Zurich University of Applied Sciences

Department Life Sciences and Facility Management

Institute of Natural Resource Sciences

BACHELOR THESIS

Deep Learning for Biodiversity Monitoring: Automated Classification of Small Mammals Captured in Foto Trap Boxes

JUNE 24, 2025

Author:
Julian Kraft[1]

Supervisor:
Dr. Stefan Glüge[2]
Dr. Matthias Nyfeler[2]

Affiliations:
[1]ZHAW - Institute of Natural Resource Sciences
[2]ZHAW - Institute of Computational Life Sciences

Imprint

Project type:	Bachelor Thesis
Title:	Deep Learning for Biodiversity Monitoring: Automated Classification of Small Mammals Captured in Foto Trap Boxes
Date:	June 24, 2025
Keywords:	Camera Trapping, Deep Learning, Transfer Learning, Object Detection, Image Classification, Small Mammals, Biodiversity Monitoring, MegaDetector
Copyright:	Zurich University of Applied Sciences

Author:	Julian Kraft[1] (kraftjul@students.zhaw.ch)
Supervisor:	Dr. Stefan Glüge[2] (glue@zhaw.ch)
	Dr. Matthias Nyfeler[2] (nyfe@zhaw.ch)
Affiliations:	[1]ZHAW - Institute of Natural Resource Sciences
	[2]ZHAW - Institute of Computational Life Sciences

Abstract

This thesis explores the use of Deep Learning (DL) to automate the classification of small mammals captured in camera trap images gathered as part of the Wildlife@Campus project. A dataset of over 400,000 labeled images, grouped into sequences, was processed using MegaDetector (MD) to filter and crop relevant regions of interest. Several model architectures were evaluated, with the pretrained EfficientNet-B0 achieving the highest balanced accuracy of 0.992 for the classification task. A comprehensive data pipeline was developed, including detection, preprocessing, cross-validation and classification on an image and sequence level, enabling efficient and reproducible model training and evaluation. While pretrained models outperformed non-pretrained variants, the results also demonstrated that smaller architectures can be accurate while saving resources. The study highlights the importance of detection quality, label accuracy and the need for a non-target class to handle unknown species such as snails or misdetections such as plant parts or simply empty images. In addition to the missing non-target class, the study also emphasizes the need for an improved detection process in order to reduce missed sightings. There is still a high dependency on large amounts of labeled data, which is a real challenge when adding additional classes. This work lays the foundation for integrating DL into the camera trap approach of the Wildlife@Campus project, aiming to reduce the associated manual effort in small mammal monitoring and contribute to ecological research.

Zusammenfassung

Diese Bachelorarbeit untersucht den Einsatz von DL zur automatisierten Klassifikation von Kleinsäugern, die mithilfe von Kamerafallen im Rahmen des Wildlife@Campus-Projekts erfasst wurden. Ein Datensatz mit über 400'000 annotierten Bildern, gruppiert in Sequenzen, wurde mithilfe von MD verarbeitet, um relevante Bildausschnitte zu filtern und zuzuschneiden. Mehrere Modellarchitekturen wurden evaluiert, wobei das vortrainierte EfficientNet-B0 die höchste Balanced Accuracy von $0{,}992$ für die Klassifikationsaufgabe erzielte. Es wurde eine umfassende Datenpipeline entwickelt, die Erkennung, Vorverarbeitung, Kreuzvalidierung sowie Klassifikation auf Bild- und Sequenzebene umfasst und ein effizientes sowie reproduzierbares Modelltraining und eine entsprechende Evaluation ermöglicht. Während vortrainierte Modelle gegenüber nicht vortrainierten Varianten bessere Leistungen zeigten, verdeutlichten die Ergebnisse auch, dass kleinere Architekturen präzise sein können und gleichzeitig Ressourcen sparen. Die Studie betont die Bedeutung der Detektionsqualität, der Labelgenauigkeit und die Notwendigkeit einer Sonstige-Klasse zur Berücksichtigung unbekannter Arten wie z.B. Schnecken oder Fehldetektionen wie z.B. Pflanzenteile oder einfach leere Bilder. Neben der fehlenden Sonstige-Klasse unterstreicht die Studie auch den Bedarf nach einem verbesserten Detektionsprozess, um verpasste Sichtungen zu reduzieren. Zudem besteht weiterhin eine hohe Abhängigkeit von grossen Mengen annotierter Daten, was eine reale Herausforderung bei der Einführung zusätzlicher Klassen darstellt. Diese Arbeit legt das Fundament für die Integration von DL in den Kamerafallen-Ansatz des Wildlife@Campus-Projekts, mit dem Ziel, den damit verbundenen manuellen Aufwand bei der Kleinsäugerüberwachung zu reduzieren und zur ökologischen Forschung beizutragen.

Acknowledgments

I am deeply grateful to my supervisor, Dr. Stefan Glüge, for his invaluable guidance and engaging discussions throughout this thesis. Besides his expertise in the field, he was exceptionally supportive and understanding — his light-hearted approach and optimism were a great motivator for me. Additional thanks go to Dr. Matthias Nyfeler, who provided the topic and ensured I had all the support needed throughout the project. I want to thank Prof. Dr. Roland Felix Graf and Nils Ratnaweera for their help in obtaining the data and all the information I needed on the Wildlife@Campus project to get started with the thesis. For his support in answering many of my questions and proofreading my thesis, I am very grateful to my brother, Dr. Basil Kraft. Finally, I would like to extend my thanks to the HPC support team for ensuring the smooth operation of the cluster and for their impressively swift responses to my questions.

Contents

Code, and LaTeX source are to be found on GitHub:
https://github.com/juliankraft/BachelorThesis

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
BalAcc	balanced accuracy
BBox	bounding box
CM	confusion matrix
CNN	Convolutional Neural Network
CSV	Comma-Separated Values
DL	Deep Learning
eDNA	environmental DNA
EXIF	Exchangeable Image File Format
GB	Giga Byte
GPU	Graphics Processing Unit
HPC	High Performance Computing
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IUNR	Institute of Natural Resource Sciences
MD	MegaDetector
MLWIC2	Machine Learning for Wildlife Image Classification 2
OCR	Optical Character Recognition
OOD	Out-Of-Distribution
ROI	region of interest
ViT	Vision Transformer
WI	Wildlife Insights
ZHAW	Zurich University of Applied Sciences

1 Introduction

The ongoing loss of biodiversity is among the most urgent environmental issues globally (Brondízio et al., 2019; Cardinale et al., 2012). Small mammals, despite their ecological importance, often receive limited attention in conservation research compared to birds and butterflies (Graf et al., 2022). This disparity persists although small mammals significantly contribute to ecosystem functions such as seed dispersal and soil aeration. In Switzerland, of the approximately 30 native small mammal species, several are endangered and require targeted conservation strategies (BAFU, 2019). In response to these challenges, the Wildlife@Campus project was initiated to establish a long-term and locally anchored monitoring program for small mammals in the urban context of the ZHAW Campus Wädenswil. By combining citizen science with scientific research, the project aims to improve the detection and understanding of small mammal populations while also raising public awareness and contributing to species conservation efforts (Graf et al., 2022).

1.1 Background

Monitoring small mammal populations traditionally involves labor-intensive methods like live trapping, which not only consume significant time and resources but also pose risks to animal welfare (Graf et al., 2022). Indirect, more efficient survey methods are a much-needed alternative to monitor less invasively and to generate more data with less effort. A proven effective method is the use of footprint tunnels, which show a very low non-detection rate (Yarnell et al., 2014). A further non-invasive method with increasing relevance is the use of environmental DNA (eDNA) for species detection with high reliability (Thomsen & Willerslev, 2015). The method in focus for this study utilizes camera traps for species detection — an approach that has seen rapid and widespread adoption in ecological and conservation research (Delisle et al., 2021). Based on findings from the Wildlife@Campus project, a combined approach using camera traps for continuous monitoring and eDNA-based methods for species-level identification appears to be a promising and scalable solution for small mammal detection in urban environments (Graf et al., 2022). As part of the Wildlife@Campus project, initial efforts were made to explore the collection of non-invasive genetic material such as hair and fecal samples. While these genetic methods were not fully implemented during the pilot phase, they were recognized as essential for identifying cryptic or morphologically similar species and are considered a key component of future monitoring strategies. Based on the design of the “Mostela” camera trap box, Aegerter (2019) developed the “MammaliaBox”, adapting it to local conditions for monitoring small mustelids and dormice in Switzerland. In addition to improving image quality, he conducted extensive field testing and produced a detailed guideline for standardized and effective deployment. During the Wildlife@Campus project, images were generated exclusively using “MammaliaBoxes” at multiple locations on the ZHAW campus. These camera trap boxes produced a substantial volume of image material, capturing a wide range of small mammal activity. Significant effort has already been invested within the Wildlife@Campus project to manually label a substantial portion of the image dataset and to explore automated approaches to species detection. In fact, a preliminary

solution for training a Deep Learning (DL) model was developed and tested, and a usable labeled dataset already exists (Ratnaweera, 2021). However, this work remained largely undocumented and was not brought to full completion. In the context of this bachelor's thesis, this dataset will be revisited to explore possibilities to leverage DL techniques for species detection in small mammal images.

1.2 Problem Statement

The aim of this thesis is to develop a reproducible and effective workflow for the automated classification of small mammals in camera trap images. To achieve this, the following tasks are addressed:

- **Dataset Preparation:** Review and analyze the existing labeled dataset, including understanding the data structure and labeling conventions.
- **Data Processing:** Develop a processing pipeline that includes identifying the region of interest (ROI) in each image, selecting appropriate training samples and preparing the data for DL model training.
- **Model Selection:** Research and select suitable DL architectures for the classification task, considering both performance and computational efficiency.
- **Model Training:** Train the selected models on the prepared dataset using appropriate optimization strategies.
- **Model Evaluation:** Assess and compare model performance using standard evaluation metrics and conduct a detailed analysis of the most promising architecture.

