

12.06.2026 – BACHELOR THESIS

Automatic recognition of birds and bats in audio samples

Department of Computer Science

OST - Ostschweizer Fachhochschule

Campus Rapperswil-Jona

AUTHOR: Oliver Dietsche

ADVISOR: Prof. Hannes Badertscher

CO-ADVISOR: Patrik Müller

PROJECT PARTNER: Dr. Paul Haverkamp
(Institute for Landscape and Open Space, ILF OST)

EXTERNAL CO-EXAMINER: Gabriel Sidler

INTERNAL CO-EXAMINER: Thomas Corbat

Abstract

The Institute for Landscape and Open Space (ILF), the industrial partner of this thesis, wants to monitor bird and bat activity on the OST Rapperswil campus. In preparation, four AudioMoth recording devices were deployed across the campus, recording 1.77 TB of audio between September and December 2025. This thesis builds a tool that lets the ILF batch-process this data to automatically detect bird and bat vocalisations: birds are to be identified by species, whereas for bats detecting their presence is sufficient. The results are to be presented as a statistical overview for the ILF to work with.

Passive Acoustic Monitoring is a well-established field, and several open-source bird and bat detection models have proven themselves in practice. Given this maturity – and the lack of human resources to fully annotate the recordings – existing tools were assessed rather than a model trained from scratch. For birds, the BirdNET model was evaluated by running inference on 30 five-minute recordings, chosen primarily to cover the 16 defined focus bird species; its predictions were then annotated to confirm or reject each. For bats, no expert was available to verify calls, so an ensemble of three existing models – BatDetect2, BattyBirdNET and BatScope4 – was used to derive a majority-agreement set on the collected recordings.

The assessment of BirdNET showed a micro precision of 98.6% across the 16 focus species, dropping to 84.9% once the non-focus species are included. The non-focus species are captured as a by-product of the deliberately low confidence threshold, which accounts for the drop. This confirms BirdNET as a strong fit: precision is high at stricter thresholds, while low-confidence predictions remain available to inspect individually.

Of the three bat models, BatDetect2 and BattyBirdNET integrate with the pipeline through its adapter architecture, letting the researcher run both and apply each where it is strongest. Evaluated against the majority-agreement set, BatDetect2 reached the higher precision and additionally reports the exact time and frequency range of each call, whereas BattyBirdNET tended towards higher recall, providing more predictions for an expert to further inspect.

The implemented research dashboard lets the researcher filter detections by location, date and minimum confidence, and view them overall or broken down by species, location or five-minute recording. The ILF intends to use the monitoring data as a biodiversity indicator and to raise visitors' awareness by presenting these insights.

Acknowledgements

I would like to express my gratitude to everyone who supported me throughout this thesis.

I am grateful to my advisor, Prof. Hannes Badertscher, for the continuous support and guidance at every major step of this thesis. Special thanks also to my co-advisor, Patrik Müller, for his guidance, support and feedback through weekly meetings.

I thank the Interdisciplinary Center for Artificial Intelligence (ICAI) for granting access to their DGX-2, which was an effective tool for large-scale inference over all recordings with various adapters.

My thanks go to Dr. Paul Haverkamp and the other members of the Institute for Landscape and Open Space (ILF) for the collaboration on this thesis – providing the recordings, sharing their expertise on domain and campus species and assessing the predictions.

Finally, I thank Dr. Ruedi Bösch from Swiss Federal Institute for Forest, Snow and Landscape Research (WSL) for taking the time to meet in person and sharing his knowledge of BatScope4.

Contents

1	Introduction	5
1.1	Initial Situation	5
1.2	Assignment	5
1.3	Project Partner	5
1.4	Limitations	5
1.5	General Conditions	6
1.6	Report Structure	6
2	Requirements	7
2.1	Functional Requirements	7
2.1.1	Actors	7
2.1.2	Use Cases	7
2.2	Non-Functional Requirements	8
3	Domain Knowledge	11
3.1	Ecological Monitoring Context	11
3.1.1	Passive Acoustic Monitoring	11
3.1.2	Regional Birds	11
3.1.3	Regional Bats	12
3.1.4	Acoustic Characteristics of Birds and Bats	13
3.1.5	Bat Echolocation and Sonotypes	14
3.2	Existing Tools	14
3.2.1	Bird Classifiers	15
3.2.2	Bat Classifiers	16
3.2.3	Selection	17
3.3	Digital Audio Processing	18
3.4	Machine Learning for Acoustic Classification	18
3.5	Evaluation Metrics	18
4	Data Collection	20
4.1	Recording Setup	20
4.1.1	AudioMoth Devices	20
4.1.2	Data Exploration	20
4.2	Reference Set Creation	24
4.2.1	BirdNET Assessment Set	24
4.2.2	Annotator Setup	25
4.2.3	Bat Consensus Set	26
5	Design	29
5.1	Key Decisions	29
5.2	System Architecture	30
5.3	Pipeline Design	34
5.3.1	Model Integration	34
5.3.2	Prediction Processing	36
5.3.3	Configuration	37
5.4	Database Design	38

5.5 Research Dashboard Design	39
6 Implementation	40
6.1 Data Pipeline	40
6.1.1 Bird Model	41
6.1.2 Bat Models	41
6.1.3 Computing	42
6.2 Configuration and Command-Line Interface	43
6.3 Database and Loading	43
6.4 Research Dashboard	44
6.5 Testing and Code Quality	47
6.6 Packaging and Handover	47
7 Validation	48
7.1 Acoustic Monitoring Performance Analysis	48
7.1.1 BirdNET Precision	48
7.1.2 Bird Species Plausibility Check	50
7.1.3 Bat Detection Consensus	51
7.1.4 Bat Focus-Species Coverage	52
7.2 Processing Benchmark	53
7.2.1 Processing Robustness	54
7.3 Validation of Requirements	54
7.3.1 Validation of FRs	54
7.3.2 Validation of NFRs	54
8 Discussion of Results and Outlook	56
8.1 Results	56
8.2 Limitations	58
8.3 Possible Further Improvements	59
8.3.1 Local Fine-Tuning of the Models	59
8.3.2 In-Dashboard Confirmation and Playback	59
8.3.3 Throughput Optimisation	59
8.3.4 Deployment and Hosting	60
8.3.5 Public Outreach Site	60
8.3.6 Live Monitoring	60
A Glossary	61
B List of Tables	64
C List of Figures	65
D List of Tools	66
E Bibliography	67
F Appendix	71
F.1 Annotation Instructions	72
F.2 BirdNET Precision Across All Species	74
F.3 Detected Bird Species and Plausibility	76
F.4 BatDetect2 Sample Calls	83
F.5 Original Assignment	99

1 Introduction

This bachelor thesis is conducted at Eastern Switzerland University of Applied Sciences (OST) in cooperation with the ILF. It develops a tool to recognise birds and bats automatically and visualises the results interactively.

1.1 Initial Situation

Prior to the beginning of this thesis, the ILF deployed four audio recorders on the OST Rapperswil campus. Over the span of a few months they collected 1.77 TB of data. Recognising bats and birds manually in such quantities of data is infeasible, making automated processing the only practical option.

Mature off-the-shelf classifiers for this task already exist, among them BirdNET for birds [1] and BatDetect2 for bat echolocation [2]. Their published performance metrics, however, originate from different geographic regions, recording setups, and acoustic environments. Their effectiveness on the campus recordings therefore has to be validated before their predictions can be relied upon.

1.2 Assignment

The goal of this bachelor thesis is to develop a tool for the ILF to recognise birds and bats in audio samples. Bird sounds must be classified by species, whereas for bats the detection of their presence is sufficient. The assignment is as follows.

First, research into spectrogram-based and raw-audio approaches to audio processing has to be conducted. Existing classifiers such as BirdNET should be explored and considered. Software to detect and classify has to be implemented and validated against a ground-truth set created in collaboration with the ILF. The processing pipeline should be efficient enough to process large amounts of files in a suitable amount of time. Finally, a research dashboard has to be implemented to visualise the results and extract statistical properties. The system should be user-friendly enough for the ILF to process new data on their own. The complete original assignment can be read in Appendix F.5.

1.3 Project Partner

The ILF is located at OST in Rapperswil and studies landscape ecology and open-space development. Their representative for this thesis is Dr. Paul Haverkamp, who has an interest in monitoring biodiversity on campus. The collected data points to areas where their landscape management can improve and will additionally be shown to visitors to encourage engagement with biodiversity. As a means to this end, he is already employing photo traps to recognise terrestrial wildlife. The current outreach is limited to images from the photo traps displayed on screens at OST in Rapperswil and at events. Audio processing should add to that by expanding monitoring to aerial wildlife.

This monitoring effort is supported by RealTransform, a swissuniversities-funded sustainability initiative led by the University of Zurich with OST as a partner [3]. Their goal is to foster a university culture of sustainable development by conducting transdisciplinary real-world experiments.

1.4 Limitations

The limitations of this thesis are listed here.

- The recordings are collected periodically by the ILF and processed on their workstations; real-time capability is out of scope for this thesis.
- Bat detection is limited to presence; species-level classification is clearly defined as optional.
- Manually annotated, species-level ground-truth sets could not be created for the local recordings: the available expert time covered only smaller assessment sets, not full ground truth.

1.5 General Conditions

This bachelor thesis has been conducted during the spring semester 2026, from 16 February to 12 June, at the OST – Eastern Switzerland University of Applied Sciences. The advisor is Prof. Hannes Badertscher, who has participated in all milestone meetings, together with Dr. Paul Haverkamp and co-advisor Patrik Müller. In addition, weekly meetings have been held with Patrik Müller. The expected workload is approximately 360 hours.

1.6 Report Structure

The remainder of this report is organised as follows. Chapter 2 derives the functional requirement (FR) and non-functional requirements (NFRs) from the assignment. Chapter 3 provides the ecological and tooling background. Chapter 4 describes the recording setup, the collected dataset and the limited reference sets created for validation. Chapter 5 presents the system architecture, and Chapter 6 its concrete realisation. Chapter 7 evaluates the system against the requirements, and Chapter 8 discusses the results and gives an outlook on further work.

2 Requirements

This chapter defines the requirements derived from the formal thesis assignment (Chapter 1.2) and the needs communicated by the ILF [4].

2.1 Functional Requirements

This section describes the FRs of the system as illustrated in Figure 1. The use cases represent externally visible interactions between the actors and the system.

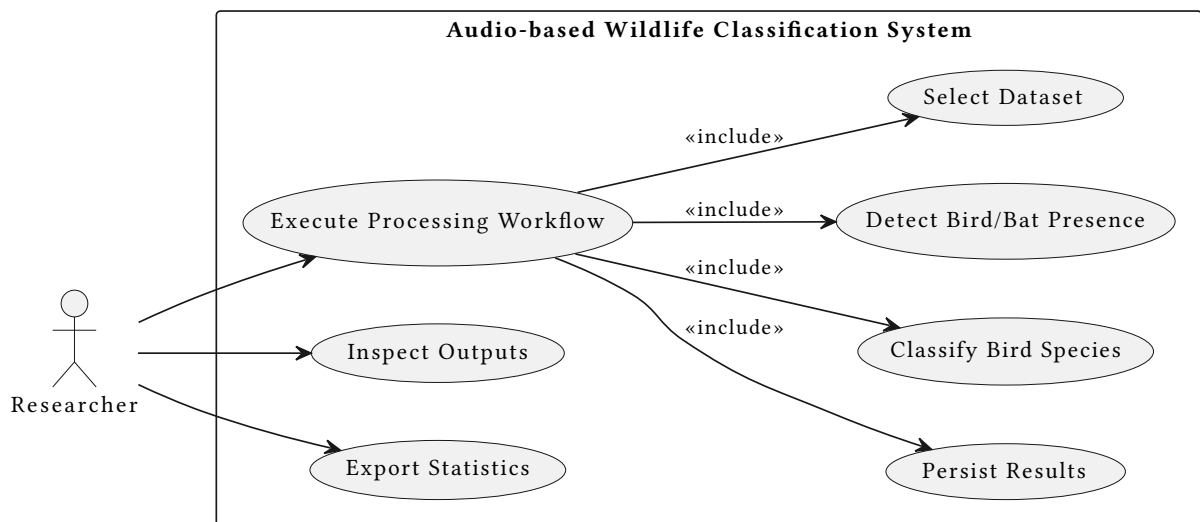


Figure 1: Use case diagram of the audio-based classification system

2.1.1 Actors

Researcher ILF member responsible for processing audio datasets, reviewing results, and exporting aggregated statistics.

2.1.2 Use Cases

The use cases below correspond to the interactions shown in Figure 1, each describing one capability the system must provide.

2.1.2.1 Execute Processing Workflow

The researcher shall be able to execute the complete processing workflow on a selected dataset of audio recordings. The workflow includes dataset selection, automated detection of bird and bat presence, bird species classification, and persistence of results.

2.1.2.2 Select Dataset

The system shall allow the researcher to specify an input directory containing audio recordings. All supported audio files within this directory, including files in subdirectories, shall be processed automatically. Non-audio files shall be ignored.

2.1.2.3 Detect Bird/Bat Presence

The system shall determine whether birds, bats, or both are present in each processed audio recording and provide structured file-level results.

2.1.2.4 Classify Bird Species

For recordings in which bird presence is detected, the system shall predict bird species labels.

2.1.2.5 Persist Results

The system shall write detection and classification results to a researcher-defined output directory.

2.1.2.6 Inspect Outputs

The researcher shall be able to inspect generated outputs on a per-file basis. This includes reviewing predicted presence, bird species labels, confidence values, and associated timestamps or time segments. Inspection may be performed through structured result files and/or the research dashboard provided by the system.

2.1.2.7 Export Statistics

The system shall compute and export aggregated statistics over processed datasets, such as bird and bat presence rates over time.

2.2 Non-Functional Requirements

This section defines the NFRs that establish the minimum acceptable performance and quality characteristics of the system.

ID	NFR-1
Category	Processing Time
Requirements	The system shall process audio recordings in batch mode with a total processing duration that does not exceed the total duration of the processed audio. Real-time capability is not required.
Acceptance Criteria	• Total processing time $\leq$ total duration of processed audio on defined reference hardware. • Measurement includes preprocessing, inference, and post-processing.
Measures	Benchmark end-to-end pipeline on representative dataset and documented reference hardware.

Table 1: NFR-1 Processing Time

ID	NFR-2
Category	Monitoring Performance
Requirements	Bird species classification and bat presence detection achieve practically useful performance on the reference sets.
Acceptance Criteria	• Bird species classification precision $\geq 80\%$ on the reference set. • At least 5 bird species achieve precision $\geq 60\%$ on the reference set. • Precision, recall, and F1 score (F1) are reported per bird species and overall. • Confusion matrix is provided for bird species classification. • Bat detection precision $\geq 80\%$ on the reference set.
Measures	Evaluation against the reference sets.

Table 2: NFR-2 Monitoring Performance

The targets named in Table 2 are framed in terms of precision rather than recall because the system is an exploration aid, not an autonomous or safety-critical classifier. Its role is to surface trustworthy candidate detections for the researcher to inspect and follow up [4]. A precision of 80% means roughly four in five surfaced detections are genuine, so the researcher discards about one for every five confirmed. Below that level the false-positive burden grows to an unacceptable amount. Recall is deliberately not the primary target – missing some calls is acceptable for an exploratory tool, and without a ground-truth set it cannot be measured in any case. The secondary target, at least five focus species reaching 60% precision, guards breadth: it keeps the tool useful across a range of the campus’ popular species rather than only the one or two dominant species.

ID	NFR-3
Category	Robustness
Requirements	The system shall execute batch processing robustly, handling corrupted or incompatible audio files without terminating the entire workflow. All relevant processing steps shall be logged.
Acceptance Criteria	• Failure of individual files does not abort batch execution. • Informative error messages are logged.
Measures	Test with corrupted and incompatible files.

Table 3: NFR-3 Robustness

ID	NFR-4
Category	Operational Usability
Requirements	The system shall provide a simple operational interface (command-line or minimal web interface) enabling the researcher to execute the full processing workflow without modifying source code.
Acceptance Criteria	• Basic usage documentation is provided. • Complete workflow can be executed using documented commands or interface controls. • No source code changes are required for standard operation. • New microphone locations integrate without code change.
Measures	Demonstration of independent execution by the ILF or a simulated researcher scenario.

Table 4: NFR-4 Operational Usability

ID	NFR-5
Category	Dataset Scalability
Requirements	The system shall support automated processing of multi-gigabyte audio datasets.
Acceptance Criteria	• Directory-based batch execution is supported.
Measures	Execution on representative multi-gigabyte dataset.

Table 5: NFR-5 Dataset Scalability

3 Domain Knowledge

This chapter provides the background needed to follow the rest of the thesis: why birds and bats are monitored acoustically, which species are expected on the OST campus, how their calls differ, which existing tools can recognise them, and the signal-processing and machine-learning foundations the methods build on.

3.1 Ecological Monitoring Context

Monitoring the activity of birds and bats does not only give insight into their population diversity and size; it draws a broader picture of biodiversity because they play a crucial role in a functioning ecosystem [5].

3.1.1 Passive Acoustic Monitoring

PAM is a non-invasive method of monitoring wildlife that emits acoustic signals – like birds and bats [6], [7]. It requires the placement of microphones at locations where activity shall be monitored, recording around the clock or on a schedule as required. The recordings can be either processed on-device or collected raw to process at a later time on devices with higher computational power. The former allows storing only the acoustic events of interest, reducing storage capacity required and therefore the maintenance overhead. The latter allows collection of data prior to the development of a model and allows for computationally heavier methods, which was done for this thesis.

3.1.2 Regional Birds

Around 40% of assessed Swiss breeding species are threatened or near threatened (as of 2021) [8]. This is a major motivation to conduct PAM as a scalable and non-invasive tool to survey their presence. This section discusses the species expected on the OST Rapperswil campus during the recording period of September to December 2025. The campus is located at the shore of Lake Zurich, and waterbirds are therefore significantly represented. The recording period covers late summer through autumn migration to the beginning of winter. The following 16 focus species were identified by [4] as those most expected at this location and form the scope of this project.

Year-round residents [9]:

- *Corvus corone* (Carrion Crow; Rabenkrähe) – an entirely black bird with a heavy beak.
- *Cyanistes caeruleus* (Blue Tit; Blaumeise) – a small bird with a bright blue cap and yellow belly; very common in parks and gardens.
- *Erithacus rubecula* (Robin; Rotkehlchen) – a small bird with a distinctive orange-red breast that sings year-round.
- *Fringilla coelebs* (Chaffinch; Buchfink) – males have a pink breast, blue-grey cap, and white wing bars; very common in woodland and gardens.
- *Milvus milvus* (Red Kite; Rotmilan) – a large raptor with a distinctive forked tail and reddish-brown plumage; often seen soaring overhead.
- *Parus major* (Great Tit; Kohlmeise) – a bold black head, white cheeks, and yellow breast with a central black stripe.
- *Passer domesticus* (House Sparrow; Haussperling) – a small, stocky brown bird found near human settlements; males have a grey cap and black bib.

- *Pica pica* (Magpie; Elster) – a large black-and-white bird with a long, distinctive tail that is noisy and easy to spot.
- *Streptopelia decaocto* (Collared Dove; Türkentaube) – a pale pinkish-grey dove with a narrow black collar; gives a distinctive three-note cooing call.
- *Turdus merula* (Blackbird; Amsel) – one of the most common urban birds in Europe; males are entirely black with a bright yellow bill, females dark brown with a mottled pale breast.

Autumn migrants passing through before departing south [10]:

- *Ciconia ciconia* (White Stork; Weissstorch) – a very large white bird with black wingtips and a long red bill and legs; unmistakable in flight.
- *Sylvia atricapilla* (Blackcap; Mönchsgrasmücke) – a medium-sized warbler; males have a distinctive black cap, females a chestnut-brown one; a rich, melodious singer.

Waterbirds on the Obersee, with wintering populations considerably larger from October to December:

- *Anas platyrhynchos* (Mallard; Stockente) – males have a glossy green head and yellow bill; the most familiar duck in Europe [9].
- *Chroicocephalus ridibundus* (Black-headed Gull; Lachmöwe) – a small gull with a red bill and legs; present on Swiss lakes from October to March [10].
- *Fulica atra* (Coot; Blässhuhn) – an all-black waterbird with a white bill and frontal shield; forms large flocks on open water and dives to feed [9].
- *Larus michahellis* (Yellow-legged Gull; Mittelmeermöwe) – a large white gull with grey wings, a yellow bill, and yellow legs; common on Swiss lakes in winter [10].

3.1.3 Regional Bats

Switzerland is home to 30 bat species, 26 of which have been evaluated for their conservation status in a collaborative effort by the Federal Office for the Environment (BAFU), West Coordination Centre for Bat Conservation (CCO), East Coordination Centre for Bat Conservation (KOF), Swiss Centre for Faunal Cartography (SZKF) and WSL [11]. The majority (22 species) are either on the Swiss Red List or classified as near threatened. Reasons for that are renovation of buildings that eliminate their roosting sites and further alteration of their habitats [11]. This precarious conservation status highlights the importance of monitoring bat populations. The four not threatened species are most represented and therefore also the most likely to be found in the campus recordings:

***Pipistrellus pipistrellus* (Common Pipistrelle; Zwergfledermaus)** A dominant urban species in Switzerland due to its light-tolerance [12]. It exploits artificial light that attracts insects for hunting. Accounted for ~91% of bat calls in a study conducted in urban green areas of Zurich, Paris and Antwerp [13].

***Pipistrellus kuhlii* (Kuhl's Pipistrelle; Weissrandfledermaus)** A species of Mediterranean origin that has been expanding north over decades [11]. It is urban-adapted and present in the region of Zurich [12].

***Pipistrellus nathusii* (Nathusius's Pipistrelle; Rauhautfledermaus)** A highly mobile, migratory species whose Swiss populations are largely sustained by annual immigration [11]. It is present in the region of Zurich [12].

***Myotis mystacinus* (Whiskered Bat; Bartfledermaus)** A widely distributed species in Switzerland [11]. As a *Myotis* it produces broadband frequency-modulated calls [14] that are hard to tell apart from those of closely related species.

Myotis daubentonii (Daubenton's Bat; Wasserfledermaus) and *Pipistrellus pygmaeus* (Soprano Pipistrelle; Mückenfledermaus) are two of the near threatened species whose habitat is near lakes, making them plausible candidates on campus too [11].

3.1.4 Acoustic Characteristics of Birds and Bats

There are two common ways to identify birds by inspecting acoustic data. The obvious first one is listening to them, because bird sounds are well interpretable by human hearing. Another method is to generate spectrograms and look at that visual representation to identify shapes and patterns that resemble their sounds. Figure 2 shows such a spectrogram of an audible bird call, which stays below 5 kHz. It is computed on audio resampled to 24 kHz with a 512-sample short-time Fourier transform (STFT) window at 50% overlap on a linear 1–8 kHz axis.

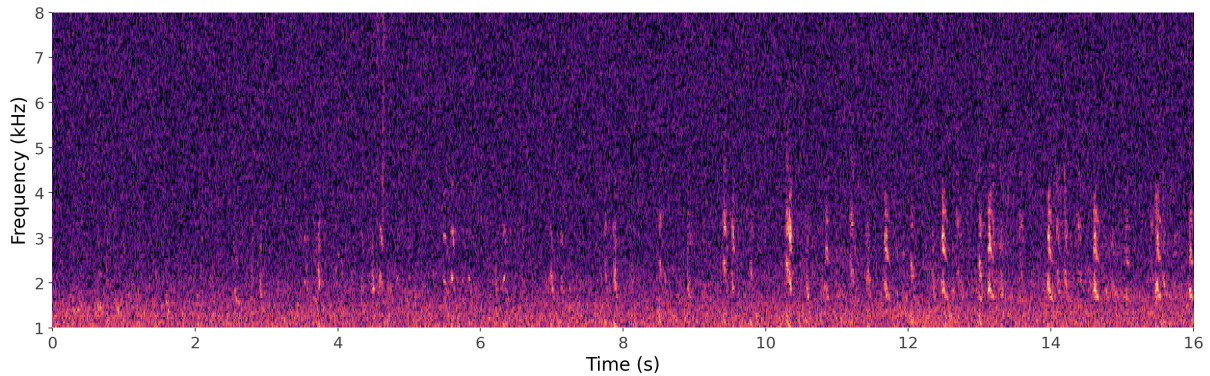


Figure 2: Spectrogram of an audible bird call (*Fulica atra*) from a recording collected for this thesis.

Recordings also contain non-biological background noise, much of it in the same low-frequency range as bird calls. Figure 3 shows one such event – a train stopping at the train station right next to the campus – whose broadband energy spreads across the bands where bird calls reside, so it can mask genuine calls or be mistaken for them. It is computed on audio resampled to 24 kHz with a 1024-sample STFT window at 50% overlap on a linear 1–8 kHz axis.

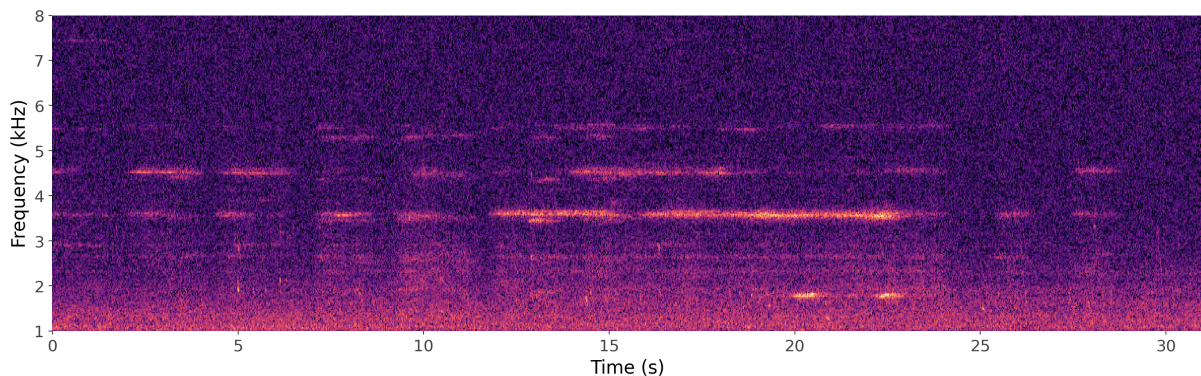


Figure 3: Spectrogram of a non-biological noise event – likely a train stopping – from a recording collected for this thesis.

For bats it is a much greater challenge. Their sounds are generally at frequencies inaudible to human hearing (ultrasonic), unless they are slowed down roughly by a factor of 10. Therefore, spectrograms are often used to identify their calls, as in Figure 4 where the echolocation pass reaches well above 40 kHz, computed on the native 384 kHz recording with a 1024-sample STFT window at 87.5% overlap. The sample rate must be at least twice the highest call frequency, so 256 kHz – which resolves frequencies up to 128 kHz – already suffices for the fundamental calls of many bat species [15]; the campus recordings at 250 and 384 kHz resolve up to 125 and 192 kHz respectively, spanning the frequencies of the European bats considered here. Such noise is not confined to the lower bands either: insects, rain, electrical noise, traffic and rivers likewise occupy the ultrasonic frequencies where bats call [2]. Bat activity varies widely from night to night at a single location [15]. Consequently, activity levels are not expected to transition smoothly between consecutive nights.

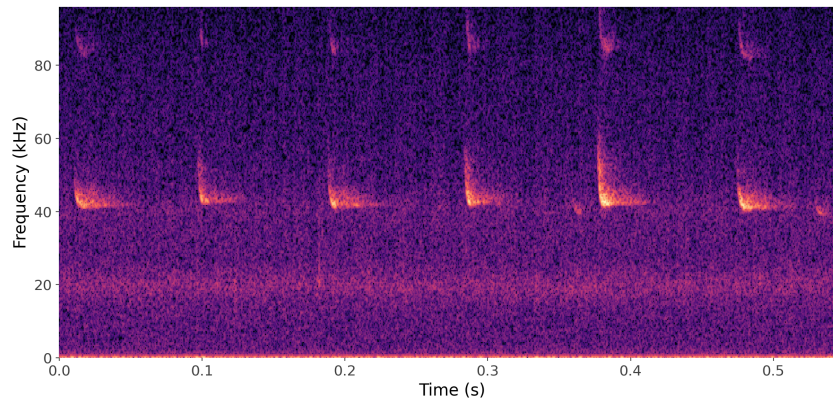


Figure 4: Spectrogram of example ultrasonic bat echolocation calls from a recording collected for this thesis.

3.1.5 Bat Echolocation and Sonotypes

Bats navigate and hunt by echolocation, emitting ultrasonic calls and reading the returning echoes [15]. Across species these calls take only a small number of recurring shapes, called sonotypes. Building on the Tadarida toolbox [16], [14] defined these categories worldwide and trained a classifier to assign them – the main reference for the sonotype approach. The principal categories are:

- *FM (frequency-modulated)* – a rapid downward sweep, short in duration and wide in bandwidth; common across most European bats, including *Myotis* and *Pipistrellus* species.
- *CF (constant-frequency)* – a sustained tone at a fixed frequency, used with Doppler-shift compensation to detect fluttering prey [14]; among the species relevant here this characterises the horseshoe bats (*Rhinolophus*), which are rare in Switzerland [11].
- *CF-FM* – a constant-frequency element followed by an FM sweep; like the pure CF call, it is produced by horseshoe bats and other constant-frequency foragers [14].
- *QCF (quasi-constant-frequency)* – a shallow, nearly flat frequency contour, used for longer-range detection, as in the Common Pipistrelle (*Pipistrellus pipistrellus*) [14].

A call's shape stays more consistent across regions than species identity does, so a sonotype can be assigned before narrowing to a species – whereas BatDetect2 [2] and BatScope4 [17] classify species directly.

The species most likely on campus (Chapter 3.1.3) are acoustically similar: the three pipistrelles – Common, Kuhl's and Nathusius's – share the combined FM and QCF sonotype that [14] exemplifies with the Common Pipistrelle, while the Whiskered Bat, a *Myotis*, uses frequency-modulated calls. Their calls overlap so closely in shape and frequency that telling the species apart reliably is difficult [11].

3.2 Existing Tools

Automated tools for analysing bat and bird recordings have been developed for several decades and continue to evolve. Earlier approaches extracted hand-crafted audio signal features – such as frequency contours or energy envelopes – and fed them into classifiers like support vector machine (SVM) or weighted k-nearest neighbours (KNN). The current generation of well-established tools, such as BirdNET and BatDetect2, uses convolutional neural networks (CNNs) operating on spectrograms. Newer research is exploring transformer-based architectures that may further improve recognition accuracy, both for birds [18], [19] and for bats [20]. All of these established tools operate on the spectrogram rather than the raw audio waveform; end-to-end models that learn directly from the waveform have been explored but are rare among the established tools surveyed here, so this project follows the spectrogram-based route. This section gives an overview of tools that were considered for

this project. Table 6 compares them at a glance – the algorithm each uses, whether it offers a Python interface and its licence – before each is described in turn.

