entropy loss (L_{WCE}) with different weighting strategies from Section 3.2.2: inverse class frequency (ICF), linear class (LC), modified linear class (MLC), and class-balanced (CB). For CB, we test different values of $\beta \in \{0.9, 0.99, 0.999, 0.9999\}$ and for L_{FL} values of $\gamma \in \{0.5, 1.0, 2.0, 5.0\}$. Training with soft F_1 loss (L_{F_1}) did not work in our classification tasks; the loss did not produce good gradients for the networks to train. A possible explanation for the failure is that the mini-batch size of 64 is significantly smaller than the total number of classes, 182 (DF20M), 770 (SnakeCLEF), or 1 604 (DF20). This leads to the classes not being represented in every mini-batch, making the approximation of L_{F_1} inaccurate [9].

Table 4.4 shows dependence of the loss functions on the classification scores of the SnakeCLEF-Reduced dataset. Fine-tuning EfficientNet-B0 with the baseline L_{CE} achieves the highest scores. For ViT-Base-224 using L_{WCE} with ICF, MLC, or CB has positive effect on scores; the highest F_1 and Top-1 scores are achieved by L_{WCE} with CB ($\beta = 0.9$).

The results on DF20M are covered in Table 4.5. For both EfficientNet-B0

				EfficientNet-B0			ViT-Base-224		
Loss	WS	β	γ	Top-1	Top-3	F_1	Top-1	Top-3	F_1
L_{CE}	×	×	×	**69.1**	80.7	**57.5**	66.4	80.1	63.3
L_{WCE}	ICF	×	×	-13.7	-0.4	-3.2	+0.3	+3.1	+0.1
L_{WCE}	LC	×	×	-10.7	-1.2	-3.4	-19.3	-11.4	-18.0
L_{WCE}	MLC	×	×	-5.7	**+5.4**	-1.6	+3.7	+3.5	+0.8
L_{WCE}	CB	0.9	×	-3.9	+0.9	-0.6	**+5.1**	+2.7	**+1.3**
L_{WCE}	CB	0.99	×	-8.9	+2.6	-1.8	+0.6	+4.3	+0.6
L_{WCE}	CB	0.999	×	-13.7	+2.5	-3.2	-5.5	+3.0	-2.6
L_{WCE}	CB	0.9999	×	-16.8	-1.7	-2.7	-1.4	+2.2	-0.3
L_{FL}	×	×	0.5	-0.4	+3.9	-0.6	-1.0	+2.5	+0.4
L_{FL}	×	×	1.0	-1.3	+4.6	-0.4	+1.1	+4.0	+1.1
L_{FL}	×	×	2.0	-6.3	+1.4	-2.1	+1.3	**+4.6**	-0.3
L_{FL}	×	×	5.0	-7.7	+2.0	-2.7	+0.8	+2.5	-0.9

Table 4.4: SnakeCLEF-Reduced dataset – Effect of training with different loss functions and weighting strategies (WS).

Loss	WS	β	γ	EfficientNet-B0 Top-1	EfficientNet-B0 Top-3	EfficientNet-B0 F_1	ViT-Base-224 Top-1	ViT-Base-224 Top-3	ViT-Base-224 F_1
L_{CE}	×	×	×	63.1	81.0	51.8	68.5	85.4	58.6
L_{WCE}	ICF	×	×	-0.8	-0.1	-0.3	-0.5	-0.6	-0.8
L_{WCE}	LC	×	×	-0.7	-0.1	-0.4	-0.6	-0.7	-0.8
L_{WCE}	MLC	×	×	**+0.2**	+0.5	+0.7	**+0.5**	+0.7	-0.6
L_{WCE}	CB	0.9	×	-0.3	+0.4	-0.6	**+0.5**	+0.3	-0.6
L_{WCE}	CB	0.99	×	-0.2	+0.5	+0.1	+0.2	+0.7	-0.4
L_{WCE}	CB	0.999	×	-1.0	+0.1	-0.4	-0.2	-0.5	-0.2
L_{WCE}	CB	0.9999	×	+0.1	+0.2	**+1.0**	-0.4	+0.0	**+0.7**
L_{FL}	×	×	0.5	-1.0	-0.1	-1.4	-1.3	-0.9	-2.9
L_{FL}	×	×	1.0	-0.8	+0.4	-1.0	+0.4	+0.4	-1.0
L_{FL}	×	×	2.0	-0.8	+0.5	-0.8	-0.3	**+0.8**	-1.2
L_{FL}	×	×	5.0	-0.8	**+0.8**	-0.7	-0.5	+0.6	-2.1

Table 4.5: Danish Fungi dataset (DF20M) – Effect of training with different loss functions and weighting strategies (WS).

and ViT-Base-224 the highest F_1 score has L_{WCE} with CB ($\beta = 0.9999$), while the highest Top-1 accuracy has L_{WCE} with MLC and Top-3 accuracy has L_{FL}.

Fine-tuning large networks using the candidate loss functions. Based on the outcomes from the previous step, we select and test the candidate loss functions for each dataset and network architecture type. For SnakeCLEF dataset, we use L_{WCE} with CB ($\beta = 0.9$) to train Vision Transformers, see Table 4.6. The loss improves the performance of BEiT in both SnakeCLEF-Reduced (Top-1=+5.9; F_1=+6.8) and SnakeCLEF (Top-1=+0.2; F_1=+0.5). ViT and DeiT have non-conclusive results. The highest score, achieved by BeIT, is increased from 84.0 to 84.6. Therefore, we use L_{WCE} with CB ($\beta = 0.9$) for fine-tuning Vision Transformers. For CNNs, with keep L_{CE}.