1.3 Related Work

Image classification utilizing DL has become a cornerstone of modern computer vision, with significant advancements in recent years. One of the earliest successful Convolutional Neural Network (CNN) architectures was LeNet-5, introduced by Lecun et al. (1998) — it was designed for handwritten digit recognition and laid the foundation for subsequent developments in DL for image classification. A major breakthrough came with the introduction of AlexNet by Krizhevsky et al. (2012), which demonstrated the power of deep convolutional networks in large-scale image classification tasks. This architecture won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012 and significantly outperformed previous methods, showcasing the potential of DL in computer vision. Following AlexNet, several other influential architectures emerged, including VGGNet (Simonyan & Zisserman, 2015), GoogLeNet (Szegedy et al., 2015) and ResNet (He et al., 2016). These architectures introduced various innovations such as deeper networks, inception modules and residual connections, which further improved classification accuracy and model efficiency. Subsequent models such as DenseNet (Huang et al., 2017) and EfficientNet (Tan & Le, 2019) focused on parameter efficiency and scaling strategies, achieving strong performance with fewer computational resources. The development of these architectures has been complemented by the creation of large-scale datasets like

ImageNet (Deng et al., 2009), which provided a rich source of labeled images for training and benchmarking DL models. The area of modern DL for image classification has continued to evolve with the introduction of transformer-based architectures like Vision Transformer (ViT) (Dosovitskiy et al., 2021) and Swin Transformers (Liu et al., 2021), which have shown promising results in various image classification tasks.

The success of DL in general image classification has led to its increasing application in wildlife monitoring, particularly in the analysis of camera trap images. The application of DL in wildlife monitoring began gaining traction around 2014, when Chen et al. (2014) used CNN to classify animal species in camera trap images. Their approach involved image segmentation via graph-cut algorithms followed by species classification using a CNN trained on 14,000 manually segmented images across 20 species. Building on this early work, Gomez Villa et al. (2017) compared various CNN architectures on a subset of the Snapshot Serengeti dataset, demonstrating the superiority of deeper models like ResNet-101 for animal identification tasks. Norouzzadeh et al. (2018) broadened the scope of DL applications in wildlife monitoring by developing a model capable of not only identifying but also counting animals in camera trap images. This work demonstrated the potential of DL to automate the labor-intensive task of manual image annotation, significantly improving the efficiency of wildlife monitoring efforts. Following this, Tabak et al. (2019) applied DL techniques to classify species in camera trap images, achieving high accuracy and demonstrating the feasibility of using DL for large-scale wildlife monitoring projects.

There is a growing number of AI platforms and tools designed to facilitate the application of DL in wildlife monitoring. Vélez et al. (2022) provide a comprehensive overview of various platforms including MegaDetector (MD), Wildlife Insights (WI), Machine Learning for Wildlife Image Classification 2 (MLWIC2) and Conservation AI. WI and MLWIC2 demonstrated low recall—many animals present in images were missed—but high precision for some species. In contrast, MD performed reliably for broad classifications such as distinguishing “animal” from “blank”. The authors conclude that while fully automated species classification is not yet reliable, DL tools are valuable for filtering blank images and supporting semi-automated workflows. MD, originally developed by Microsoft, has since become one of the most widely adopted tools for wildlife image filtering and is actively maintained as a standalone detection model (Beery et al., 2019). Building on this foundation, a highly regarded research team supported by Microsoft is now developing the PyTorch Wildlife project (Hernandez et al., 2024). This framework incorporates MD as a core component and extends its functionality into a full-featured DL platform tailored to wildlife monitoring. A study applying MD for image preprocessing is presented by Schneider et al. (2024), who demonstrate its effectiveness in filtering blank images and identifying animals in camera trap datasets. A particularly notable contribution of their work is the taxonomic classification approach, enabling hierarchical predictions beyond species level to genus, family or order.

While most existing studies have focused on larger mammals, there is a growing interest in applying DL techniques to small mammal species. Camera traps are a proven effective method for monitoring small mammal populations (Aegerter, 2019; Clucas & McCluskey, 2025; Littlewood et al., 2021). Despite these developments, there remains a lack of comprehensive DL applications specifically targeting small mammal classification. A notable

exception is the work by Hopkins et al. (2024), who successfully applied a transfer learning approach to fine-tune a YOLOv5 model for detecting six rodent species from camera trap images. Their study highlights the feasibility of using DL for small mammal monitoring and demonstrates how existing detection models can be adapted to new species with relatively limited training data. This indicates a promising direction for future research aiming to extend AI-powered wildlife monitoring beyond the currently dominant focus on large mammals.

There are more complex approaches leveraging the often sequential nature of camera trap data to improve detection and classification accuracy. As an example, Zotin and Proskurin (2019) utilize sequences for a non-DL approach to detect animals in camera trap images taken under complex shooting conditions. Another study by Muhammad et al. (2024) introduces a sequence-aware and metadata-driven approach to camera trap image classification by adapting the Swin Transformer architecture (Swin-FPN Net) to process image sequences rather than individual frames. By incorporating temporal dynamics and leveraging metadata such as timestamps, their method enhances classification accuracy and reduces processing time, enabling a more context-sensitive and efficient analysis of wildlife imagery.

2 Methods

2.1 Dataset

As part of the Wildlife@Campus project, a labeled dataset was created to train a deep learning algorithm. The information about the dataset is derived from the dataset itself and a progress report available on GitHub (Ratnaweera, 2021). This dataset is divided into seven sessions — indicating the source of the images — the sessions and their origin are listed in Table 1. The dataset provides two kinds of labels: the original annotations from each session and a standardized version created when the sessions were integrated into the dataset. For this project, only the standardized labels were used. Images are grouped into sequences, with each sequence representing a single animal sighting. The sequences were created by the Wildlife@Campus team, utilizing the EXIF information of the images to group them based on the time and date of capture. Sequence lengths are not fixed — the distribution of sequence lengths is shown in Figure 1. To get an overview of the available sequences per label, refer to Figure 2. The category `other` represents sequences containing more than one species; this is a result of the process creating the sequences. Furthermore, the category `NaN` represents sequences not labeled — both were excluded from the dataset. The category `glis_glis` is represented in only four sequences, which is simply not enough to train a model to detect it. For this reason, it was excluded from the dataset as well. This leaves four categories for the classification task, which are summarized in Table 2.

Table 1: Information about the origin of the different sessions in the dataset.

Session	Description
1	Data from 'the wild', collected durch Wiesel&Co 2019
2	Data from 'the wild', collected during WILMA (SummerSchool) 2020
3	Data from 'the wild', collected by Vogelwarte 2020
4	Data from 'the wild', collected by WILMA (Bachelorthesis) 2020
5	Data from 'the wild', collected by WILMA (Roland) 2020 (Contains images and videos of stoats (*Mustela erminea*))
6	Data from an enclosure, collected by Nils (Contains only images of stoats)
7	Data gathered from Nathalie Straub in her Bachelor Thesis

2.2 Data Processing

The data processing is done in two main steps — after the initial detection, the ROIs are selected and processed. This part is illustrated in the upper part of Figure 3. Furthermore, a custom data splitting was implemented to create five stratified folds for cross-validation.

Table 2: Overview of the classification categories in this study.

Label	Designation	Description
`apodemus_sp`	Apodemus species	Genus of wood mice
`cricetidae`	Cricetidae	Family including voles and hamsters
`soricidae`	Soricidae	Family of shrews
`mustela_erminea`	Stoat / Ermine	Species of weasel (*Mustela erminea*)
Excluded:		
`glis_glis`	Edible dormouse	Species of dormouse (*Glis glis*)

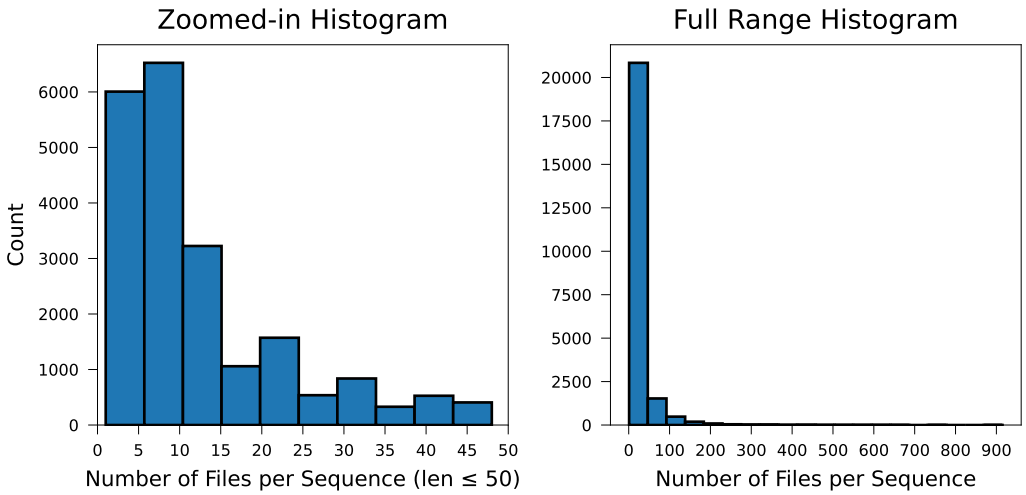

Figure 1: Distribution of sequence lengths per label. More than 90% of the sequences are between 1 and 50 images long. There are longer ones up to a length of 915 images, but these are outliers.

While the detection was performed once for the whole dataset, the selection, image processing and data splitting were implemented to be done on the fly.