Tool	Algorithm	Pipeline integration	Licence
Merlin	CNN	No (mobile app)	Proprietary
BirdNET	CNN (EfficientNet)	Yes (Python)	MIT
Perch	CNN (EfficientNet)	No (model only)	Apache 2.0
Tadarida	Random forest (hand-crafted features)	No (C++/R tools)	GPL / LGPL
BatScope4	KKNN + SVM	No (desktop app)	Proprietary
BattyBirdNET	CNN (transfer learning)	Yes (Python)	CC BY-NC-SA 4.0
BatDetect2	CNN	Yes (Python)	CC BY-NC 4.0

Table 6: Overview of the bird and bat classifiers considered for this project.

3.2.1 Bird Classifiers

Existing bird classifiers discussed here are Merlin Bird ID, BirdNET and Perch, all of which use CNNs. They are selected because they are well established in the community and free for non-commercial use.

3.2.1.1 Merlin Bird ID

Merlin [21] is a free, closed-source mobile application developed by Cornell Lab of Ornithology (Cornell Lab) for members of the general public who want to identify birds they hear or see. It supports identification by audio (songs and calls) or image. The audio is converted to a spectrogram – a visual representation. This makes both identification methods an image classification problem for which they use CNNs. They are trained on the Macaulay Library, the world’s largest and oldest scientific archive of wildlife recordings [22]. The library contains more than 99 million photos, audio and videos, amongst them more than 10’000 species of birds. It is a curated set of all the contributed recordings from various sources, one of them is Merlin Bird ID [23].

A typical bird identification process starts with an audio recording done right in the Merlin app – it will list detected species in real time. Once the recording is stopped, the detections can be verified against best matches of the reference data. Once confirmed, the app supports reporting the recording to eBird, a citizen-science online database of bird observations created by Cornell Lab. To contribute, a few more details like location and date have to be added. This database is powering Merlin and enabling scientists, researchers and amateur naturalists to monitor birds globally [21].

3.2.1.2 BirdNET

BirdNET [1] is free software developed by Cornell Lab for automated bird classification. Like Merlin, it uses a CNN to recognise bird songs and calls in audio recordings, but unlike Merlin’s multi-input approach, it only supports audio. Both classifiers are trained on the Macaulay Library, but BirdNET additionally uses Xeno-Canto [24], [25] – a community-driven collection of recordings around the world [1]. A comparative study on passive recordings found that the two tools trade off differently: Merlin identified more species correctly but with more false positives, whereas BirdNET produced far fewer false positives at the cost of more missed detections [26].

BirdNET Version 2.4 – the latest release as of 5 June 2026 – covers more than 6’000 species worldwide [27], making it broadly applicable across geographic regions. Additionally, a GeoModel can be used to pre-select potential species by their occurrence probability. The input is coordinates and based on the

eBird database it will calculate the occurrence probabilities. The species predictions are always made on a 3-second window of 48 kHz audio. The software comes both as a graphical user interface (GUI) for easy usage and a Python library to build custom workflows. Both are published on GitHub under the MIT licence [28].

BirdNET-Analyzer has documentation on how to train a custom classifier in case BirdNET does not include some species needed. However, they recommend to first check if other BirdNET labels can act as proxies for the custom species if their mapping is consistent and unique [29].

To improve BirdNET's performance specifically on the locally used microphones and present species, it can be fine-tuned. Schiavo et al. [29] fine-tuned BirdNET to monitor the Italian Alpine forests, also using AudioMoth recorders. Their results show significant improvements over both original BirdNET and a reference vanilla CNN across all averaged F1 scores. Surprisingly, the original BirdNET outperforms the fine-tuned variants at the micro-average precision. However, that advantage diminishes when comparing weighted- and sample-averages, where the fine-tuned variants' metrics remain stable, consistently outperforming the original. This is a clear demonstration of how fine-tuning BirdNET on region-specific data can yield valuable performance gains.

3.2.1.3 Perch

Perch [30] is a bioacoustic embedding model developed by Google DeepMind under the Apache License 2.0. The comparison of Perch 1.0 to BirdNET 2.3 made in [30] shows that its embedding model takes roughly twice as long to process audio as BirdNET's; both used the EfficientNetB1 architecture. In the table of results, the improvement over BirdNET is marginal. Perch 2.0 [31] shows further improvements compared to Perch 1.0 and uses the larger EfficientNetB3 architecture. It was trained on the XenoCanto [32], iNaturalist, the Tierstimmenarchiv, and FSD50K datasets. BirdNET 2.4 in comparison uses an EfficientNetB0-like architecture that in turn is smaller than before [27]. The classifications are made for 5-second windows at 32 kHz.

Unlike BirdNET, Perch was primarily designed as an embedding model with the flexibility to attach various classifying heads. The linear embedding drastically reduces the complexity of the problem and reduces labelled data needed to train. This transfer learning method even outperformed models that are specialised on marine training data, despite having had almost no marine training data itself [31]. Perch is distributed as a model; there is no pip package as simple as for BirdNET.

3.2.2 Bat Classifiers

The first two bat classifiers discussed – Tadarida and BatScope4 – are pre-deep-learning and rely on hand-engineered features. In comparison follow BattyBirdNET and BatDetect2, both models with a CNN architecture. This selection should provide a broad overview of the automated bat detection landscape.

3.2.2.1 Tadarida

Tadarida [16] is an open-source bat classifier pipeline split into two GitHub repositories under the GPL and LGPL licences. The first repository (Tadarida-D) for detection and feature extraction is written in C++ and produces .ta files. The second repository (Tadarida-C) for classification is entirely written in R. Roemer et al. [14] built on top of that architecture by inserting sonotype classification between the two. This improves accuracy by grouping species by call shape, so the classifier's set of possible species to predict gets reduced. Additionally, it supports region-independent classification because call shapes are determined by bat physiology and foraging strategy.

3.2.2.2 BatScope4

BatScope4 [17] is free, closed-source software developed by WSL – a Swiss institute – that allows to organise bat recordings in a local database. It is used as a GUI on macOS or Windows and offers tools to cut the recordings into calls, inspect and then classify them automatically. Trained on 19'636 echolocation calls of 27 European bat species, it is geographically suited well for this thesis. Automation with a Python-based scripting interface was considered too error-prone for many users and is therefore not available. Data can be imported in the proprietary format of BatLogger or Batcorder as well as directly as WAV files [33].

3.2.2.3 BattyBirdNET

BattyBirdNET [34] – as the name suggests – is a transfer learning project based on BirdNET to classify bats. It uses BirdNET's audio feature extraction capabilities to generate embeddings. The classifying head is replaced by one provided in the repository that predicts bats instead of birds. It passes 256 kHz audio to BirdNET as if it were the 48 kHz it expects, which naturally moves the frequency ranges of bat activities to the ones BirdNET is trained on for birds. This trick results in predictions per $\left(\frac{48 \text{ kHz}}{256 \text{ kHz}}\right) * 3 \text{ s} = 0.5625 \text{ s}$. The variant geographically most relevant to this thesis, due to its proximity, is trained on regional data from Bavaria. It is released under the CC BY-NC-SA 4.0 licence, which allows non-commercial use.

3.2.2.4 BatDetect2

BatDetect2 [2] is an open-source bat detection CNN framework developed by Mac Aodha et al. Unlike the other CNNs mentioned that predict on a fixed window of audio, it generates bounding boxes around individual calls while simultaneously predicting the species. The software is implemented in Python, offering both a package and a command-line interface (CLI). Mac Aodha et al. train and evaluate it on five datasets from four regions (the United Kingdom, Mexico, Australia and Brazil); the pretrained model shipped with the package, and used here, is the UK model, covering the 17 species that breed in the United Kingdom, which makes it relevant for European applications. It is released under the CC BY-NC 4.0 licence, free for non-commercial use.

3.2.3 Selection

Of the surveyed tools, four were carried forward. The selection criteria followed from the assignment: a tool had to be an established solution, free to use for non-commercial purposes, and scriptable – so that it could be driven automatically over the whole dataset. BirdNET meets all three for birds and BatDetect2 for bats, both maintained CNN models with a Python package. BattyBirdNET is taken on as a second bat model trained on data from neighbouring Bavaria, making it the regionally closest of the candidates. BatScope4 is a deliberate exception to the integration criterion: it exposes no scripting interface and therefore cannot be a pipeline adapter. However, it is developed by the WSL and trained on European bats of this region, which makes it valuable as an independent reference to compare the other adapters' bat predictions against – it is therefore run by hand for that purpose alone.

The other three tools were not adopted. Merlin is a closed-source mobile application aimed at the general public, with no interface to drive from a pipeline. Tadarida is accessible through C++ / R, which did not fit with the Python integrations of the others. Perch is the closest call – a capable embedding model with strong transfer-learning results – but it is distributed as a bare model with no package as convenient as BirdNET's, and BirdNET already covered the bird side at a precision the ILF judged sufficient, so a second bird model promised little return for the integration effort it would have required.

Having established the ecological context and surveyed the existing tools, the remainder of this chapter sets out the signal-processing and machine-learning foundations that the methods in this thesis build on.

3.3 Digital Audio Processing

A recording is sound sampled at a fixed *sample rate*. By the Nyquist criterion [35], a sample rate of f_s can only represent frequencies up to $\frac{f_s}{2}$; anything higher is lost. Therefore, the sample rate of the recordings must be at least double the highest frequency that needs to be captured.

The selected models do not work on the raw waveform but on a spectrogram: a time-frequency image computed with the STFT, with time on one axis, frequency on the other, and intensity showing how much energy is present [36]. In this image a call becomes a visible shape and similar calls look similar, which turns species recognition into an image-classification problem. Both human analysts and these models tell species apart by these shapes.

3.4 Machine Learning for Acoustic Classification

The models used here are supervised classifiers: from labelled examples they learn to assign an input to a species (multi-class classification) [36]. This thesis applies such models off the shelf rather than training its own. The models chosen to integrate in the pipeline are CNNs, the standard architecture for image-like inputs [36]. A CNN reads the spectrogram, detects local call patterns wherever they occur, and combines them layer by layer into a compact feature vector, from which it predicts the species. That feature vector is the model's embedding [30].

Because models are trained so that similar-sounding inputs map to nearby vectors, an embedding is a compact, reusable summary of a sound. A simple classifier trained on embeddings can then recognise classes the original model never saw, using only a few labelled examples [31] – the mechanism behind transfer learning.

Transfer learning reuses a model pre-trained on a large dataset instead of training one from scratch [30]. It takes two forms: *feature extraction*, where the pre-trained model is frozen and only a small classifier is added on top of its embeddings [30], and *fine-tuning*, where the model itself is trained further on in-domain data [29]. This thesis uses a model off the shelf that replaces just the classifying head; the supervised pre-training they rely on has been shown to outperform self-supervised alternatives in bioacoustics [30], [31].

3.5 Evaluation Metrics

The models are scored by comparing their predictions against a reference set. A prediction that matches the reference is a true positive (TP), a predicted positive with no match is a false positive (FP), and a positive in the reference that the model missed is a false negative (FN). A true negative (TN) is a negative the model correctly leaves unpredicted. The three metrics below are built only from TPs, FPs and FNs and do not use TNs, which this thesis does not report.

Precision is the ratio of TPs to all the positives the model predicted. In other words, the greater the precision, the fewer false alarms it raises:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall is the ratio of TPs to all the positives actually present. In other words, the greater the recall, the fewer positives missed:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

The F1 combines precision and recall into a single value as their harmonic mean. In other words, it is only high when both are high:

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

4 Data Collection

This chapter introduces the data the thesis is built on. It covers the recording hardware and its deployment, an exploration of the collected dataset's structure and properties, and how the limited reference sets for birds and bats were established.

4.1 Recording Setup

This section describes the recording setup used by the ILF to collect the data this project is built on. It explores the data's structure and properties to inform subsequent analysis.

4.1.1 AudioMoth Devices

The recording setup consists of four AudioMoth 1.2.0 devices deployed as displayed in Figure 5. Of the available microphone devices to monitor birds and bats simultaneously, AudioMoth was chosen for its budget-friendly pricing [4]. It is equipped with a micro-electro-mechanical systems (MEMS) microphone capable of capturing both audible and ultrasonic audio at sample rates up to 384 kHz [37]. MEMS microphones are susceptible to electromagnetic interference and background noise [38], which can make bird and bat call detection more difficult. The British Trust for Ornithology (BTO) guide for commonly used recorders reports loss of weaker end calls and the production of self-noise artefacts in recordings. This can make it difficult to detect bat calls, especially those of *Myotis* and *Rhinolophus* species. BTO also recommends to record at a sample rate of 384 kHz to detect bat calls [39].



Figure 5: AudioMoth device deployed on tree. Provided by ILF [4].

4.1.2 Data Exploration

The data provided covers the date range from 17 September to 12 December 2025. The devices were spread across campus; locations are annotated in Figure 6 (their numbers were assigned manually in an anti-clockwise direction).

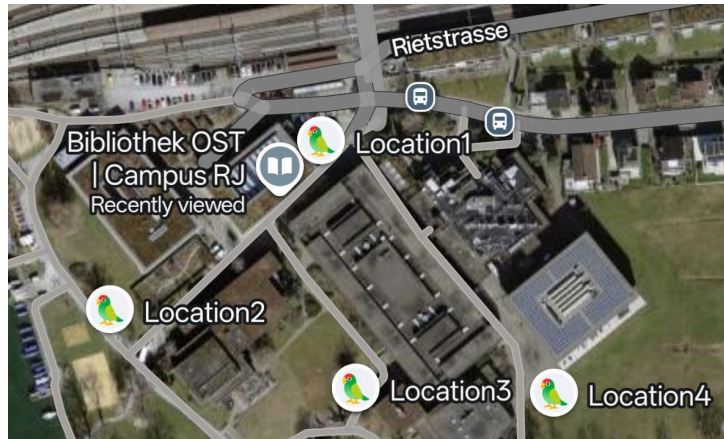


Figure 6: OST Rapperswil campus with microphone locations annotated. Screenshot taken on Google Maps [40].

[4] chose the locations to cover as much of the campus area as possible, including both high-traffic and low-traffic areas. They configured the devices, set them up at the locations and retrieved them periodically when the storage was full to collect the recordings.

4.1.2.1 Folder Convention

The folder structure begins with a file named `README.txt` and a folder named `Audiomoths`. The file briefly explains the structure of the data and the folder is considered the input root. The input root contains 7 folders named in `YYYYMMDD` format. The children thereof are folders named like `Audiomoth1` where the suffix number can range from 1 to 4. Such a folder in turn contains a file named like `Audiomoth1_ort.txt` (single line of LV95 coordinates), a file named `CONFIG.TXT` (AudioMoth's device configuration) and more folders in `YYYYMMDD` format. The latter are referred to as *leaf folders*. Inside these folders are the wav files named like `248D9B016487A731_20250917_180000.WAV`, where the first part is the device id, second the date (`YYYYMMDD`) and last the time (`HHMMSS`).

One sample path to a wav file:

`Audiomoths/20250923/Audiomoth2/20250917/248D9B016487A731_20250917_180000.WAV`

4.1.2.2 Date and Time Coverage

To reduce the intervals between which they had to do this, the devices record for 5 minutes and then pause for 10 minutes on repeat. Another measure taken is the schedule that limited recordings to time windows each day. Figure 7 shows how many devices were recording each time of day, colour-coded by the number of locations recording simultaneously (the 10-minute pause is treated as covered time).

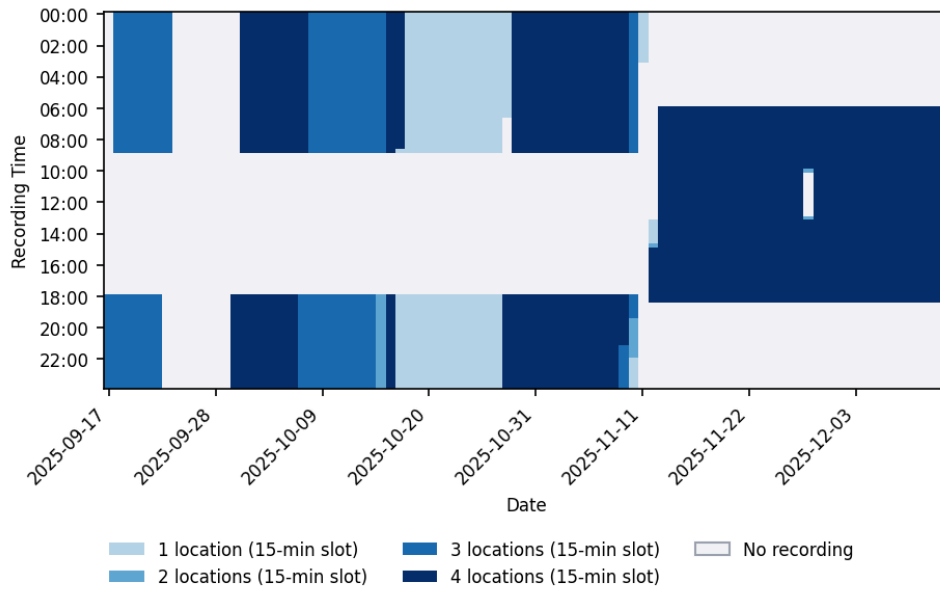


Figure 7: Date and time coverage of the recordings.

Up until 11 November 2025, recordings were generally from 17:30 until 07:00 the next morning, with the goal of capturing the twilight hours plus the night for bat calls. According to [4], the most distinctive calls occur around dawn and dusk – roughly from one hour before to two hours after sunrise, and from two hours before to one hour after sunset – which the schedule is designed to span. [4] chose to stop recording during the night and at high frequencies because bats are hibernating. Therefore, they changed the schedule to be from 06:00 to 18:30. When time schedules slightly differ on a single day, that is likely due to the manual process of collecting the devices and extracting the recordings.

4.1.2.3 Location Coverage

Figure 7 already makes it apparent that not all four devices were always deployed simultaneously. Figure 8 shows the deployed days per location and how many recordings were captured each day. The single days of fewer recordings indicate that the device has been temporarily retrieved to fetch recordings. Location2 has an anomaly around 15 October 2025 which is likely a misconfiguration of the device’s recording interval or schedule.

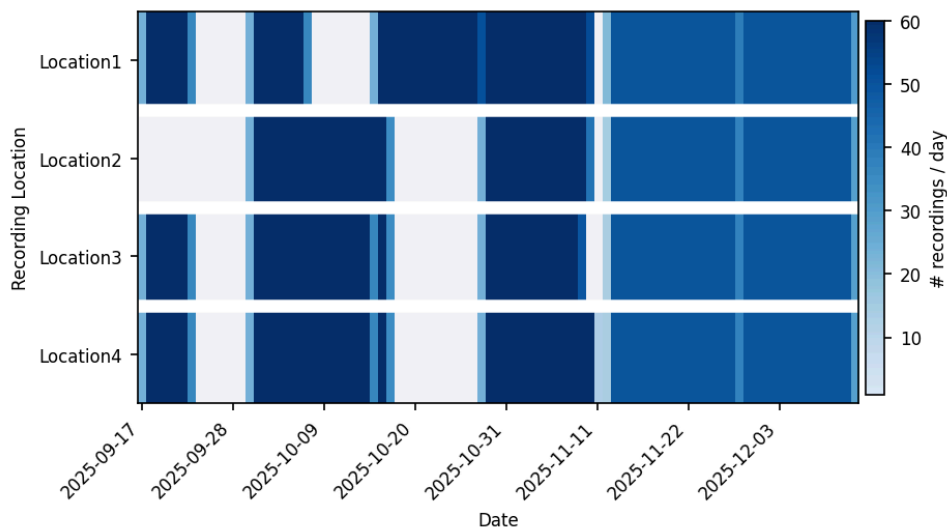


Figure 8: Number of recordings per recording location and date.

The lack of consistent deployments where all four locations are covered simultaneously makes relative activity comparison between locations notoriously difficult. Figure 8 must be consulted when making statements regarding that, to check what locations were recorded during the period of interest.

4.1.2.4 Sample Rate Coverage

Figure 9 shows that the recordings were initially recorded at a sample rate of 384 kHz, which got reduced to 250 kHz on 16 October 2025. As described in Chapter 3.1.4, 250 kHz is likely still enough to capture bat calls at a reduced file size per recording. 12 November 2025 onwards, the sample rate was reduced further to 48 kHz, correlating with the schedule switch mentioned above due to bat hibernation.

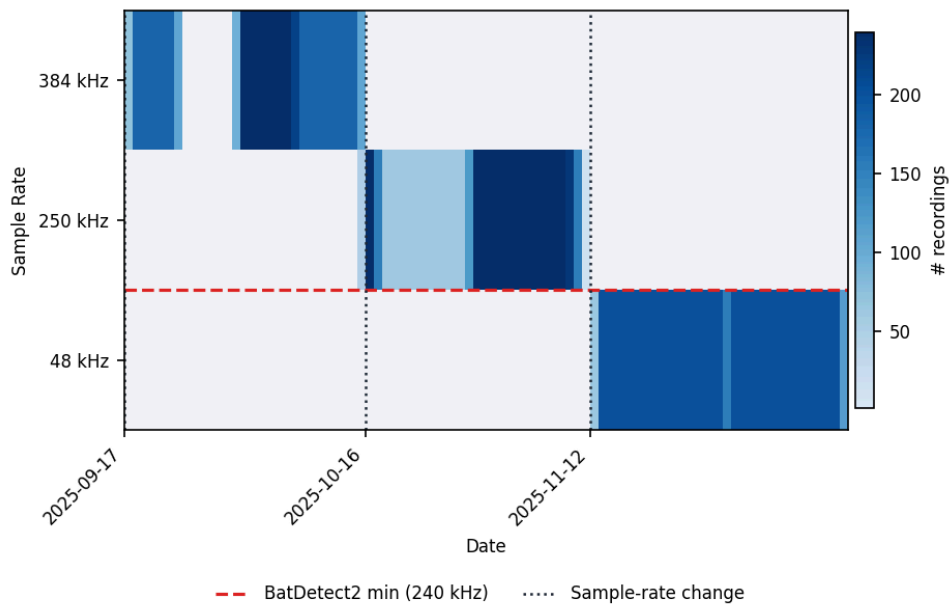


Figure 9: Number of recordings per sample rate and date.

4.1.2.5 Filesize

The sum of 1.77 TB in audio recordings is spread amongst 14'333 individual 5-minute wav files. The significant differences in file size per sample rate can be observed in Figure 10. This confirms that lower sample rates bring a significant storage saving.

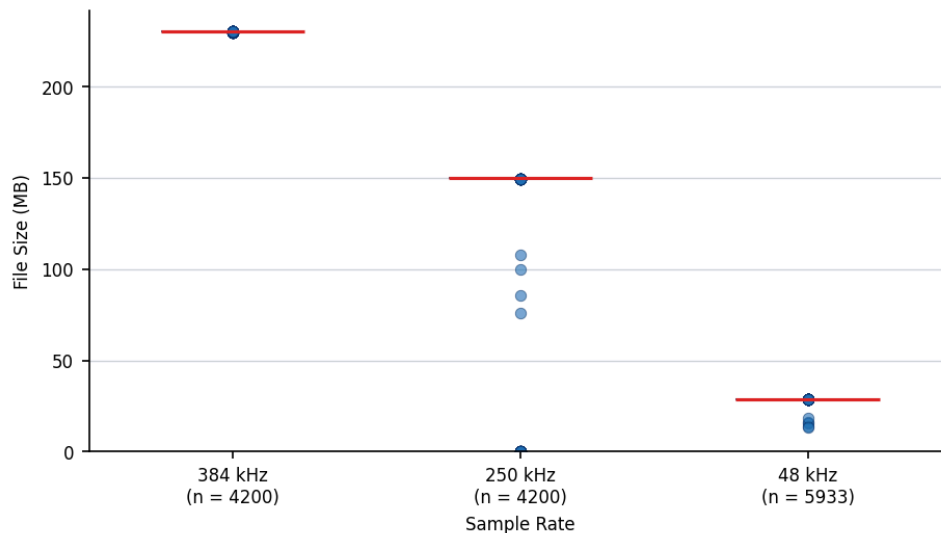


Figure 10: File sizes per sample rate and outliers.

4.2 Reference Set Creation

The limitations stated in Chapter 1.4 make an annotated ground-truth set – one with every call labelled – infeasible for either birds or bats. Reference sets were therefore created in their place: two limited sets that support precision evaluation without claiming to capture every call. For birds, the BirdNET assessment set of 30 selected recordings was created to evaluate its precision. For bats, where no person was available to confirm calls, the bat consensus set is constructed as an ensemble of the BatDetect2, BattyBirdNET and BatScope4 adapters to serve as the validation reference. The bird predictions are generally cut off at 0.2 confidence, below which most predictions no longer showed clear signs of birds. Bat predictions were likewise cut off at 0.3 for the same reason.

4.2.1 BirdNET Assessment Set

In order to focus bird recognition validation on a list of species that are known to be present around campus, the ILF provided 16 species listed in Chapter 3.1.2.

The ideal ground-truth set would consist of all species calls annotated per 5-minute recording. The absence of calls must be guaranteed at times of no annotation to measure how many calls were left undetected. Therefore, the primary target was to establish such a dataset on the assessment recordings – 30 recordings curated from the thousands collected. Each being 5 minutes long, that totals 2.5 hours of audio to carefully listen to and annotate. This has to be done by an expert in identifying birds by their calls. Unfortunately, the ILF was not able to provide that, which led to an alternative – the assessment of BirdNET’s individual performance.

The predictions of BirdNET to be assessed still originate from these 30 recordings. The assessment recordings were selected based on five criteria in priority order descending. All 14’333 recordings were first processed as described in Chapter 6.1 and then considered candidates to choose from.

1. **New species coverage (most important):** Ensure that all 16 focus species are covered if detected at all.
2. **Rarity bonus:** Prioritise focus species that are least represented up to 10 calls.
3. **Date balance:** Prioritise recordings on dates that are covered the least yet.
4. **Species balance:** Prioritise species that are represented less.
5. **Model confidence (tiebreaker):** Choose higher max confidence of focus-species detection.

As a consequence of the date balance criterion, the 30 recordings have spread across 30 distinct days, ranging from 20 September to 10 December 2025. Additionally, the first two species coverage criteria ensured that each focus species is part of mostly four recordings as Table 7 illustrates. For each focus species, the table gives the number of recordings it appears in, the prediction segments included and the median confidence over all predicted segments – each value as assessment set / full dataset (processed BirdNET predictions, confidence ≥ 0.2) – ordered by in-set segment count. The species occurring less in the assessment set also occur less in the full dataset.

Species	Recordings (set / all)	Segments (set / all)	Median confidence (set / all)
<i>Fulica atra</i>	4 / 4'738	25 / 16'632	0.927 / 0.600
<i>Pica pica</i>	4 / 1'133	22 / 3'077	0.978 / 0.693
<i>Erithacus rubecula</i>	4 / 2'887	16 / 8'339	0.842 / 0.407
<i>Passer domesticus</i>	3 / 113	12 / 225	0.336 / 0.312
<i>Corvus corone</i>	4 / 1'597	11 / 3'585	0.409 / 0.461
<i>Cyanistes caeruleus</i>	4 / 1'132	10 / 2'640	0.473 / 0.407
<i>Parus major</i>	4 / 952	9 / 2'287	0.380 / 0.386
<i>Chroicocephalus ridibundus</i>	4 / 822	8 / 1'555	0.491 / 0.424
<i>Fringilla coelebs</i>	4 / 509	8 / 1'006	0.814 / 0.431
<i>Turdus merula</i>	4 / 880	7 / 1'906	0.567 / 0.381
<i>Anas platyrhynchos</i>	4 / 1'465	5 / 2'089	0.537 / 0.294
<i>Larus michahellis</i>	4 / 135	5 / 216	0.826 / 0.482
<i>Streptopelia decaocto</i>	4 / 201	5 / 321	0.676 / 0.388
<i>Sylvia atricapilla</i>	2 / 7	4 / 10	0.483 / 0.355
<i>Ciconia ciconia</i>	3 / 7	3 / 7	0.636 / 0.475
<i>Milvus milvus</i>	2 / 3	2 / 4	0.300 / 0.309

Table 7: BirdNET predictions of the 16 focus species in the assessment recordings.

All predictions – including non-focus species – were processed as described in Chapter 6.1 and then prepared as an annotation kit (folder and usage instructions of Appendix F.1). The annotator [4] went through all 230 predictions and annotated them using the setup as described in the following chapter, additionally verifying them against spectrograms generated in Audacity.

Although the recordings were curated to cover primarily the focus species, BirdNET surfaced a further 30 non-focus species across them. These account for 78 of the 230 predictions (34%) and have generally lower confidences than the focus detections: non-focus species included 2.6 segments on average at a median confidence of 0.39, against 9.5 segments at 0.70 for the focus species. This is a direct consequence of the deliberately low confidence threshold, which allows to draw a confidence retention chart like Figure 28.

The resulting annotated data is the BirdNET assessment set, a limited reference set that is used to evaluate the model's precision. The recall, however, is not representative because calls missed by BirdNET would wrongly count as the absence of such.

4.2.2 Annotator Setup

The annotator setup was designed to address concerns regarding annotation quality and quantity. Regarding quality, the concerns were that species names would be typed inconsistently and that with

the complex folder convention (see Chapter 4.1.2.1) the risk of opening the wrong recording was non-negligible. The former is mitigated by providing a searchable dropdown of species to choose from, with the option to add additional ones which will thereafter also appear in the dropdown. The latter is mitigated by script code matching path names and the interface displaying the respective recording player directly. To help with the handling of recordings as files, e.g. to open in other software, they are named like clip-001 in the annotation kit.

Regarding quantity, the interface is an attempt of supporting the annotator to have fewer time spent per prediction. This is done by loading an audio player ready to listen to the respective audio segment, rather than the annotator opening the audio in a second window, skipping forward to the segment start and careful to stop when the segment finishes.

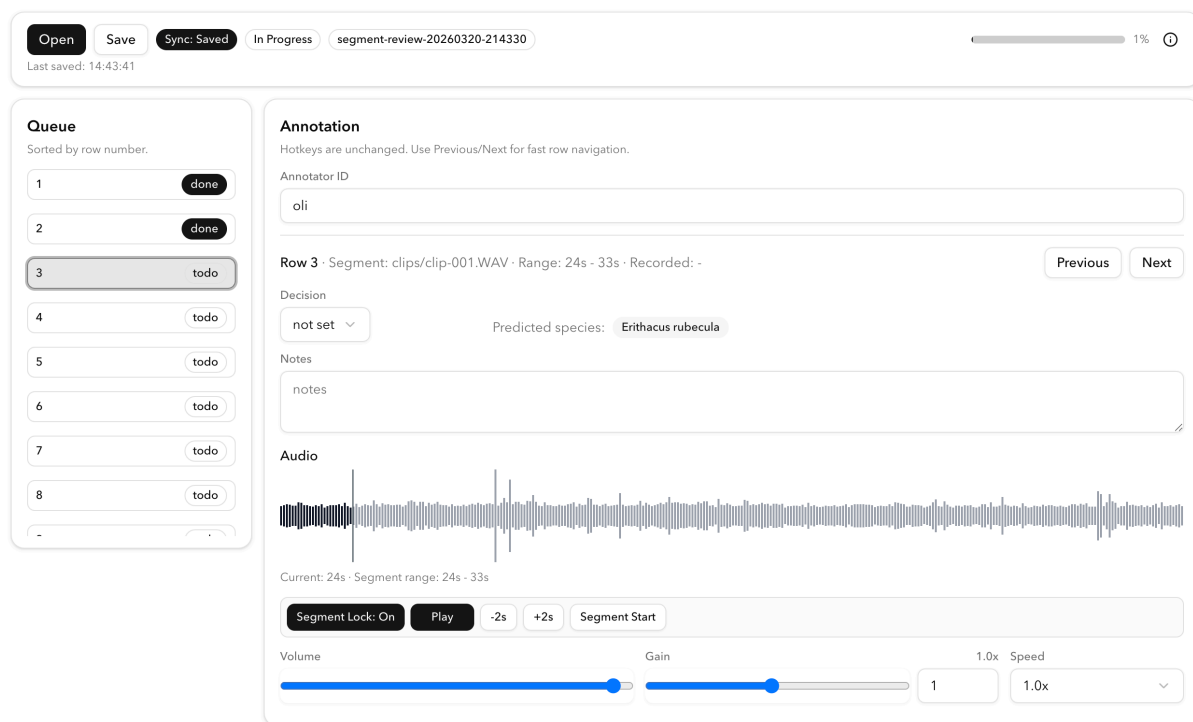


Figure 11: The purpose-built annotation interface used to annotate the candidate clips.