Architecture	Top-1 L_{CE}	Top-1 CB	Top-1 Diff	Top-3 L_{CE}	Top-3 CB	Top-3 Diff	F_1 L_{CE}	F_1 CB	F_1 Diff
SnakeCLEF-Reduced									
ViT-Base-224	66.4	71.5	+5.1	80.1	82.8	+2.7	63.3	64.7	+1.3
ViT-Base-384	79.5	78.5	-1.0	88.6	92.8	+4.2	73.5	73.0	-0.5
DeiT-Base-384	77.6	77.9	+0.3	94.8	94.6	-0.2	72.7	72.2	-0.5
BEiT-Base-384	77.6	83.5	+5.9	91.8	92.7	+0.9	67.9	74.8	+6.8
ViT-Large-384	85.1	**84.7**	-0.3	**95.6**	94.6	-0.9	75.5	**76.4**	+0.9
SnakeCLEF									
ViT-Base-224	89.9	87.5	-2.3	95.6	94.3	-1.3	76.0	75.5	-0.5
ViT-Base-384	91.0	91.7	+0.6	97.3	97.3	+0.0	81.2	82.0	+0.7
DeiT-Base-384	92.0	90.6	-1.4	96.9	96.7	-0.2	81.3	81.2	-0.1
BEiT-Base-384	93.5	**93.7**	+0.2	97.7	**97.8**	+0.1	84.0	**84.6**	+0.5
ViT-Large-384	91.9	92.0	+0.1	97.7	97.5	-0.1	83.0	82.9	-0.0

Table 4.6: SnakeCLEF dataset – Improvement of Vision Transformers trained with weighted cross entropy and class-balanced (CB) weighting ($\beta = 0.9$) compared to standard cross entropy (L_{CE}).

Architecture	Top-1 L_{CE}	Top-1 CB	Top-1 Diff	Top-3 L_{CE}	Top-3 CB	Top-3 Diff	F_1 L_{CE}	F_1 CB	F_1 Diff
DF20M									
EfficientNet-B0	63.1	63.2	+0.1	81.0	81.2	+0.2	51.8	52.8	+1.0
ViT-Base-224	**68.5**	68.1	-0.4	**85.4**	**85.4**	+0.0	58.6	**59.3**	+0.7
EfficientNet-B4	68.4	67.5	-0.8	85.1	85.0	-0.2	58.2	57.8	-0.5
NoisyStudent-B4	**70.2**	69.5	-0.7	86.1	**86.5**	+0.4	59.7	**61.0**	+1.3
EfficientNetV2-S	70.0	69.4	-0.6	85.6	86.0	+0.4	59.4	60.5	+1.1
ViT-Base-384	**74.7**	73.2	-1.5	**88.5**	88.4	-0.1	**65.7**	65.0	-0.7
DeiT-Base-384	73.0	71.2	-1.9	87.6	86.8	-0.8	63.4	61.8	-1.6
BEiT-Base-384	73.2	66.3	-6.9	88.1	83.3	-4.8	64.3	56.6	-7.7
ViT-Large-384	**75.5**	75.0	-0.5	**89.3**	88.9	-0.3	66.1	**66.1**	+0.1
DF20									
EfficientNet-B0	67.7	66.7	-1.0	83.3	83.0	-0.3	58.2	59.3	+1.1
ViT-Base-224	**73.4**	71.6	-1.7	**87.0**	86.1	-0.9	**66.0**	65.0	-1.0
EfficientNet-B4	74.8	74.1	-0.7	88.2	88.0	-0.2	66.1	67.5	+1.4
NoisyStudent-B4	76.3	75.0	-1.3	89.0	88.8	-0.2	68.1	68.4	+0.3
EfficientNetV2-S	**76.7**	75.5	-1.2	**89.2**	88.7	-0.5	68.0	**69.1**	+1.1
ViT-Base-384	**78.7**	77.4	-1.4	**90.5**	89.9	-0.6	**71.7**	71.3	-0.5
DeiT-Base-384	78.0	76.4	-1.5	89.9	88.9	-1.1	70.9	70.1	-0.8
BEiT-Base-384	77.1	76.4	-0.7	89.6	89.3	-0.3	69.5	70.3	+0.8
ViT-Large-384	**80.6**	79.4	-1.3	**91.8**	91.0	-0.8	**74.5**	73.4	-1.1

Table 4.7: Danish Fungi dataset – Improvement of the networks trained with weighted cross entropy and class-balanced (CB) weighting ($\beta = 0.9999$) compared to standard cross entropy (L_{CE}).

We test L_{WCE} with CB ($\beta = 0.9999$) on all networks on Danish Fungi data, see Table 4.7. F_1 score improved for CNNs: NoisyStudent-B4 achieves the highest score on DF20M (Top-1=69.5; F_1=61.0) with increase +1.3; EfficientNetV2-S on DF20 (Top-1=75.5; F_1=69.1) with increase +1.1. On the other hand, Vision Transformers have a negative impact: ViT-Large-384 with the highest scores on DF20 decreased -1.1 in F_1 score and -1.3 in Top-1 accuracy. Top-1 accuracy decreased for most of the networks. Because of the negative impacts, we keep fine-tuning Vision Transformers using L_{CE}; and we adopt L_{WCE} with CB ($\beta = 0.9999$) for CNNs.

4.3 Evaluation of Side Information-based Classification

The section evaluates the results of using images with side information for species classification. As described in Section 3.3, we employ two different approaches for metadata inclusion. SnakeCLEF dataset comes with a country-species presence mapping that we use to remove predictions of the species not occurring in the country of the given image. For the Danish Fungi metadata, we estimate priors using relative frequency or kernel density estimation methods and adjust image-based predictions using Equation 3.11.