2.2.1 Detection and Selection

In this project, the MD (Beery et al., 2019) is used to identify ROIs on all the images. The MD outputs a list of bounding boxes (BBoxes) for detected objects labeled `animal`, `human` or `vehicle` with a corresponding confidence value. Only BBoxes with a confidence score above 0.25 were stored for further processing. This detection step was done sequence-wise and the outputs were saved to json files for each sequence. An example of what these detections look like is shown in Figure 4. Only images with a detection labeled `animal` and a confidence score above a certain threshold were retained. The percentage of images discarded per category due to this threshold was determined by inspecting Figure 5. In order to reduce noise in the data and eliminate blank inputs to the model, a

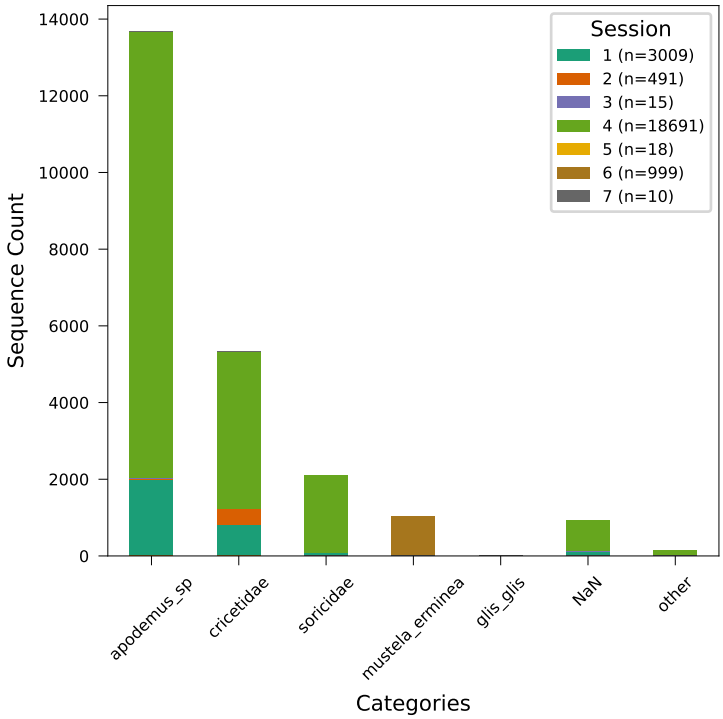

Figure 2: Available sequences per category colored by session.

threshold of 0.5 was chosen. The trade-off was to discard around 20% of the images for the `soricidae` and `mustela_erminea` categories. For images with multiple detections, only the BBox with the highest confidence score was kept.

2.2.2 Image Processing

To process the images, a custom transformation pipeline was implemented using transform version 2 from the torchvision library and a custom crop function. This transformation was applied on the fly by the PyTorch DataLoader. Cropping was done using the BBoxes from the detection, extending them to match the ratio expected by the model. In cases where the extended BBox surpassed the image border, the image was padded with black pixels. After cropping, each image was resized to the model's expected input size, i.e. 224×224 pixels. Each image was first converted into a tensor of shape (C, H, W). The pixel values were then normalized using the global channel-wise mean and standard deviation of the dataset itself. This mean and standard deviation were calculated on the whole dataset, not just the training set, to ensure consistency across all folds. Furthermore, only the best BBox area per image was used to calculate the mean and standard deviation.

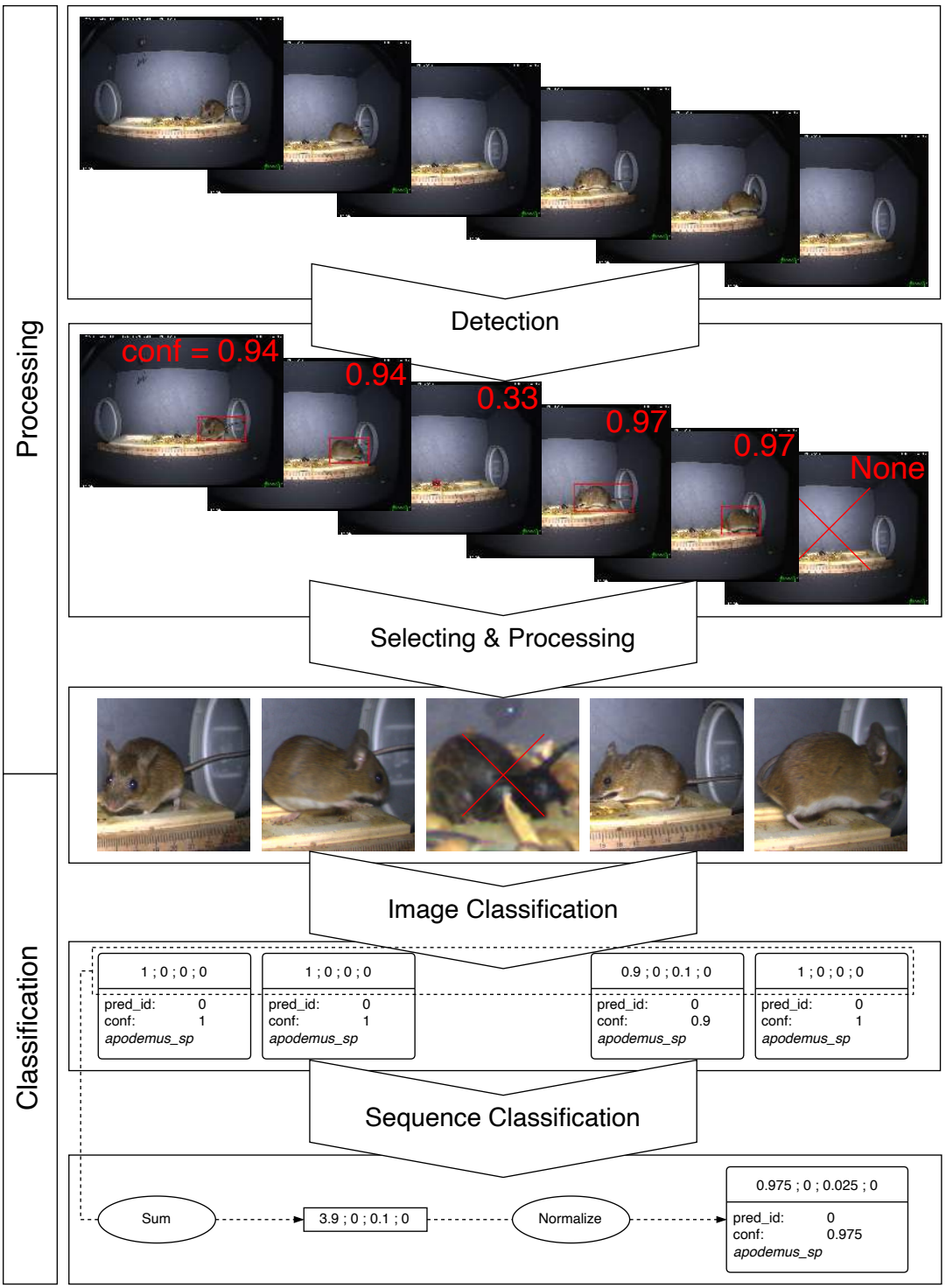

Figure 3: Flow chart illustrating the workflow for one sequence of the dataset from the raw images to a sequence classification. Selected was the sequence 4,002,072, which is a sample from the `apodemus_sp` category.

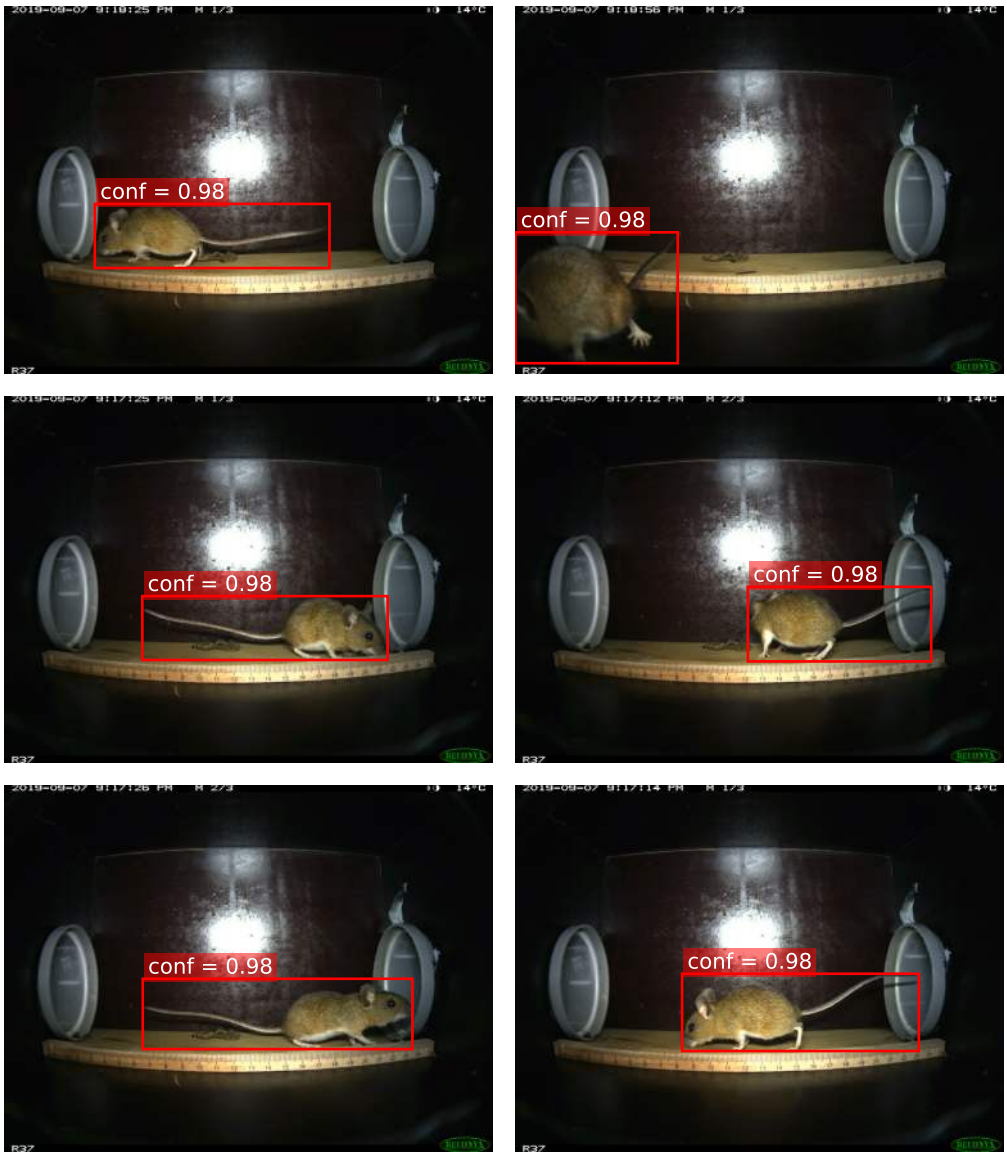

Figure 4: Example of how the detections look like. The BBoxes are the six highest confidence detections for the sequence 1,001,824 a sample from the `apodemus_sp` category.