The interface has been developed in form of a website to make it easily accessible to the annotator without any installation. It uses the File System Access API – an experimental feature only supported by certain browsers [41] – to read and write to a folder on the annotator’s device. The annotator uses a compatible browser, making this feature the right choice to have a lean frontend-only interface.

Figure 11 displays the annotation interface (see Appendix F.1 for usage instructions). It was intentionally developed rapidly to increase annotation quantity and quality and is not expected to be further used. Because of that, its code quality was deprioritised in favour of other tasks and is not included in the final repository.

4.2.3 Bat Consensus Set

The bat consensus set likewise comprises 30 curated recordings drawn from the processed predictions. This sample size matches the BirdNET assessment set and is in any case limited by BatScope4, which – unlike the two pipeline adapters – is a GUI tool with no scripting interface, making a larger set impractical. As the evaluation concerns only presence and not species, recordings were prioritised by how many confident detections they contain – the segments above a detection confidence of 0.75. Both adapters supported by the pipeline, BatDetect2 and BattyBirdNET, were run over all recordings, and

to balance their influence the 15 strongest recordings of each were taken. Where a recording ranks among the best for both, it is attributed to a single adapter so that the set holds 30 distinct recordings while keeping the two evenly represented. Table 8 lists the 30 recordings with the metrics that drove the selection: the adapter each was attributed to and, per adapter, the segment counts at confidence ≥ 0.75 alongside the total counts at confidence ≥ 0.3 . The recordings are grouped by their attributed adapter and, within each group, ordered by descending ≥ 0.75 count of that adapter.¹

No.	Location	Recorded	Via	BatDetect2 (≥ 0.75 / all)	BattyBirdNET (≥ 0.75 / all)
1	Location2	2025-09-30 20:00	BatDetect2	37 / 58	418 / 448
2	Location2	2025-09-30 20:15	BatDetect2	30 / 60	488 / 511
3	Location2	2025-10-10 03:00	BatDetect2	28 / 60	403 / 448
4	Location2	2025-10-10 01:45	BatDetect2	22 / 53	418 / 450
5	Location2	2025-10-10 20:45	BatDetect2	21 / 57	361 / 407
6	Location2	2025-10-10 03:45	BatDetect2	20 / 60	384 / 435
7	Location2	2025-09-30 19:30	BatDetect2	20 / 66	424 / 483
8	Location2	2025-10-10 20:30	BatDetect2	19 / 61	348 / 397
9	Location2	2025-10-09 23:15	BatDetect2	19 / 53	407 / 433
10	Location2	2025-10-09 23:30	BatDetect2	18 / 62	487 / 507
11	Location2	2025-10-09 22:30	BatDetect2	17 / 62	442 / 463
12	Location3	2025-10-10 23:15	BatDetect2	17 / 49	208 / 274
13	Location2	2025-10-09 23:00	BatDetect2	16 / 62	406 / 442
14	Location2	2025-10-10 03:15	BatDetect2	16 / 56	407 / 443
15	Location2	2025-10-10 03:30	BatDetect2	16 / 56	380 / 430
16	Location2	2025-09-30 19:45	BattyBirdNET	8 / 66	489 / 516
17	Location2	2025-10-10 05:15	BattyBirdNET	29 / 65	345 / 428
18	Location1	2025-10-05 19:30	BattyBirdNET	4 / 58	298 / 391
19	Location4	2025-09-20 18:00	BattyBirdNET	11 / 64	273 / 351
20	Location4	2025-10-05 18:15	BattyBirdNET	16 / 61	241 / 313
21	Location1	2025-10-05 18:15	BattyBirdNET	0 / 57	220 / 329
22	Location4	2025-09-30 19:30	BattyBirdNET	13 / 47	183 / 231
23	Location4	2025-09-30 20:00	BattyBirdNET	13 / 46	167 / 236
24	Location3	2025-10-10 01:45	BattyBirdNET	2 / 41	152 / 241
25	Location3	2025-10-10 05:15	BattyBirdNET	15 / 59	137 / 214
26	Location4	2025-09-30 19:45	BattyBirdNET	7 / 44	132 / 197
27	Location3	2025-10-13 05:15	BattyBirdNET	8 / 34	123 / 180
28	Location2	2025-10-04 05:00	BattyBirdNET	9 / 52	0 / 0
29	Location1	2025-10-15 18:00	BattyBirdNET	0 / 42	0 / 0
30	Location2	2025-10-07 05:15	BattyBirdNET	8 / 57	0 / 0

Table 8: The 30 recordings of the bat consensus set with the metrics behind the selection.

The selection concentrates the set on the recordings of highest bat activity, which was primarily at a single location and during a narrow part of the season. 18 of the 30 recordings originate from Location2, the location that contributes by far the most confident detections, and all 30 fall between 20 September and 15 October 2025. This follows the activity of the local bat species, which falls off as they enter

¹A few recordings attributed via BattyBirdNET carry low or zero BattyBirdNET counts because their recording locations were misassigned when the set was selected and corrected only afterwards.

hibernation in the following months [11]. The set therefore characterises bat detection on high-activity autumn nights at the busiest location, rather than uniformly across the campus and the full recording period.

In order to have three independent models for the ensemble, BatScope4 was included as a local variant of bat detection. Together they combine three complementary approaches: BatScope4 applies statistical machine learning trained on local species that detects individual calls only milliseconds long; BatDetect2 is a CNN developed specifically for bats that also predicts individual calls, including their frequency ranges; and BattyBirdNET is a CNN originally trained for birds, adapted to bat sounds with a fixed window of 0.5625 seconds. To let them vote fairly, all outputs are binned to the smallest mutual window length of 0.5625 seconds. BatScope4 and BatDetect2 produce much shorter outputs and activate the bin they fall in. When a detection lies between two bins, only the bin containing its centre is activated. Multiple detections in a single bin are combined into a single detection, retaining the maximum confidence value.

The predictions that are allowed to vote are gated by a minimum confidence threshold to reduce noise. This threshold is configured per adapter because their confidence distributions vary widely, as shown in Figure 12. BatDetect2's confidence distribution is bell-shaped, peaking at 0.6 and decaying quickly to a maximum of 0.898. Therefore the minimum confidence of predictions considered for the consensus is set to 0.7 to keep only the most certain. BattyBirdNET's and BatScope4's confidences concentrate near the top of the scale. That is why, for both, only predictions with confidence 0.9 or higher are kept for the consensus set.

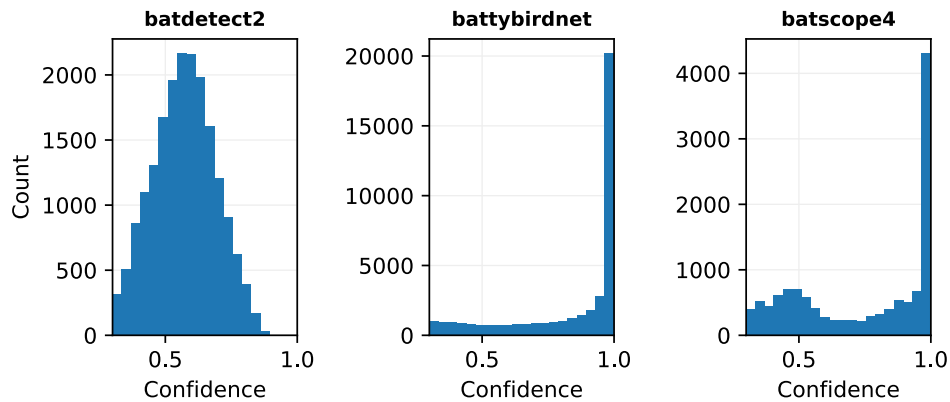


Figure 12: Confidence distribution and quantity of bat adapter predictions on all recordings.

With the remaining predictions, every adapter gets to vote for each 0.5625-second window. The consensus set contains annotations for all windows where the majority of adapters voted for. This means if at least two adapters vote for a detected bat presence, that window is marked as bat present.

4.2.3.1 Validity as a Reference

Because the reference is assembled from the detectors themselves, it captures inter-model agreement rather than ground truth: a genuine bat call that fewer than two of the adapters recognise never enters the set, so the reference represents only the presence the detectors agree on, not the true presence of bats. Precision can be assessed against it, and a recall can be computed, but that recall is measured relative to the consensus set and not to the calls truly present in the recordings. Because it records agreed presence rather than absence, the reference supports these presence-based metrics but not a true-negative count.

5 Design

The system was named *ost-wildlife* to indicate its scope to the OST campus while leaving room for future extensions of further locations than Rapperswil or other sensors like the camera traps. The adapters integrating the models are deliberately designed to be interchangeable because the underlying models are constantly improving. With the ground-truth limitations stated in Chapter 1.4, the focus lay on integrating and validating existing models.

5.1 Key Decisions

Building on that framing, a few further decisions shape the architecture below.

The first decision separates a low-confidence prediction stage from a later processing stage and keeps each adapter's raw, model-native output on disk. Filtering and canonicalisation happen in processing (Chapter 5.3.2), so thresholds can be re-tuned without repeating the expensive inference pass. Had filtering been applied during prediction, every change of mind would cost a full re-run.

The second decision is to hold the results in a relational database. The predictions are naturally tabular and the research dashboard relies on flexible filtering and aggregation across them (Chapter 5.4), which a relational store serves directly.

The third decision concerns the exploration surface: it is delivered as a web application that reads from the database directly, with no separate backend service (Chapter 5.5). This keeps it easy to deploy, shareable and operating-system independent, and leaves nothing extra to build or operate between the dashboard and its data.

5.2 System Architecture

In order to draw a clear picture of the system architecture, the C4 model for visualising software architecture is used [42]. Level 3 and 4 diagrams were omitted where the inclusion of such would be bloating rather than yielding clarity.

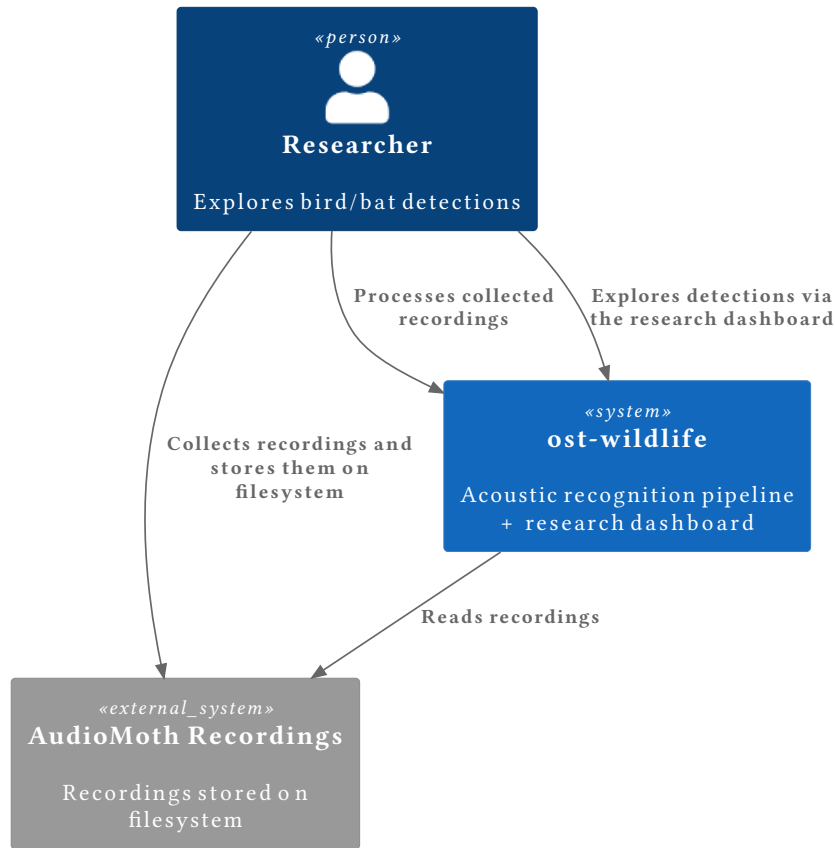


Figure 13: C4 Level 1 – system context: *ost-wildlife* in relation to its users and external systems.

Figure 13 displays the *Researcher*, primarily an actor of the ILF. They interact with the external system *AudioMoth Recordings* by writing the collected recordings to a filesystem in the given structured format. Using the *ost-wildlife* system, they process the recordings which the system reads directly from the filesystem. They also use the *ost-wildlife* system to explore the detections made on the processed recordings using the research dashboard.

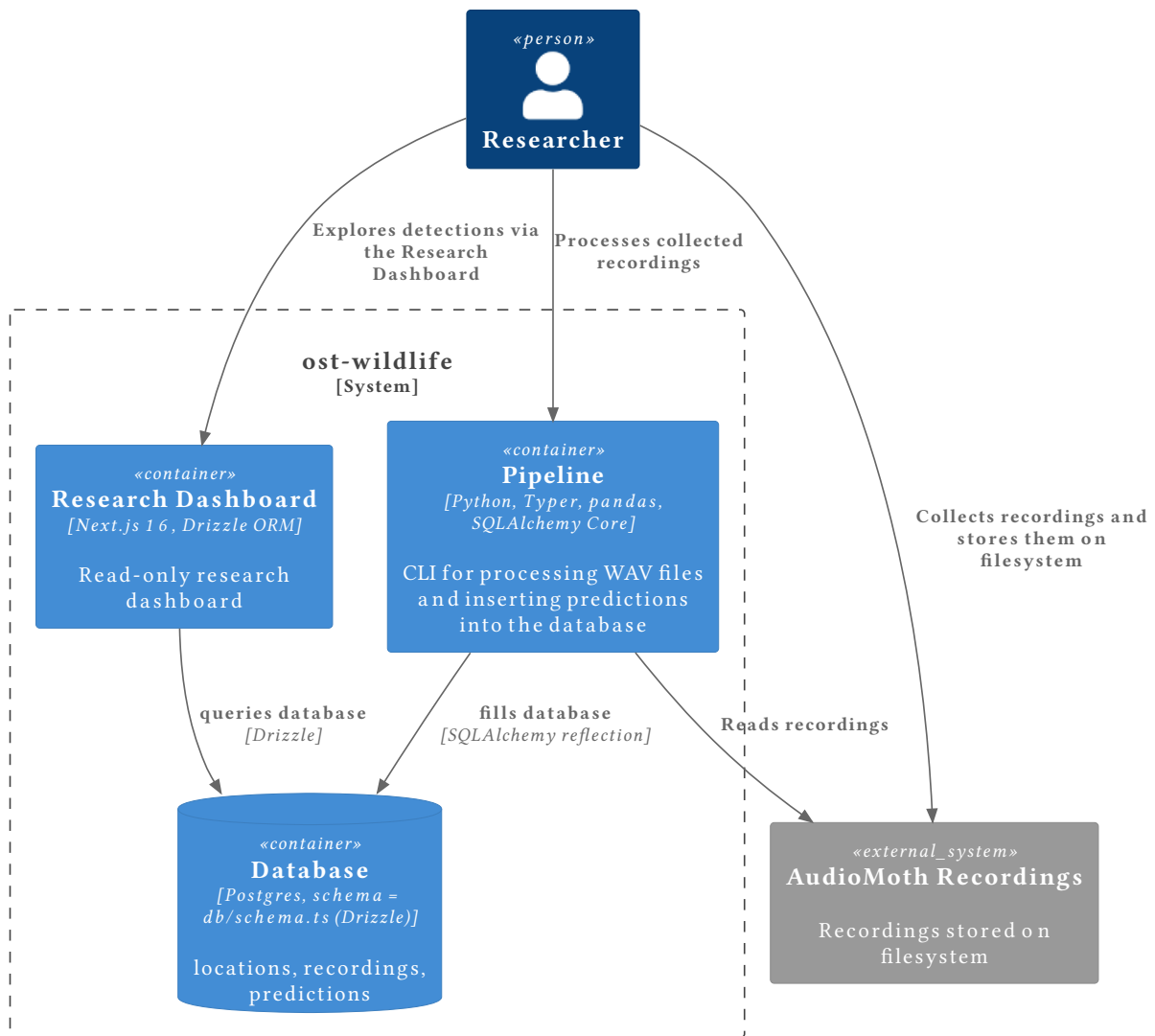


Figure 14: C4 Level 2 – containers: the three containers and their interactions.

Figure 14 visualises the split of the *ost-wildlife* system into three containers. The first to note is *Pipeline*, which is invoked by the *Researcher* to process the recordings of the filesystem and to fill the Postgres *Database* with the locations, recordings and predictions. The predictions can be explored using the *Research Dashboard* which queries its data from the *Database*.

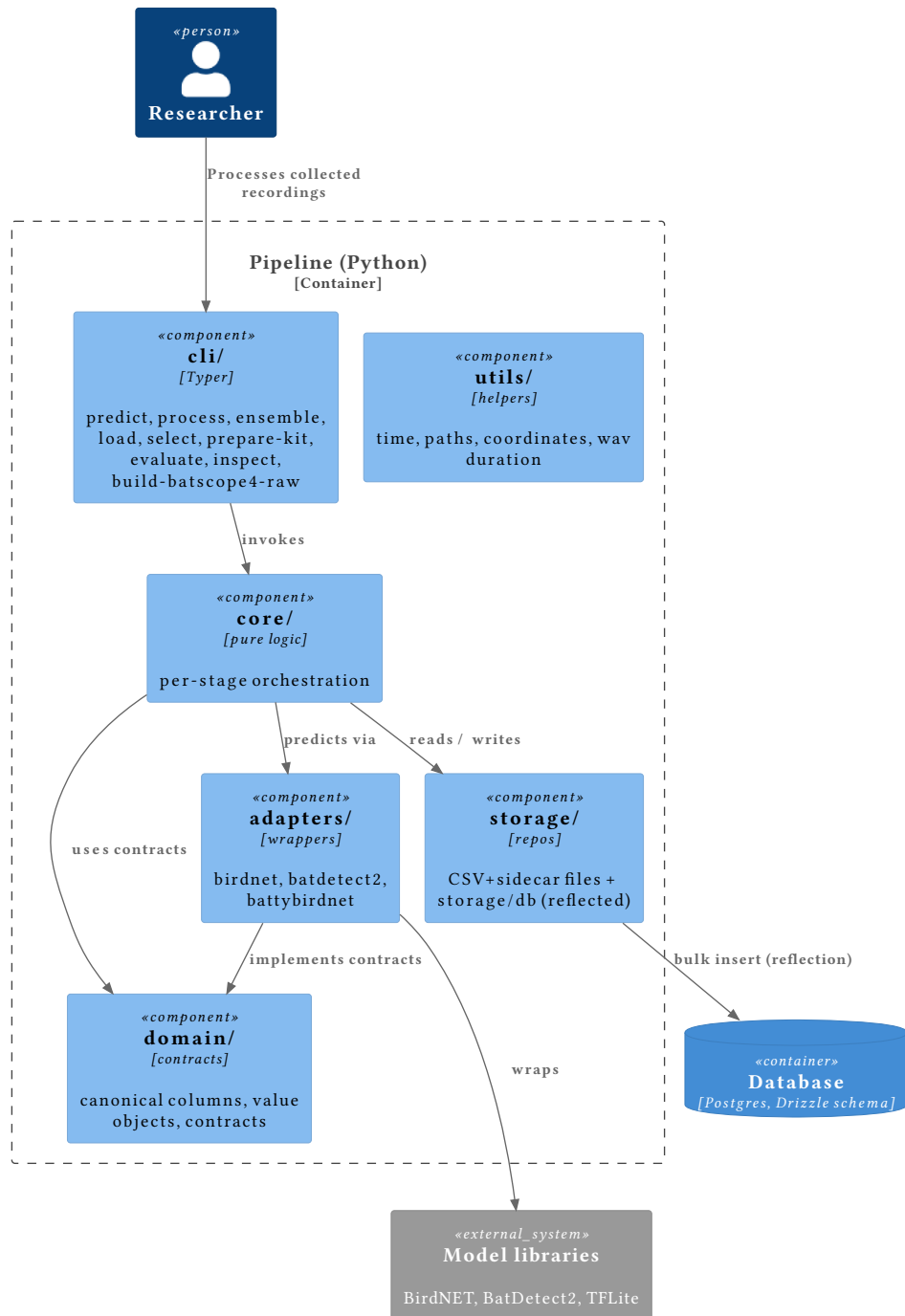


Figure 15: C4 Level 3 – pipeline components: CLI commands, core orchestration, adapters, and storage layers.

Figure 15 displays the components of the Python *Pipeline* container and how they interact. The *cli/* component defines the CLI commands and their arguments and invokes the logic of *core/* accordingly. The *core/* orchestrates a command by combining the logic it contains with contracts from *domain/* to invoke the *adapters/* in a uniform manner and the *storage/* repositories to read and write data. The *adapters/* wrap the model libraries and implement the contracts from *domain/*. Predictions are stored in the filesystem and reflected to the *Database* for the *Research Dashboard* to query. The *utils/* are helpers that may be used by any other component within this container.

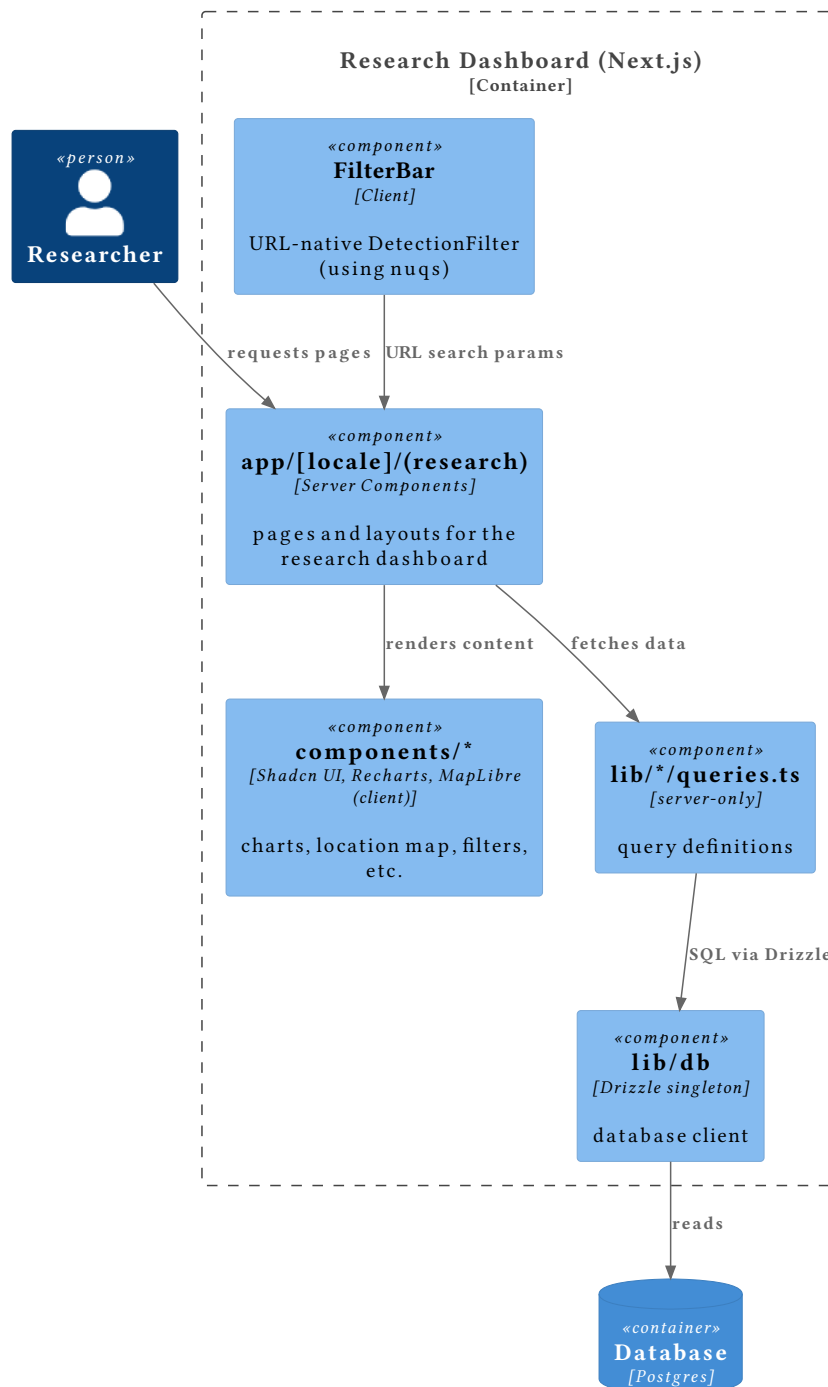


Figure 16: C4 Level 3 – research dashboard components: route groups, server components, and database access layer.

Figure 16 shows the components of the Next.js *Research Dashboard* container. It uses directory-based routing of `app/[locale]/(research)` to serve localised pages. The content is rendered as UI-components of the `components/*` directory for content blocks like charts, a location map and filter items. The data is fetched server-side using query definitions of `lib/*queries.ts` via a database client in `lib/db` that reads from the *Database*. *FilterBar* is a global filter bar that persists in the URL search params.

5.3 Pipeline Design

The end-to-end data flow is illustrated in Figure 17. It begins with AudioMoth recordings following the folder convention described in Chapter 4.1.2.1. The predict command traverses the filesystem and predicts in batches of one day and one device, mapping each device to a location code derived from the coordinates in its `_ort.txt` file. Each batch produces the adapter's raw, native output together with a sidecar that records which recordings were processed. The minimal confidence at the prediction stage is deliberately set low to allow inspection of low confidence detections and how they compare to high confidence ones. The location-code scheme and the on-disk output layout are detailed in Chapter 6.1.

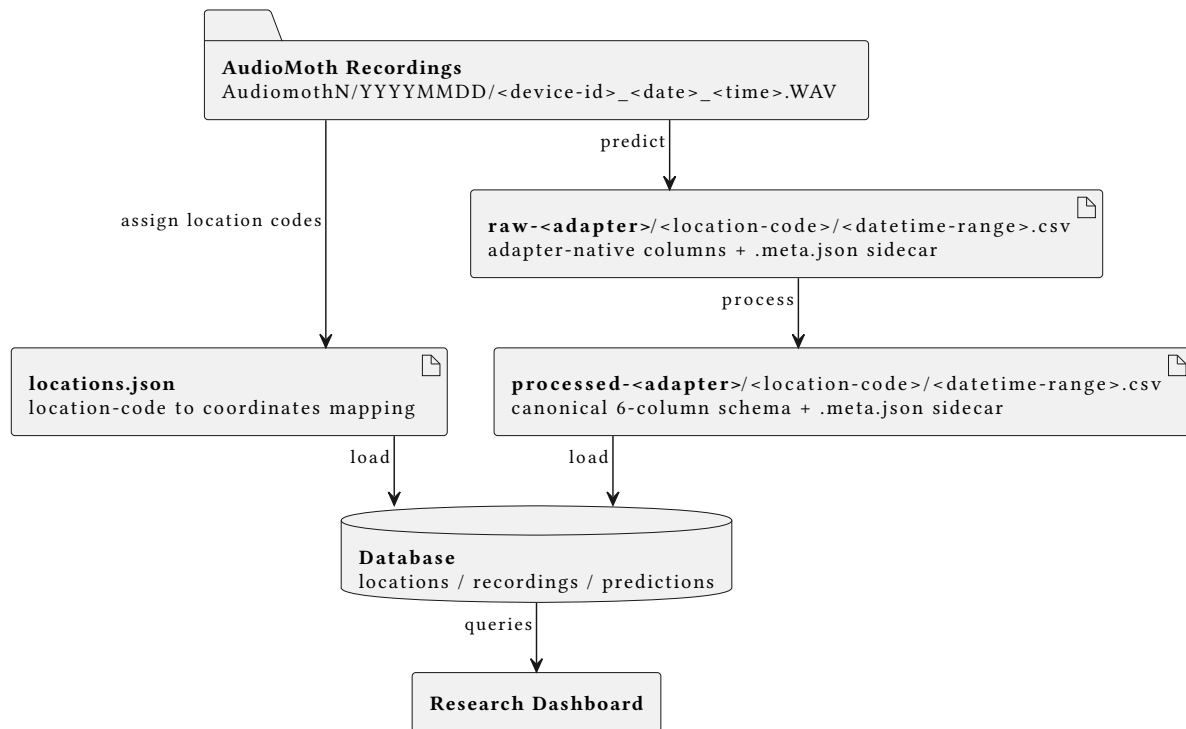


Figure 17: Data flow from the raw recordings to the research dashboard.

The next step process is there to canonicalise the output as described in Chapter 5.3.2.1 and filter the data further. The processed output file structure mirrors the one of prediction output. Because the raw prediction output is stored, the processing step can be repeated with various configuration values. The load step fills the processed predictions, their recordings' information and the locations into the database. Lastly, the research dashboard will query its data from the database.

5.3.1 Model Integration

The implementation of adapters is designed for them to be interchangeable, so that future improved models or custom trained ones can be easily plugged into the existing ecosystem. To achieve this abstraction, a Predictor interface is used as displayed in Figure 18. It includes a static method `from_config(config)` that provides the adapter implementations to access their config variables at initialisation. The `load()` method is for computationally expensive code, like loading the model, and therefore only called once in the beginning of the adapter's lifecycle. The `predict(files)` method is designed to predict a batch of files.

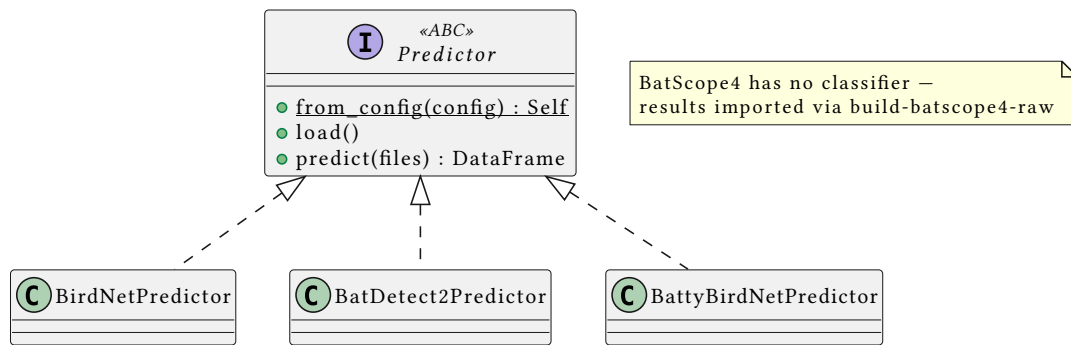


Figure 18: Class diagram of the Predictor interface with implementations per adapter.