Architecture	Top-1 Orig	Top-1 Adj	Top-1 Diff	Top-3 Orig	Top-3 Adj	Top-3 Diff	F_1 Orig	F_1 Adj	F_1 Diff
SnakeCLEF-Reduced									
EfficientNet-B0	69.1	**76.0**	+6.9	80.7	**89.4**	+8.6	57.5	64.8	+7.3
ViT-Base-224	71.5	75.9	+4.4	82.8	87.2	+4.4	64.7	**70.3**	+5.6
EfficientNet-B4	73.5	79.5	+6.0	89.2	94.4	+5.2	68.7	74.1	+5.4
NoisyStudent-B4	80.2	83.8	+3.6	91.6	**96.2**	+4.6	71.0	**75.3**	+4.3
EfficientNetV2-S	79.4	**84.1**	+4.7	92.1	95.6	+3.5	70.3	75.0	+4.7
ViT-Base-384	78.5	82.4	+3.8	92.8	94.6	+1.8	73.0	77.8	+4.8
DeiT-Base-384	77.9	80.2	+2.3	94.6	**95.7**	+1.1	72.2	76.2	+4.0
BEiT-Base-384	83.5	**84.9**	+1.3	92.7	94.3	+1.6	74.8	**78.1**	+3.3
ViT-Large-384	84.7	**85.9**	+1.2	94.6	**96.4**	+1.7	76.4	**79.4**	+3.1
SnakeCLEF									
EfficientNet-B0	85.1	88.0	+2.9	94.2	95.4	+1.2	70.6	75.9	+5.3
ViT-Base-224	87.5	**88.6**	+1.0	94.3	96.5	+2.2	75.5	**79.1**	+3.6
EfficientNet-B4	91.4	92.2	+0.8	96.9	97.2	+0.3	80.2	82.5	+2.3
NoisyStudent-B4	91.7	**92.6**	+0.9	97.4	**97.5**	+0.1	81.0	83.5	+2.5
EfficientNetV2-S	91.6	**92.6**	+1.0	96.0	97.0	+1.0	81.3	**84.0**	+2.8
ViT-Base-384	91.7	92.9	+1.2	97.3	97.5	+0.2	82.0	84.6	+2.6
DeiT-Base-384	90.6	91.7	+1.0	96.7	97.2	+0.5	81.2	83.8	+2.6
BEiT-Base-384	93.7	**93.9**	+0.2	**97.8**	97.7	-0.2	84.6	**85.8**	+1.2
ViT-Large-384	92.0	92.6	+0.6	97.5	97.6	+0.0	82.9	84.9	+2.0

Table 4.8: SnakeCLEF dataset – Improvement when adjusting image-based predictions using country information.

4.3.1 Country-specific Removal of Predictions in SnakeCLEF Dataset

Table 4.8 compares classification scores of original and adjusted predictions on the SnakeCLEF dataset. The adjustment works for all architectures: larger improvement have smaller networks like EfficientNet-B0 and ViT-Base-224 and networks trained on fewer data (SnakeCLEF-Reduced). The approach helps to increase the highest F_1 score, obtained by BEiT-Base-384, from 84.6 to 85.8 and the highest Top-1 accuracy from 93.7 to 93.9.

4.3.2 Adjusting Predictions using Side Information Priors in Danish Fungi Dataset

For the Danish Fungi data, we estimate side information priors: habitat (H), substrate (S), and month (M) as relative frequencies using m-estimate of probability; and geographical location (G) (latitude, longitude) using kernel density estimation (KDE). We use parameter values $m = 0.1$ for m-estimates and exponential kernel with bandwidth $h = 0.2$ for KDE.

Table 4.9 shows effect of different metadata attributes on the classification scores of EfficientNet-B0. To avoid over-fitting, we remove images in DF20M and DF20 that belong to the same observation and have the same metadata. Hence the baseline scores slightly differ compared to EfficientNet-B0 scores

H	S	G	M	DF20M Top-1	DF20M Top-3	DF20M F_1	DF20 Top-1	DF20 Top-3	DF20 F_1
×	×	×	×	63.0	81.4	52.2	67.1	83.4	59.5
✓	×	×	×	+2.0	+2.0	+2.3	+1.3	+1.0	+1.9
×	✓	×	×	+1.0	+0.9	+1.1	+1.3	+1.4	+1.8
×	×	✓	×	+1.3	+0.9	+2.1	+0.9	+0.7	+1.4
×	×	×	✓	+1.1	+1.2	+1.3	+1.2	+1.0	+1.1
✓	✓	×	×	+2.7	+2.4	+3.1	+2.4	+2.0	+3.4
✓	×	✓	×	+2.9	+2.3	+4.2	+1.9	+1.3	+3.0
✓	×	×	✓	+3.3	+2.8	+3.7	+2.4	+1.8	+3.0
×	✓	✓	×	+2.6	+1.9	+3.8	+2.0	+1.9	+3.1
×	✓	×	✓	+2.0	+2.1	+2.3	+2.6	+2.2	+3.1
×	×	✓	✓	+2.4	+1.7	+3.7	+2.1	+1.7	+2.6
✓	✓	✓	×	+3.1	+3.2	+5.2	+2.5	+2.0	+4.0
✓	✓	×	✓	+4.4	+3.2	+5.9	+3.1	+2.5	+4.4
✓	×	✓	✓	+4.6	+3.1	+6.6	+2.8	+2.0	+3.5
×	✓	✓	✓	+3.5	+2.9	+5.2	+3.2	+2.6	+4.1
✓	✓	✓	✓	+4.8	+3.9	+7.1	+3.6	+2.8	+5.0
✓	✓	✓	✓	67.7	85.3	59.2	70.7	86.2	64.6

Table 4.9: Danish Fungi dataset – Effect of different side information attributes on the classification scores of EfficientNet-B0.

from Table 4.7. When evaluating single attributes, the highest F_1 score percentage point increase on DF20M has habitat (Top-1=+2.0; F_1=+2.3), followed by geographical location (Top-1=+1.3; F_1=+2.1), month (Top-1=+1.1; F_1=+1.3), and substrate (Top-1=+1.0; F_1=+1.1). Similarly, on DF20, the highest improvement has habitat (Top-1=+1.3; F_1=+1.9), followed by substrate (Top-1=+1.3; F_1=+1.8), geographical location (Top-1=+0.9; F_1=+1.4), and month (Top-1=+1.2; F_1=+1.1). The approach has a higher effect when combining multiple attributes: the highest scores achieves the combination of all four attributes on DF20M (Top-1=67.7; F_1=59.2) and

m	DF20M Top-1	DF20M Top-3	DF20M F_1	DF20 Top-1	DF20 Top-3	DF20 F_1
×	63.0	81.4	52.2	67.1	83.4	59.5
0	+2.8	+0.9	+4.0	+1.3	-0.8	+0.7
0.01	+4.1	+3.5	+6.2	+2.9	+2.0	+3.7
0.05	+4.7	+3.7	+7.1	+3.4	+2.6	+4.6
0.1	+4.8	+3.9	+7.1	+3.6	+2.8	+5.0
0.5	+4.4	+4.0	+6.6	+3.6	+3.0	+4.9
1	+4.0	+3.6	+6.0	+3.3	+2.8	+4.5
2	+3.4	+3.1	+5.3	+2.7	+2.3	+3.5

Table 4.10: Danish Fungi dataset – Effect of different m values in m-estimate of probability when estimating side information priors.