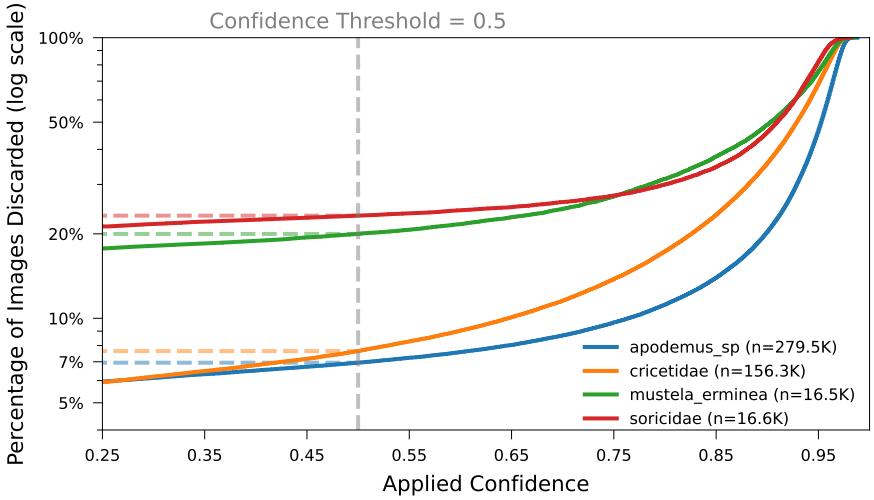

Figure 5: Fraction of images discarded at various detection confidence thresholds. The selected threshold of 0.5, used in this project, is indicated.

2.2.3 Data Splitting

The dataset was split into five folds using a stratified split based on the classes. A custom helper function was implemented to ensure the splits are done on a sequence level, meaning no sequence is ever split between folds. The fold size was determined by the number of images instead of the number of sequences, as the sequences vary in length. This approach ensures that the images are evenly distributed across the folds while maintaining the sequence integrity. For each class in the dataset, all the sequences were shuffled using a fixed seed for reproducibility, resulting in two lists: one with the sequence IDs and one with the corresponding sequence lengths. The list with sequence lengths was used to determine the cut-points for the folds, while the list with sequence IDs was used to assign the sequences to the folds.

2.3 Model

In this project, a selection of models from the torchvision library was tested to classify the images. Each model was both trained from scratch and fine-tuned using the weights of a model pre-trained on ImageNet (Deng et al., 2009). A custom helper function was implemented to adapt the last layer of the model to fit the number of classes in the dataset when the model is loaded. The models used in this project are:

- **EfficientNet-B0** (Tan & Le, 2019): A Convolutional Neural Network architecture that uses a compound scaling method to uniformly scale network width, depth and resolution. The B0 variant is the baseline model from which larger EfficientNets are derived.

- **DenseNet-169** (Huang et al., 2017): A densely connected convolutional network with 169 layers, in which each layer receives feature maps from all preceding layers, fostering feature reuse and improved gradient flow.
- **ResNet-50** (He et al., 2016): A 50-layer Residual Network that introduces skip connections (residual blocks) to alleviate the vanishing gradient problem, enabling training of very deep models.
- **ViT-B_16** (Dosovitskiy et al., 2021): The "Base" Vision Transformer model which splits an image into 16×16 patches, linearly embeds them and processes the resulting sequence with a standard Transformer encoder.

2.4 Training

The training process was divided into four main steps repeated for each fold of the cross-validation:

1. Loading the dataset and applying the processing steps described above.
2. Initializing the model and adapting the last layer to match the classes.
3. Training the model for the current fold using the training set.
4. Validating the model on the validation set and saving the best version of the model based on the lowest validation loss.
5. The best version of the model is loaded to predict the whole dataset for later evaluation.

During training, the loss was calculated using the cross-entropy loss function with the class weights computed on the current training set. To adjust the model parameters, the AdamW optimizer (Loshchilov & Hutter, 2019) was used with a weight decay of 10^{-5} and an initial learning rate of 10^{-4}. The learning rate was adjusted using a cosine annealing scheduler (Loshchilov & Hutter, 2017) over 50 epochs. A maximum of 50 epochs was trained, but an early stopping callback was implemented, monitoring the validation loss with a patience of 10 epochs. Logging was done using the TensorBoard logger, which is integrated into PyTorch Lightning, and an additional custom CSV logger for easier log access for evaluation. A batch size of 64 was used for training — and doubled for validation and prediction.

2.5 Classification

After the training process was completed, the dataset was predicted using the best version model. To the model's output logits, a `softmax` function was applied to obtain the predicted class confidences. The predicted class was determined by selecting the class with the highest confidence score using the `argmax` function. Both the predicted class and the confidence scores for each class were saved in a CSV file for later evaluation. In order to classify the sequences from the image-level predictions, some additional steps were performed. This step was only performed after the model had been trained and the

predictions for the whole dataset were available. These steps are illustrated in the lower part of Figure 3.

For this calculation, a sequence could be viewed as a matrix P of shape $(n, 4)$, where n is the number of images in the sequence and 4 is the number of classes. Since the model outputs a normalized confidence value over classes, the values in each row sum to one, i.e., $\sum_{j=1}^{4} p_{i,j} = 1$ for all $i \in \{1, \ldots, n\}$.

$$P = \begin{bmatrix} p_{1,1} & p_{1,2} & p_{1,3} & p_{1,4} \\ p_{2,1} & p_{2,2} & p_{2,3} & p_{2,4} \\ \vdots & \vdots & \vdots & \vdots \\ p_{n,1} & p_{n,2} & p_{n,3} & p_{n,4} \end{bmatrix} \tag{1}$$

Each row corresponds to an image in the sequence, and each column to one of the four classes.

The scores for each class in the sequence were calculated by summing the scores across all images in the sequence, resulting in a class score vector S:

$$S = [s_1 \; s_2 \; s_3 \; s_4] = \left[\sum_{i=1}^{n} p_{i,1}, \; \sum_{i=1}^{n} p_{i,2}, \; \sum_{i=1}^{n} p_{i,3}, \; \sum_{i=1}^{n} p_{i,4} \right] \tag{2}$$

This class score vector S is then normalized to obtain the normalized class scores $\hat{S}$:

$$\hat{S} = [\hat{s}_1 \; \hat{s}_2 \; \hat{s}_3 \; \hat{s}_4] = \left[\frac{s_1}{S}, \; \frac{s_2}{S}, \; \frac{s_3}{S}, \; \frac{s_4}{S} \right] \quad \text{with } S = s_1 + s_2 + s_3 + s_4 \tag{3}$$

Finally, the predicted class label $\hat{y}$ for the sequence is determined by selecting the class with the highest normalized score:

$$\hat{y} = \arg\max \{\hat{s}_1 \; \hat{s}_2 \; \hat{s}_3 \; \hat{s}_4\} \tag{4}$$

2.6 Evaluation

There are three levels of evaluation in this project. The first is the evaluation of the detection output, which is done using the MD output. The second is the evaluation of the model architecture performance, determining the best performing model architecture based on the balanced accuracy score. The third is the evaluation of the best performing model architecture in more detail, using additional metrics such as precision, recall, F1-score and support.

2.6.1 Detection Output

To evaluate the detection output, two approaches were used. On one hand, a strictly numerical evaluation was done by counting the number of images and sequences per category with detections above the thresholds of 0.25 and 0.5. On the other hand, a visual evaluation was done by plotting series of detections on images from the dataset, creating views like Figure 6 and Figure 7. To do this, a Python class was implemented to iterate over a dataframe, outputting fixed-size chunks of the dataframe. This class was used to browse through a dataframe —— with different filters applied to it —— visualizing the chunks with a custom plot function.

2.6.2 Comparison of Model Architecture Performance

The evaluation is done using the test set predictions created by the best version of each fold's model, which were saved at the end of the training process. To compare model architectures, both the image-level and sequence-level predictions are evaluated using the highest balanced accuracy (BalAcc) score as the metric for best performance. For every model architecture, the BalAcc is calculated for each fold and then averaged over all folds. This evaluation was done for every model architecture both with and without pretraining. The BalAcc score is calculated using the following formula:

$$\text{BalAcc} = \frac{1}{K} \sum_{c=1}^{K} \frac{TP_c}{TP_c + FN_c} \tag{5}$$

where:

K is the total number of classes.
TP_c (true positives for class c) is the count of samples whose true label is c and whose predicted label is also c.
FN_c (false negatives for class c) is the count of samples whose true label is c but whose predicted label is not c.

2.6.3 Best Performing Model Architecture

To further evaluate the best performing model architecture, some additional metrics were computed on the full dataset by combining the predictions from each cross-validation fold test set. On this data, precision, recall, F1-score and support for each class were computed. In the way the data is combined, support (true positives per class) is the same as the number of available samples per class in the whole dataset. Additionally, the normalized confusion matrix (CM) was calculated to visualize the performance of the model across all classes. These metrics were all calculated using the respective functions from the `sklearn` library (Pedregosa et al., 2011).

To assess the rank-based relationship between detection and classification confidence scores, Spearman's rank correlation coefficient was calculated. This analysis was performed separately for correctly classified and misclassified images. The p-value was not reported, as it becomes uninformative with large sample sizes, where even negligible correlations tend to appear statistically significant.

2.7 Hardware and Software

This project was processed on the IUNR HPC cluster using node 301, an HPE Apollo 6500 Gen10+ node running Rocky Linux 8. The node is equipped with 8 NVIDIA L40S GPUs (48 GB each), dual AMD EPYC 7742 processors, 512 cores and 5800 GB of storage, providing the computational power needed for high-performance tasks.

The software environment was set up using micromamba, a lightweight version of conda, to manage the dependencies and packages required for the project. An environment file is provided in the GitHub repository to reproduce the environment. The Python version and used packages are as follows:

- Python 3.10.16
- NumPy 2.2.4
- pandas 2.2.3
- Matplotlib 3.10.1
- scikit-learn 1.6.1
- PyTorch 2.5.1
- PyTorch Lightning 2.5.1
- Pillow 9.4.0

3 Results

3.1 Detection

One of the main goals of the initial detection through the dataset was to identify which images were suitable for the training routine. The selection of images and the discarded ones are shown in Table 3; the data is divided into four categories. The effect of the selection is displayed for detection thresholds of 0.25 and 0.5, with the latter being the one finally applied. On the image level, the most affected category is `soricidae`, with 23% of the images being discarded, followed by `mustela_erminea` with 20%. The other categories, `apodemus_sp` and `cricetidae`, were only affected by 7% and 8%, respectively. On the sequence level, the situation is very different, with only the `mustela_erminea` category being seriously affected, with 23% of the sequences being discarded. For the other categories, only around 1% of the sequences were discarded. Some visual examinations of the results of the MD were performed to gain a better understanding of the outcomes — some of these insights are presented in the following, focusing on the `mustela_erminea` category. Some of the highest-confidence detections for this category are shown in Figure 6; all the images display a properly detected animal. The inspection of images with no detection within a threshold of 0.25 is shown in Figure 7; this is a hand-picked selection of images that were interesting for the analysis. Most of the images with no detection were actually empty or showed only the tip of the animal's tail, as seen in examples (c)–(g). Some showed forms of obstruction, such as examples (a) and (b), with the dragged-in tube in (b) being a common one. There were also several images with an animal clearly visible but no detection, such as in examples (h)–(l). This occurred far more often for individuals with white fur than for those with brown fur — this observation is based on visual inspection and has not been statistically evaluated.