5.3.1.1 Implemented Adapters

The four tools selected in Chapter 3.2.3 are taken on as adapters; the implementation-specific reason for each is noted below:

- **BirdNET**: Implemented first, chosen for its popularity and Python package that allows quick integration. Its precision was validated by the ILF and deemed sufficient, so that no further bird adapter was considered.
- **BatDetect2**: First bat adapter implemented, because it predicts bounding boxes of individual calls. This provided guidance to recordings of high probability of activity and the bounding boxes directly pointed to areas of interest. An excerpt of each species, even with low confidence, can be found in Appendix F.4.
- **BattyBirdNET**: Given the satisfactory performance of BirdNET, this transfer learning approach is a promising variant with a regionally trained classifier of Bavaria.
- **BatScope4**: This tool is not compatible with the Python pipeline, however it is a valuable model to compare predictions of other adapters against. It is developed by the WSL institute located in the canton of Zurich, an indicator that it is used and tested on bats of this region and has been around and continuously developed for years [43]. It was used to process selected recordings in the GUI, query the predictions from their database and canonicalised to make it comparable and usable for the bat consensus set (see Chapter 4.2.3).

An adapter that simply measures energy levels in the audio recordings is considered susceptible to noise on the same frequency bands that are only distinguishable by details like their shape. Existing models are more sophisticated and less susceptible in that regard. Therefore, such primitive adapters are not implemented.

5.3.1.2 Prediction Flow

The flow of batch predicting a stem of collected recordings is specified in detail in Figure 19. It begins by collecting all leaf folders, which are the single batch unit (one day, one device). Next the `predictor.load()` method is called once initially. Next, the loop of predicting batches begins and only ends once all folders have been predicted. The first steps per folder are to collect all WAV files within and filter out faulty ones. At this point, the BirdNET implementation geo-restricts possible species based on season. In the bat lane on the other hand, files that are outside the configured active window or recorded below a configured minimal sample rate, are filtered out, avoiding unnecessary computing on recordings where bats are inactive or not within the recorded frequencies. This repeats until all leaf folders have been predicted.

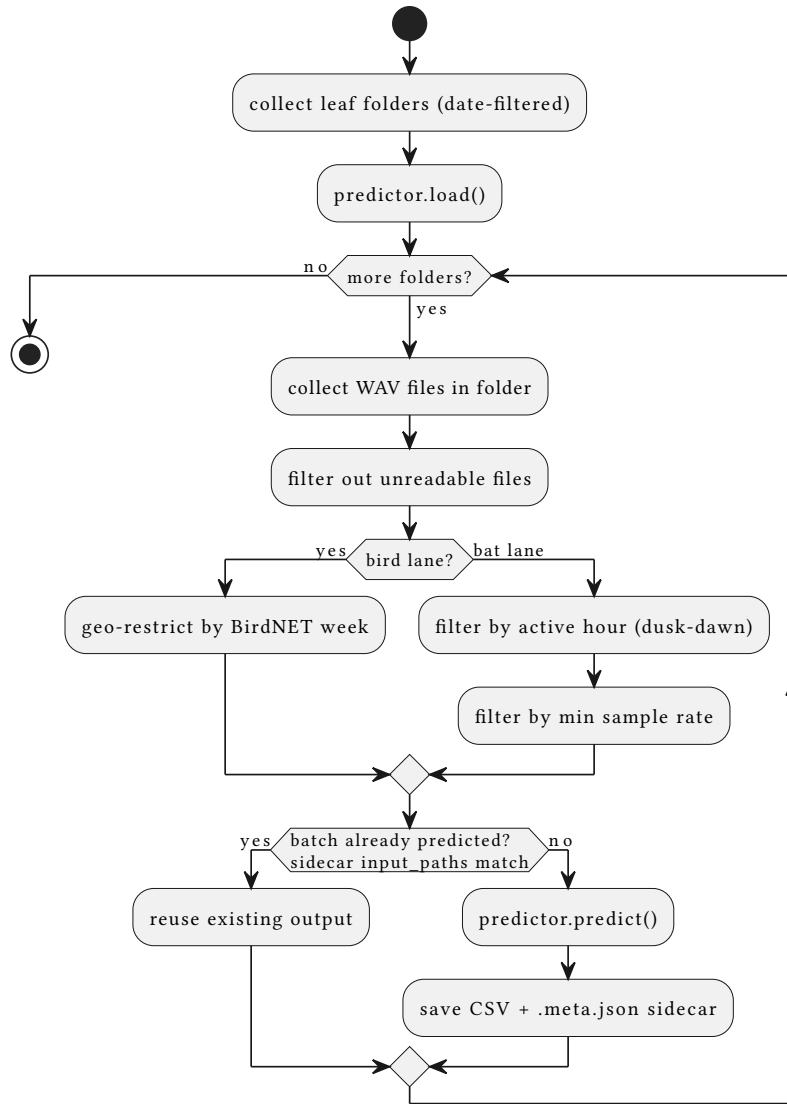


Figure 19: Flow chart of batch predictions.

Now, having the final list of files to predict, it is checked if that batch has already been predicted on the respective adapter. Only if the list of WAV files matches exactly is the batch skipped. Otherwise the `predictor.predict()` is called and the predictions are written to the output, overwriting potential prior predictions because the underlying data has changed.

5.3.2 Prediction Processing

Processing transforms the raw, adapter-native output into a uniform, filtered form before it is loaded into the database. It canonicalises the differing column schemas into one shared layout and applies further filtering, such as a minimum-confidence threshold and the merging of adjacent segments. Because the raw output is retained, this step can be repeated with different filter values.

5.3.2.1 Canonical Prediction Schema

Column names of the output predictions vary between the adapters. In order to compare them against each other and to enable the research dashboard to offer all functionality for birds and bats alike, a canonical prediction schema is defined. Table 9 shows the mapping of native adapter columns to the canonical form. BatDetect2 is the only adapter that predicts a frequency range additional to the time

range; these `high_freq` and `low_freq` columns are kept during processing as optional extra fields for analysis purposes.

The schema also distinguishes two confidences: a detection confidence – whether a vocalisation is present at all – and a classification confidence – whether it is the named species. Only BatDetect2 reports the two separately, as its detection score and class probability; the other three adapters (BirdNET, BattyBirdNET and BatScope4) emit a single score that is mapped to both fields.

Adapter	input	start_time	end_time	species_name	confidence_ detection	confidence_ classification
BirdNET	input	start_time	end_time	species_name	confidence	confidence
BattyBirdNET	input	start_time	end_time	species_name	confidence	confidence
BatDetect2	input	start_time	end_time	class	det_prob	class_prob
BatScope4	input	start_time	end_time	species_name	confidence	confidence

Table 9: Native output columns per adapter mapped to the canonical prediction schema.

Further canonicalisation includes the normalisation of the path to the recording by making it relative to the input root. This way the paths of two recordings equal even if they were predicted from different environments.

5.3.2.2 Binning and Merging

A prediction with a start and end time is called a *segment*. Segments can first be binned to a defined width; the bat adapters do this to achieve a uniform grid for the ensemble method as described in Chapter 4.2.3. This is optional, just like the merging that follows. The merging allows a maximum gap duration to be defined; if two segments are separated by less than that duration, they are combined into a single one with the maximum confidence as the new confidence.

5.3.3 Configuration

The YAML configuration is chosen for its slim structure that allows the researcher to easily make adjustments if needed. The `config/default.yaml` is the base configuration that sets default values for any setup. Override files like `ilf.yaml` – the configuration for the researcher’s workstation – are applied on top of the base configuration if selected. The effective configuration has some global configurations for computation and paths. The bird and bat lanes are separate, both with a defaults object each. Then, each adapter can overwrite fields where specialised values are needed. The `batensemble` config field is separate, because it does not follow the same predict and process steps. A visual representation of this can be seen in Figure 20.

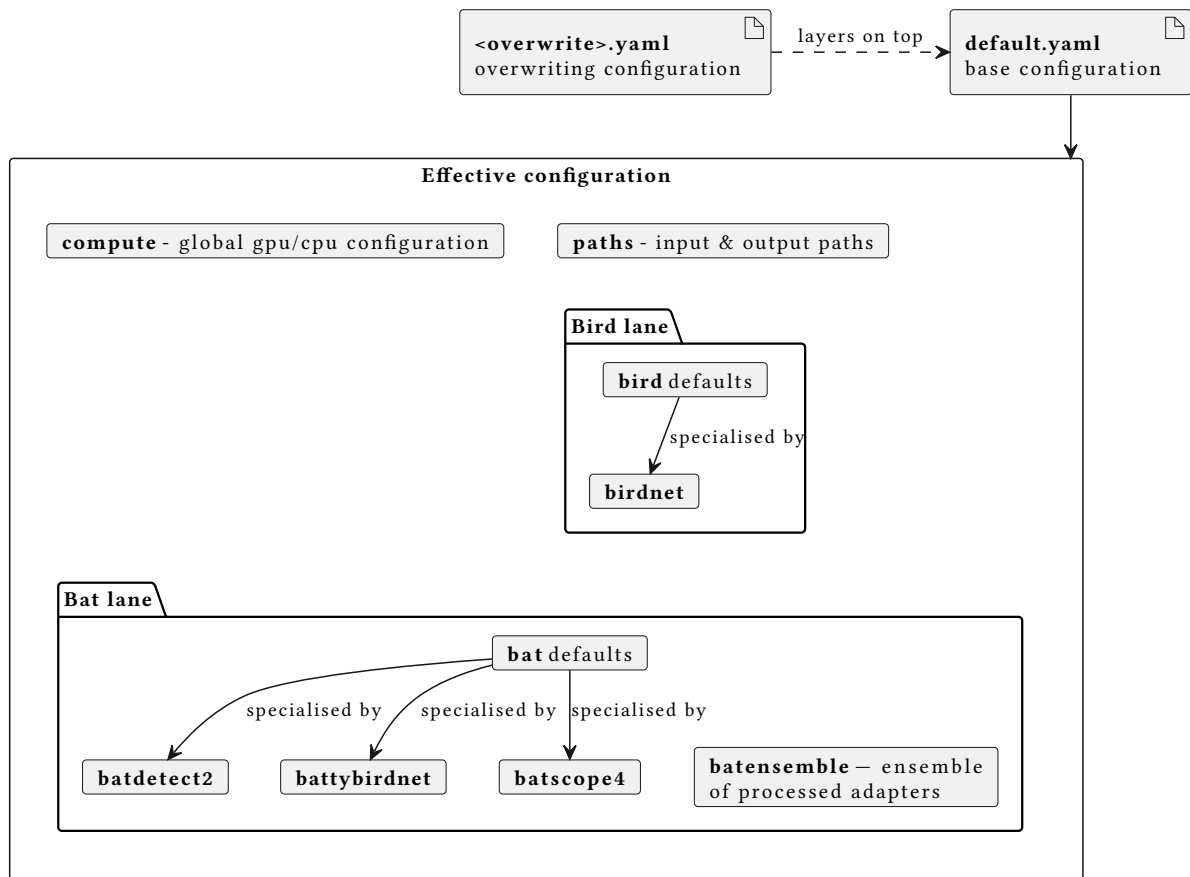


Figure 20: The YAML configuration architecture.

5.4 Database Design

A relational database was chosen to hold the results, as the prediction data is naturally tabular and the research dashboard relies on flexible filtering and aggregation across it. The concrete database and how its schema is defined are described in Chapter 6.3.

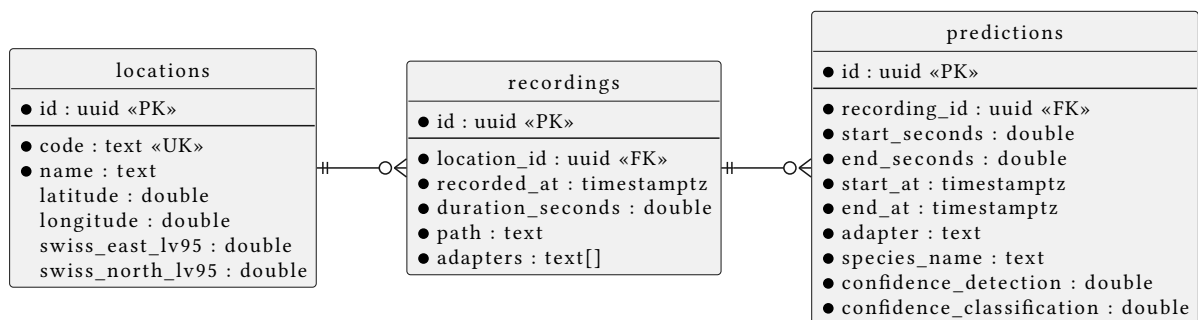


Figure 21: Database schema.

The database schema (Figure 21) is designed to reflect the processed *predictions* including their metadata like *recordings* and *locations* to explore the data using the research dashboard. The *locations* include both the international WGS 84 coordinates because they are accepted more broadly and the Swiss LV95, because those are the original coordinates stored on the filesystem with the recordings. The code is auto-assigned during processing as an easier way to handle location assignment and a more human friendly distinction than coordinates. The location name currently equals the code but is deliberately left non-unique so the schema would allow custom names for the research dashboard.

The *recordings* are assigned to a single location and are assigned themselves a list of predictions. They also carry more details like the `recorded_at` timestamp, the duration in seconds, the path relative to the input root and a list of adapters that were run on this recording. The list of adapters cannot be inferred from their predictions because a recording might not have a single prediction even though it was processed.

The *predictions* get their location through their relationship with a recording. They hold the start- and end-seconds of the prediction which are relative to the start of the recording. For convenience, the start- and end-timestamp are computed during insertion by adding offset seconds to the `recorded_at` timestamp. The adapter that made the prediction, the species they predicted, confidence of detection and confidence of classification are all stored on the *predictions* table to filter by them.

5.5 Research Dashboard Design

The research dashboard is the system's exploration surface: it lets the researcher analyse the detections. It is delivered as a web application so that it is easy to deploy and share and it is operating-system independent. To keep the architecture lean it runs no additional backend service – each page reads from the database directly when it is requested, so there is nothing extra to build or operate between the dashboard and the data.

The views allow the researcher to drill down from various interconnected contexts: an overview, a per-species catalogue, a per-location view, a recordings browser and a comparison view, whose component structure is shown in Figure 16. A single global filter – by adapter, location, confidence and date – scopes every view and is carried in the page address, so any filtered view can be bookmarked or shared.

6 Implementation

This chapter describes how the design (Chapter 5) was realised in concrete technology. It covers the data pipeline and its model adapters, the command-line interface and configuration, the database, and the research dashboard, followed by the testing setup and the handover to the ILF.

6.1 Data Pipeline

The predict stage walks the ILF folder convention (Chapter 4.1.2.1) directly. From the input root it descends through each device folder into the day folders – the leaf folders – that hold the recordings. Each leaf folder is one device’s recordings for a single day and forms one prediction batch; a batch must cover exactly one calendar day, which it does according to the convention, because a prediction row stores only a time of day and takes its date from the folder it came from. This stage can be scoped to a date range that may be open on either end.

A batch’s location is resolved from coordinates, not from the device that recorded it. Every device folder carries an `_ort.txt` file holding the recorder’s LV95 coordinates. The pipeline maps those coordinates to a location code through a `locations.json` registry, reusing the code when the coordinates are already known and otherwise appending a new entry with the next code in sequence (`Location1`, `Location2`, and so on). The four campus locations were seeded manually, numbered anti-clockwise (Figure 6). Any further location will automatically be assigned a new code, so adding a microphone needs no code change.

For each leaf folder the configured adapter runs once over the batch and writes its raw, model-native predictions as CSV to the output tree, beside a JSON sidecar listing the files that were processed alongside any that proved unreadable and were left unprocessed. The bat lane first discards recordings that cannot carry usable bat calls – those recorded outside the night-time window of 18:00 to 08:59, and those below a 240 kHz sample rate (Figure 9) – so no inference is spent on them. The 240 kHz gate follows from the Nyquist limit: the birds-only 48 kHz recordings represent frequencies only up to 24 kHz, far below bat echolocation, whereas the 250 and 384 kHz recordings reach well into the ultrasonic and clear the gate. A batch already predicted from the same set of files is skipped when the stage is re-run, keeping prediction, the expensive step, from repeating.

The process stage turns this raw, model-native output into the uniform form the rest of the system consumes. Its central step is canonicalisation: each adapter maps its native columns onto one shared prediction schema (Table 9), so bird and bat predictions become directly comparable no matter which model produced them. Recording paths are rewritten relative to the input root at the same time, so a recording resolves to a single path regardless of where it was processed.

On that shared schema the stage then filters and tidies the predictions, where the two lanes – bird and bats – differ slightly while keeping the same shape. Both drop predictions below a minimum confidence – 0.2 for birds and 0.3 for bats – and the bat models’ noise and reject classes are removed here. The bird lane merges consecutive same-species segments no more than three seconds apart – one BirdNET window – into one, so a call detected across neighbouring windows is recorded as a single vocalisation rather than several; the bat lane instead snaps each segment onto the shared 0.5625-second grid used for the consensus (Chapter 4.2.3). Because the raw predictions are kept, this stage can be re-run with different thresholds without predicting again. The processed output – a parallel tree mirroring the raw one – is then loaded into the database (Chapter 6.3) for the research dashboard to query.

6.1.1 Bird Model

BirdNET 2.4 is run through its official `birdnet` Python package, which handles its preprocessing internally: the adapter passes raw file paths, and the package resamples each recording to the 48 kHz BirdNET expects, splits it into consecutive 3-second windows, and computes for each the mel spectrogram BirdNET classifies – whose layout, the STFT window and mel bins, is fixed by the model. The windows are left non-overlapping, the package’s default. Before classifying, the adapter narrows the species BirdNET may predict to those plausible on campus using BirdNET’s GeoModel: given the campus coordinates, a week of the year and a minimum occurrence confidence of 0.01, the GeoModel returns the species whose estimated occurrence at that place and time exceeds the threshold.

The GeoModel expects the week not as a calendar date but as one of 48 – BirdNET divides the year into twelve months of four weeks each. The adapter approximates this from a recording’s day of the year d as $w = \left\lfloor \frac{(d-1) \cdot 48}{365} \right\rfloor + 1$, clamped to the range $[1, 48]$. Recordings are grouped by this week, and the corresponding species list is generated once per distinct week and reused for every recording that falls in it.

Supplying this list is optional, but restricting the candidates to species plausible for the location and season reduces false positives – a species that does not occur on campus can no longer be predicted at all. This follows BirdNET’s own best-practice guidance for species lists [44].

The occurrence floor itself is set just above zero, at 0.01, so the GeoModel acts as a permissive plausibility prefilter rather than a hard ecological gate: any species with a non-negligible modelled occurrence stays predictable. This is deliberate. With no prior species list for the campus, and no certainty about how conservatively the GeoModel scores a present-but-uncommon species, a higher floor risked silently excluding a species that is in fact present – an exclusion nothing downstream could recover. A near-zero floor avoids that, and the resulting over-inclusion is handled where it is cheaper: the per-prediction confidence threshold and the expert plausibility review remove implausible species afterwards. The precision/recall trade-off is thus moved from the prefilter, where a wrong exclusion is irreversible, to the confidence threshold, where it is tunable.

BirdNET’s acoustic model is queried at a low confidence floor of 0.1. Keeping every prediction down to this floor lets the raw output be re-filtered to any stricter threshold later in processing, without repeating the costly inference pass; anything below 0.1 is assumed too unlikely to be a genuine vocalisation and is dropped at the source. After listening to a few samples, it was even raised to 0.2 at processing.

6.1.2 Bat Models

BatDetect2 is run through its `batdetect2` package (version 2.0.0b1). The adapter passes the recording paths to the package, which does its own reading and preprocessing – resampling each recording to its 256 kHz target internally. Unlike the window-based, bird-derived classifiers, BatDetect2 detects individual echolocation calls: each detection carries a start and end time, a frequency range and a detection score alongside the predicted species. A call lasts only milliseconds, and these detections are normalised onto the shared grid in processing.

BattyBirdNET reuses BirdNET’s embedding model as a feature extractor with a small bat-classifier head on top. The head is the full BattyBirdNET-Bavaria-256kHz model, trained on regional Bavaria data. A -high variant with fewer noise-rejection classes also ships, but the full model is used: background-noise classes serve to reduce false positives [45], and it carries a far broader set of them. Several correspond to the non-bat sounds these recordings actually contain – among them `electrical`, `rain` and even an `audiomoth` class for the recorder used here – whereas the -high variant keeps only `insect` and `mechanical`. Switching variants is a configuration change that needs no code.

The ultrasonic trick that lets this bird model read bat calls is realised by resampling each recording to BattyBirdNET’s native 256 kHz in the adapter and feeding the raw windows to the embedding model tagged as 48 kHz. Each 0.5625-second window is then read as the model’s usual 3-second window, moving the ultrasonic calls down into the 0–15 kHz band BirdNET was trained on; the timestamps are scaled back to real time afterwards.

Because BatScope4 has no scripting interface (Chapter 5.3.1.1), the selected recordings were processed by hand in its GUI, following the workflow advised by [43]. Calls were first cut from the recordings with the *Period Cutter*, which splits calls more cleanly than the alternative *Peak Search Cutter*. They were then reviewed in the *Bat Inspector* with a 7.5 kHz high-pass filter, below which no bat calls are expected. For classification, both available classifiers – KNN and SVM – were run and their confidence values averaged. BatScope4 additionally reports a 95% confidence interval as a measure of statistical certainty, and only predictions that lie within it are retained; the rest are filtered out.

Species labels are deliberately left in each adapter’s own vocabulary rather than mapped onto a shared one. BatDetect2’s predictions in particular retain its short class codes (e.g. pipnat) rather than scientific names, so the stored output mirrors exactly what the model emitted and the full prediction detail stays available to the researcher. Table 10 lists, per focus bat species, the native label each adapter produces, alongside the scientific name each is mapped to for the final cross-adapter comparison (Table 13). BatDetect2 emits six-letter codes (– marks a species absent from its label set), BattyBirdNET appends an English common name after _, and BatScope4 emits the scientific name directly; the labels were verified against the BatDetect2 checkpoint, the BattyBirdNET-Bavaria label file and BatScope4 output.

Scientific name	BatDetect2	BattyBirdNET	BatScope4
<i>Pipistrellus pipistrellus</i>	pippip	Pipistrellus pipistrellus_Common pipistrelle	Pipistrellus pipistrellus
<i>Pipistrellus kuhlii</i>	–	Pipistrellus kuhlii_Kuhls or Nathusius pipistrelle	Pipistrellus kuhlii
<i>Pipistrellus nathusii</i>	pipnat	Pipistrellus nathusii_Nathusius or Kuhls pipistrelle	Pipistrellus nathusii
<i>Myotis mystacinus</i>	myomys	Myotis mystacinus_Whiskered or Daubentons bat	Myotis mystacinus
<i>Myotis daubentonii</i>	myodau	Myotis daubentonii_Daubentons or Whiskered bat	Myotis daubentonii
<i>Pipistrellus pygmaeus</i>	pippyg	Pipistrellus pygmaeus_Soprano pipistrelle	Pipistrellus pygmaeus

Table 10: Native output label per species-level adapter for each focus bat species.

6.1.3 Computing

Which processor each model runs on is fixed once in the configuration. A single `compute.device` value – auto, cpu, gpu, or gpu:N for a specific physical card – is translated into the `CUDA_VISIBLE_DEVICES` setting the underlying frameworks honour. Before any framework initialises, a requested card is

checked against those `nvidia-smi` reports, so an index that does not exist on the machine fails loudly instead of quietly running somewhere unexpected.

The two model families – PyTorch and TensorFlow – need different handling on the GPU. BatDetect2 runs on PyTorch and takes the device pin directly. The BirdNET-derived adapters are more delicate: the `birdnet` package pins the device inside its own worker processes and assumes it alone owns a single card, which breaks if the parent process has already claimed the GPU. For those adapters the parent therefore does not claim the GPU and lets each worker claim the card. On the GPU, BirdNET also switches from its CPU-only TensorFlow Lite model to the TensorFlow SavedModel that can run on the card.

6.1.3.1 DGX Remote

To run inference of the adapters over the whole database efficiently, the DGX-2 [46] provided by the ICAI was used. The machine was reached over secure shell (SSH): an interactive session launched the inference, and recordings were moved to and from the server with `rsync`, a CLI tool that transfers only the changed files over that same connection.

The DGX-2 is part of the ICAI’s shared high-performance computing (HPC) infrastructure, whose provided setup requires workloads to be supplied as an Apptainer image. The pipeline was therefore packaged as such an image so that it would run unchanged in that environment, following the system requirements of the remote as set up by the ICAI. The image is built from a slim Python 3.12 base, adds the system libraries the audio stack needs (`ffmpeg`, `libsndfile`), and installs the project’s pinned Python dependencies with `uv`. It is run with the `--nv` flag so the container can reach the host GPU.

6.2 Configuration and Command-Line Interface

The CLI has primarily three commands to be used by the researcher and many more that were created specifically as helpers for this thesis and only left in there for further development purposes. The CLI goes under the name `ost-wildlife` and is built using Typer, a Python library for CLIs.

A global `--override` option selects the configuration to apply on top of `default.yaml`. The commands of the pipeline for researchers are:

1. **predict**: has mandatory option to specify which `--adapter` to predict; `--adapter all` is useful to predict all at once e.g. over night. Optional arguments to limit the dates are `--from-date` and `--to-date`. If there is a sidecar listing all the same files of the batch to be processed, the batch will be skipped as *already processed*. However, if they do not match up, the command will be aborted with a descriptive error message, because already-processed data changed since the last run. Running it with `--force` will overwrite existing predictions.
2. **process**: has mandatory option to specify which `--adapter` to process; includes `batscope4` that is usable in combination with manual data imports. It always clears the existing processed data before processing to ensure no old artefacts remain and because it is quick enough to do a clean re-process.
3. **load**: has also mandatory `--adapter` option. It will upsert the processed predictions, recordings and locations to the database.

6.3 Database and Loading

The database is Postgres 18. Its schema is defined once in TypeScript with Drizzle, a lightweight TypeScript-first object-relational mapping (ORM), in `db/schema.ts`. That single source of truth generates the SQL and any migrations, doubling as the type-safe definitions for an SQL-like query builder used by the research dashboard.

The pipeline does not declare the schema a second time. The Drizzle definitions in `db/schema.ts` are the single source: the migrations that build the Postgres schema are generated from them, and at run time the Python loader reflects the tables straight from the live database rather than from a schema of its own. The dashboard thus works against the Drizzle definitions directly and the loader against the database those same definitions produced, so the two cannot drift apart. This keeps the data layer thin – there is no parallel schema to declare or maintain on the Python side – and fails safely. A renamed or dropped column surfaces as an error when the reflected table is queried, rather than as a silently wrong result.

All tables use `uuidv7` as their `id` primary key column – a time-ordered globally unique value [47]. It was chosen to gather experience applying it rather than for its necessity; incrementing numbers would have been sufficient as well. Every table additionally has columns for `createdAt` and `updatedAt`, which allow tracing back to when the data was loaded. `createdAt` is initialised with the current date when a row is inserted, `updatedAt` is defined as a trigger on each table, that fires on changes and updates the rows' date.

The recordings table has a uniqueness constraint on the pair: `location_id` and `recorded_at`. This is a safeguard, because no location should be able to have two recordings at the same time. Because database use is read-heavy, indexes are created on the predictions table to aid query performance. B-tree is used as the index type for its general-purpose applicability:

- **`predictions_recording_id_idx`** on `recording_id`: Every recording-scoped view joins predictions to recordings on this key, and re-ingestion deletes a recording's predictions by it. The index turns these lookups into a direct range read instead of a full table scan.
- **`predictions_species_start_at_idx`** on `(species_name, start_at)`: Serves the per-species views. The researcher picks one or more species and a date range. Leading with `species_name` isolates the species; the trailing `start_at` then satisfies the time window and the chronological ordering from the same index.
- **`predictions_adapter_species_idx`** on `(adapter, species_name)`: Supports filtering predictions to a given model adapter. `adapter` is low-cardinality, so pairing it with `species_name` makes the index more selective.

The load stage fills the database in three ordered steps that mirror the schema (Figure 21). Locations come first: each entry of the on-disk registry becomes a row carrying both its original LV95 coordinates and the WGS 84 pair derived from them. Recordings follow, gathered from the processed sidecars and de-duplicated by location and recording time across all adapters. A recording is kept even when an adapter predicted nothing on it, so each row also accumulates the set of adapters that processed it. Predictions load last, each attached to its recording. Their start and end seconds are stored as given, and the absolute timestamps are computed by adding them to the recording's start time. Loading an adapter is idempotent: by default a recording that already holds that adapter's predictions is left untouched. A forced reload deletes only that adapter's rows for the recording before re-inserting, so one adapter can be reprocessed without disturbing the others.

6.4 Research Dashboard

The research dashboard is built with Next.js 16, a React framework chosen for its file-based routing and server-side rendering (SSR). Its pages live under the localised route group `app/[locale]/(research)`, and all database access happens in server components and server actions, so the server fetches the data and returns fully rendered HTML with no separate client-side fetching or API layer. It is written in TypeScript and shares the `schema.ts` Drizzle definitions used by the pipeline, so its queries are type-checked against the database schema. Charts are rendered with Recharts and the geographic location map with MapLibre GL. All pages are served in two languages through the localised `app/[locale]`

route group. The dashboard is organised as a set of interlinked views: an overview, a species catalogue with per-species pages, a list of locations with per-location pages, a recordings browser leading to a per-recording inspection page and a page to compare species’ hour-of-day activity to each other. Their recurring components and design choices are described below.

A single sticky filter bar scopes every view (Figure 22). Its state is held entirely in the URL query string, so any filtered or drilled-down view can be bookmarked or shared and reopens exactly as left. The first control selects birds or bats – one of the two is always active – whereas the bat adapter itself is chosen on a separate settings page to keep the bar uncluttered. The remaining controls restrict the recordings considered to a single location, a date range and a minimum confidence, and a species multi-select narrows the view to chosen species; selecting none means all. Only the controls that make sense for the current view are shown – on a single recording, for instance, the location and date controls fall away.



Figure 22: The sticky global filter bar shared by every dashboard view.

The recordings browser lists every recording under the active filters, adds a free-text search over the table, and lets most columns be sorted (Figure 23). It leads to a per-recording page that lays each prediction out on a timeline of the file with its confidence and time range.

Recordings

Every audio file the pipeline has ingested.

Search path...