Architecture	Top-1 Orig	Top-1 Adj	Top-1 Diff	Top-3 Orig	Top-3 Adj	Top-3 Diff	F_1 Orig	F_1 Adj	F_1 Diff
DF20M									
EfficientNet-B0	63.0	67.7	+4.8	81.4	85.3	+3.9	52.2	59.2	+7.1
ViT-Base-224	68.7	**71.5**	+2.8	85.6	**87.7**	+2.1	58.4	**61.7**	+3.3
EfficientNet-B4	68.0	71.8	+3.8	85.2	88.3	+3.1	58.0	62.2	+4.1
NoisyStudent-B4	70.0	**73.6**	+3.6	86.9	88.6	+1.7	61.3	64.9	+3.7
EfficientNetV2-S	69.8	73.0	+3.2	86.4	**88.7**	+2.3	60.7	**65.4**	+4.7
ViT-Base-384	75.2	**76.9**	+1.7	88.8	90.0	+1.2	65.8	67.9	+2.1
DeiT-Base-384	73.6	75.8	+2.2	87.9	89.8	+1.9	63.4	66.6	+3.3
BEiT-Base-384	73.6	**76.9**	+3.3	88.4	**90.7**	+2.3	64.2	**69.0**	+4.8
ViT-Large-384	76.0	**78.3**	+2.4	89.7	90.6	+0.9	66.0	**69.0**	+3.0
DF20									
EfficientNet-B0	67.1	70.7	+3.6	83.4	86.2	+2.8	59.5	64.6	+5.0
ViT-Base-224	73.7	**76.4**	+2.6	87.3	**89.1**	+1.8	66.2	**69.1**	+2.9
EfficientNet-B4	74.6	77.1	+2.5	88.4	90.0	+1.6	67.7	71.0	+3.3
NoisyStudent-B4	75.5	**78.6**	+3.1	89.1	**91.1**	+1.9	68.6	**72.7**	+4.0
EfficientNetV2-S	76.0	78.2	+2.3	89.0	90.6	+1.6	69.2	72.1	+2.9
ViT-Base-384	79.0	**80.6**	+1.5	90.8	**91.6**	+0.8	71.9	**73.1**	+1.2
DeiT-Base-384	78.3	80.0	+1.7	90.3	91.2	+0.9	70.9	72.5	+1.6
BEiT-Base-384	77.5	79.5	+2.1	89.9	91.3	+1.4	69.6	72.1	+2.6
ViT-Large-384	81.0	**81.9**	+0.9	92.1	**92.1**	+0.1	74.6	**74.8**	+0.2

Table 4.11: Danish Fungi dataset – Improvement when adjusting image-based predictions using side infromation priors.

DF20 (Top-1=70.7; F_1=64.6).

Table 4.10 shows effect of different m values in m-estimate of probability when estimating habitat, substrate, and month. The scores are computed on a combination of all four attributes, including geographical location. The increase is the lowest when estimating relative frequencies without pseudo-counts ($m = 0$). The highest impact on Danish Fungi dataset have m values on interval $(0; 1]$.

The results of the method on all networks are shown in Table 4.11. Similar to the SnakeCLEF results from Table 4.8, the adjustment of predictions works for all architectures and smaller networks like EfficientNet-B0 and ViT-Base-224 have a larger improvement. The highest scores after adjustment are achieved by ViT-Large-384 in DF20M (Top-1=78.3; F_1=69.0) and DF20 (Top-1=81.9; F_1=74.8).

4.4 Evaluation of Informed Augmentation

We apply the weakly supervised saliency-based object localization method from Section 3.4 to detect "saliency regions" capturing snake or fungi in each image. The technique employs fine-tuned CNN to generate saliency map. We crop images using saliency regions to enrich our data and run experiments on training and evaluation with full-size and cropped images. As

a baseline we fine-tune and evaluate EfficientNet-B0 on the full-size images (Full) and compare the classification scores with networks fine-tuned on training strategies from Section 3.4.2:

- Crop – training on cropped images,
- Full & Crop – training on 15 epochs with full-size images and 15 epochs with cropped images in alternating order,
- Full & Crop (rand.) – training on full-size and cropped images where for each sample the system randomly selects whether to use a full-size or cropped image with probability 0.5,
- 6-channels – training and evaluation on concatenated full-size and cropped images,
- Mixup – training on a mixup-based method where parameter α controls the strength of interpolation between full-size and cropped images and parameter p sets the probability of applying the method on one sample.

We evaluate fine-tuned checkpoints on the full-size (Full) or cropped (Crop) images for all training strategies except 6-channels.