3.2 Classification Performance

All classification models performed well on their respective test sets. The BalAcc scores for each model architecture are presented in Table 4 for the pretrained and non-pretrained variants. Generally, smaller architectures achieved slightly higher scores than larger ones, although these differences remained within one standard deviation. In particular, the pretrained EfficientNet-B0 reached the highest BalAcc of 0.992 ± 0.004. Applying sequence-level classification to the image-level predictions improved BalAcc for every model architecture, but only by 0.001 to 0.005, which again falls within a single standard deviation.

Pretrained versions consistently outperformed those trained from scratch, as shown in Figure 8 and Table 4. EfficientNet-B0 with pretraining performed uniformly well across all folds, with only one outlier, while the non-pretrained version scored slightly lower on average and showed greater variability. In contrast, DenseNet-169 showed less of a gap between pretrained and non-pretrained variants, with the pretrained version performing slightly better but also showing increased spread across folds. ResNet-50 values were more dispersed for both variants, yet the pretrained model still held a clear advantage.

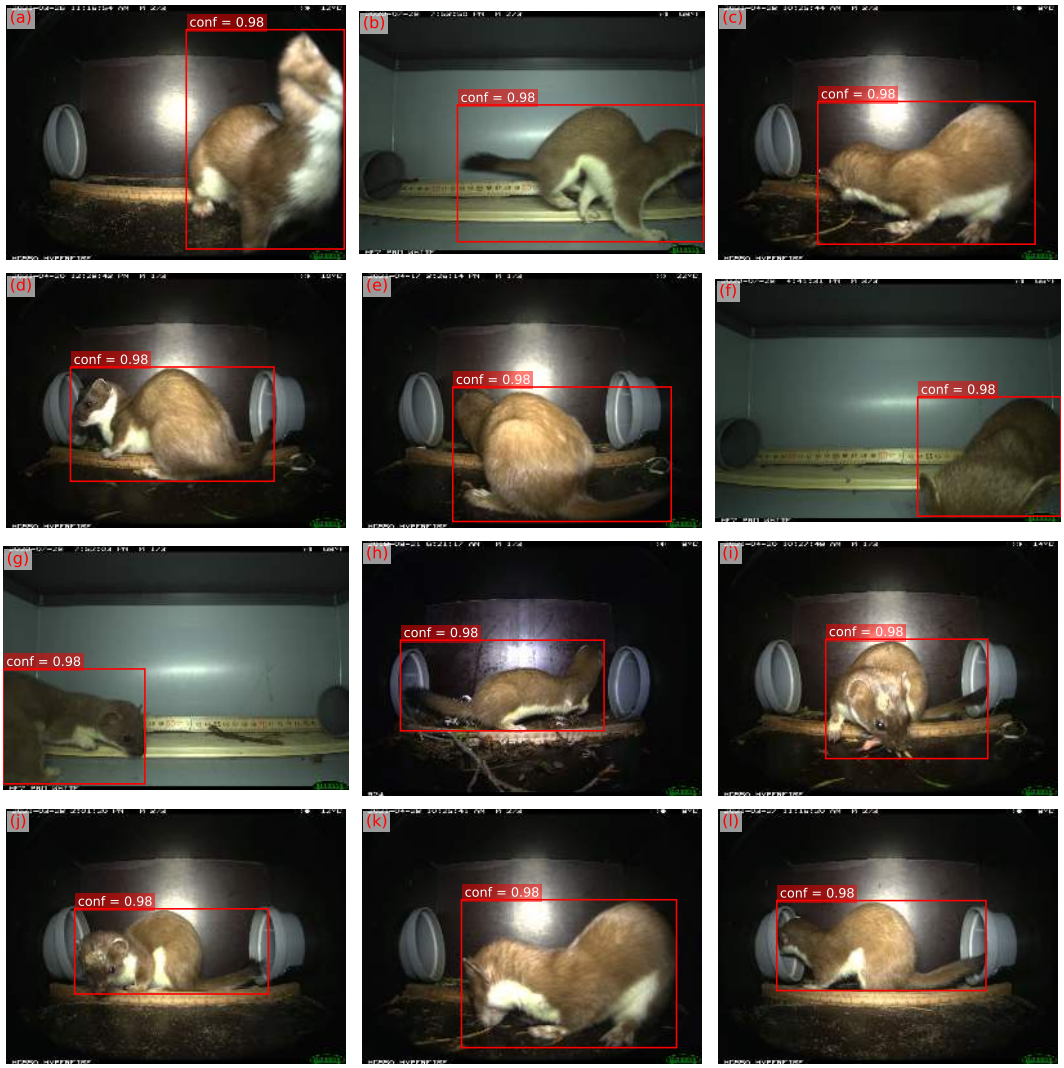

Figure 6: Highest-confidence detections for the `mustela_erminea` category.

Figure 7: Hand-picked selection of `mustela_erminea` images with no detection at a threshold of 0.25.

Table 3: Data loss dew to MD output for confidence thresholds of 0.25 and 0.5 by category for the image and the sequence level.

	apodemus_sp	cricetidae	mustela_erminea	soricidae
Image Level				
Total available	279,492	156,350	16,465	16,645
Threshold of 0.25:				
Lost	16,609	9,248	2,914	3,529
Percentage lost	6%	6%	18%	21%
Available	262,883	147,102	13,551	13,116
Threshold of 0.5:				
Lost	19,417	11,948	3,290	3,865
Percentage lost	7%	8%	20%	23%
Available	260,075	144,402	13,175	12,780
Sequence Level				
Total available	13,669	5,329	1,035	2,107
Threshold of 0.25:				
Lost	51	40	211	7
Percentage lost	0%	1%	20%	0%
Available	13,618	5,289	824	2,100
Threshold of 0.5:				
Lost	56	46	239	13
Percentage lost	0%	1%	23%	1%
Available	13,613	5,283	796	2,094

Finally, ViT-B/16 displayed the largest benefit from pretraining, alongside the greatest fold-to-fold variability in both its pretrained and non-pretrained versions.

3.3 Best Performing Model Architecture

The model architecture that achieved the best performance, as measured by BalAcc, was EfficientNet-B0 with pretraining. Figure 9 shows the validation metrics for each cross-validation fold across all epochs; note that accuracy is reported in this figure instead of BalAcc. For all folds, the best version, defined by the lowest validation loss, occurred within the first 3 epochs, while the accuracy kept increasing. Its performance per category is shown in Table 5. The class it performed best on was `mustela_erminea`, with a value of 0.999 for all metrics — this happens to be one of the classes with relatively few samples available. It performed the worst on the other underrepresented class, `soricidae`, with a precision of 0.971, recall of 0.979 and F1-score of 0.975. For the other, more represented classes, the model achieved very high scores above 0.99 for all metrics.

Table 4: BalAcc of all models — mean ± standard deviation; best values highlighted.

Model	Pretrained	Params (M)	Image BalAcc	Sequence BalAcc
efficientnet_b0	Yes	4	0.9921 ± 0.004	0.9947 ± 0.002
densenet169	Yes	12	0.9904 ± 0.004	0.9939 ± 0.002
resnet50	Yes	23	0.9899 ± 0.004	0.9934 ± 0.002
vit_b_16	Yes	85	0.9885 ± 0.005	0.9933 ± 0.002
efficientnet_b0	No	4	0.9856 ± 0.005	0.9898 ± 0.003
densenet169	No	12	0.9863 ± 0.006	0.9899 ± 0.002
resnet50	No	23	0.9850 ± 0.004	0.9888 ± 0.003
vit_b_16	No	85	0.9767 ± 0.006	0.9856 ± 0.004

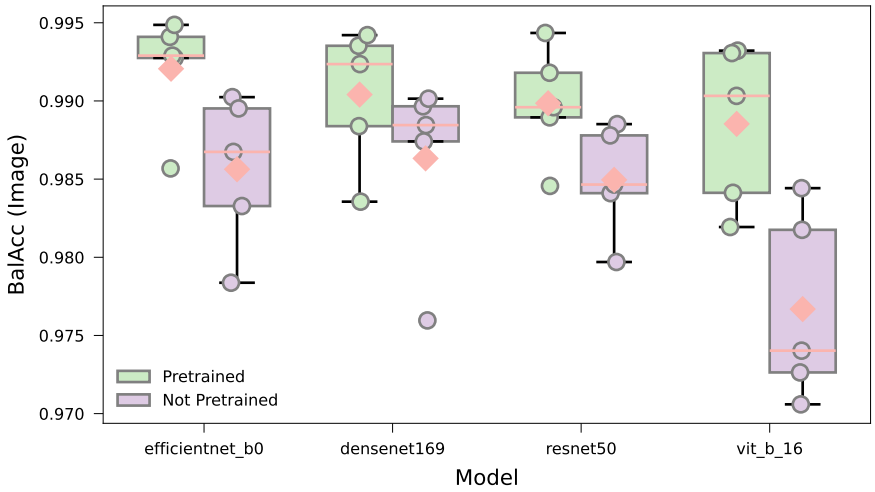

Figure 8: BalAcc of each model at image level across folds, shown separately for pretrained and non-pretrained variants. Individual fold results are plotted as points, the mean BalAcc is marked by a diamond, and the median is indicated by a horizontal line.

The normalized CM for the best model is shown in Figure 10. It shows that there were essentially no false positives for the class `mustela_erminea`, while it was rarely confused with other classes. The most frequent misclassification occurred with `soricidae` being falsely classified as `apodemus_sp`, with a value of 0.0184. All other classes had a false positive rate of less than 0.006.