Path	Recorded at	Duration	Detections	Max. confidence	Location	Species
20251212/Audiomoth2/20251212/24809B016487A731_20251212_13...	Dec 12, 2025, 01:00 PM	5:00	3	93%	Location3	Fulica atra
20251212/Audiomoth3/20251212/243B1F026487A7FE_20251212_12...	Dec 12, 2025, 12:45 PM	5:00	1	96%	Location4	Carduelis carduelis
20251212/Audiomoth2/20251212/24809B016487A731_20251212_12...	Dec 12, 2025, 12:45 PM	5:00	10	98%	Location3	Corvus frugilegus Fulica atra
20251212/Audiomoth1/20251212/243B1F026487A7CB_20251212_12...	Dec 12, 2025, 12:45 PM	5:00	4	74%	Location2	Fulica atra
20251212/Audiomoth2/20251212/24809B016487A731_20251212_12...	Dec 12, 2025, 12:30 PM	5:00	7	99%	Location3	Fulica atra
20251212/Audiomoth1/20251212/243B1F026487A7CB_20251212_12...	Dec 12, 2025, 12:30 PM	5:00	2	92%	Location2	Corvus corone Fulica atra
20251212/Audiomoth1/20251212/243B1F026487A7CB_20251212_12...	Dec 12, 2025, 12:15 PM	5:00	3	91%	Location2	Corvus corone Fulica atra
20251212/Audiomoth2/20251212/24809B016487A731_20251212_12...	Dec 12, 2025, 12:15 PM	5:00	6	90%	Location3	Corvus corone Fulica atra
20251212/Audiomoth3/20251212/243B1F026487A7FE_20251212_12...	Dec 12, 2025, 12:00 PM	5:00	3	63%	Location4	Anas acuta Erithacus rubecula

Figure 23: The recordings browser with free-text search and sortable columns.

The location map draws one marker per recording site over a swisstopo map served through MapLibre GL (Figure 24). Each marker’s area – not its radius – scales linearly with the number of detections at that site relative to the others currently shown, so that a site with twice the detections occupies twice the visual area rather than four times. The map can be read for all species at once or for a single species.

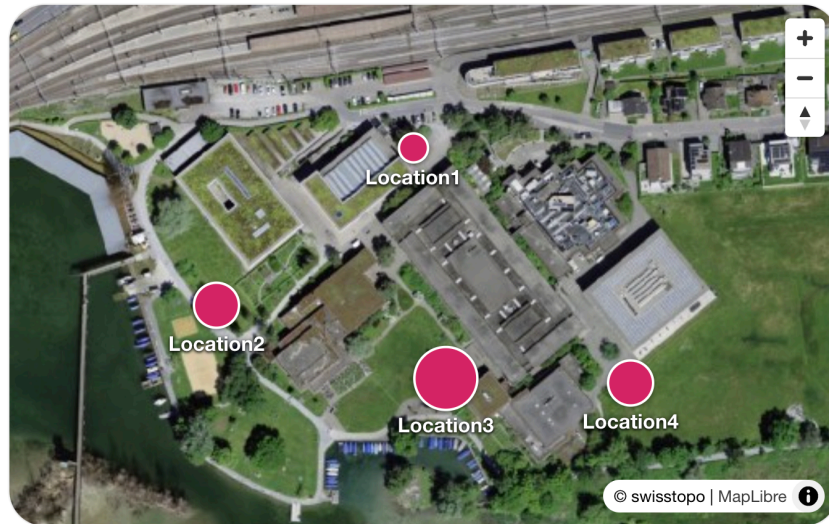


Figure 24: The location map of the recording sites.

Two time charts share the same normalisation choice. The detection rate over time (Figure 25) and the daily activity pattern by hour of day (Figure 26) both plot detections per recording rather than raw counts, because the recording schedule covers neither every hour nor every day equally; dividing by the number of recordings in each bucket removes that coverage bias and makes the buckets comparable. On the detection rate over time, days with no recordings are shaded rather than plotted as zero, so missing coverage is not mistaken for a recorded day on which nothing was detected. The detection rate can be viewed per species, over the whole campus or for a single location, and the daily activity pattern for one species or several at once for comparison.



Figure 25: Detection rate over time for the most-detected bird, *Fulica atra*.

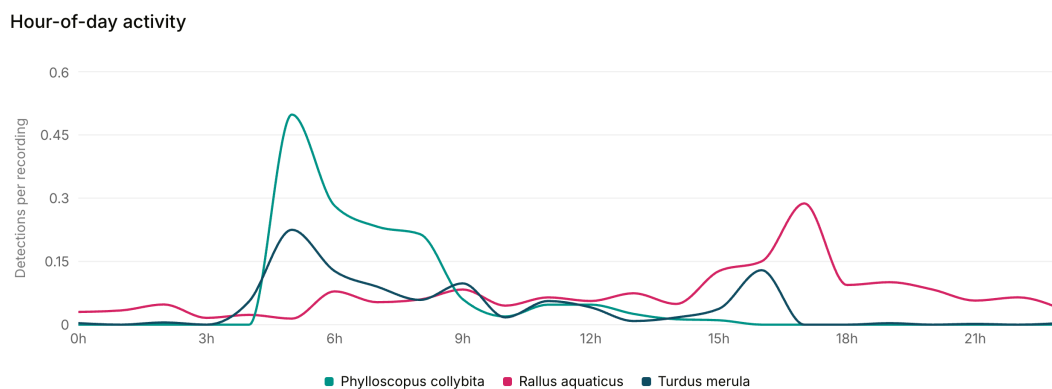


Figure 26: Daily activity by hour of day, comparing several species.

6.5 Testing and Code Quality

The system was developed under automated tests that acted as a regression safety net: as the pipeline and research dashboard were refactored and extended, the suite verified that existing behaviour continued to hold rather than breaking unnoticed. The Python pipeline has a pytest suite run with coverage, is type-checked under Pyright in strict mode, and is linted and formatted with Ruff. The TypeScript dashboard is type-checked, linted with ESLint and formatted with Prettier, with unit and integration tests run under Vitest – the integration tests exercising the database queries against a real database. A pre-commit hook runs the relevant checks on the changed files, so formatting, linting and type errors are caught before they enter the history.

Beyond guarding against regressions, one of these tests supplies acceptance evidence for a requirement. The corrupt-file handling of Chapter 7.2.1 has a batch read each recording's duration up front and skip an unreadable file on its own, recording it in the sidecar as unprocessed rather than aborting the day's batch. The batch is skipped in full only when every one of its files is unreadable. This is exactly the case NFR-3 calls for, and a dedicated test covers it.

6.6 Packaging and Handover

The submitted system [48] is hosted as a code repository on the OST GitLab, spanning all three components: the pipeline, the database and the research dashboard. The researcher clones it from there and pulls any later changes, and it is also where further development can be carried on by others. Predictions are deliberately committed to the repository so that fresh clones can set the dashboard up without needing to run their own inference on already processed data.

To process a new dataset, the researcher runs the pipeline stages in turn – predict, process and load – after which the research dashboard serves the results. Adjustments are made by editing the YAML configuration files (Figure 20) rather than the source, and moving a recorder to a new location needs no intervention: an incremental location code is automatically assigned (Chapter 6.1). The repository's README documents the usage instructions.

7 Validation

This chapter evaluates the system against the requirements of Chapter 2. It first analyses the acoustic monitoring performance – BirdNET’s bird classification precision and the bat detection consensus – then benchmarks the processing pipeline, and finally checks each FR and NFR against the evidence gathered.

7.1 Acoustic Monitoring Performance Analysis

This section assesses how well the integrated models recognise birds and bats on the reference sets created in Chapter 4. BirdNET is evaluated for bird species precision against the BirdNET assessment set, and the three bat adapters are compared against the bat consensus set; a separate check reviews how ecologically plausible the overall species lists are.

7.1.1 BirdNET Precision

Each prediction in the BirdNET assessment set has received one of the following decisions:

- **Confirmed:** Predicted species is correct.
- **Mismatch:** Prediction is wrong, but correct species is named.
- **Unknown:** Bird is present, but species cannot be identified reliably.
- **Rejected:** No bird vocalisation in this segment.

Species	Predicted	Confirmed	Mismatch	Rejected	Unknown	Precision
<i>Fulica atra</i>	25	21	0	0	4	100.0%
<i>Pica pica</i>	22	21	0	1	0	95.5%
<i>Erithacus rubecula</i>	16	16	0	0	0	100.0%
<i>Passer domesticus</i>	12	12	0	0	0	100.0%
<i>Corvus corone</i>	11	10	0	0	1	100.0%
<i>Cyanistes caeruleus</i>	10	10	0	0	0	100.0%
<i>Parus major</i>	9	9	0	0	0	100.0%
<i>Chroicocephalus ridibundus</i>	8	7	0	0	1	100.0%
<i>Fringilla coelebs</i>	8	7	0	0	1	100.0%
<i>Turdus merula</i>	7	6	1	0	0	85.7%
<i>Larus michahellis</i>	5	5	0	0	0	100.0%
<i>Streptopelia decaocto</i>	5	5	0	0	0	100.0%
<i>Sylvia atricapilla</i>	4	4	0	0	0	100.0%
<i>Anas platyrhynchos</i>	4	3	0	0	1	100.0%
<i>Ciconia ciconia</i>	3	3	0	0	0	100.0%
<i>Milvus milvus</i>	2	2	0	0	0	100.0%

Table 11: Per-species classification precision for the 16 focus species.

Confirmed predictions are assigned to TPs, mismatched and rejected ones are both assigned to FPs. Unknown ones are neither confirmed nor rejected and are therefore ignored; this assignment yields the species classification precision. The presence detection precision is scored differently: every prediction in which a bird is genuinely present – confirmed, mismatched or unknown alike – counts as a TP, leaving only the rejected predictions as FPs. Table 11 reports the results for the assessed focus species, ordered by number of predicted segments; the complete list including non-focus species is in Appendix F.2. Micro-averaged precisions for the assessed focus-species predictions are: 99.3% for presence detection and 98.6% for species classification.

These are consistently high precision figures. In comparison, the detection precision of the non-focus species in the same assessment recordings is 82.1% and the species precision just 53.2%. This can be explained by the fact that the included predictions have confidences all the way down to 0.2. The selection process favoured high confidence predictions of focus species – see curated median markers in Figure 27 – and non-focus species were included as a by-product. The inclusion of low confidences is deliberate in order to infer the ideal confidence threshold based on the assessment.

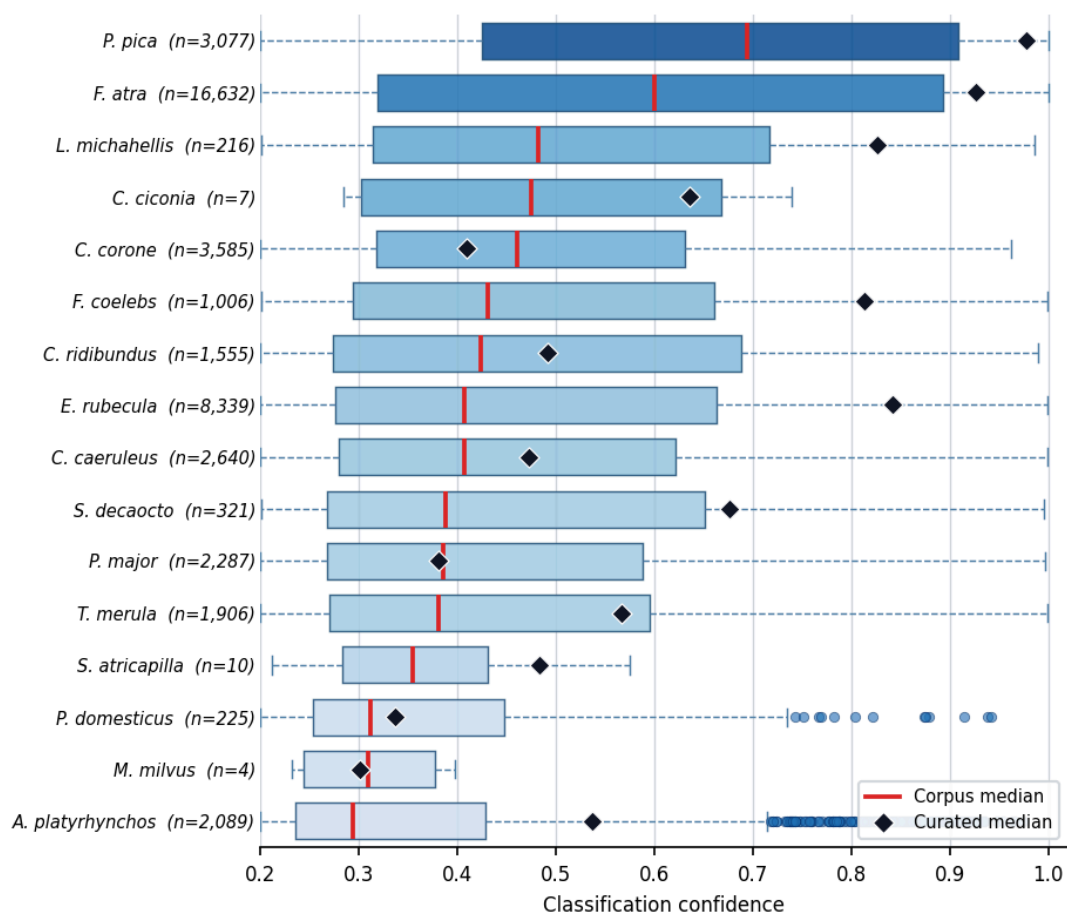


Figure 27: Distribution of BirdNET's prediction confidences for the focus species.

Figure 27 also shows that prediction confidences vary widely both across species and within; the n next to each species name is the number of predictions BirdNET made on all available data. Interesting candidates are *P. domesticus* and *A. platyrhynchos*, which have a median confidence of roughly 0.3 and a narrow box from the first to the third quartile, but their outliers reach to 0.8 and above. The other focus species with n>10 are all also spread out, but their inner 50% already span a bigger range so they do not contain outliers.

To turn this into a concrete recommendation, Figure 28 traces how raising the minimum confidence threshold prunes the assessment predictions: the share of confirmed predictions that is retained and the share of rejected predictions that is removed, as the threshold sweeps upward from 0.2. The two move together – a higher threshold discards more false positives but also more genuine detections – so the setting is a trade-off rather than an optimum. Three operating points illustrate the range. A minimum confidence of 0.42 balances the two, retaining roughly 70% of the confirmed predictions while removing roughly 70% of the rejected ones, and is recommended as a starting point. Where false positives are less tolerable, 0.6 removes more than 90% of the rejected predictions while still keeping about 60% of the confirmed ones. At the strict extreme, admitting only near-certain detections, 0.9 removes every rejected prediction on the assessment set but retains only about 30% of the confirmed ones.

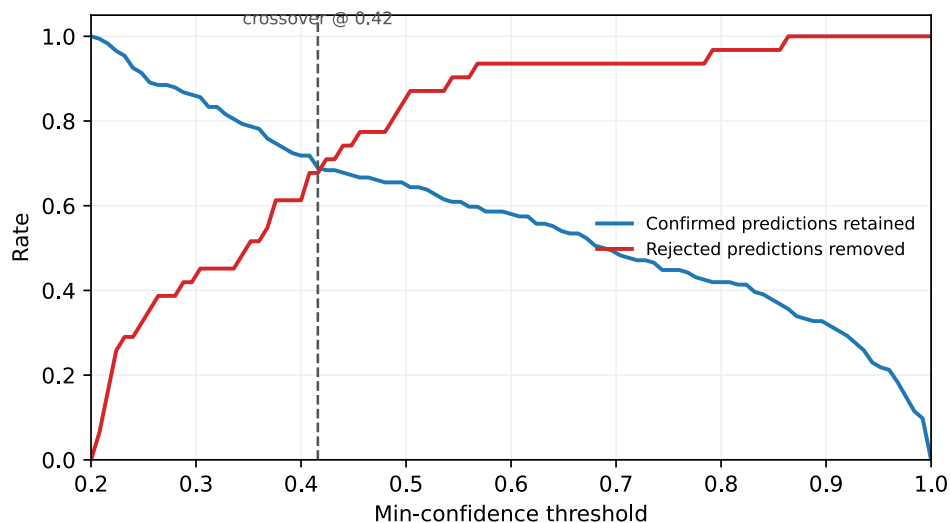


Figure 28: Retention of confirmed and removal of rejected BirdNET predictions on the BirdNET assessment set as the minimum confidence threshold varies.

7.1.2 Bird Species Plausibility Check

The precision assessment in Chapter 7.1.1 judges whether individual predictions are correct, not whether the overall set of species BirdNET reports for the campus is ecologically believable. To check the latter, the ILF [4] had two domain experts review the complete list of 173 species BirdNET detected across the dataset, ranked by number of detections and maximum detection confidence. They rated each species' plausibility for the site and season, the verdicts are grouped in Table 12 and listed per species in Appendix F.3, where the expert notes also flag conservation status with International Union for Conservation of Nature (IUCN) Red List categories.

Plausibility verdict	Species
Plausible	106
Possible	24
Unlikely	25
Does not occur or vocalise	9
Likely misidentification	9

Table 12: Expert plausibility verdicts on the 173 bird species BirdNET detected across the campus.

Three quarters of the species (130 of 173) are judged plausible or possible, and the implausible verdicts fall almost entirely in the long tail of rarely detected species. The frequently detected species at the top

of the list are, with a single exception, all rated plausible – they are the campus’s common residents and waterbirds such as the Eurasian Coot (*Fulica atra*), European Robin (*Erithacus rubecula*) and Carrion Crow (*Corvus corone*). The exception is the Hooded Crow (*Corvus cornix*, 1’519 detections), which the experts attribute more likely to the Carrion Crow, a well-known regional confusion. The remaining corrections and “does not occur” verdicts sit far lower in the ranking, consistent with these being false positives that a confidence threshold would largely remove.

The review also surfaced ecologically notable detections worth manual follow-up, among them red-listed species such as the Critically Endangered Eurasian Curlew (*Numenius arquata*), rated plausible. Whether or not each proves genuine, they show how the research dashboard can direct expert attention to species of interest.

This is a judgement on the species list rather than a verification of individual detections: being plausible does not confirm every detection of that species, and an implausible verdict hints at a likely false positive rather than a certain one. Read together with Chapter 7.1.1, it indicates that BirdNET’s common detections form an ecologically coherent picture of the campus avifauna, while its rare detections call for expert scrutiny.

7.1.3 Bat Detection Consensus

No expert was available to confirm bat calls (Chapter 1.4). In the absence of confirmed calls, the bat consensus set serves as the reference. The three adapters’ predictions are first aligned onto the shared 0.5625-second grid so they can be compared window by window. Each adapter is then scored against the consensus set per window. True negatives are not reported, since the consensus set is a presence reference (Chapter 4.2.3.1).

Because the reference is assembled from the adapters themselves, these metrics measure agreement among the adapters rather than verified correctness. Under the two-vote rule an adapter’s precision is the share of its segments that at least one other adapter also predicted, so the consensus is dominated by whichever adapters fire most densely. The result is a cross-adapter plausibility check rather than an absolute precision, and the 80% bat detection precision target is read in that light.

Figure 29 reports, per adapter, the TP, FP and FN counts against the consensus together with the resulting detection precision, recall and F1.

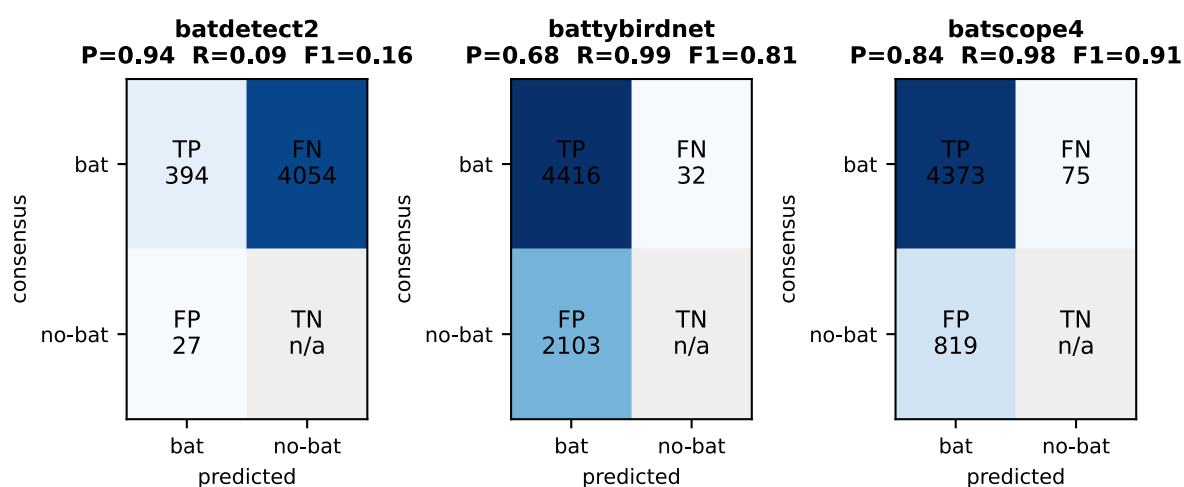


Figure 29: Per-adapter detection confusion against the bat consensus set.

BattyBirdNET and BatScope4 overlap heavily against the consensus: each contributes more than 4’000 TPs with almost no FNs, so the two agree on most of the detections that make up the consensus.

BatScope4's significantly lower FP of 819, compared to BattyBirdNET's 2'103, makes it the more precise instrument here. BatDetect2 is the most precise with a precision of 94%, however the recall is unusable at 9%.

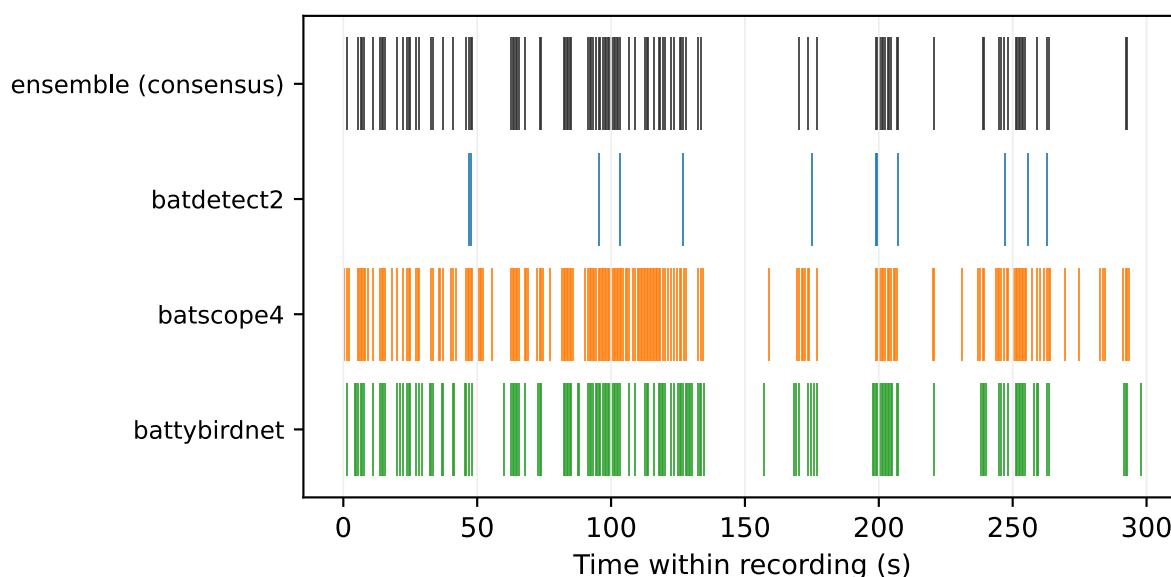


Figure 30: The bat consensus set against each adapter's segments for one representative recording (Location2, 2025-09-30 19:30).

The timeline in Figure 30 shows one representative recording of medium segment density, chosen so the comparison reflects a typical rather than an extreme case. The consensus set runs along the top row, with each adapter's segments shown beneath it. It exposes how unevenly the adapters contribute: BatScope4 and BattyBirdNET fire densely across the recording and between them effectively define the consensus band, whereas BatDetect2 contributes only a handful of segments – which nonetheless land on the band, few but well placed.

These characteristics suggest practical minimum-confidence settings depending on the researcher's goal. When only few but rather certain detections are wanted, BatDetect2 is the natural choice: its precision is already strong at a minimum confidence of 0.7, and the threshold can be lowered to surface more candidate calls – at the cost of admitting noise that then warrants closer inspection. When the goal is instead not to miss bats, the high recall of BattyBirdNET and BatScope4 makes them better suited, with BatScope4 the more precise of the two.

7.1.4 Bat Focus-Species Coverage

The bat species expected on campus (Chapter 3.1.3) can be set against what the three species-level adapters actually predicted over the dataset – each adapter's native label for these species is given in Table 10. Table 13 classifies each adapter's output per species as *Exact* (the adapter output the species), *Genus only* (it output one or more same-genus relatives, listed, but not the species itself) or *Absent* (neither); for an *Exact* match it also reports the share of that adapter's total predicted bat segments the species represents, with the raw segment count in brackets. These are model outputs rather than confirmed presence (Chapter 1.4).

Species	BatDetect2	BattyBirdNET	BatScope4
<i>P. pipistrellus</i>	Exact – 5.2% (985)	Exact – 2.0% (1'526)	Exact – 7.7% (973)
<i>P. kuhlii</i>	Genus only (<i>P. nathusii</i> , <i>P. pipistrellus</i> , <i>P. pygmaeus</i>)	Exact – 6.8% (5'204)	Exact – 0.4% (48)
<i>P. nathusii</i>	Exact – 88.4% (16'776)	Exact – 74.6% (56'939)	Exact – 53.9% (6'792)
<i>M. mystacinus</i>	Genus only (<i>M. nattereri</i> , <i>M. daubentonii</i>)	Exact – 0.3% (246)	Genus only (<i>M. capaccinii</i> , <i>M. myotis</i> , <i>M. nattereri</i>)
<i>M. daubentonii</i>	Exact – 0.1% (15)	Exact – <0.1% (4)	Genus only (<i>M. capaccinii</i> , <i>M. myotis</i> , <i>M. nattereri</i>)
<i>P. pygmaeus</i>	Exact – 0.5% (94)	Exact – 5.2% (3'966)	Genus only (<i>P. nathusii</i> , <i>P. pipistrellus</i> , <i>P. kuhlii</i>)

Table 13: Coverage of the campus focus bat species by the three species-level adapters.

P. pipistrellus and *P. nathusii* are output exactly by every adapter. *P. nathusii* is by a wide margin the most predicted species overall: it alone makes up 88.4% of BatDetect2's segments and 74.6% of BattyBirdNET's. *P. kuhlii* is never produced by BatDetect2. Its 17-species United Kingdom label set omits the species, leaving another *Pipistrellus* as the closest match. The two *Myotis* species barely register. *M. mystacinus* is output exactly only by BattyBirdNET, and only at 0.3%. *M. daubentonii* appears only in BatDetect2 and BattyBirdNET, at noise level (15 and 4 segments). That matches how hard *Myotis* calls are to separate by species (Chapter 3.1.5). BatScope4 was run on fewer recordings than the other two adapters. Its narrower coverage and lower counts are therefore partly a sampling effect, not a property of the model. The per-adapter shares normalise for this.

7.2 Processing Benchmark

To judge whether the pipeline is fast enough to be practical (NFR-1) and scales to the full dataset (NFR-5), both processing stages were timed. Bulk inference across the whole dataset was run on the DGX-2, but to measure usability by the ILF, the benchmark was taken on the reference workstation in Table 14 during an in-person session on 9 June 2026. In that session the system was set up from scratch and run together with Dr. Paul Haverkamp [4] on campus, executing the full workflow in sequence – predict, process and load – over the benchmark recordings and a few newly collected ones, after which the research dashboard was started and the resulting detections browsed interactively.

CPU	Intel Core i7-14700 (2.10 GHz)
Memory	32 GB
GPU	NVIDIA GeForce RTX 4060 (8 GB)
GPU driver / CUDA	581.57 / 13.0
Operating system	Windows 11 Enterprise 24H2 (64-bit, x64-based)

Table 14: Reference workstation used for the processing benchmark.

The predict stage was measured on every device recording on 10 October 2025: three day-device batches totalling 180 recordings and 15 hours of audio. Those recordings were captured at 384 kHz – the highest sample rate the AudioMoths used (Figure 9) and therefore the most demanding to process – with a file size of about 230 MB, so the predict benchmark processed roughly 41 GB of audio. The far

cheaper process stage was measured across the entire dataset. Table 15 reports, per adapter, the wall-clock time per 5-minute recording and the real-time factor – how many times faster than the recorded duration the stage runs.

Adapter	Time / recording	Real-time factor
BirdNET	4.0 s	74×
BatDetect2	3.5 s	85×
BattyBirdNET	3.5 s	86×

Table 15: Predict-stage throughput per adapter on the reference workstation.

Prediction dominates the runtime yet still runs far faster than real time: 3.5 to 4.0 seconds per 5-minute recording, or 74 to 86 times the recorded duration. The process stage – canonicalisation and filtering – is negligible beside it, handling each recording in well under a millisecond. The total processing time is therefore far below the total audio duration, satisfying NFR-1 with a wide margin. A snapshot taken during inference showed about 5.6 GB of the 8 GB of GPU memory in use, so the workload fits comfortably on a single mid-range GPU.

For scalability (NFR-5), the full 1.77 TB dataset of 14'333 recordings has been processed end to end through directory-based batches of one day and one device, demonstrating that the pipeline runs on a dataset of that size. Bat detectors automatically skip recordings below 240 kHz that would not capture bats appropriately. No manual per-file handling is involved.

7.2.1 Processing Robustness

Batch processing is built to tolerate individual file failures (NFR-3). Each recording's duration is read up front, so a truncated or zero-byte WAV is skipped on its own and recorded in the sidecar as unprocessed rather than aborting the whole day-device batch; the batch is skipped in full only when every one of its files is unreadable. An automated test exercises this path by injecting a corrupt recording and confirming the batch completes with the faulty file logged and excluded. It has also proven to work on the thesis data, which contained such corrupt recordings.

7.3 Validation of Requirements

This section checks the requirements of Chapter 2 against the evidence gathered above.

7.3.1 Validation of FRs

All functional use cases of Figure 1 are realised by the system, as summarised in Table 16. The processing workflow – dataset selection, presence detection, species classification and persistence – runs end to end through the pipeline, and the research dashboard provides per-file inspection and aggregated statistics such as presence over time.

7.3.2 Validation of NFRs

Table 17 summarises the NFRs: four are met and NFR-2 is partially met.

Requirement	Result	Reason
NFR-1 Processing Time	Met	Processing runs 74–86 times faster than real time, well under the audio duration (Table 15).
NFR-2 Monitoring Performance	Partially met	Precision targets are met for both birds and bats on respective reference sets (Chapter 7.1.1, Chapter 7.1.3). Recall, F1 and a confusion matrix for birds need a recall-capable set the project lacks (Chapter 1.4).
NFR-3 Robustness	Met	Unreadable files are skipped so the batch continues, covered by a test (Chapter 7.2.1), and the user gets informed “ <i>Skipped N unreadable file(s)</i> ”.
NFR-4 Operational Usability	Met	The workflow runs via documented commands and configuration, and new locations get codes without code changes (Chapter 6.1); it was also set up and run end to end on the reference workstation together with the ILF in the 9 June session (Chapter 7.2).
NFR-5 Dataset Scalability	Met	The full 1.77 TB dataset was processed through directory-based batches.

Table 17: Validation of the non-functional requirements.

NFR-2 is only partially met. The precision targets are met: bird species precision is 98.6% on the focus species and 84.9% across all species – far more than five species exceed 60% (Chapter 7.1.1). Bat detection precision clears 80% for both BatDetect2 and BatScope4 – only BattyBirdNET falls short (Chapter 7.1.3). What the requirement additionally asks for cannot be produced, however: without a recall-capable, expert-labelled set (Chapter 1.4), per-species recall and F1 are not reportable and a bird species confusion matrix is not meaningful. The bat figures are also measured against the bat consensus set rather than against confirmed calls.