The results on the SnakeCLEF-Reduced dataset are covered in Table 4.12. The baseline method achieves 57.5 F_1 score and 69.1 Top-1 accuracy. The F_1 score is improved by +2.6 when fine-tuning and evaluating the network on cropped images. Fine-tuning on the mix of full-size and cropped images has smaller improvement: +0.5 for Full & Crop evaluated on cropped images and +1.8 for Full & Crop (rand.) evaluated on full-size images. The highest F_1 score increase (+2.8) achieves mixup-based method with $\alpha = 0.2$ and $p = 0.25$ evaluated on cropped images. Most of the mixup variants with different parameters have a positive impact when evaluated on cropped images, while evaluating on full images decreases F_1 score. Interestingly, different methods increased in Top-1 accuracy compared to the baseline: mixup-based method with $\alpha = 0.2$ and $p = 0.5$ (+2.9) and the baseline checkpoint (+1.9),

			Top-1		Top-3		F_1	
Training Strategy	α	p	Full	Crop	Full	Crop	Full	Crop
Full	×	×	69.1	**71.0**	80.7	85.1	57.5	53.9
Crop	×	×	61.3	68.9	**87.8**	81.8	53.7	**60.1**
Full & Crop	×	×	58.7	61.6	81.5	83.1	57.3	58.0
Full & Crop (rand.)	×	×	69.0	67.2	85.0	86.0	59.3	59.2
6 channels	×	×	62.9		87.4		56.7	
Mixup	0.2	0.25	60.4	67.6	81.1	85.4	58.0	**60.3**
Mixup	0.5	0.25	55.4	70.0	80.5	**89.4**	51.5	57.6
Mixup	0.8	0.25	58.6	63.1	78.5	80.3	54.7	59.4
Mixup	0.2	0.50	65.1	**72.0**	88.7	89.2	57.5	59.8
Mixup	0.5	0.50	58.4	65.5	83.1	89.5	54.0	56.7
Mixup	0.8	0.50	68.7	66.5	83.9	87.0	53.9	58.4

Table 4.12: SnakeCLEF-Reduced dataset – Improvement when training and evaluating EfficientNet-B0 with cropped images created using saliency-based localization.

Training Strategy	α	p	Top-1 Full	Top-1 Crop	Top-3 Full	Top-3 Crop	F_1 Full	F_1 Crop
Full	×	×	**63.2**	59.0	81.2	78.7	52.8	48.8
Crop	×	×	57.5	61.4	77.7	80.5	46.6	50.7
Full & Crop	×	×	62.3	**63.1**	**82.1**	**82.7**	52.6	**53.5**
Full & Crop (rand.)	×	×	62.9	62.9	81.6	81.7	52.9	**53.3**
6 channels	×	×	62.4		80.7		53.0	
Mixup	0.2	0.25	62.0	63.0	81.3	81.9	52.4	52.4
Mixup	0.5	0.25	59.8	**63.1**	80.2	81.6	49.7	52.7
Mixup	0.8	0.25	58.4	62.7	78.5	81.1	47.1	52.2
Mixup	0.2	0.50	60.4	61.5	81.1	81.6	50.6	52.5
Mixup	0.5	0.50	60.0	63.0	81.0	81.3	50.0	53.0
Mixup	0.8	0.50	58.3	62.0	78.0	80.9	47.0	51.1

Table 4.13: Danish Fungi dataset (DF20M) – Improvement when training and evaluating EfficientNet-B0 with cropped images created using saliency-based localization.

both evaluated on cropped images. Training and evaluating on 6-channel images negatively impact both F_1 score and Top-1 accuracy.

Table 4.13 shows results on DF20M dataset. The baseline achieves 52.8 F_1 score and 63.2 Top-1 accuracy. The highest increase (+0.7) in F_1 score obtains Full & Crop strategy, followed by Full & Crop (rand.) with increase +0.5, both evaluated on cropped images. Training using mixup-based methods achieves inferior F_1 score, while 6-channel strategy has a small increase (+0.2). The highest Top-1 accuracy has the baseline.

4.5 Discussion

The section summarizes the experiment results of the snake and fungi species recognition tasks:

Vision Transformers outperform CNNs. On the Danish Fungi dataset, the performance of ViT-Base-384 is better than EfficientNet-B4, NoisyStudent-B4, and EfficientNetV2-S, as well as DeiT-Base-384 and BEiT-Base-384. As expected, larger network ViT-Large-384 achieves higher scores than ViT-Base-384 but runs slower. On SnakeCLEF data, the highest scores has BEiT-Base-384 followed by ViT-Base-384, DeiT-Base-384, NoisyStudent-B4, and EfficientNetV2-S with similar scores.

Loss functions proposed for imbalanced class distributions achieve minor improvement. In some setups, loss functions like class-balanced cross entropy loss outperform standard cross entropy loss. However, the increase is minor, and the procedure requires testing different functions and their hyperparameters. On the other hand, the standard cross entropy loss requires little effort, and the trained networks achieve good results.

Side information improves classification. Both metadata inclusion approaches used in this work have positive results, but well-performing

networks have a relatively small increase in the F_1 score. BEiT-Base-384, with the highest scores on the SnakeCLEF dataset, has a +1.2 percentage point increase, while the best performer on the DF20, ViT-Large-384, has only +0.2.

Saliency-based object localization improves classification. Using cropped images for training and test time improves the F_1 score on both tasks. The approach has a larger effect on the SnakeCLEF dataset, with the highest F_1 score increase +2.8, while the highest increase on the Danish Fungi dataset is +0.7. We assume the difference depends on how people take photos: photos of snakes are taken from a safe distance and contain unnecessary background, while fungi are captured at close range.

Chapter 5

Real-world Application

The previous chapters describe the species recognition task. In practice, however, the problem can be different. For example, in emergency treatment of a snake bite, it is more relevant to know whether the biting snake is venomous and which anti-venom to administer instead of determining the name of the snake species. This chapter shows how the proposed method can be used for venomous/non-venomous species classification. For simplicity, we assume that the universal anti-venom exists and can treat the snakebite of any venomous snake; the goal is thus to decide if the snake is venomous or non-venomous[1].