The Spearman's rank correlation coefficient between the detection and the classification confidence is:

For correctly classified samples: $\rho = 0.092$
For misclassified samples: $\rho = 0.276$

This suggests that there is a very weak positive correlation between detection confidence and classification confidence for correctly classified images, and a weak positive correlation for misclassified images.

Table 5: Class-wise precision, recall, F1-score, and support for the best-performing model.

Class	Precision	Recall	F1-Score	Support
`apodemus_sp`	0.996	0.997	0.996	260075
`mustela_erminea`	0.999	0.999	0.999	13175
`cricetidae`	0.995	0.993	0.994	144402
`soricidae`	0.971	0.979	0.975	12780

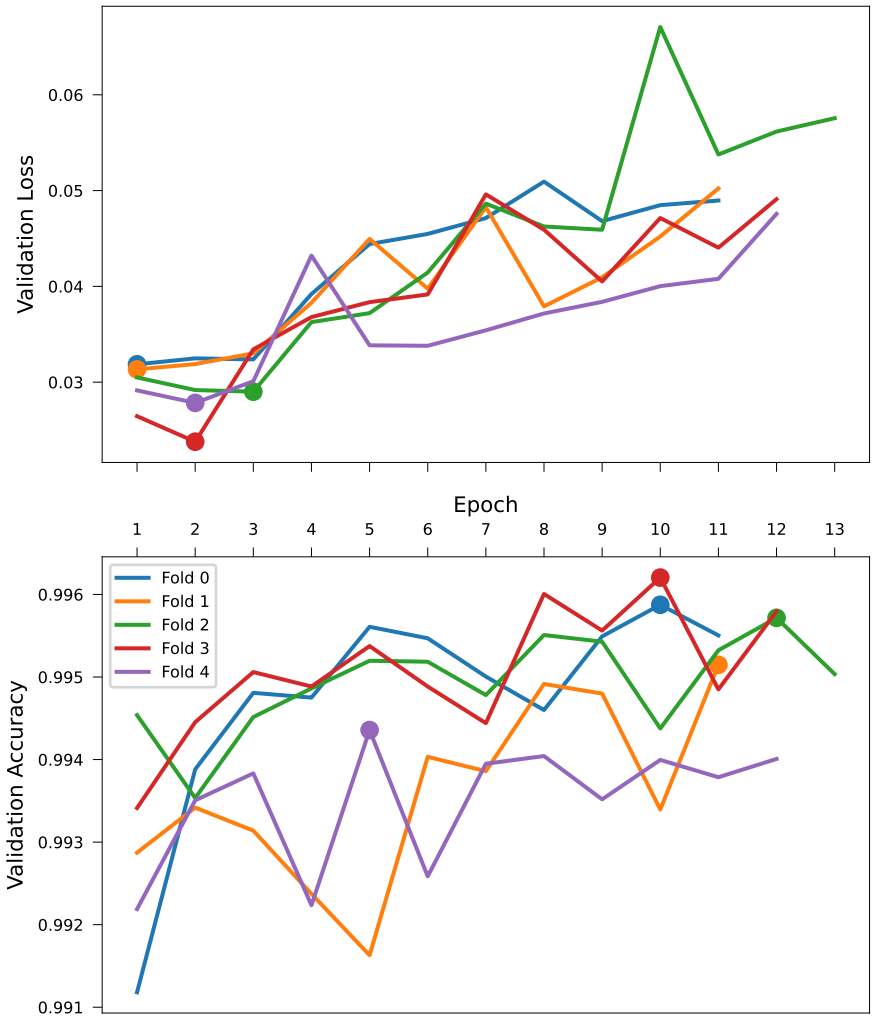

Figure 9: Validation loss and accuracy of the pretrained EfficientNet-B0 across all cross-validation folds. In the loss subplot, the lowest value per fold is marked with a dot, indicating the best-performing epoch. In the accuracy subplot, the highest value per fold is similarly marked.

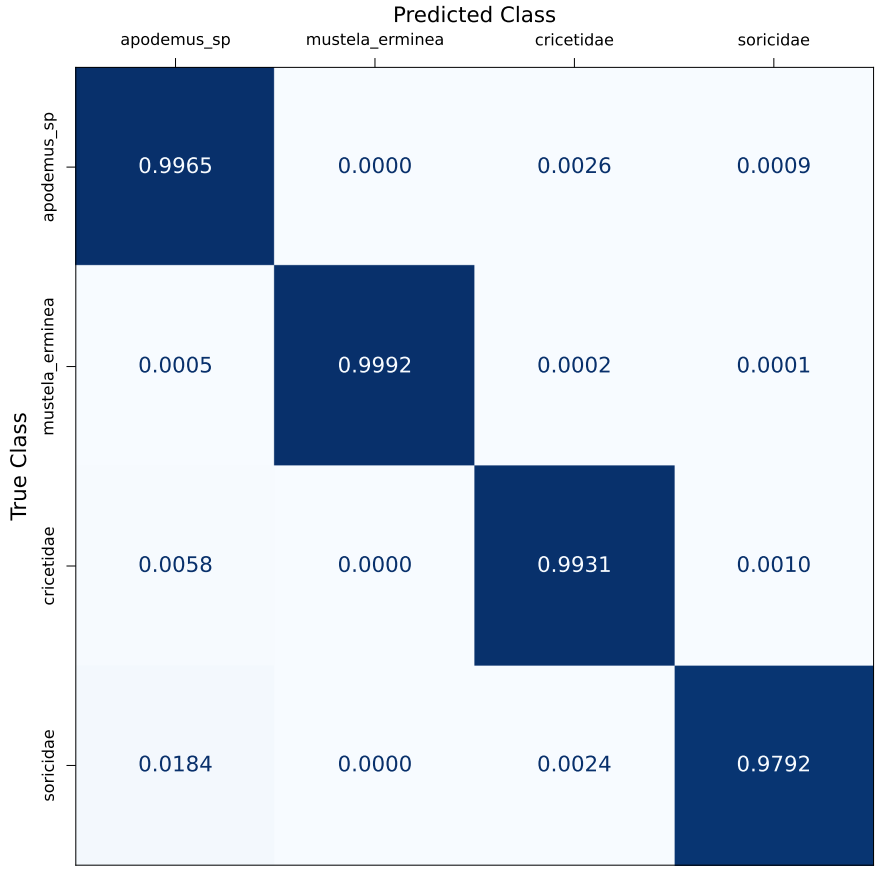

Figure 10: Normalized CM for the best-performing model.

4 Discussion

4.1 Detection

The detection process significantly influenced the dataset composition, particularly on the sequence level for the `mustela_erminea` category. This category experienced higher data loss, likely due to the relative size of the species —– it is much larger than the other species captured by these MammaliaBox camera traps. This larger size may have resulted in more frequent bad shots due to closeness to the camera or the speed at which they pass through the box — many images showed only the tip of the animal's tail disappearing out of the box. The difficulty of properly detecting stoats using camera traps is a known issue examined, for example, by Croose et al. (2022). Another interesting observation was that the white-fur individuals were more often not detected than the brown-fur individuals. An explanation has yet to be found for why this was only the case on the sequence level, where the `mustela_erminea` category was disproportionately affected. On the image level, the `soricidae` category was the most affected, with 23% of the images lost but only 1% of the sequences.

Visual inspection revealed a mixture of exemplary BBox detections alongside notable inaccuracies, such as missed detections in images with obstructions or partial visibility of animals. Looking through misclassified images with high confidence values revealed some interesting insights. One was that quite a few false positive detections were made, leading to surprisingly high-confidence classifications. Some of these images were empty or contained no animal at all, as shown in Figure 11, which includes examples where plant parts were misdetected, animals possibly appeared in very dark areas, or very small, not clearly visible objects were present. Another interesting finding was that quite a few of the images contained a snail in the highest-confidence BBox — refer to Figure 12 for a hand-picked selection of these examples. These misclassifications highlight the need for enhanced object discrimination capabilities or the introduction of an additional class for non-target species.

The dilemma of not missing potentially relevant detections while also not polluting the dataset with image noise remains a challenge. Simply lowering the detection threshold to reduce missed animals is not a viable solution. As Leorna and Brinkman (2022) showed, reducing the MD confidence threshold substantially increased recall, but at the cost of a significant increase in false positives, such as empty frames or vegetation. This supports the need for a more sophisticated approach to reduce missed detections while maintaining a clean dataset.

4.2 Model Performance

All tested models achieved high performance in image classification tasks, demonstrating their suitability for automating small mammal identification. Pretrained models slightly outperformed those trained from scratch, underscoring the value of transfer learning. The

Figure 11: Examples of false positives with high classification confidence but no animal present.

slightly better balanced accuracy of pretrained models is just one of the benefits of using pretrained models, as they also require less training time and computational resources and could potentially be trained with less data. There are various studies exploring or applying transfer learning to camera trap images, supporting its potential (Beery et al., 2018; Doan & Le-Thi, 2024; Hopkins et al., 2024; Ramesh et al., 2025; Stančić et al., 2022).

Interestingly, the smaller the models, the better they performed on the image-level classification task, highlighting the fact that bigger is not always better. Smaller models, if complex enough for the task, are always preferable since they require fewer computational resources and are faster to train. The proper scaling of a model is an important aspect of model selection, as emphasized, for example, by Tan and Le (2019). Examining Figure 8 again, one could speculate about a trend for smaller models to perform more consistently across folds when trained from scratch — but for some reason, this is not the case for the EfficientNet-B0 model. The observations from this study suggest that smaller, computationally efficient models are sufficient and therefore preferable for the classification tasks at hand.

4.3 Best Model Architecture

The pretrained EfficientNet-B0 emerged as the best-performing architecture, achieving the highest BalAcc. Figure 9 shows the validation metrics for each cross-validation fold across all epochs; note that accuracy is reported in the figure instead of BalAcc. For all folds, the best version, defined by the lowest validation loss, occurred within the first few epochs, while accuracy continued to improve. While an increase in accuracy indicates more correct predictions, the loss can still worsen because it also penalizes correct predictions made with low confidence and incorrect predictions made with high confidence.