Use case	Result	Reason
Execute Processing Workflow	Met	The predict, process and load stages run end to end and were benchmarked (Chapter 7.2).
Select Dataset	Met	The input directory is traversed recursively.
Detect Bird/Bat Presence	Met	Per-segment presence is produced and validated for birds (Chapter 7.1.1) and bats (Chapter 7.1.3).
Classify Bird Species	Met	BirdNET assigns species labels, assessed per species (Chapter 7.1.1).
Persist Results	Met	Results are written to the output directory and loaded into the database.
Inspect Outputs	Met	The dashboard gives per-file inspection of predictions, confidences and timestamps (Chapter 6.4).
Export Statistics	Met	Aggregated statistics such as presence over time are surfaced in the dashboard (Chapter 6.4).

Table 16: Validation of the FRs.

8 Discussion of Results and Outlook

First, this chapter draws the results together – what the finished tool reveals and the acoustic picture of the campus that emerges from the data – then sets out the limitations that bound those findings, and closes with the further work they motivate.

8.1 Results

The resulting system lets the researcher orient themselves on the bird species present on the OST Rapperswil campus: which species occur, the locations they visit most frequently, the time of day at which they are most active, and which recordings contain their calls – so those recordings can be listened to and inspected closely. Figure 31 shows the overview page, which brings these views together.

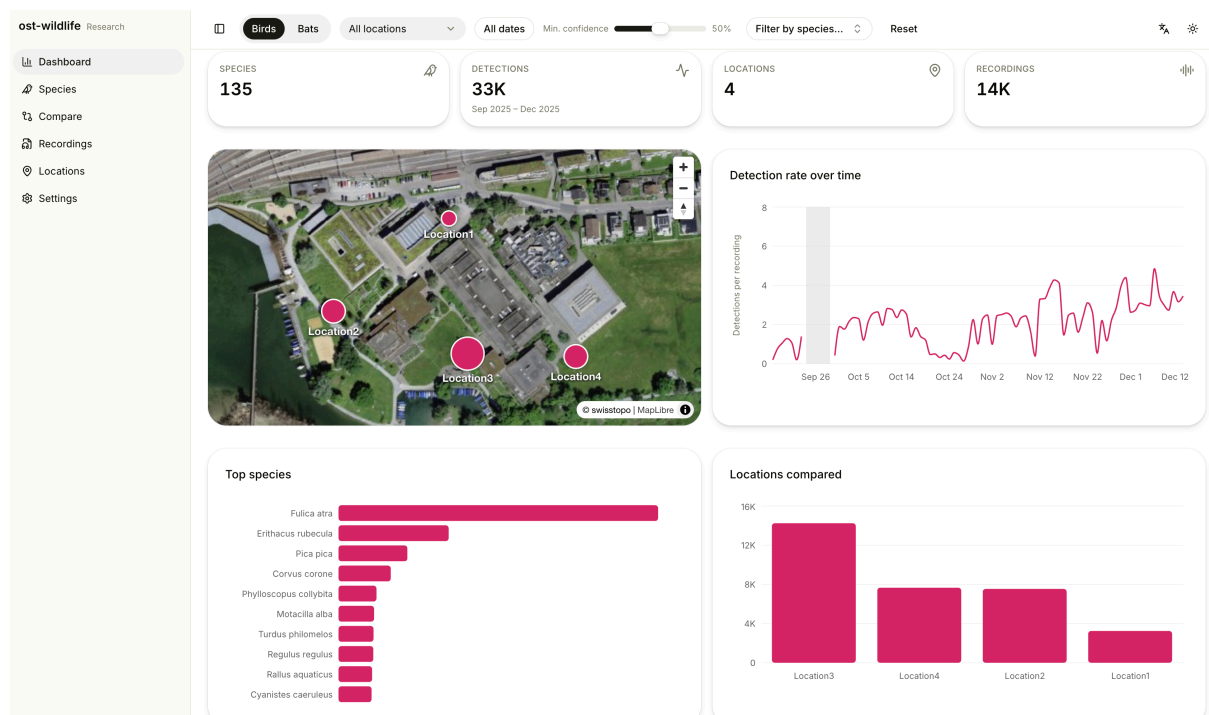


Figure 31: Overview page of the research dashboard.

The bird species predicted by the chosen model, BirdNET, proved highly precise on the BirdNET assessment set, particularly once the adjustable minimum confidence threshold is raised. A reasonable starting point is a minimum confidence of 0.42, which on the BirdNET assessment set retains about 70% of the confirmed predictions while removing about 70% of the rejected ones. It can be raised to 0.6 when false positives are less tolerable, or to 0.9 to admit only near-certain detections, trading recall for precision accordingly; Figure 28 quantifies the full trade-off. The higher the threshold, the fewer false positives are to be expected. Predictions start at 0.2, giving the researcher full control in the dashboard.

Across the dataset, BirdNET predicted 106 species that experts consider plausible and a further 24 they rated possible (Table 12). However, most of them need expert confirmation before any claim can be made. The research dashboard lets the ILF easily identify the audio segments of the species they want to look at more closely.

Confirming bat calls proved far more difficult. Their calls are inaudible to human hearing and the ultrasonic bands they occupy are crowded with noise, which makes individual calls hard to visualise and verify. With no expert available to confirm them, this thesis instead combined three existing bat detectors into an ensemble and derived a bat consensus set by majority vote. For the same reason, the consensus set – and the validation built on it – covers presence detection only, not species identification (Chapter 7.1.3).

The three adapters behave very differently against this consensus. BatDetect2, which predicts a call's frequency band in addition to its temporal span, detects far fewer calls than the other two; the few it does are mostly matched by them, giving it the highest precision against the consensus at 94% – though its recall is just 9%. A likely factor is that the pretrained BatDetect2 model used here was trained on a UK dataset of 17 species [2], so it generalises less well to the local fauna than the other two. BattyBirdNET's classifier is trained on regional data from Bavaria [34] and BatScope4 on European species by the Swiss institute WSL [17].

This gap in detection volume holds across the whole deployment (Figure 32). Both panels share the same time axis over the deployment; the vertical axis, however, runs from 0 to 80 detections per recording for BattyBirdNET (left) and only from 0 to 8 for BatDetect2 (right). At a minimum confidence of 0.6, BattyBirdNET reports around ten times as many detections per recording as BatDetect2 – up to roughly 60 against fewer than six – yet the two distributions over time are near-identical in shape. The two detectors therefore agree closely on which days bats were active, even as they disagree on how many calls each of those days holds.

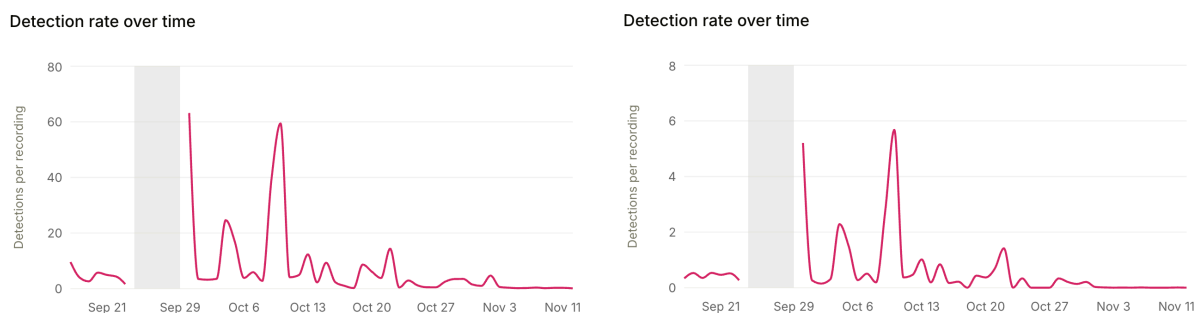


Figure 32: Detection rate per recording across the deployment for BattyBirdNET and BatDetect2.

BattyBirdNET and BatScope4 detect very similar segments and between them define the consensus, with almost no false negatives and recall approaching 100%. BatScope4 is the more precise of the two – its precision clears the 80% target that BattyBirdNET falls short of (Figure 29). BatScope4 is, however, only available as a GUI and cannot be driven automatically through the pipeline, so of the two high-recall detectors only BattyBirdNET is usable in the research dashboard. The dashboard therefore lets the researcher switch between BatDetect2 and BattyBirdNET; although validation covers presence only, both adapters still output species labels, which are displayed and explorable there.

The recording devices, AudioMoth, have proven to produce satisfactory audio recordings that capture both birds and bats. They can be relocated at will: the system automatically assigns them new location codes and displays them accordingly on the map. The research dashboard is available in both English and German, so members of the ILF can work in their preferred language.

As a worked example, *Fulica atra* – the Eurasian Coot and the most-detected bird on campus – shows how these views combine into an ecological reading. It is a water bird, and its detections concentrate accordingly: the largest marker on the map sits at Location3, the recording site by the water (Figure 33).

Fulica atra

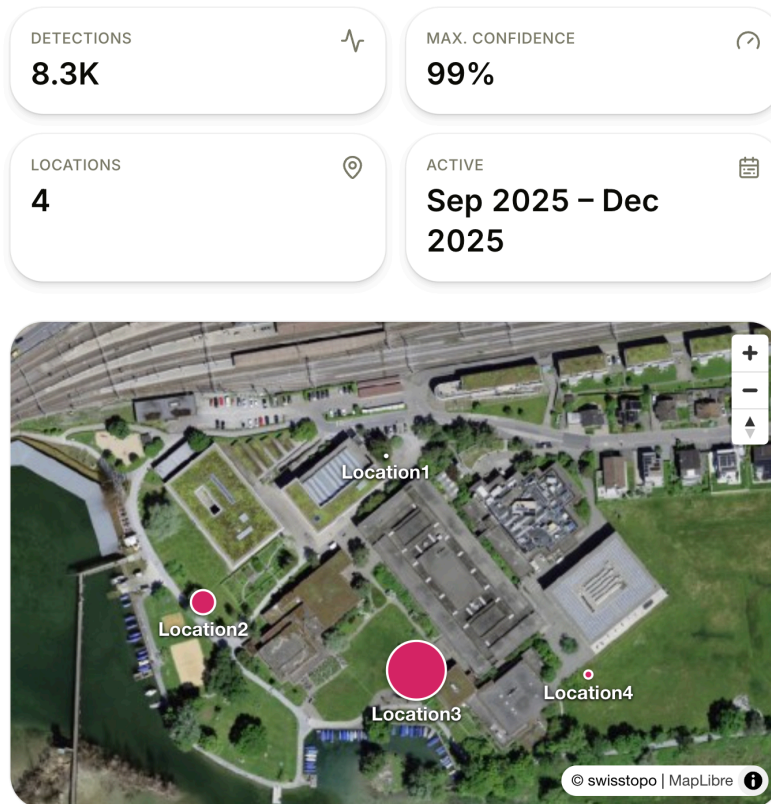


Figure 33: Detection summary and location map for *Fulica atra*, the most-detected species.

8.2 Limitations

The most consequential limitation is the absence of a ground-truth set for the local recordings. No expert-confirmed, species-level reference could be created for the campus data. The available expert time stretched only to the small BirdNET assessment set, not to a complete annotation of the dataset; that assessment was moreover carried out by a single annotator, so its figures are not backed by inter-annotator agreement. The validation could therefore measure precision – how many of the model’s predictions hold up – but not recall or F1, and a bird-species confusion matrix is not meaningful without such a set, which is why NFR-2 is only partially met (Table 17).

Bat detection has no expert reference at all. The species the bat adapters output are surfaced in the research dashboard as a starting point for inspection, but none have been confirmed by an expert – they are model output to look at, not validated identifications. The presence validation itself rests on a consensus assembled from the adapters (Chapter 7.1.3), so it measures agreement between them rather than verified correctness.

Prediction throughput was not a priority and so was never refined. The pipeline meets its processing-time requirement comfortably (Table 15), but it was not tuned for efficient use of the hardware. Doing so was out of reach within the project’s time and is taken up as an improvement below (Chapter 8.3.3).

Finally, the research dashboard is built for a single researcher at the ILF working locally. It provides no authentication and no public deployment. The architecture was designed with that extension in mind, however, so moving to a hosted, authenticated, multi-user setup – the subject of the improvements that follow – would build on the existing design rather than require reworking it.

8.3 Possible Further Improvements

With the core system built and validated (Chapter 7), the directions below are extensions rather than fixes. They fall into two groups. The first three sharpen the recognition and the evidence behind it – adapting the models to the campus, growing a set of confirmed detections, and making fuller use of the hardware. The last three widen the system’s reach, carrying it from an internal research tool toward a hosted, public-facing platform and, eventually, live monitoring, each building on the one before. Every entry is weighed by its expected effort and payoff.

8.3.1 Local Fine-Tuning of the Models

Every model integrated here is used as released, trained on audio from other regions and recording setups: BirdNET globally, BatDetect2 on a UK dataset of 17 species [2], BattyBirdNET on Bavarian data [34] and BatScope4 on European species [17]. The validation exposed where this regional gap hurts – most visibly BatDetect2’s 9% recall and its inability to name *P. kuhlii* at all. Fine-tuning any of these models on labelled campus audio would adapt them to the local species and to the AudioMoth microphones, closing that gap; for BirdNET this can be as light as training a thin classifier on its embeddings rather than retraining the backbone [29]. The effort is high – it depends on labelled in-domain data, which the project does not have (Chapter 1.4) – but the payoff is models fitted to the site rather than borrowed from elsewhere.

8.3.2 In-Dashboard Confirmation and Playback

The project lacks a recall-capable set of confirmed detections (Chapter 1.4), and the natural place to build one is the research dashboard itself. While inspecting predictions, the researcher could mark each as confirmed or rejected, so a verified set grows as a by-product of normal review rather than as a separate annotation campaign. Pairing this with in-dashboard audio playback – letting the researcher listen to a segment and decide on the spot – would make confirmation quick and keep it in one place. Such a set could be grown more efficiently still by surfacing the most informative segments first, the idea behind active learning [49].

This fits birds far better than bats. Bird calls are audible, so the researcher can confirm them simply by listening. Bats are harder on two counts: no one at the ILF is able to confirm their calls in the first place, and because the calls are ultrasonic, confirmation rests on reading spectrograms rather than listening – a slower and more specialised judgement. The effort is moderate and the payoff compounding: a growing confirmed set would unlock the per-species recall and F1 figures the validation could not report, and would supply exactly the labelled data the fine-tuning above needs.

8.3.3 Throughput Optimisation

The predict stage is already running 74 to 86 times faster than real time (Table 15), so throughput was never a blocker. It was also never optimised: limited time, and limited access to the reference workstation the ILF will ultimately run on, left no room to tune it. There is nevertheless clear headroom for larger or multi-site datasets. The stage currently processes one day-and-device batch at a time – reading a batch, running inference, then writing results before the next begins – so disk reads and inference never overlap; loading the next batch while the current one is still being processed would hide much of that I/O. Two configuration levers are also left untuned: the inference batch size and the number of workers. The effort is moderate and the payoff is shorter wall-clock time on very large runs and on the more modest hardware the ILF operates.

8.3.4 Deployment and Hosting

Today the system runs locally. It can be shared in three ways: by setting it up on another machine, sharing the IP address so colleagues on the same network can open the research dashboard, or by gathering in person around a single device. This is sufficient for the time being, however it would be convenient to have it hosted centrally. That would make the dashboard reachable over the web without any local setup and is the foundation the two extensions below build on. The effort is moderate and the payoff immediate: a system that is simply available rather than handed around.

8.3.5 Public Outreach Site

The research dashboard is at present a single inspection tool. A natural next step is to split it into two spaces. The research area would sit behind authentication and keep exactly the analysis functions it has today, so the researcher continues to explore detections as before. Alongside it, a curation workflow would let the researcher promote individual species and calls for publication – renaming a species to its German common name, adding a short description and further facts, and attaching a representative image or audio snippet. That curated selection would feed a public area of the site that anyone can visit to browse the species recognised on campus.

This serves the ILF's outreach goal of raising biodiversity awareness beyond the photo-trap screens already in use, and, as part of the RealTransform initiative [3], it would make that outreach measurable: analytics on site visits and navigation behaviour give a concrete gauge of whether a campaign reached people. QR codes on posters or the campus screens could link directly to the curated site. The effort is substantial – it amounts to a second product layered on the research tool – but the payoff is the public-engagement dimension that partly motivated the project.

8.3.6 Live Monitoring

Real-time processing was deliberately out of scope (Chapter 1.4); the ILF collects recordings periodically and processes them in batches. A forward-looking extension would make the recordings live-capable – giving the AudioMoths network connectivity so they stream audio continuously to a server – and feed the results straight into the dashboard.

On the public site this would turn passive browsing into a live experience. A visitor opening the site on their phone could see which birds or bats were recognised at their location in the last minutes or hours. Drawing on past data, the site could extend this into a short forecast – for example that a given species is likely to call around the approaching sunset – paired with a sample call so the visitor knows what to listen for. This is the most ambitious extension and depends on both the hosting and the public site above; its payoff is an engagement loop that connects visitors directly to the wildlife around them.

A Glossary

A.1 Audio & Bioacoustics

MEMS – **micro-electro-mechanical systems**: A tiny mechanical microphone etched onto a silicon chip, offering low cost and small size but reduced sensitivity at high ultrasonic frequencies.

PAM – **passive acoustic monitoring**: Recording an environment with autonomous sensors and analysing the audio offline to study species presence, activity, and ecological dynamics.

sonotype: A recurring bat-call shape that is more consistent across regions than species identity [14].

spectrogram: A time-frequency image of an audio signal, computed with a short-time Fourier transform.

STFT – **short-time Fourier transform**: Time-frequency representation computed by applying the Fourier transform to short, overlapping windows of a signal.

A.2 Evaluation Metrics

F1 – **F1 score**: Harmonic mean of precision and recall.

FN – **false negative**: A positive instance the classifier failed to detect.

FP – **false positive**: A negative instance the classifier incorrectly flagged as positive.

TN – **true negative**: A negative instance the classifier correctly identified as negative.

TP – **true positive**: A positive instance the classifier correctly identified as positive.

A.3 Machine Learning

CNN – **convolutional neural network**: Neural network architecture that uses convolutional filters to analyse images.

embedding: A fixed-length feature vector a model produces from its input.

fine-tuning: Continuing the training of a pre-trained model on in-domain data to adapt it.

KKNN – **weighted k-nearest neighbours**: Distance-weighted variant of k-nearest-neighbours classification, where closer training examples count more towards the vote.

SVM – **support vector machine**: Supervised classifier that separates classes with a maximum-margin hyperplane in a feature space.

transfer learning: Reusing a model pre-trained on a large dataset for a new task rather than training from scratch.

A.4 Organisations

BAFU – **Federal Office for the Environment**: Swiss federal environmental authority; publisher of the national Red List of bats [11].

BTO – **British Trust for Ornithology**: UK non-profit ornithological research organisation; cited in this thesis for its guidance on AudioMoth and bat-detector recording limitations [39].

CCO – West Coordination Centre for Bat Conservation: Swiss bat-conservation coordination centre for the western regions; co-author of the national Red List of bats [11].

Cornell Lab – Cornell Lab of Ornithology: Ornithological research institute at Cornell University; hosts the Macaulay Library and developed the BirdNET and Merlin bird-identification tools.

ICAI – Interdisciplinary Center for Artificial Intelligence: Research centre at OST focused on applied artificial intelligence across disciplines.

ILF – Institute for Landscape and Open Space: Research institute at OST and the industrial partner for this thesis; manages the campus AudioMoth recorders and provided the recordings analysed here. Contact: Dr. Paul Haverkamp.

IUCN – International Union for Conservation of Nature: International authority on the conservation status of species; maintains the IUCN Red List.

KOF – East Coordination Centre for Bat Conservation: Swiss bat-conservation coordination centre for the eastern regions; co-author of the national Red List of bats [11].

OST – Eastern Switzerland University of Applied Sciences: The university where this thesis is carried out (German: Ostschweizer Fachhochschule, hence OST); its Rapperswil campus hosts ILF and is where the analysed recordings originate.

SZKF – Swiss Centre for Faunal Cartography: Swiss centre mapping species distributions; co-author of the national Red List of bats [11].

WSL – Swiss Federal Institute for Forest, Snow and Landscape Research: Eidgenössische Forschungsanstalt für Wald, Schnee und Landschaft (Birmensdorf ZH); produced the BatScope classifier [17] and was approached for advice on bat detection during this thesis.

A.5 Software & Computing

Apptainer: A container runtime designed for shared HPC systems.

CLI – command-line interface: Text-based interface for invoking a program through typed commands and arguments.

DGX-2: An NVIDIA DGX-2, a multi-GPU deep-learning server.

FR – functional requirement: Defines what the system must do, specifying concrete features, behaviours, and interactions to fulfil user and business needs.

GUI – graphical user interface: Visual, windowed interface for operating a program through on-screen controls.

HPC – high-performance computing: Shared, multi-user compute infrastructure for large workloads; such environments commonly provide unprivileged container runtimes such as Apptainer for supplying workloads.

NFR – non-functional requirement: Defines how the system should perform or behave.

ORM – object-relational mapping: Technique that maps relational database tables to application-level objects, allowing queries to be expressed in the host language rather than raw SQL.

SSH – secure shell: Encrypted network protocol for logging into and running commands on a remote machine.

SSR – server-side rendering: Web rendering strategy where page HTML is generated on the server per request rather than in the browser.

A.6 Thesis Terms

adapter: A wrapper that integrates a model behind the uniform Predictor interface; the interchangeable unit of the processing pipeline.

bat consensus set: The bat validation set used in place of expert annotation, formed by an agreement vote across the BatDetect2, BattyBirdNET and BatScope4 adapters.

BirdNET assessment set: The annotated set of BirdNET predictions over the 30 selected assessment recordings, used to measure BirdNET's precision.

leaf folder: The innermost date folder in the AudioMoth folder convention; the pipeline's batch unit of one day and one device.

research dashboard: The web application container through which the researcher explores the detections made on the processed recordings.

segment: A predicted time interval within a recording, delimited by a start and end time.

B List of Tables

Table 1	NFR-1 Processing Time	8
Table 2	NFR-2 Monitoring Performance	9
Table 3	NFR-3 Robustness	9
Table 4	NFR-4 Operational Usability	10
Table 5	NFR-5 Dataset Scalability	10
Table 6	Overview of the bird and bat classifiers considered for this project.	15
Table 7	BirdNET predictions of the 16 focus species in the assessment recordings.	25
Table 8	The 30 recordings of the bat consensus set with the metrics behind the selection.	27
Table 9	Native output columns per adapter mapped to the canonical prediction schema.	37
Table 10	Native output label per species-level adapter for each focus bat species.	42
Table 11	Per-species classification precision for the 16 focus species.	49
Table 12	Expert plausibility verdicts on the 173 bird species BirdNET detected across the campus.	50
Table 13	Coverage of the campus focus bat species by the three species-level adapters.	53
Table 14	Reference workstation used for the processing benchmark.	53
Table 15	Predict-stage throughput per adapter on the reference workstation.	54
Table 16	Validation of the FRs.	54
Table 17	Validation of the non-functional requirements.	55

C List of Figures

Figure 1	Use case diagram of the audio-based classification system	7
Figure 2	Spectrogram of an audible bird call (<i>Fulica atra</i>) from a recording collected for this thesis.	13
Figure 3	Spectrogram of a non-biological noise event – likely a train stopping – from a recording collected for this thesis.	13
Figure 4	Spectrogram of example ultrasonic bat echolocation calls from a recording collected for this thesis.	14
Figure 5	AudioMoth device deployed on tree. Provided by ILF [4].	20
Figure 6	OST Rapperswil campus with microphone locations annotated. Screenshot taken on Google Maps [40].	21
Figure 7	Date and time coverage of the recordings.	22
Figure 8	Number of recordings per recording location and date.	22
Figure 9	Number of recordings per sample rate and date.	23
Figure 10	File sizes per sample rate and outliers.	24
Figure 11	The purpose-built annotation interface used to annotate the candidate clips.	26
Figure 12	Confidence distribution and quantity of bat adapter predictions on all recordings.	28
Figure 13	C4 Level 1 – system context: ost-wildlife in relation to its users and external systems. . .	30
Figure 14	C4 Level 2 – containers: the three containers and their interactions.	31
Figure 15	C4 Level 3 – pipeline components: CLI commands, core orchestration, adapters, and storage layers.	32
Figure 16	C4 Level 3 – research dashboard components: route groups, server components, and database access layer.	33
Figure 17	Data flow from the raw recordings to the research dashboard.	34
Figure 18	Class diagram of the Predictor interface with implementations per adapter.	35
Figure 19	Flow chart of batch predictions.	36
Figure 20	The YAML configuration architecture.	38
Figure 21	Database schema.	38
Figure 22	The sticky global filter bar shared by every dashboard view.	45
Figure 23	The recordings browser with free-text search and sortable columns.	45
Figure 24	The location map of the recording sites.	46
Figure 25	Detection rate over time for the most-detected bird, <i>Fulica atra</i>	46
Figure 26	Daily activity by hour of day, comparing several species.	46
Figure 27	Distribution of BirdNET’s prediction confidences for the focus species.	49
Figure 28	Retention of confirmed and removal of rejected BirdNET predictions on the BirdNET assessment set as the minimum confidence threshold varies.	50
Figure 29	Per-adapter detection confusion against the bat consensus set.	51
Figure 30	The bat consensus set against each adapter’s segments for one representative recording (Location2, 2025-09-30 19:30).	52
Figure 31	Overview page of the research dashboard.	56
Figure 32	Detection rate per recording across the deployment for BattyBirdNET and BatDetect2. .	57
Figure 33	Detection summary and location map for <i>Fulica atra</i> , the most-detected species.	58

D List of Tools

Task	Tools
Research	Google, NotebookLM, ChatGPT, Claude
Generation of ideas	ChatGPT, Claude
Text creation, optimisation and grammar checks	ChatGPT, Claude
Report typesetting	Typst
Diagram creation	PlantUML, matplotlib
Code editing and assistance	Visual Studio Code, Claude Code
Pipeline implementation	Python, uv, Typer, pandas, pyarrow, SciPy, Pydantic, SoundFile, soxr, SQLAlchemy
Detection models	BirdNET, BatDetect2, BattyBirdNET, BatScope4
Machine-learning frameworks	TensorFlow, PyTorch, CUDA
Audio inspection and annotation	Audacity
Web dashboard	Next.js, React, TypeScript, Tailwind CSS, shadcn/ui, Recharts, MapLibre GL, Drizzle ORM, pnpm
Database	PostgreSQL, Docker, Drizzle Kit
Testing and code quality	pytest, Ruff, Pyright, Vitest, ESLint, Prettier, Storybook
Version control and hosting	Git, GitHub, OST GitLab, Vercel, lefthook

E Bibliography

- [1] S. Kahl, C. M. Wood, M. Eibl, and H. Klinck, “BirdNET: A deep learning solution for avian diversity monitoring,” *Ecological Informatics*, vol. 61, 2021, doi: [10.1016/j.ecoinf.2021.101236](https://doi.org/10.1016/j.ecoinf.2021.101236).
- [2] O. Mac Aodha *et al.*, “Towards a General Approach for Bat Echolocation Detection and Classification,” *bioRxiv*, 2022, doi: [10.1101/2022.12.14.520490](https://doi.org/10.1101/2022.12.14.520490).
- [3] University of Zurich Sustainability Hub, “Projekt RealTransform: Realexperimente als Transformationsräume für nachhaltige Entwicklung in Hochschule und Gesellschaft.” Accessed: May 17, 2026. [Online]. Available: <https://www.sustainability.uzh.ch/de/forschung-lehre/forschung/realtransform.html>
- [4] P. Haverkamp, “Personal communication, ILF OST.” 2026.
- [5] C. H. Sekercioglu, D. G. Wenny, and C. J. Whelan, Eds., *Why Birds Matter: Avian Ecological Function and Ecosystem Services*. University of Chicago Press, 2016. doi: [10.7208/chicago/9780226382777.001.0001](https://doi.org/10.7208/chicago/9780226382777.001.0001).
- [6] S. R. P.-J. Ross *et al.*, “Passive acoustic monitoring provides a fresh perspective on fundamental ecological questions,” *Functional Ecology*, vol. 37, 2023, doi: [10.1111/1365-2435.14275](https://doi.org/10.1111/1365-2435.14275).
- [7] K.-H. Frommolt, “Anwendungsfelder des akustischen Monitorings von Arten,” *Natur und Landschaft*, vol. 98, 2023, doi: [10.19217/NuL2023-06-03](https://doi.org/10.19217/NuL2023-06-03).
- [8] P. Knaus, S. Antoniazza, V. Keller, T. Sattler, H. Schmid, and N. Strebel, “Rote Liste Brutvögel. Gefährdete Arten der Schweiz, Stand 2021,” Bundesamt für Umwelt (BAFU), Bern & Schweizerische Vogelwarte, Sempach. Umwelt-Vollzug Nr. 2124, 2021. Accessed: May 25, 2026. [Online]. Available: <https://www.vogelwarte.ch/wp-content/uploads/2023/11/Knaus-et-al-2021-Rote-Liste-der-Brutvoegel.pdf>
- [9] P. Knaus *et al.*, *Swiss Breeding Bird Atlas 2013–2016: Distribution and Population Trends of Birds in Switzerland and Liechtenstein*. Swiss Ornithological Institute, 2018. Accessed: May 25, 2026. [Online]. Available: https://www.vogelwarte.ch/wp-content/uploads/2025/08/Swiss-Atlas-2013-2016_E_low.pdf
- [10] T. Sattler, P. Knaus, H. Schmid, and B. Volet, “The State of Birds in Switzerland: Report 2017,” Swiss Ornithological Institute, Sempach, 2017. Accessed: May 25, 2026. [Online]. Available: https://www.vogelwarte.ch/wp-content/uploads/2024/05/Zustand-der-Vogelwelt-2017_en_web.pdf
- [11] T. Bohnenstengel *et al.*, “Rote Liste Fledermäuse. Gefährdete Arten der Schweiz, Stand 2011,” Bundesamt für Umwelt (BAFU), Bern. Umwelt-Vollzug Nr. 1412, 2014. Accessed: May 25, 2026. [Online]. Available: <https://www.dora.lib4ri.ch/wsl/item/wsl:10270>
- [12] Stadt Zürich, “Fledermäuse.” Accessed: May 25, 2026. [Online]. Available: <https://www.stadt-zuerich.ch/de/umwelt-und-energie/natur/tiere-in-der-stadt/fledermaeuse.html>
- [13] L. Villarroya-Villalba *et al.*, “Response of bats and nocturnal insects to urban green areas in Europe,” *Basic and Applied Ecology*, vol. 51, 2021, doi: [10.1016/j.baae.2021.01.006](https://doi.org/10.1016/j.baae.2021.01.006).
- [14] C. Roemer *et al.*, “An automatic classifier of bat sonotypes around the world,” *Methods in Ecology and Evolution*, vol. 12, 2021, doi: [10.1111/2041-210X.13721](https://doi.org/10.1111/2041-210X.13721).