In the snake species recognition, the network estimates classifier confidence $P(c|I)$ of a class c given an image I. We transform the initial task into the venomous/non-venomous species classification by computing $P(v|I)$ where v denotes whether the species is venomous or non-venomous. We use information about medical importance of a species – information whether the specific species is venomous or non-venomous. The medical importance data are available online[2]. We consider the following methods for obtaining $P(v|I)$:

- A baseline method (Method 1) takes a snake species top-1 prediction and checks whether the species is venomous. Formally, the method sets

$$\begin{aligned} P(v = \text{"venomous"}|I) &= 1, \\ P(v = \text{"non-venomous"}|I) &= 0, \end{aligned} \tag{5.1}$$

  if the predicted species is venomous. And vice-versa for non-venomous species.
- Improved method (Method 2) recognizes if the snake is venomous by aggregating (sum) the classifier confidence of all venomous species (S_v) and all non-venomous species (S_n):

$$\begin{aligned} P(v = \text{"venomous"}|I) &= \sum_{c_v \in S_v} P(c_v|I), \\ P(v = \text{"non-venomous"}|I) &= \sum_{c_n \in S_n} P(c_n|I). \end{aligned} \tag{5.2}$$

[1] The problem can be extended to decide on treatment of snakebite victim: no treatment (if the snake is non-venomous) or treatment with anti-venom A, B, C, etc. For this, a mapping of venomous snakes with appropriate anti-venoms for treatment is required.

[2] `https://github.com/chamidullinr/fine-grained-visual-recognition`

Additionally, we consider a method (Method 3) designed to make optimal decisions based on assigned costs. This method is a general form of Method 2, defined as

$$\begin{aligned} P(v = \text{"venomous"}|I) &= \sum_{c_v \in S_v} P(c_v|I) * w_v, \\ P(v = \text{"non-venomous"}|I) &= \sum_{c_n \in S_n} P(c_n|I) * w_n, \end{aligned} \tag{5.3}$$

where weights w_v and w_n enable assigning a cost value for both types of error, false positives and false negatives.

5.1 Evaluation of Venomous/non-venomous Species Classification

The section evaluates performance of Methods 1-3 to obtain venomous/non-venomous species decision $P(v|I)$ from the network outputs. We utilize two different networks: small EfficientNet-B0 and large BEiT-Base-384; both are fine-tuned on the SnakeCLEF dataset with cross entropy loss and have adjusted predictions using the country origin of observations. We use temperature scaling from Section 3.3.2 to calibrate classifier confidence $P(c|I)$; the temperature parameter is tuned on the validation set. For Method 3, we assign 1000× larger cost for misclassifying venomous species as non-venomous, i.e. $w_v = 1000$ and $w_n = 1$.

Table 5.1 shows the results of venomous/non-venomous species classification evaluated on Top-1 accuracy, precision, recall, and F_1 score. EfficientNet-B0 trained on the species recognition task achieves 97.2 F_1 score and 98.8 Top-1 accuracy on the venomous/non-venomous classification task when applying Method 2, and BEiT-Base-384 has 98.9 F_1 score and 99.5 Top-1 accuracy. On both networks, Method 2 is superior to Method 1. While low in Top-1 accuracy and F_1 score Method 3 with $w_v = 1000$ and $w_n = 1$ maximizes recall. Confusion matrix in Table 5.2 illustrates differences in the methods. Method 1 applied on EfficientNet-B0 has 171 false negatives and 162 false positives; Method 2 improves false positives (107) and slightly worsens false

	Top-1	Precision	Recall	F_1
EfficientNet-B0				
Method 1	98.6	96.8	96.7	96.7
Method 2	**98.8**	**97.9**	96.6	**97.2**
Method 3	95.4	82.5	**99.3**	90.1
BEiT-Base-384				
Method 1	**99.5**	99.0	98.7	98.8
Method 2	**99.5**	**99.2**	98.7	**98.9**
Method 3	98.4	93.5	**99.4**	96.4

Table 5.1: Scores on venomous/non-venomous species classification problem.

EfficientNet-B0							
		Method 1		Method 2		Method 3	
		V	N	V	N	V	N
Actual	V	4 934	171	4 931	174	5 067	38
	N	162	18 960	107	19 015	1 072	18 050
BEiT-Base-224							
		Method 1		Method 2		Method 3	
		V	N	V	N	V	N
Actual	V	5 037	68	5 037	68	5 076	29
	N	51	19 071	42	19 080	350	18 772

Table 5.2: Confusion matrix on classifying venomous (V) and non-venomous (N) species.

negatives (174); Method 3 minimizes false negatives (38) for the cost of false positives (1 072). On BEiT-Base-384, Method 2 has 68 false negatives and 42 false positives, and Method 3 decreases false negatives (13) for the cost of false positives (469).

The use of Method 3 with different weights demonstrates a way to adjust classification based on the misclassification cost requirements of the real-world problem. For example, misclassifying venomous species would make the wrong decision not to administer antivenom to a snakebite victim, leading to serious health issues or even death; on the other hand, misclassifying non-venomous species may lead to an allergic reaction to antivenom. Therefore, weights w_v and w_n can represent the medical costs of treating snakebite victims after a wrong diagnosis.

Chapter 6

Conclusion

This thesis presents a method for fine-grained recognition, in particular snake and fungi species identification, for which annotated data with side information are available. The proposed method is based on state-of-the-art deep neural networks for classification – EfficientNet, ViT, and their variants – fine-tuned from ImageNet-1k pre-trained checkpoints. We evaluate the method on SnakeCLEF 2021 and Danish Fungi 2020 datasets with imbalanced class distribution; we use standard metrics, Top-1 and Top-3 accuracy, and a metric suitable for imbalanced datasets, macro averaged F_1 score.

The initial experiments show that CNNs and Vision Transformers can be successfully trained to identify snake and fungi species. Vision Transformers show superior performance on both tasks: BEiT-Base-384 on snake recognition and ViT-Large-384 on fungi recognition. We apply the following approaches to achieve performance improvements:

We train the classifiers with loss functions proposed to address the class imbalance and compare the performance with the baseline cross entropy loss. We found out that the weighted cross entropy loss with class-balanced weighting increases scores of BEiT-Base-384 on snakes, while the standard cross entropy loss is superior for ViT-Large-384 on fungi.