Since the training dataset was quite large, contained only four classes and was intensively vetted using the MD, the model was able to learn the task quite quickly. The initial supposition that a higher detection confidence would lead to a higher classification confidence could not be confirmed by the Spearman's rank correlation coefficient. It even suggested that the correlation is stronger for incorrect classifications. Examining Figure 13, which shows the relationship between detection confidence and classification confidence, reveals that classification confidence is generally very high — across all detection confidence values and even for incorrect classifications. There seem to be more values located in the upper right corner, which would suggest that higher detection confidence leads to higher classification confidence. However, since there are so many samples with classification confidence close to 1 across the range of detection confidence values, Spearman's rank may not be a good measure for this relationship.

These many very high classification confidence values could be explained by the model's architecture. In the last layers of the model, the learned features are mapped to the four classes and then normalized so the values sum to one. If the model is entirely confident that an input does not belong to the first three classes and assigns a very small value to the fourth class (e.g., $[0\ ;\ 0\ ;\ 0\ ;\ 10^{-5}]$), this distribution will be normalized to $[0\ ;\ 0\ ;\ 0\ ;\ 1]$, resulting in a prediction for the fourth class with maximum confidence. As shown by Hendrycks and Gimpel (2018), softmax-based confidence scores often fail to meaningfully reflect true uncertainty, especially for Out-Of-Distribution (OOD) inputs. Therefore, this evaluation lacks meaningfulness. It could be repeated using logits without normalization. Nonetheless, the assumption that a non-target class would benefit the classification task remains valid.

4.4 Limitations

The primary limitation of the current methodology is the absence of an explicit non-target class, forcing models into potentially incorrect predictions, exemplified by the snail misclassification issue shown in Figure 12. Further, despite improvements, the MD still misses a significant number of potentially relevant detections. This limitation is particularly pronounced for rare species, where insufficient data reduces detection reliability. The loss of sequences for the `mustela_erminea` category, as shown in Table 3, illustrates this issue. Every sequence completely lost is a potentially missed sighting of a rare species, which could have provided valuable insights for conservation efforts.

The step of sequence classification is performed on the model's outputted classification confidence values per class and image. Since the information from the logits is heavily distorted by the normalization step, sequence classification does not yet reach its full potential. This approach might need to be revisited to improve the reliability of sequence-level results. The current approach also depends on manual preprocessing of the dataset to group the images into sequences — this process would benefit from automation.

Moreover, the current approach is heavily dependent on data availability. Rare species inherently have fewer data points, constraining model training and potentially biasing predictions. To include an additional class would require a sufficient number of samples to ensure reliable model performance. Generally, data acquisition is the most resource-intensive part of most machine learning projects.

Figure 12: Hand-picked selection of misclassified images where a snail appears in the highest-confidence BBox.

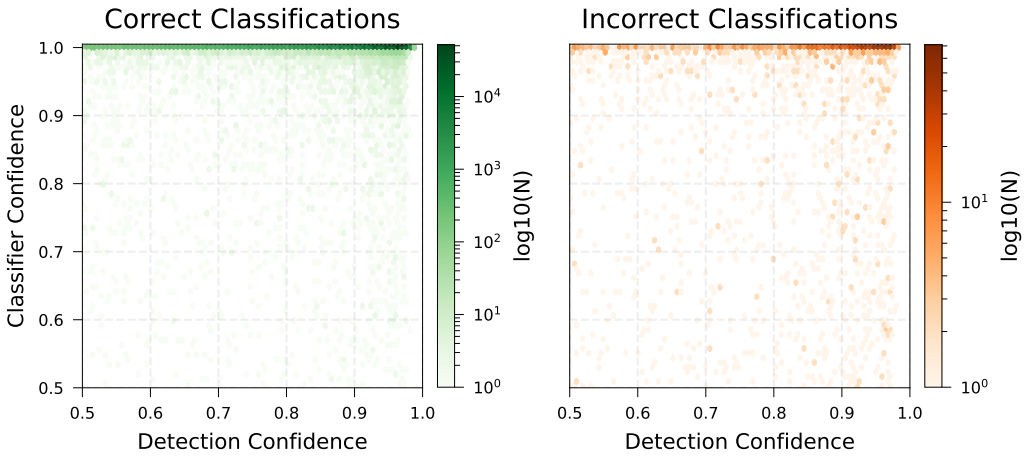

Figure 13: Hexbin plot of detection confidence vs. classification confidence for correct and incorrect predictions. Color scale indicates $\log_{10}$-binned counts.

5 Conclusion and Outlook

5.1 Conclusion

This thesis demonstrated the effectiveness of deep learning models for detecting and classifying small mammals from camera trap images. The pretrained EfficientNet-B0 model provided superior classification accuracy, quickly converging and demonstrating robustness across validation folds. The integration of automated detections utilizing MD, while beneficial, revealed some room for improvement, particularly concerning misdetections and missed out detections. It was found that some finetuning of the MD to the specific MammaliaBox camera trap setup could improve the results. Despite these limitations, the processing pipeline and the trained models provide a promising start for developing an applicable tool and a workflow to reduce manual effort.

5.2 Outlook

Future enhancements should focus on addressing current limitations by introducing an explicit category for non-target species to improve classification accuracy and reduce false predictions with high confidence. A possible approach could be the OOD detection suggested by Hendrycks and Gimpel (2018), which could be used to identify images that do not belong to any of the known categories. To allow for broader application of the model, additional categories would be needed — such as the here explicitly ignored *Glis glis*, a common small mammal in Switzerland on the IUCN (2025) Red List of Threatened Species. To improve the model's robustness while adding more categories — possibly with limited data availability — data augmentation techniques could be implemented, as they have been shown to enhance model performance and generalization (Shorten & Khoshgoftaar, 2019).

The sequence-based classification did not improve classification performance significantly as it was done in this thesis. It still seems a very promising approach, since camera trap images are often taken in sequences and the sequence information could be used to improve classification performance. Further research could explore different options for sequence-based classification. As a first step, the model's output could be evaluated on the logits level to determine how this information could be utilized to improve classification performance. More sophisticated approaches could involve utilizing temporally aware models as demonstrated by Muhammad et al. (2024). Since the initial detection process using MD could still be improved, utilizing sequence information for the detection process could be explored further. Zotin and Proskurin (2019) demonstrated a promising non-DL approach for detecting animals in camera trap images using sequence information. There is still a better approach needed to determine the sequence length than the method applied by the Wildlife@Campus team using the EXIF timestamp. Better sequence information is available visually imprinted on top of the images, which could be extracted using an Optical Character Recognition (OCR) model — refer to Figure 14. Since the information is imprinted on the images with high contrast and no distortion, this should be a

straightforward task for an OCR model. The fact that the first information imprinted on the image is the timestamp — which is also available in the EXIF metadata — could be used to match the OCR output with the EXIF information to quickly determine the reliability of the OCR output.

Figure 14: Top 5% of a camera trap image — it was processed with the `Tesseract` OCR model. The output string was: `2019-09-04 1:02:09 AM M 1/3 #9 10°C`.

Another important step toward practical application is to develop an interface or integrated software solution. This would allow researchers to actually use the model in their workflows for monitoring small mammal populations. This could be done in a way that integrates manual review — which is still necessary for reliable results — as a means of further improving the model. Currently, the data processing pipeline still depends on manual preprocessing of the image metadata to extract sequence information. This step would benefit from automation to streamline the workflow — ideally, the input for the classification task would be just the raw images as retained from the camera trap.

6 Declaration

6.1 Declaration of AI Usage

GitHub Copilot was active during all coding tasks as well as text writing in this thesis. While it was primarily intended to assist with LaTeX formatting, it also provided suggestions for the text itself. For the actual coding tasks, it was used to help solve problems, generate code snippets and played a significant role in debugging.

ChatGPT Model o4-mini was used to assist with coding, problem-solving and debugging. **ChatGPT GPT-4o** and **GPT-4.5** were used to support content research, thesis structuring, writing and refining the text. The Abstract was initially generated and translated using **ChatGPT GPT-4o**. Both versions where thoroughly reviewed and manually edited. Every paragraph was finally fed into **GPT-4o** for a final spelling and grammar check — all changes were manually reviewed using a git diff viewer to ensure that no unwanted edits were introduced.

Zürcher Hochschule
für Angewandte Wissenschaften

N-FO-Formular Statement of Authorship for

Student Work

Life Sciences und Facility Management

Education Unit

Statement of Authorship for Student Work at the School of Life Sciences and Facility Management

By submitting the enclosed

☐ Project
☐ Literature review
☐ Course work
☐ Minor paper
☒ Bachelor's thesis
☐ Master's thesis (tick as appropriate)

the student affirms independent completion of the(ir) work without outside help.

The undersigned student declares that all printed and electronic sources used in the text and the bibliography are correctly indicated, i.e., that the work does not contain any plagiarism (no parts that have been taken in whole or in part from another's or his/her own text or from another's or his/her own work without clear identification and without stating the source).

In the event of misconduct of any kind, Paragraph 39 and Paragraph 40 of the General Academic Regulations for Bachelor's and Master's degree programmes at the Zurich University of Applied Sciences (dated 29 January 2008) and the provisions of the Disciplinary Measures of the University Regulations shall apply.

Location, date: Student signature:

Neuhausen, 2025-06-23 Julian Kraft

Note on submitting the Statement of Authorship:

Direct submission of the work: This Statement of Authorship is to be inserted in the appendix of the ZHAW version of all work with original signatures and date (copies will not be accepted).

Submission of the work via Complesis: The Statement of Authorship should be made directly in Complesis by clicking as directed and should not be inserted in the appendix of the work.

Erlassverantwortliche/-r	LeiterIn Stabsbereich Bildung	Ablageort	2.05.00 Lehre Studium
Beschlussinstanz	LeiterIn Stab	Publikationsort	Public
Genehmigungsinstanz			

Version	Beschluss	Beschlussinstanz	Inkrafttreten	Beschreibung Änderung
1.0.0	15.03.2022	LeiterIn Stab	15.03.2022	Originalversion
1.1.0	15.04.2024	Leiter:in Stab	01.5.2024	Adding clarification on self-plagiarism

References

Aegerter, S. (2019). *Monitoring von Kleinmusteliden, Schläfern und anderen Kleinsäugern : Weiterentwicklung der Nachweismethoden mit Fotofalle*. ZHAW Zürcher Hochschule für Angewandte Wissenschaften. https://doi.org/10.21256/zhaw-19209

BAFU. (2019). *Liste der National Prioritären Arten und Lebensräume. In der Schweiz zu fördernde prioritäre Arten und Lebensräume* (Umwelt-Vollzug Nr. 1709). Bundesamt für Umwelt. Bern. https://www.bafu.admin.ch/uv-1709-d

Beery, S., Horn, G. van, & Perona, P. (2018, July 25). *Recognition in Terra Incognita*. arXiv: 1807.04975 [cs]. https://doi.org/10.48550/arXiv.1807.04975

Beery, S., Morris, D., & Yang, S. (2019, July 15). *Efficient Pipeline for Camera Trap Image Review*. arXiv: 1907.06772 [cs]. https://doi.org/10.48550/arXiv.1907.06772

Brondízio, E. S., Settele, J., Díaz, S., & Ngo, H. T. (Eds.). (2019). *The global assessment report of the intergovernmental science-policy platform on biodiversity and ecosystem services*. Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services (IPBES).
OCLC: 1336011247.