- [15] M. Obrist, *Bat echolocation research: A handbook for planning and conducting acoustic studies*. Bat Conservation International, 2020. Accessed: May 25, 2026. [Online]. Available: <https://www.dora.lib4ri.ch/wsl/item/wsl:24877>
- [16] Y. Bas, D. Bas, and J.-F. Julien, “Tadarida: A toolbox for animal detection on acoustic recordings,” *Journal of Open Research Software*, vol. 5, 2017, doi: [10.5334/jors.154](https://doi.org/10.5334/jors.154).
- [17] M. K. Obrist and R. Boesch, “BatScope manages acoustic recordings, analyses calls, and classifies bat species automatically,” *Canadian Journal of Zoology*, vol. 96, 2018, doi: [10.1139/cjz-2017-0103](https://doi.org/10.1139/cjz-2017-0103).
- [18] L. Rauch, R. Schwinger, M. Wirth, B. Sick, S. Tomforde, and C. Scholz, “Active Bird2Vec: Towards End-to-End Bird Sound Monitoring with Transformers,” *arXiv preprint*, 2023, doi: [10.48550/arXiv.2308.07121](https://doi.org/10.48550/arXiv.2308.07121).
- [19] D. Yi and X. Shen, “Investigation of Bird Sound Transformer Modeling and Recognition,” *Electronics*, vol. 13, 2024, doi: [10.3390/electronics13193964](https://doi.org/10.3390/electronics13193964).
- [20] F. Fundel, D. A. Braun, and S. Gottwald, “Automatic bat call classification using transformer networks,” *Ecological Informatics*, vol. 78, 2023, doi: [10.1016/j.ecoinf.2023.102288](https://doi.org/10.1016/j.ecoinf.2023.102288).
- [21] Cornell Lab of Ornithology, “Merlin Bird ID.” Accessed: June 05, 2026. [Online]. Available: <https://merlin.allaboutbirds.org/>
- [22] Cornell Lab of Ornithology, “About the Macaulay Library.” Accessed: June 05, 2026. [Online]. Available: <https://www.birds.cornell.edu/landtrust/macaulay-library>
- [23] Cornell Lab of Ornithology, “Macaulay Library.” Accessed: June 05, 2026. [Online]. Available: <https://www.macaulaylibrary.org/>
- [24] W.-P. Vellinga and R. Planque, “The Xeno-canto collection and its relation to sound recognition and classification,” in *Working Notes of CLEF 2015 — Conference and Labs of the Evaluation forum*, CEUR-WS, 2015. Accessed: Apr. 02, 2026. [Online]. Available: <https://ceur-ws.org/Vol-1391/166-CR.pdf>
- [25] Xeno-canto Foundation for Nature Sounds, “Xeno-canto: Sharing wildlife sounds from around the world.” Accessed: Mar. 26, 2026. [Online]. Available: <https://xeno-canto.org/>
- [26] J. E. A. Godinez and Ø. Steifetten, “A comparative analysis of Merlin and BirdNet applications for accurate bird species identification through passive acoustic monitoring,” University of South-Eastern Norway, 2024. Accessed: June 05, 2026. [Online]. Available: <https://hdl.handle.net/11250/3137413>
- [27] BirdNET-Team, “Models — BirdNET-Analyzer documentation.” Accessed: June 05, 2026. [Online]. Available: <https://birdnet-team.github.io/BirdNET-Analyzer/models.html>
- [28] BirdNET-Team, *BirdNET-Team — The AI toolkit for conservation*. (2022). GitHub. Accessed: June 05, 2026. [Online]. Available: <https://github.com/birdnet-team>
- [29] G. Schiavo, A. Portaccio, and A. Testolin, “Fine-Tuning BirdNET for the Automatic Ecoacoustic Monitoring of Bird Species in the Italian Alpine Forests,” *Information*, vol. 16, 2025, doi: [10.3390/info16080628](https://doi.org/10.3390/info16080628).
- [30] B. Ghani, T. Denton, S. Kahl, and H. Klinck, “Global birdsong embeddings enable superior transfer learning for bioacoustic classification,” *Scientific Reports*, 2023, doi: [10.1038/s41598-023-49989-z](https://doi.org/10.1038/s41598-023-49989-z).
- [31] B. van Merriënboer, V. Dumoulin, J. Hamer, L. Harrell, A. Burns, and T. Denton, “Perch 2.0: The Bittern Lesson for Bioacoustics,” 2025. Accessed: June 05, 2026. [Online]. Available: <https://arxiv.org/abs/2508.04665>

- [32] W.-P. Vellinga, “Xeno-canto: Bird sounds from around the world,” Xeno-canto Foundation for Nature Sounds, 2026. doi: [10.15468/qv0ksn](https://doi.org/10.15468/qv0ksn).
- [33] Swiss Federal Institute for Forest, Snow and Landscape Research WSL, “Getting Started — BatScope 4 4.2.4 documentation.” Accessed: May 22, 2026. [Online]. Available: <https://static.wsl.ch/batscope/manual-4.2/getting-started.html>
- [34] R. D. Zinck, *BattyBirdNET - Bat Sound Analyzer*. (2023). GitHub. Accessed: Mar. 26, 2026. [Online]. Available: <https://github.com/rdz-oss/BattyBirdNET-Analyzer>
- [35] C. E. Shannon, “Communication in the Presence of Noise,” *Proceedings of the IRE*, vol. 37, 1949, doi: [10.1109/JRPROC.1949.232969](https://doi.org/10.1109/JRPROC.1949.232969).
- [36] D. Stowell, “Computational bioacoustics with deep learning: a review and roadmap,” *PeerJ*, vol. 10, 2022, doi: [10.7717/peerj.13152](https://doi.org/10.7717/peerj.13152).
- [37] P. Prince, “Using software-based acoustic detection and supporting tools to enable large-scale environmental monitoring,” Doctoral thesis, University of Southampton, 2019. Accessed: May 06, 2026. [Online]. Available: <https://eprints.soton.ac.uk/438947/>
- [38] S. Alonso-Rollán, G. Redondo-Galán, P. Peramato-Benito, L. González-Hernández, and A. Carrera, “Microphone Optimization and Development for AI-Based Detection of Flying Wildlife,” in *Distributed Computing and Artificial Intelligence, Special Sessions II, 22nd International Conference (DCAI 2025)*, Springer, Cham, Apr. 2026. doi: [10.1007/978-3-032-05748-8_6](https://doi.org/10.1007/978-3-032-05748-8_6).
- [39] British Trust for Ornithology, “Advice for commonly used recorders and bat detectors.” Accessed: May 06, 2026. [Online]. Available: <https://www.bto.org/data/tools-products/acoustic-pipeline/support-hub/advice-commonly-used>
- [40] Google LLC, “Google Maps.” Accessed: Mar. 13, 2026. [Online]. Available: <https://maps.google.com/>
- [41] MDN Contributors, “Window: showOpenFilePicker() method.” Accessed: May 23, 2026. [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/API/Window/showOpenFilePicker>
- [42] S. Brown, “The C4 model for visualising software architecture.” Accessed: June 06, 2026. [Online]. Available: <https://c4model.com/>
- [43] R. Bösch, “Personal communication, WSL.” June 02, 2026.
- [44] BirdNET-Team, “Creating Your Own Species List — BirdNET-Analyzer documentation.” Accessed: June 05, 2026. [Online]. Available: <https://birdnet-team.github.io/BirdNET-Analyzer/best-practices/species-lists.html>
- [45] BirdNET-Team, “Training Custom Classifiers — BirdNET-Analyzer documentation.” Accessed: June 05, 2026. [Online]. Available: <https://birdnet-team.github.io/BirdNET-Analyzer/best-practices/training.html>
- [46] NVIDIA Corporation, “NVIDIA DGX-2: AI Servers for Solving Complex AI Challenges.” Accessed: June 10, 2026. [Online]. Available: <https://www.nvidia.com/en-in/data-center/dgx-2/>
- [47] K. R. Davis, B. Peabody, and P. J. Leach, “Universally Unique Identifiers (UUIDs),” Internet Engineering Task Force (IETF), May 2024. doi: [10.17487/RFC9562](https://doi.org/10.17487/RFC9562).
- [48] O. Dietsche, *ost-wildlife — companion code for the OST bird and bat acoustic recognition thesis*. (2026).

- [49] D. Li, Z. Wang, Y. Chen, R. Jiang, W. Ding, and M. Okumura, “A Survey on Deep Active Learning: Recent Advances and New Frontiers,” *IEEE Transactions on Neural Networks and Learning Systems*, 2024, doi: [10.1109/TNNLS.2024.3396463](https://doi.org/10.1109/TNNLS.2024.3396463).

F Appendix

F.1 Annotation Instructions

Use this guide to review BirdNET-predicted segments in the deployed UI:

- <https://ilf-audiomoth.vercel.app/>

F.1.1 Requirements

- Browser: Chrome or Edge (desktop recommended)
- A segment-review export folder containing:
 - `manifest.json`
 - `labels_template.csv`
 - `clips/*.wav`

F.1.2 Before You Start

Expected layout after unzip:

```
instructions.pdf
dataset-1/
  manifest.json
  labels_template.csv
  clips/*.wav
```

1. Do not rename files inside `dataset-1`.
2. Keep `labels_template.csv` in the `dataset-1` root.

F.1.3 Open the Dataset

1. Open the deployed web UI.
2. Click **Open Export Folder**.
3. Select the export folder root (the folder that directly contains `manifest.json`), i.e. select `dataset-1`.
4. Enter your **Annotator ID**.

F.1.4 Label Each Segment

For each row, listen to the audio and set one decision:

- `confirm` – predicted species is correct
- `correct` – prediction is wrong, but you can provide exactly one corrected species
- `unknown` – bird is present, but species cannot be identified reliably
- `reject` – no bird vocalisation in this segment

Rules for `corrected_species`:

- Required only when `decision=correct`
- Must contain exactly one species
- Use Latin species name
- Do not use `;` separators

`notes` is optional and can be used for uncertainty or context.

F.1.5 Navigation and Shortcuts

- Use the queue on the left to jump between rows.
- Use **Previous** / **Next** for sequential review.
- Keyboard shortcuts:
 - `C` – set `confirm`

- X – set correct
- U – set unknown
- V – set reject
- R – toggle needs_review

F.1.6 Save Behaviour

- Changes auto-save to labels_template.csv.
- You can click **Save** any time for a manual save.
- The app may also create helper files in the same folder:
 - .annotation_state.json (resume position/status)
 - custom_species.txt (species typed during correction)
 - .backup/labels_template-YYYYMMDD-HHMMSS.csv (periodic backups)

F.1.7 Completion Checklist

1. Progress is at 100%.
2. No rows remain in todo unless intentionally marked needs_review.
3. All correct rows have a valid corrected_species.
4. Annotator ID is set.
5. labels_template.csv is updated in the export folder.

F.1.8 Handover

Return the full dataset-1 folder (or at minimum the updated labels_template.csv) to the project owner.

F.2 BirdNET Precision Across All Species

The table below reports the BirdNET precision assessment results across all species. The decisions and the TP / FP assignment are the same as described in Chapter 7.1.1. Across all species the detection precision is 93.4% and the species precision is 84.9%.

Species	Predicted	Confirmed	Mismatch	Rejected	Unknown	Precision
<i>Fulica atra</i>	25	21	0	0	4	100.0%
<i>Pica pica</i>	22	21	0	1	0	95.5%
<i>Erithacus rubecula</i>	16	16	0	0	0	100.0%
<i>Passer domesticus</i>	12	12	0	0	0	100.0%
<i>Phylloscopus collybita</i>	11	10	1	0	0	90.9%
<i>Corvus corone</i>	11	10	0	0	1	100.0%
<i>Cyanistes caeruleus</i>	10	10	0	0	0	100.0%
<i>Mareca penelope</i>	10	7	1	0	2	87.5%
<i>Parus major</i>	9	9	0	0	0	100.0%
<i>Chroicocephalus ridibundus</i>	8	7	0	0	1	100.0%
<i>Fringilla coelebs</i>	8	7	0	0	1	100.0%
<i>Rallus aquaticus</i>	8	0	0	6	2	0.0%
<i>Turdus merula</i>	7	6	1	0	0	85.7%
<i>Motacilla alba</i>	5	5	0	0	0	100.0%
<i>Corvus frugilegus</i>	5	1	2	0	2	33.3%
<i>Larus michahellis</i>	5	5	0	0	0	100.0%
<i>Streptopelia decaocto</i>	5	5	0	0	0	100.0%
<i>Turdus philomelos</i>	4	1	0	1	2	50.0%
<i>Sylvia atricapilla</i>	4	4	0	0	0	100.0%
<i>Anas platyrhynchos</i>	4	3	0	0	1	100.0%
<i>Ciconia ciconia</i>	3	3	0	0	0	100.0%
<i>Gallinula chloropus</i>	3	0	0	0	3	n/a
<i>Prunella modularis</i>	3	3	0	0	0	100.0%
<i>Corvus cornix</i>	2	1	1	0	0	50.0%
<i>Milvus milvus</i>	2	2	0	0	0	100.0%
<i>Strix aluco</i>	2	0	0	2	0	0.0%
<i>Tringa ochropus</i>	2	0	2	0	0	0.0%
<i>Phoenicurus ochruros</i>	2	1	0	0	1	100.0%
<i>Anthus trivialis</i>	2	0	2	0	0	0.0%
<i>Aythya ferina</i>	2	0	1	1	0	0.0%
<i>Aegithalos caudatus</i>	2	2	0	0	0	100.0%
<i>Certhia brachydactyla</i>	1	1	0	0	0	100.0%
<i>Pandion haliaetus</i>	1	0	0	0	1	n/a
<i>Motacilla cinerea</i>	1	0	0	0	1	n/a
<i>Melanitta nigra</i>	1	0	1	0	0	0.0%
<i>Anser albifrons</i>	1	0	0	0	1	n/a
<i>Turdus iliacus</i>	1	0	0	1	0	0.0%

Species	Predicted	Confirmed	Mismatch	Rejected	Unknown	Precision
<i>Pluvialis apricaria</i>	1	0	0	1	0	0.0%
<i>Asio otus</i>	1	1	0	0	0	100.0%
<i>Coccothraustes coccothraustes</i>	1	0	1	0	0	0.0%
<i>Certhia familiaris</i>	1	0	1	0	0	0.0%
<i>Spinus spinus</i>	1	0	1	0	0	0.0%
<i>Ardea cinerea</i>	1	0	0	0	1	n/a
<i>Columba livia</i>	1	0	0	1	0	0.0%
<i>Podiceps cristatus</i>	1	0	0	1	0	0.0%
<i>Emberiza schoeniclus</i>	1	0	1	0	0	0.0%

F.3 Detected Bird Species and Plausibility

The complete list of all 173 bird species BirdNET detected across the dataset is reproduced below, ranked by number of detections and maximum confidence, with the per-species plausibility notes of the two ILF experts, Jasmin Joshi and Nicolas Auchli [4]; it is the source for the summary in Table 12. The detection counts and maximum confidence are BirdNET's own; the notes are reproduced verbatim in the original German. The status codes in these notes are IUCN Red List categories, as applied in the Swiss Red List of Breeding Birds [8]: CR (Critically Endangered), EN (Endangered), VU (Vulnerable), and NT (Near Threatened).

No.	Species	Common name	Max. conf.	Detections	Month	Einschätzung Jasmin Joshi	Einschätzung Nicolas Auchli
1	<i>Fulica atra</i>	Blässhuhn	1.0	16'578	November		Passt
2	<i>Erithacus rubecula</i>	Rotkehlchen	1.0	8'248	November		Passt
3	<i>Rallus aquaticus</i>	Wasserralle	1.0	3'869	November		Passt
4	<i>Corvus corone</i>	Aaskrähe	1.0	3'557	November		Passt
5	<i>Pica pica</i>	Elster	1.0	3'064	November		Passt
6	<i>Turdus philomelos</i>	Singdrossel	1.0	2'980	October		Passt
7	<i>Regulus regulus</i>	Wintergoldhähnchen	1.0	2'713	November		Passt
8	<i>Cyanistes caeruleus</i>	Blaumeise	1.0	2'612	October		Passt
9	<i>Strix aluco</i>	Waldkauz	1.0	2'538	October	cool	Passt
10	<i>Parus major</i>	Kohlmeise	1.0	2'254	October		Passt
11	<i>Gallinula chloropus</i>	Teichhuhn	1.0	2'162	November		Passt
12	<i>Phylloscopus collybita</i>	Zilpzalp	1.0	2'112	October		Passt
13	<i>Anas platyrhynchos</i>	Stockente	1.0	2'078	November		Passt
14	<i>Turdus merula</i>	Amsel	1.0	1'878	November		Passt
15	<i>Turdus iliacus</i>	Rotdrossel	1.0	1'785	November	cool	Passt
16	<i>Motacilla alba</i>	Bachstelze	1.0	1'625	October		Passt
17	<i>Chroicocephalus ridibundus</i>	Lachmöwe	1.0	1'554	November		Passt
18	<i>Aegithalos caudatus</i>	Schwanzmeise	1.0	1'525	November		Passt
19	<i>Corvus cornix</i>	Nebelkrähe	1.0	1'519	November		Wohl eher Rabenkrähe/Aaskrähe
20	<i>Fringilla coelebs</i>	Buchfink	1.0	994	November		Passt
21	<i>Corvus frugilegus</i>	Saatkrähe	1.0	702	November		Passt
22	<i>Coccothraustes coccothraustes</i>	Kernbeisser	1.0	694	November		Passt
23	<i>Troglodytes troglodytes</i>	Zaunkönig	1.0	558	November		Passt

No.	Species	Common name	Max. conf.	Detections	Month	Einschätzung Jasmin Joshi	Einschätzung Nicolas Auchli
24	<i>Carduelis carduelis</i>	Stieglitz	1.0	539	October		Passt
25	<i>Certhia brachydactyla</i>	Gartenbaumläufer	1.0	515	October		Passt
26	<i>Motacilla cinerea</i>	Gebirgsstelze	1.0	489	November	cool	Passt
27	<i>Anser anser</i>	Graugans	1.0	488	November		Passt
28	<i>Ardea cinerea</i>	Graureiher	1.0	474	October		Passt
29	<i>Numenius arquata</i>	Grosser Brachvogel	1.0	386	October	cool, CR	Passt
30	<i>Tringa ochropus</i>	Waldwasserläufer	1.0	336	November	cool	Passt
31	<i>Prunella modularis</i>	Heckenbraunelle	1.0	325	October		Passt
32	<i>Streptopelia decaocto</i>	Türkentaube	1.0	321	November		Passt
33	<i>Phoenicurus ochruros</i>	Hausrotschwanz	1.0	298	October		Passt
34	<i>Anthus trivialis</i>	Baumpieper	1.0	248	October		Passt
35	<i>Alcedo atthis</i>	Eisvogel	1.0	221	November	cool	Passt
36	<i>Larus michahellis</i>	Mittelmeermöwe	1.0	202	November		Passt
37	<i>Mareca strepera</i>	Schnatterente	1.0	176	November	cool, VU	Passt, auch wenn Fehlbestimmungen häufig sind
38	<i>Netta rufina</i>	Kolbenente	1.0	170	November		Passt
39	<i>Emberiza schoeniclus</i>	Rohrhammer	1.0	162	October	cool, VU	Passt
40	<i>Scolopax rusticola</i>	Waldschnepfe	1.0	134	December	cool, EN	Evtl. Fehlbestimmung?
41	<i>Spinus spinus</i>	Erlenzeisig	1.0	128	October		Passt
42	<i>Chloris chloris</i>	Grünfink	1.0	115	November		Passt
43	<i>Poecile palustris</i>	Sumpfmehse	1.0	115	October		Passt
44	<i>Dendrocopos major</i>	Buntspecht	1.0	111	November		Passt
45	<i>Turdus pilaris</i>	Wacholderdrossel	1.0	111	November		Passt
46	<i>Grus grus</i>	Kranich	1.0	96	November	cool	Passt
47	<i>Asio otus</i>	Waldohreule	1.0	84	November	cool	Passt
48	<i>Periparus ater</i>	Tannenmeise	1.0	79	October		Passt
49	<i>Phasianus colchicus</i>	Fasan	1.0	77	November		Kommt nicht vor
50	<i>Fringilla montifringilla</i>	Bergfink	1.0	76	November		Passt
51	<i>Serinus serinus</i>	Girlitz	1.0	71	October		Passt
52	<i>Dryocopus martius</i>	Schwarzspecht	1.0	62	November	cool	Passt

No.	Species	Common name	Max. conf.	Detections	Month	Einschätzung Jasmin Joshi	Einschätzung Nicolas Auchli
53	<i>Mareca penelope</i>	Pfeifente	1.0	60	December		Passt
54	<i>Gallinago gallinago</i>	Bekassine	1.0	57	November	cool, EN	Passt
55	<i>Buteo buteo</i>	Mäusebussard	1.0	47	November		Passt
56	<i>Calidris alpina</i>	Alpenstrandläufer	1.0	33	November	cool	Passt
57	<i>Anthus pratensis</i>	Wiesenpieper	1.0	24	November		Passt
58	<i>Picus viridis</i>	Grünspecht	1.0	18	October		Passt
59	<i>Alopochen aegyptiaca</i>	Nilgans	1.0	16	December	not so cool	Passt
60	<i>Bucephala clangula</i>	Schellente	1.0	8	December		Passt
61	<i>Melanitta nigra</i>	Trauerente	0.9	933	November	cool	Spannend! Einfache Art für NocMig, darum wohl korrekt?
62	<i>Muscicapa striata</i>	Grauschnäpper	0.9	351	October	cool	Passt
63	<i>Regulus ignicapilla</i>	Sommergoldhähnchen	0.9	337	November		Passt
64	<i>Apus melba</i>	Alpensegler	0.9	304	October		Eher spät, aber passt
65	<i>Anser albifrons</i>	Blässgans	0.9	244	November		Unwahrscheinlich
66	<i>Passer domesticus</i>	Hausperling	0.9	225	October		Passt
67	<i>Anas crecca</i>	Krickente	0.9	168	November		Passt
68	<i>Corvus monedula</i>	Dohle	0.9	163	November		Passt
69	<i>Botaurus stellaris</i>	Rohrdommel	0.9	104	December	cool, CR	Lohnt sich, nachzuhören. Flugruf ist typisch, Gesang wird oft falsch bestimmt
70	<i>Lophophanes cristatus</i>	Haubenmeise	0.9	77	November		Passt
71	<i>Emberiza cia</i>	Zippammer	0.9	64	November		Lohnt sich, nachzuhören
72	<i>Falco peregrinus</i>	Wanderfalke	0.9	45	November	cool	Möglich
73	<i>Perdix perdix</i>	Rebhuhn	0.9	31	November	cool, CR	Kommt nicht vor
74	<i>Pyrrhula pyrrhula</i>	Gimpel	0.9	29	November		Passt
75	<i>Glaucidium passerinum</i>	Sperlingskauz	0.9	26	November	cool, rare	Unwahrscheinlich
76	<i>Cettia cetti</i>	Seidensänger	0.9	23	October	cool	Unwahrscheinlich
77	<i>Passer montanus</i>	Feldperling	0.9	19	October		Passt
78	<i>Dendrocytes medius</i>	Mittelspecht	0.9	18	November	cool	Unwahrscheinlich
79	<i>Alectoris rufa</i>	Rothuhn	0.9	11	November	cool	Kommt nicht vor

No.	Species	Common name	Max. conf.	Detections	Month	Einschätzung Jasmin Joshi	Einschätzung Nicolas Auchli
80	<i>Tringa nebularia</i>	Grünschenkel	0.9	11	October	cool	Möglich
81	<i>Lullula arborea</i>	Heidelerche	0.9	5	November	cool, VU	Möglich
82	<i>Branta canadensis</i>	Kanadagans	0.8	326	October	not so cool	Wohl Fehlbestimmung
83	<i>Columba livia</i>	Felsentaube	0.8	259	November		Passt
84	<i>Cygnus columbianus</i>	Zwergschwan	0.8	127	November	cool, rare	Wohl Fehlbestimmung
85	<i>Falco tinnunculus</i>	Turmfalke	0.8	102	November		Passt
86	<i>Actitis hypoleucos</i>	Flussuferläufer	0.8	74	November	cool	Passt
87	<i>Aix galericulata</i>	Mandarinente	0.8	67	November		Passt
88	<i>Ficedula hypoleuca</i>	Trauerschnäpper	0.8	52	October		Passt
89	<i>Garrulus glandarius</i>	Eichelhäher	0.8	43	October		Passt
90	<i>Cygnus olor</i>	Höckerschwan	0.8	35	November		Passt
91	<i>Turdus viscivorus</i>	Misteldrossel	0.8	26	November		Passt
92	<i>Corvus corax</i>	Kolkrabe	0.8	25	October		Passt
93	<i>Phoenicurus phoenicurus</i>	Gartenrotschwanz	0.8	23	October	cool	Passt
94	<i>Anthus spinoletta</i>	Bergpieper	0.8	18	October		Passt
95	<i>Motacilla flava</i>	Schafstelze	0.8	13	October		Passt
96	<i>Ichthyaeetus melanocephalus</i>	Schwarzkopfmöwe	0.8	11	December	cool	Passt
97	<i>Remiz pendulinus</i>	Beutelmeise	0.8	8	October	cool	Passt
98	<i>Bubulcus ibis</i>	Kuhreiher	0.8	7	December		Eher unwahrscheinlich
99	<i>Certhia familiaris</i>	Waldbaumläufer	0.7	104	November		Passt
100	<i>Pluvialis apricaria</i>	Goldregenpfeifer	0.7	37	November	cool	Cool! Passt
101	<i>Athene noctua</i>	Steinkauz	0.7	34	October	cool, EN	Kommt nicht vor
102	<i>Emberiza cirrus</i>	Zaunammer	0.7	34	November	cool	Eher unwahrscheinlich
103	<i>Delichon urbicum</i>	Mehlschwalbe	0.7	29	October		Passt
104	<i>Luscinia svecica</i>	Blaukehlchen	0.7	24	October	cool, EN	Cool! Passt
105	<i>Emberiza citrinella</i>	Goldammer	0.7	17	November		Passt
106	<i>Sturnus vulgaris</i>	Star	0.7	17	September		Passt
107	<i>Milvus migrans</i>	Schwarzmilan	0.7	9	December		Zu spät, wohl keiner
108	<i>Ciconia ciconia</i>	Weissstorch	0.7	7	December		Passt
109	<i>Tringa totanus</i>	Rotschenkel	0.7	7	October	cool	Passt

No.	Species	Common name	Max. conf.	Detections	Month	Einschätzung Jasmin Joshi	Einschätzung Nicolas Auchli
110	<i>Curruca melanocephala</i>	Samtkopfgrasmücke	0.7	5	November	cool	Unwahrscheinlich
111	<i>Linaria cannabina</i>	Bluthänfling	0.7	5	November		Passt
112	<i>Ptyonoprogne rupestris</i>	Felsenschwalbe	0.7	5	October		Passt
113	<i>Pernis apivorus</i>	Wespenbussard	0.7	3	October		Unwahrscheinlich, im Oktober Rufe zu haben
114	<i>Larus fuscus</i>	Heringsmöwe	0.6	39	December		Unwahrscheinlich
115	<i>Columba palumbus</i>	Ringeltaube	0.6	29	October		Passt
116	<i>Anas acuta</i>	Spießente	0.6	27	December		Passt
117	<i>Phoenicopterus roseus</i>	Rosaflamingo	0.6	12	October	cool	Unwahrscheinlich
118	<i>Vanellus vanellus</i>	Kiebitz	0.6	11	November	cool	Passt
119	<i>Emberiza calandra</i>	Grauammer	0.6	10	November	cool, CR	Unwahrscheinlich
120	<i>Falco columbarius</i>	Merlin	0.6	10	November	cool	Unwahrscheinlich
121	<i>Phalacrocorax carbo</i>	Kormoran	0.6	10	December		Passt wohl, auch wenn nicht ruf-freudig ausserhalb Brutsaison
122	<i>Sylvia atricapilla</i>	Mönchsgrasmücke	0.6	10	September		Passt
123	<i>Hirundo rustica</i>	Rauchschwalbe	0.6	9	October		Passt
124	<i>Ardea alba</i>	Silberreiher	0.6	6	November		Passt
125	<i>Numenius phaeopus</i>	Regenbrachvogel	0.6	6	September	cool	Möglich
126	<i>Apus apus</i>	Mauersegler	0.6	5	September		Passt
127	<i>Accipiter nisus</i>	Sperber	0.6	3	November		Möglich
128	<i>Aythya ferina</i>	Tafelente	0.5	63	November		Passt
129	<i>Nycticorax nycticorax</i>	Nachtreiher	0.5	33	September	cool	Passt
130	<i>Tachybaptus ruficollis</i>	Zwergtaucher	0.5	31	November	cool, NT	Passt
131	<i>Carduelis citrinella</i>	Zitronenzeisig	0.5	26	November	cool, NT	Unwahrscheinlich
132	<i>Himantopus himantopus</i>	Stelzenläufer	0.5	23	October	cool	Unwahrscheinlich
133	<i>Charadrius dubius</i>	Flussregenpfeifer	0.5	18	October		Möglich
134	<i>Poecile montanus</i>	Weidenmeise	0.5	17	November	cool	Eher unwahrscheinlich
135	<i>Accipiter gentilis</i>	Habicht	0.5	15	November		Eher unwahrscheinlich
136	<i>Saxicola rubicola</i>	Schwarzkehlchen	0.5	15	December	cool, NT	Möglich
137	<i>Dryobates minor</i>	Kleinspecht	0.5	11	November	cool	Möglich

No.	Species	Common name	Max. conf.	Detections	Month	Einschätzung Jasmin Joshi	Einschätzung Nicolas Auchli
138	<i>Picus canus</i>	Grauspecht	0.5	7	November	cool, EN	Möglich
139	<i>Columba oenas</i>	Hohltaube	0.5	5	October		Möglich
140	<i>Gavia stellata</i>	Sternaucher	0.5	5	December		Eher unwahrscheinlich
141	<i>Ardea purpurea</i>	Purpureiher	0.5	3	October	cool, CR	Eher unwahrscheinlich
142	<i>Psittacula krameri</i>	Halsbandsittich	0.5	3	September	not so cool	Unwahrscheinlich
143	<i>Podiceps cristatus</i>	Haubentaucher	0.4	10	November		Passt
144	<i>Egretta garzetta</i>	Seidenreiher	0.4	7	November		Eher unwahrscheinlich
145	<i>Larus canus</i>	Sturmmöwe	0.4	6	October		Passt
146	<i>Calidris alba</i>	Sanderling	0.4	5	December	cool	Möglich
147	<i>Turdus torquatus</i>	Ringdrossel	0.4	5	October	cool, NT	Wohl Fehlbestimmung
148	<i>Calidris pugnax</i>	Kampfläufer	0.4	4	November	cool	Ruft nicht
149	<i>Charadrius hiaticula</i>	Sandregenpfeifer	0.4	4	October	cool	Möglich
150	<i>Milvus milvus</i>	Rotmilan	0.4	4	November		Passt
151	<i>Pandion haliaetus</i>	Fischadler	0.4	4	October	cool, RE	Ruft nicht
152	<i>Panurus biarmicus</i>	Bartmeise	0.4	4	November	cool, VU	Unwahrscheinlich
153	<i>Galerida cristata</i>	Haubenlerche	0.4	2	October	cool, RE	Wohl Heideleerche
154	<i>Tadorna tadorna</i>	Brandgans	0.4	2	November	cool, VU	Möglich
155	<i>Aquila chrysaetos</i>	Steinadler	0.4	1	November	cool, NT	Kommt nicht vor
156	<i>Circus aeruginosus</i>	Rohrweihe	0.4	1	December	cool, VU	Ruft nicht
157	<i>Phylloscopus trochilus</i>	Fitis	0.3	10	October		Möglich
158	<i>Pluvialis squatarola</i>	Kiebitzregenpfeifer	0.3	4	November	cool	Möglich
159	<i>Anser indicus</i>	Streifengans	0.3	3	December	seltsam!	Unwahrscheinlich
160	<i>Larus cachinnans</i>	Steppenmöwe	0.3	3	December		Wahrscheinlich Mittelmeermöwe
161	<i>Tringa glareola</i>	Bruchwasserläufer	0.3	3	October	cool	Passt
162	<i>Loxia curvirostra</i>	Fichtenkreuzschnabel	0.3	2	October		Häufige Fehlbestimmung
163	<i>Oenanthe oenanthe</i>	Steinschmätzer	0.3	1	October		Möglich
164	<i>Passer italiae</i>	Italiensperling	0.3	1	December	cool, NT	Hausperling
165	<i>Sitta europaea</i>	Kleiber	0.2	3	October		Möglich
166	<i>Nucifraga caryocatactes</i>	Tannenhäher	0.2	2	October		Unwahrscheinlich
167	<i>Sterna hirundo</i>	Flussseeschwalbe	0.2	2	October		Spät

No.	Species	Common name	Max. conf.	Detections	Month	Einschätzung Jasmin Joshi	Einschätzung Nicolas Auchli
168	<i>Alauda arvensis</i>	Feldlerche	0.2	1	October	cool, VU	Möglich
169	<i>Curruca curruca</i>	Klappergrasmücke	0.2	1	October		Möglich
170	<i>Falco subbuteo</i>	Baumfalke	0.2	1	October	cool, NT	Ruft nicht
171	<i>Limosa lapponica</i>	Pfuhschnepfe	0.2	1	October	cool	Möglich
172	<i>Podiceps grisegena</i>	Rothalstaucher	0.2	1	November		Möglich
173	<i>Spatula clypeata</i>	Löffelente	0.2	1	November	cool, VU	Möglich

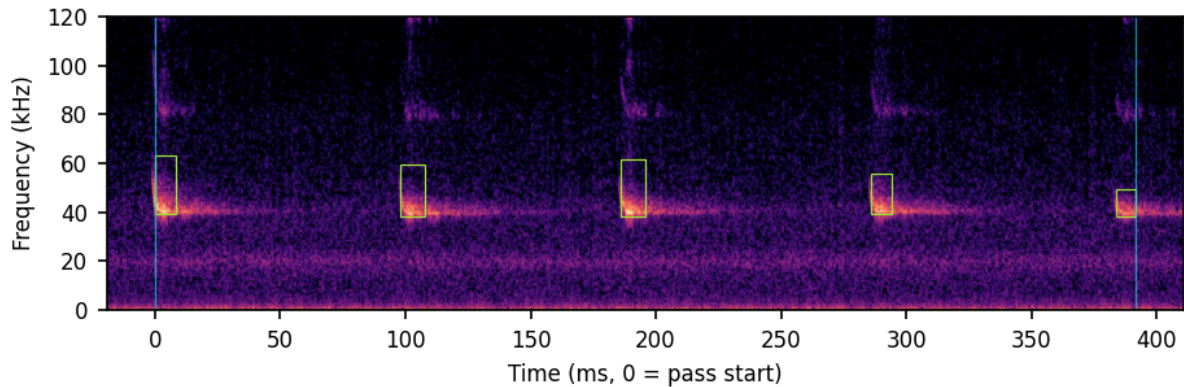
F.4 BatDetect2 Sample Calls

For each species, the three calls (merged passes) with the highest classification confidence from the processed predictions (confidence threshold 0.3; passes shorter than 20 ms excluded). The cyan lines mark the start and end of the pass; the green boxes are the individual raw BatDetect2 detections (time x frequency extent) merged into the pass. Swiss Red List categories follow [11].