We include side information into image-based classification. For snake species recognition, the method drops the predictions of the species not occurring in the country of the given image. For fungi species recognition, the method calibrates and adjusts the predictions by the prior probabilities of side information like habitat, substrate, location, and the time of observation. Both approaches improve classification scores, but well-performing networks like BEiT-Base-384 and ViT-Large-384 have relatively small increase.

On CNNs, we apply a weakly supervised object localization method using a saliency map to detect regions with snakes or fungi in images. Then, we crop images based on the detected regions to enrich the training data. The experiments show that training on a mix of full-size and cropped images and classifying cropped images improves the results.

The proposed method achieves 93.9% in Top-1 accuracy and F_1 score of 85.8 with BEiT-Base-384 on the SnakeCLEF dataset; and 81.9% in Top-1 accuracy and F_1 score of 74.8 with ViT-Large-384 on the Danish Fungi dataset.

Finally, we demonstrate an example of the real-world application of the

proposed method. We use medical information about species to transform predictions of species into decision whether the biting snake is venomous and whether the antivenom should be administered. The approach enables to assign different cost values for misclassification; this is important because misclassifying a venomous species for the non-venomous is a more significant issue than the opposite.

For future work regarding the weakly supervised object localization, we suggest generating a saliency map with Vision Transformers. The procedure [59], used in this work, relies on CNNs, which process full images. Applying the procedure on Vision Transformers that processes images as sequences of non-overlapping patches may have shortcomings. Instead, the self-attention mechanism could be leveraged. Bao et al. [47] show that the self-attention mechanism in BEIT can separate objects; the self-supervised pre-training enables BEiT to distinguish semantic regions. Using the "self-attention map" from BEiT would require a post-processing step to identify the semantic region with the object of interest.

Bibliography

[1] M. Šulc, *Fine-grained Recognition of Plants and Fungi from Images*. PhD thesis, Czech Technical University in Prague, 2021.

[2] Y. Cui, F. Zhou, Y. Lin, and S. Belongie, "Fine-Grained Categorization and Dataset Bootstrapping Using Deep Metric Learning with Humans in the Loop," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1153–1162, 2016.

[3] J. Troudet, P. Grandcolas, A. Blin, R. Vignes Lebbe, and F. Legendre, "Taxonomic bias in biodiversity data and societal preferences," *Scientific Reports*, vol. 7, Dec 2017.

[4] "Danish Mycological Society." `https://svampe.databasen.org/`; visited on 02.12.2021.

[5] L. Picek, A. M. Durso, R. Ruiz De Castañeda, and 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] L. Picek, M. Šulc, J. Matas, J. Heilmann-Clausen, T. S. Jeppesen, T. Læssøe, and T. Frøslev, "Danish Fungi 2020 - Not Just Another Image Recognition Dataset," 2021.

[7] P. Uetz and J. Hallermann, "The Reptile Database." visited on 02.12.2021.

[8] M. E. Valverde, T. Hernández-Pérez, and O. Paredes-López, "Identifying the snake: First scoping review on practices of communities and health-care providers confronted with snakebite across the world," vol. 2015, Jan 2015.

[9] R. Chamidullin, M. Šulc, J. Matas, and L. Picek, "A Deep Learning Method for Visual Recognition of Snake Species," in *CLEF working notes 2021, CLEF – Conference and Labs of the Evaluation Forum*, 2021.

[10] I. Bolon, A. M. Durso, S. Botero Mesa, N. Ray, G. Alcoba, F. Chappuis, and 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*, vol. 15, Mar 2020.

[11] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," in *Proceedings of the 34th International Conference on Machine Learning* (D. Precup and Y. W. Teh, eds.), vol. 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330, PMLR, Aug 2017.

[12] L. Picek, I. Bolon, A. M. Durso, and R. Ruiz De Castañeda, "Overview of the SnakeCLEF 2020: Automatic Snake Species Identification Challenge," in *CLEF task overview 2020, CLEF: Conference and Labs of the Evaluation Forum*, 2020.

[13] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *Proceedings of the 36th International Conference on Machine Learning* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 6105–6114, PMLR, Jun 2019.

[14] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," in *International Conference on Learning Representations*, 2021.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016.

[17] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated Residual Transformations for Deep Neural Networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[18] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, H. Lin, Z. Zhang, Y. Sun, T. He, J. Mueller, R. Manmatha, M. Li, and A. Smola, "ResNeSt: Split-Attention Networks," 2020.

[19] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and Efficient Object Detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2020.

[20] L. G. Coca, A. T. Popa, R. C. Croitoru, L. P. Bejan, and A. Iftene, "UAIC-AI at SnakeCLEF 2021: Impact of convolutions in snake species recognition," in *CLEF working notes 2021, CLEF – Conference and Labs of the Evaluation Forum*, 2021.

[21] K. Desingu, M. Palaniappan, and J. Kumar, "Snake Species Classification using Transfer Learning Technique," in *CLEF working notes 2021, CLEF – Conference and Labs of the Evaluation Forum*, 2021.

[22] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

[23] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *CoRR*, vol. abs/1409.1556, 2015.

[24] J. H. Friedman, "Greedy function approximation: A gradient boosting machine.," *Annals of Statistics*, vol. 29, pp. 1189–1232, 2001.

[25] M. A. Ottom and N. A. Alawad, "Classification of Mushroom Fungi Using Machine Learning Techniques," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 8, pp. 2378–2385, Oct 2019.

[26] M. Sulc, L. Picek, J. Matas, T. Jeppesen, and J. Heilmann-Clausen, "Fungi Recognition: A Practical Use Case," pp. 2305–2313, Mar 2020.

[27] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, p. 4278–4284, AAAI Press, 2017.

[28] 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, and 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.

[29] T. G. Frøslev, J. Heilmann-Clausen, C. Lange, T. Læssøe, J. H. Petersen, U. Søchting, T. S. Jeppesen, and J. Vesterholt, "Danish Mycological Society, fungal records database," 2019.

[30] "Scikit-learn User Guide: 3.1. cross-validation: Evaluating estimator performance." library version 1.0.1.