Cardinale, B. J., Duffy, J. E., Gonzalez, A., Hooper, D. U., Perrings, C., Venail, P., Narwani, A., Mace, G. M., Tilman, D., Wardle, D. A., Kinzig, A. P., Daily, G. C., Loreau, M., Grace, J. B., Larigauderie, A., Srivastava, D. S., & Naeem, S. (2012). Biodiversity loss and its impact on humanity. *Nature*, *486*(7401), 59–67. https://doi.org/10.1038/nature11148

Chen, G., Han, T. X., He, Z., Kays, R., & Forrester, T. (2014). Deep convolutional neural network based species recognition for wild animal monitoring. *2014 IEEE International Conference on Image Processing (ICIP)*, 858–862. https://doi.org/10.1109/ICIP.2014.7025172

Clucas, B., & McCluskey, S. L. (2025). Camera trap method effectively identifies small mammal species in forested habitats. *California Fish and Wildlife Journal*, *111*(2). https://doi.org/10.51492/cfwj.111.8

Croose, E., Hanniffy, R., Hughes, B., McAney, K., MacPherson, J., & Carter, S. P. (2022). Assessing the detectability of the Irish stoat Mustela erminea hibernica using two camera trap-based survey methods. *Mammal Research*, *67*(1), 1–8. https://doi.org/10.1007/s13364-021-00598-z

Delisle, Z. J., Flaherty, E. A., Nobbe, M. R., Wzientek, C. M., & Swihart, R. K. (2021). Next-Generation Camera Trapping: Systematic Review of Historic Trends Suggests Keys to Expanded Research Applications in Ecology and Conservation. *Frontiers in Ecology and Evolution*, *9*. https://doi.org/10.3389/fevo.2021.617996

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 248–255. https://doi.org/10.1109/CVPR.2009.5206848

Doan, T.-N., & Le-Thi, D.-N. (2024). Wildlife Species Classification from Camera Trap Images Using Fine-tuning EfficientNetV2. *International Journal of Intelligent Engineering and Systems*, *17*(6), 624–638. https://doi.org/10.22266/ijies2024.1231.48

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021,

June 3). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. arXiv: 2010.11929 [cs]. https://doi.org/10.48550/arXiv.2010.11929

Gomez Villa, A., Salazar, A., & Vargas, F. (2017). Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks. *Ecological Informatics*, *41*, 24–32. https://doi.org/10.1016/j.ecoinf.2017.07.004

Graf, R. F., Dietrich, A., Honetschläger, N., Kryszczuk, K., Palmisano, M., Pothier, J. F., Ratnaweera, N., Reifler-Bächtiger, M., Rhyner, N., & Treichler, R. (2022). Wildlife@Campus: Kleine Säugetiere im Fokus.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition, 770–778. Retrieved June 2, 2025, from https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html

Hendrycks, D., & Gimpel, K. (2018, October 3). *A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks*. arXiv: 1610.02136 [cs]. https://doi.org/10.48550/arXiv.1610.02136

Hernandez, A., Miao, Z., Vargas, L., Beery, S., Dodhia, R., Arbelaez, P., & Ferres, J. M. L. (2024, November 29). *Pytorch-Wildlife: A Collaborative Deep Learning Framework for Conservation*. arXiv: 2405.12930 [cs]. https://doi.org/10.48550/arXiv.2405.12930

Hopkins, J., Santos-Elizondo, G. M., & Villablanca, F. (2024). Detecting and monitoring rodents using camera traps and machine learning versus live trapping for occupancy modeling. *Frontiers in Ecology and Evolution*, *12*. https://doi.org/10.3389/fevo.2024.1359201

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2261–2269. https://doi.org/10.1109/CVPR.2017.243

IUCN. (2025). *The IUCN Red List of Threatened Species*. The IUCN Red List of Threatened Species. Retrieved June 23, 2025, from https://www.iucnredlist.org

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, *25*. Retrieved June 19, 2025, from https://proceedings.neurips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Leorna, S., & Brinkman, T. (2022). Human vs. machine: Detecting wildlife in camera trap images. *Ecological Informatics*, *72*, 101876. https://doi.org/10.1016/j.ecoinf.2022.101876

Littlewood, N. A., Hancock, M. H., Newey, S., Shackelford, G., & Toney, R. (2021). Use of a novel camera trapping approach to measure small mammal responses to peatland restoration. *European Journal of Wildlife Research*, *67*(1), 12. https://doi.org/10.1007/s10344-020-01449-z

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021, March 25). *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. arXiv.org. Retrieved June 19, 2025, from https://arxiv.org/abs/2103.14030v2

Loshchilov, I., & Hutter, F. (2017, May 3). *SGDR: Stochastic Gradient Descent with Warm Restarts*. arXiv: 1608.03983 [cs]. https://doi.org/10.48550/arXiv.1608.03983

Loshchilov, I., & Hutter, F. (2019, January 4). *Decoupled Weight Decay Regularization*. arXiv: 1711.05101 [cs]. https://doi.org/10.48550/arXiv.1711.05101

Muhammad, S., Xiang, W., Mann, S., Han, K., & Nair, S. (2024). TemporalSwin-FPN Net: A Novel Pipeline for Metadata-Driven Sequence Classification in Camera Trap Imagery. *2024 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, 616–623. https://doi.org/10.1109/DICTA63115.2024.00094

Norouzzadeh, M. S., Nguyen, A., Kosmala, M., Swanson, A., Palmer, M. S., Packer, C., & Clune, J. (2018). Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences*, *115*(25). https://doi.org/10.1073/pnas.1719367115

Pedregosa, F., Pedregosa, F., Varoquaux, G., Varoquaux, G., Org, N., Gramfort, A., Gramfort, A., Michel, V., Michel, V., Fr, L., Thirion, B., Thirion, B., Grisel, O., Grisel, O., Blondel, M., Prettenhofer, P., Prettenhofer, P., Weiss, R., Dubourg, V., … Cournapeau, D. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Ramesh, K., Darwish, M., Zibli, A. S. A., Miller, N. C., Sajun, A. R., Zualkernan, I., Habib, A., & Gardner, A. (2025). Exploring the Generalizability of Transfer Learning for Camera Trap Animal Image Classification. In R. Meo & F. Silvestri (Eds.), *Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (pp. 212–227). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-74627-7_15

Ratnaweera, N. (2021). *Wildlife @ Campus: ProgressReports*. Wildlife @ Campus: ProgressReports. Retrieved June 3, 2025, from https://github.zhaw.ch/pages/Wildlife-Campus/ProgressReports/

Schneider, D., Lindner, K., Vogelbacher, M., Bellafkir, H., Farwig, N., & Freisleben, B. (2024). Recognition of European mammals and birds in camera trap images using deep neural networks. *IET Computer Vision*, *18*(8), 1162–1192. https://doi.org/10.1049/cvi2.12294

Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, *6*(1), 60. https://doi.org/10.1186/s40537-019-0197-0

Simonyan, K., & Zisserman, A. (2015, April 10). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv: 1409.1556 [cs]. https://doi.org/10.48550/arXiv.1409.1556

Stančić, A., Vyroubal, V., & Slijepčević, V. (2022). Classification Efficiency of Pre-Trained Deep CNN Models on Camera Trap Images. *Journal of Imaging*, *8*(2), 20. https://doi.org/10.3390/jimaging8020020

Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1–9. https://doi.org/10.1109/CVPR.2015.7298594

Tabak, M. A., Norouzzadeh, M. S., Wolfson, D. W., Sweeney, S. J., Vercauteren, K. C., Snow, N. P., Halseth, J. M., Di Salvo, P. A., Lewis, J. S., White, M. D., Teton,

B., Beasley, J. C., Schlichting, P. E., Boughton, R. K., Wight, B., Newkirk, E. S., Ivan, J. S., Odell, E. A., Brook, R. K., … Miller, R. S. (2019). Machine learning to classify animal species in camera trap images: Applications in ecology. *Methods in Ecology and Evolution*, *10*(4), 585–590. https://doi.org/10.1111/2041-210X.13120

Tan, M., & Le, Q. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *Proceedings of the 36th International Conference on Machine Learning*, 6105–6114. Retrieved June 2, 2025, from https://proceedings.mlr.press/v97/tan19a.html

Thomsen, P. F., & Willerslev, E. (2015). Environmental DNA – An emerging tool in conservation for monitoring past and present biodiversity. *Biological Conservation*, *183*, 4–18. https://doi.org/10.1016/j.biocon.2014.11.019

Vélez, J., Castiblanco-Camacho, P. J., Tabak, M. A., Chalmers, C., Fergus, P., & Fieberg, J. (2022, February 4). *Choosing an Appropriate Platform and Workflow for Processing Camera Trap Data using Artificial Intelligence*. arXiv: 2202.02283 [cs]. https://doi.org/10.48550/arXiv.2202.02283

Yarnell, R. W., Pacheco, M., Williams, B., Neumann, J. L., Rymer, D. J., & Baker, P. J. (2014). Using occupancy analysis to validate the use of footprint tunnels as a method for monitoring the hedgehog Erinaceus europaeus. *Mammal Review*, *44*(3–4), 234–238. https://doi.org/10.1111/mam.12026

Zotin, A. G., & Proskurin, A. V. (2019). ANIMAL DETECTION USING A SERIES OF IMAGES UNDER COMPLEX SHOOTING CONDITIONS. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLII-2-W12*, 249–257. https://doi.org/10.5194/isprs-archives-XLII-2-W12-249-2019