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[32] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

[33] J. Hu, L. Shen, and G. Sun, "Squeeze-and-Excitation Networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, 2018.

[34] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-Training With Noisy Student Improves ImageNet Classification," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, 2020.

[35] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "RandAugment: Practical automated data augmentation with a reduced search space," 2019.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, Jul 2014.

[37] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep Networks with Stochastic Depth," in *Computer Vision – ECCV 2016* (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), pp. 646–661, Springer International Publishing, 2016.

[38] M. Tan and Q. Le, "EfficientNetV2: Smaller Models and Faster Training," in *Proceedings of the 38th International Conference on Machine Learning* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 10096–10106, PMLR, Jul 2021.

[39] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond Empirical Risk Minimization," in *International Conference on Learning Representations*, 2018.

[40] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[41] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, Jun 2019.

[42] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jegou, "Training data-efficient image transformers & distillation through attention," in *Proceedings of the 38th International Conference on Machine Learning* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 10347–10357, PMLR, Jul 2021.

[43] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," *ArXiv*, vol. abs/1503.02531, 2015.

[44] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random Erasing Data Augmentation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 13001–13008, Apr 2020.

[45] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2019.

[46] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, and P. Dollár, "Designing Network Design Spaces," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10425–10433, 2020.

[47] B. Hangbo, D. Li, and W. Furu, "BEiT: BERT Pre-Training of Image Transformers," 2021.

[48] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), pp. 8024–8035, Curran Associates, Inc., 2019.

[49] R. Wightman, "PyTorch Image Models." `https://github.com/rwightman/pytorch-image-models`, 2019. visited on 28.06.2021.

[50] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, (Online), pp. 38–45, Association for Computational Linguistics, Oct 2020.

[51] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, and H. Wu,

"Mixed Precision Training," in *International Conference on Learning Representations*, 2018.

[52] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2999–3007, 2017.

[53] L. Bloch, A. Boketta, C. Keibel, E. Mense, A. Michailutschenko, O. Pelka, J. Rückert, L. Willemeit, and C. M. 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.

[54] L. Bloch and C. M. Friedrich, "EfficientNets and Vision Transformers for Snake Species Identification Using Image and Location Information," in *CLEF working notes 2021, CLEF – Conference and Labs of the Evaluation Forum*, 2021.

[55] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-Balanced Loss Based on Effective Number of Samples," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9260–9269, 2019.

[56] E. Parzen, "On Estimation of a Probability Density Function and Mode," *The Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 1065 – 1076, 1962.

[57] Y.-C. Chen, "A tutorial on kernel density estimation and recent advances," *Biostatistics & Epidemiology*, vol. 1, no. 1, pp. 161–187, 2017.

[58] J. Vaicenavicius, D. Widmann, C. Andersson, F. Lindsten, J. Roll, and T. Schön, "Evaluating model calibration in classification," in *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics* (K. Chaudhuri and M. Sugiyama, eds.), vol. 89 of *Proceedings of Machine Learning Research*, pp. 3459–3467, PMLR, Apr 2019.

[59] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps," *preprint*, Dec 2013.

Appendix A

Danish Fungi Metadata

A.1 Habitat values in the Danish Fungi Dataset

Table A.1 shows a list of the habitat values in the DF20 training set.

Habitat	Number of images	Percentage of images
Deciduous woodland	58 393	22.0%
Mixed woodland (with coniferous and deciduous trees)	44 779	16.9%
Unmanaged deciduous woodland	31 267	11.8%
coniferous woodland/plantation	27 724	10.4%
park/churchyard	17 474	6.6%
natural grassland	16 843	6.3%
garden	7 551	2.8%
Unmanaged coniferous woodland	7 108	2.7%
roadside	6 562	2.5%
Thorny scrubland	6 290	2.4%
lawn	5 025	1.9%
hedgerow	4 063	1.5%
Willow scrubland	3 918	1.5%
Bog woodland	3 772	1.4%
Forest bog	3 694	1.4%
heath	3 240	1.2%
dune	3 166	1.2%
bog	2 397	0.9%
wooded meadow, grazing forest	2 275	0.9%
salt meadow	2 024	0.8%
other habitat	1 940	0.7%
meadow	1 572	0.6%
gravel or clay pit	1 361	0.5%
Acidic oak woodland	1 036	0.4%
rock	432	0.2%
improved grassland	387	0.1%
fallow field	317	0.1%
ditch	284	0.1%
fertilized field in rotation	221	0.1%
masonry	148	0.1%
roof	106	<0.1%

Table A.1: List of the habitat values in the DF20 training set.

A.2 Substrate values in the Danish Fungi Dataset

Table A.2 shows a list of the substrate values in the DF20 training set.

Substrate	Number of images	Percentage of images
soil	121 280	45.8%
dead wood (including bark)	69 316	26.2%
wood	20 694	7.8%
bark of living trees	10 213	3.9%
leaf or needle litter	10 089	3.8%
bark	4 911	1.9%
wood and roots of living trees	3 773	1.4%
mosses	3 659	1.4%
stems of herbs, grass etc	3 053	1.2%
wood chips or mulch	2 963	1.1%
living leaves	2 292	0.9%
siliceous stone	2 075	0.8%
fungi	2 022	0.8%
faeces	1 526	0.6%
cones	1 120	0.4%
stone	1 052	0.4%
other substrate	937	0.4%
dead stems of herbs, grass etc	688	0.3%
living stems of herbs, grass etc	598	0.2%
building stone (e.g. bricks)	498	0.2%
peat mosses	481	0.2%
fruits	423	0.2%
calcareous stone	327	0.1%
lichens	277	0.1%
insects	258	0.1%
fire spot	153	0.1%
catkins	124	<0.1%
living flowers	79	<0.1%
liverworts	59	<0.1%
remains of vertebrates (e.g. feathers and fur)	34	<0.1%
mycetozoans	2	<0.1%

Table A.2: List of the substrate values in the DF20 training set.