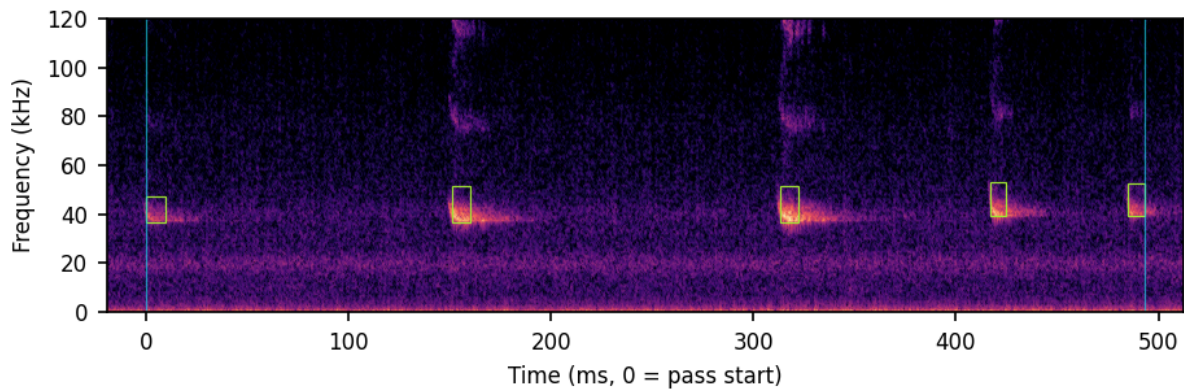
Each panel is a STFT spectrogram of the recording window around the pass (512-sample analysis window, 90% overlap). Magnitude is shown in decibels over a 60 dB dynamic range normalised to each panel's peak, on a linear frequency axis in kHz capped at 120 kHz, using the magma colour map.

F.4.1 Nathusius's pipistrelle / Rauhautfledermaus (pipnat)

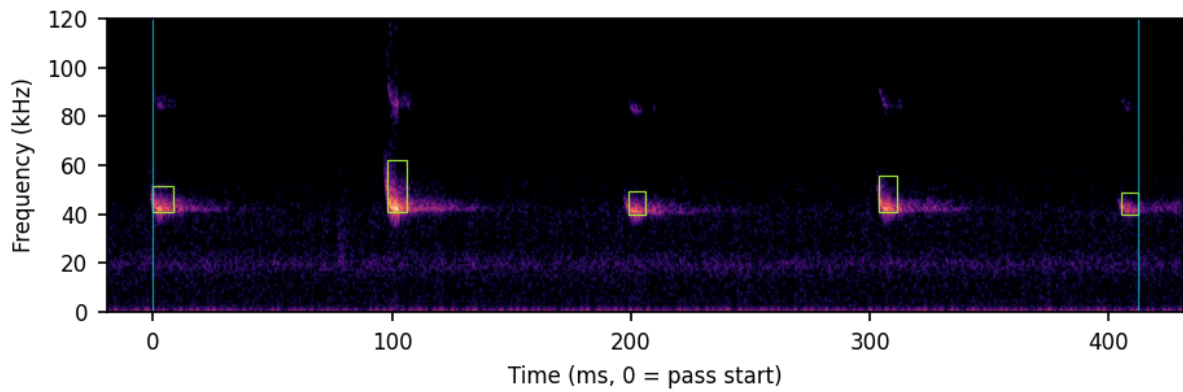
Pipistrellus nathusii · Swiss Red List: LC · detected 10008 times



File	Audiomoths/20251007/Audiomoth3/20251005/243B1F026487A7FE_20251005_183000.WAV
Time	02:53.523 - 02:53.915 (392 ms)
Species	pipnat - Nathusius's pipistrelle / Rauhautfledermaus (<i>Pipistrellus nathusii</i>)
Confidence	Classification 89% / Detection 90%
Frequency band	38.4 - 63.0 kHz
Swiss Red List	LC (Least Concern)



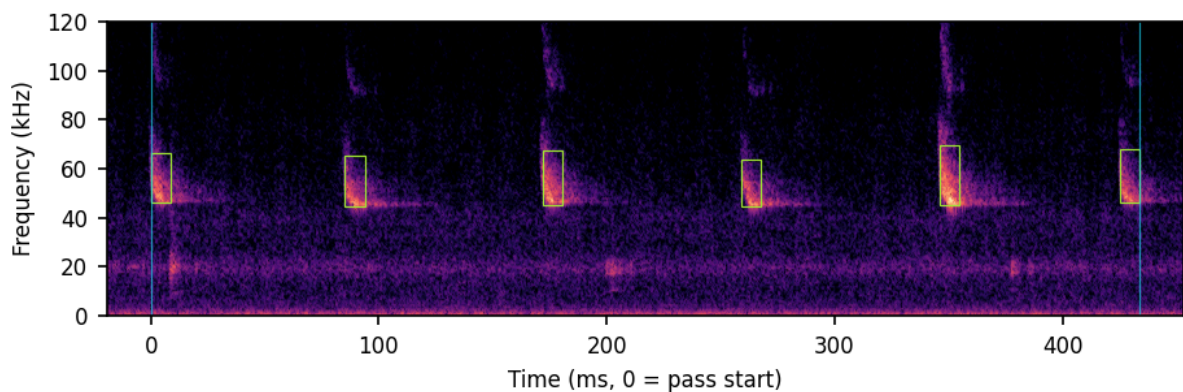
File	Audiomoths/20251007/Audiomoth3/20251005/243B1F026487A7FE_20251005_180000.WAV
Time	00:37.503 - 00:37.996 (493 ms)
Species	pipnat - Nathusius's pipistrelle / Rauhautfledermaus (<i>Pipistrellus nathusii</i>)
Confidence	Classification 87% / Detection 88%
Frequency band	36.6 - 53.0 kHz
Swiss Red List	LC (Least Concern)



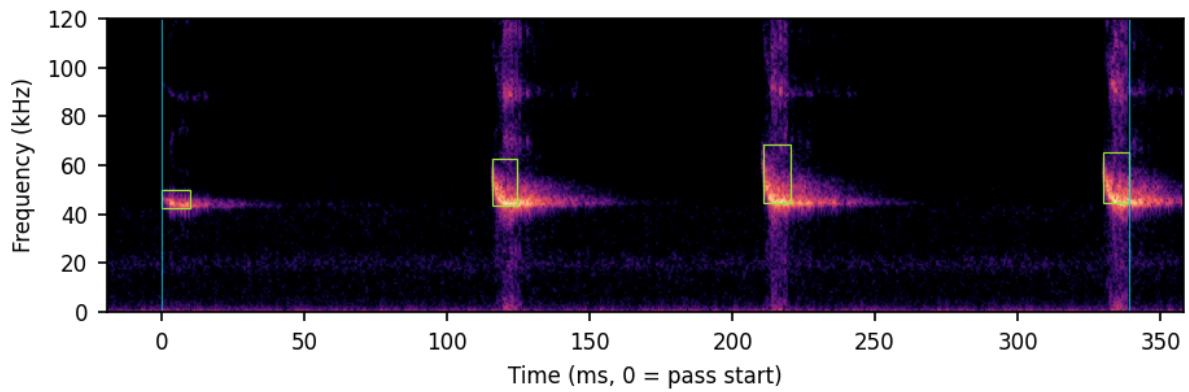
File	Audiomoths/20251015/Audiomoth1/20251010/243B1F026487A7CB_20251010_034500.WAV
Time	01:03.032 - 01:03.444 (412 ms)
Species	pipnat - Nathusius's pipistrelle / Rauhautfledermaus (<i>Pipistrellus nathusii</i>)
Confidence	Classification 87% / Detection 89%
Frequency band	40.1 - 62.3 kHz
Swiss Red List	LC (Least Concern)

F.4.2 Common pipistrelle / Zwergfledermaus (pippip)

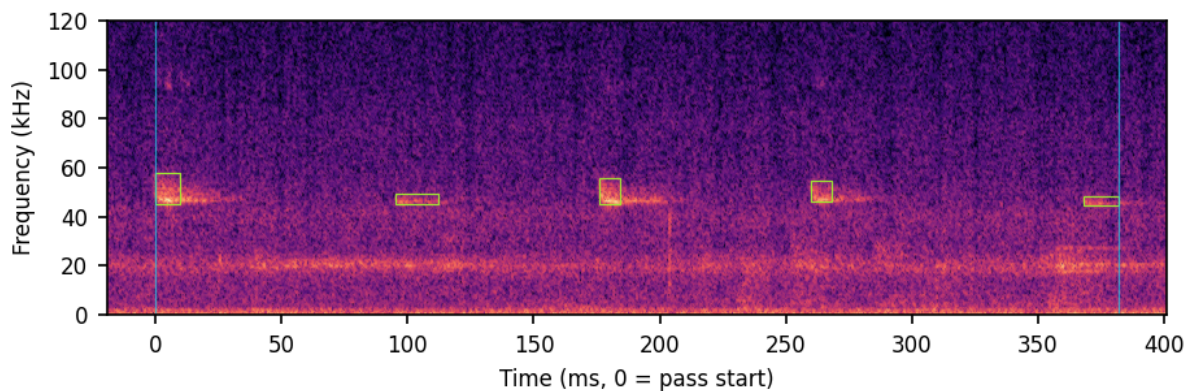
Pipistrellus pipistrellus · Swiss Red List: LC · detected 901 times



File	Audiomoths/20251007/Audiomoth1/20251005/243B1F026487A7CB_20251005_181500.WAV
Time	03:04.509 - 03:04.943 (434 ms)
Species	pippip - Common pipistrelle / Zwergfledermaus (<i>Pipistrellus pipistrellus</i>)
Confidence	Classification 75% / Detection 76%
Frequency band	44.4 - 69.6 kHz
Swiss Red List	LC (Least Concern)



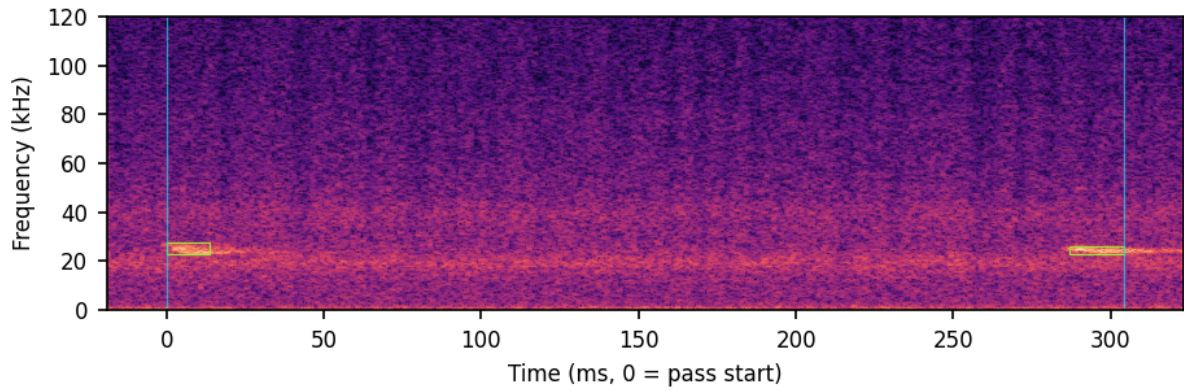
File	Audiomoths/20251007/Audiomoth3/20251005/243B1F026487A7FE_20251005_191500.WAV
Time	04:25.527 - 04:25.866 (339 ms)
Species	pippip - Common pipistrelle / Zwergfledermaus (Pipistrellus pipistrellus)
Confidence	Classification 72% / Detection 75%
Frequency band	42.7 - 68.4 kHz
Swiss Red List	LC (Least Concern)



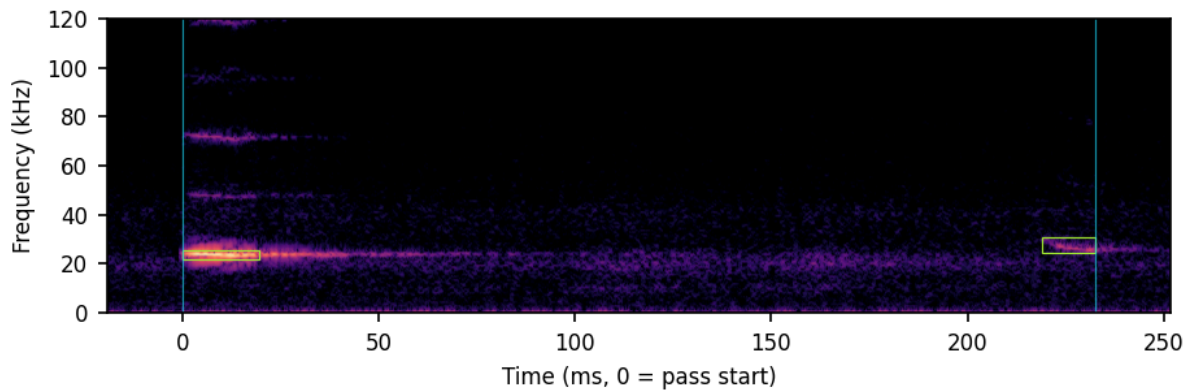
File	Audiomoths/20250923/Audiomoth4/20250922/242A260564877FAB_20250922_184500.WAV
Time	03:10.576 - 03:10.958 (382 ms)
Species	pippip - Common pipistrelle / Zwergfledermaus (Pipistrellus pipistrellus)
Confidence	Classification 71% / Detection 78%
Frequency band	44.4 - 57.9 kHz
Swiss Red List	LC (Least Concern)

F.4.3 Lesser noctule / Kleiner Abendsegler (nyclei)

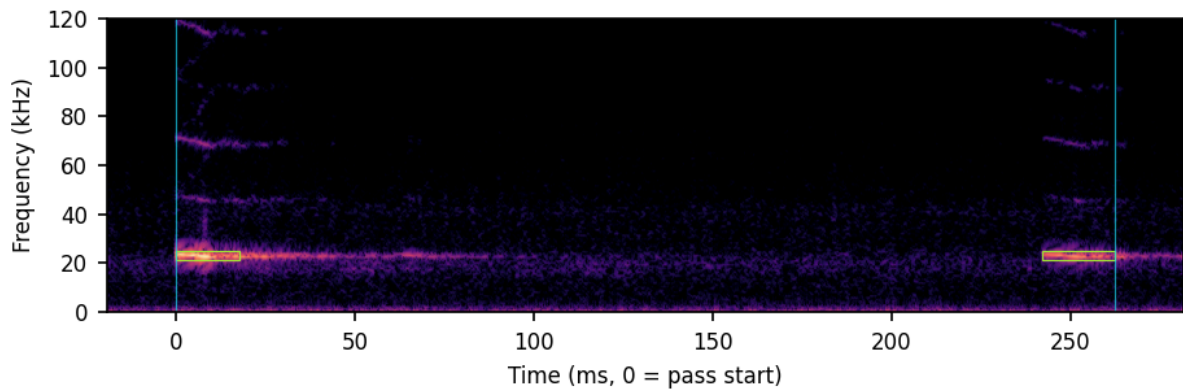
Nyctalus leisleri · Swiss Red List: NT · detected 300 times



File	Audiomoths/20251028/Audiomoth1/20251015/243B1F026487A7CB_20251015_211500.WAV
Time	02:39.512 - 02:39.816 (304 ms)
Species	nyclei - Lesser noctule / Kleiner Abendsegler (Nyctalus leisleri)
Confidence	Classification 71% / Detection 79%
Frequency band	22.9 - 27.8 kHz
Swiss Red List	NT (Near Threatened)



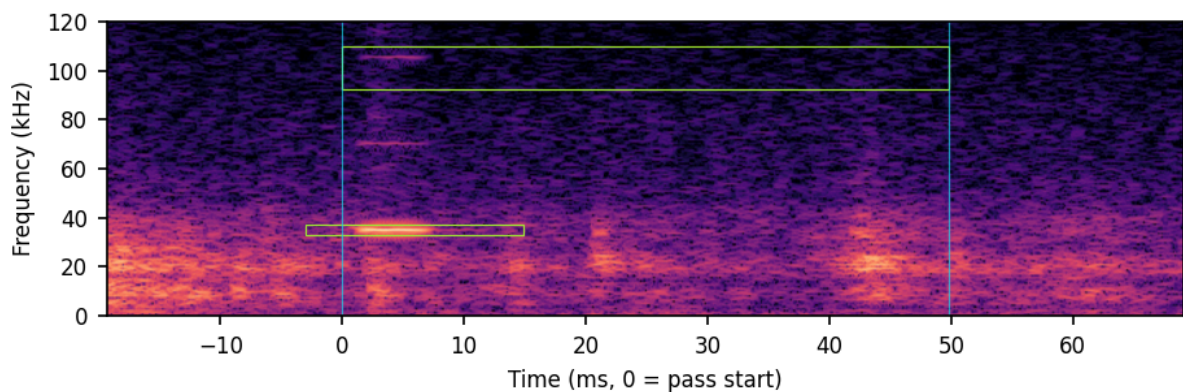
File	Audiomoths/20251015/Audiomoth3/20251009/243B1F026487A7FE_20251009_201500.WAV
Time	03:14.571 - 03:14.804 (233 ms)
Species	nyclei - Lesser noctule / Kleiner Abendsegler (Nyctalus leisleri)
Confidence	Classification 70% / Detection 79%
Frequency band	22.0 - 30.6 kHz
Swiss Red List	NT (Near Threatened)



File	Audiomoths/20250923/Audiomoth3/20250919/243B1F026487A7FE_20250919_220000.WAV
Time	01:45.617 - 01:45.879 (262 ms)
Species	nyclei - Lesser noctule / Kleiner Abendsegler (<i>Nyctalus leisleri</i>)
Confidence	Classification 69% / Detection 75%
Frequency band	21.2 - 25.2 kHz
Swiss Red List	NT (Near Threatened)

F.4.4 Lesser horseshoe bat / Kleine Hufeisennase (rhihip)

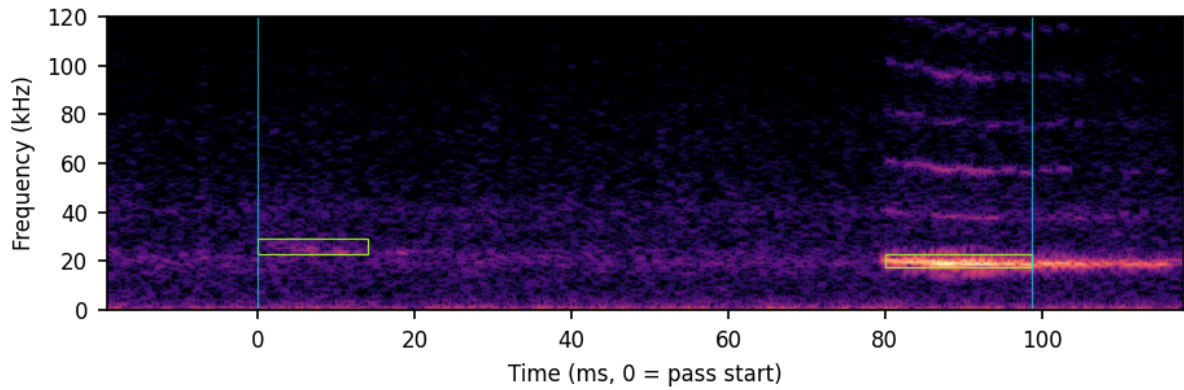
Rhinolophus hipposideros · Swiss Red List: EN · detected 295 times



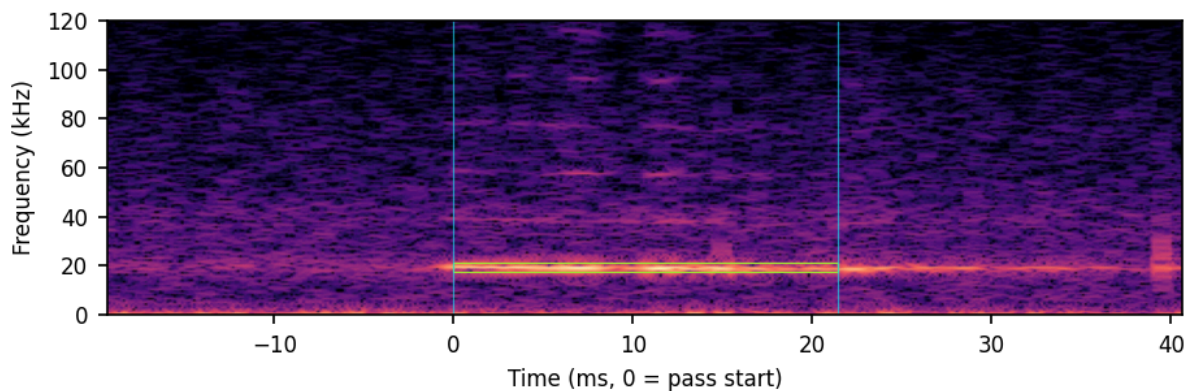
File	Audiomoths/20251007/Audiomoth1/20250930/243B1F026487A7CB_20250930_214500.WAV
Time	04:03.467 - 04:03.517 (50 ms)
Species	rhihip - Lesser horseshoe bat / Kleine Hufeisennase (<i>Rhinolophus hipposideros</i>)
Confidence	Classification 53% / Detection 53%
Frequency band	92.5 - 109.7 kHz
Swiss Red List	EN (Endangered)

F.4.5 Common noctule / Grosser Abendsegler (nycnoc)

Nyctalus noctula · Swiss Red List: NT · detected 78 times



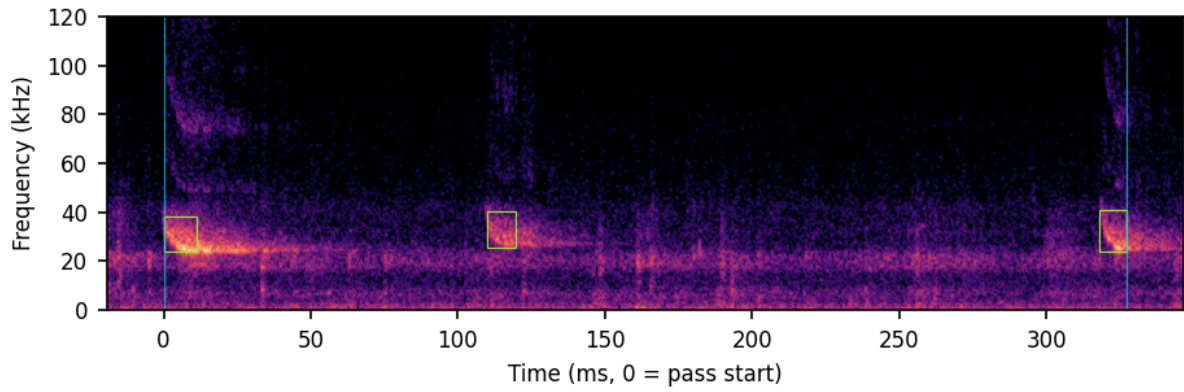
File	Audiomoths/20251015/Audiomoth3/20251013/243B1F026487A7FE_20251013_051500.WAV
Time	02:44.156 - 02:44.255 (99 ms)
Species	nycnoc - Common noctule / Grosser Abendsegler (Nyctalus noctula)
Confidence	Classification 58% / Detection 67%
Frequency band	17.7 - 29.2 kHz
Swiss Red List	NT (Near Threatened)



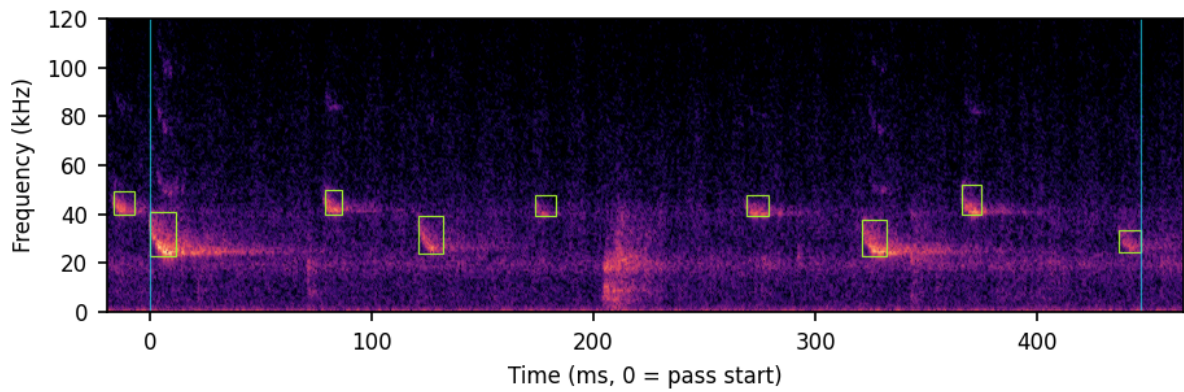
File	Audiomoths/20251015/Audiomoth3/20251011/243B1F026487A7FE_20251011_044500.WAV
Time	03:12.153 - 03:12.174 (21 ms)
Species	nycnoc - Common noctule / Grosser Abendsegler (Nyctalus noctula)
Confidence	Classification 53% / Detection 60%
Frequency band	17.7 - 21.1 kHz
Swiss Red List	NT (Near Threatened)

F.4.6 Serotine bat / Breitflügelfledermaus (cneser)

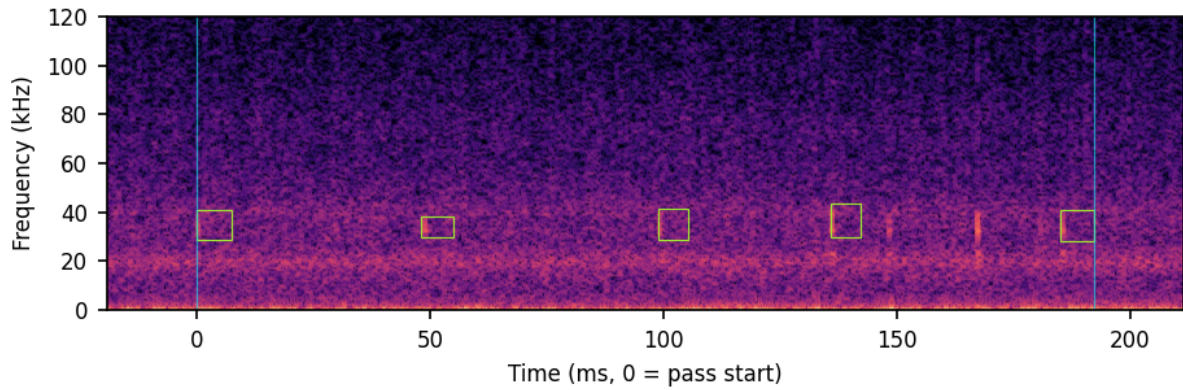
Eptesicus serotinus · Swiss Red List: VU · detected 76 times



File	Audiomoths/20250923/Audiomoth3/20250921/243B1F026487A7FE_20250921_190000.WAV
Time	04:56.520 - 04:56.847 (327 ms)
Species	cneser - Serotine bat / Breitflügelfledermaus (<i>Eptesicus serotinus</i>)
Confidence	Classification 46% / Detection 65%
Frequency band	23.8 - 41.1 kHz
Swiss Red List	VU (Vulnerable)



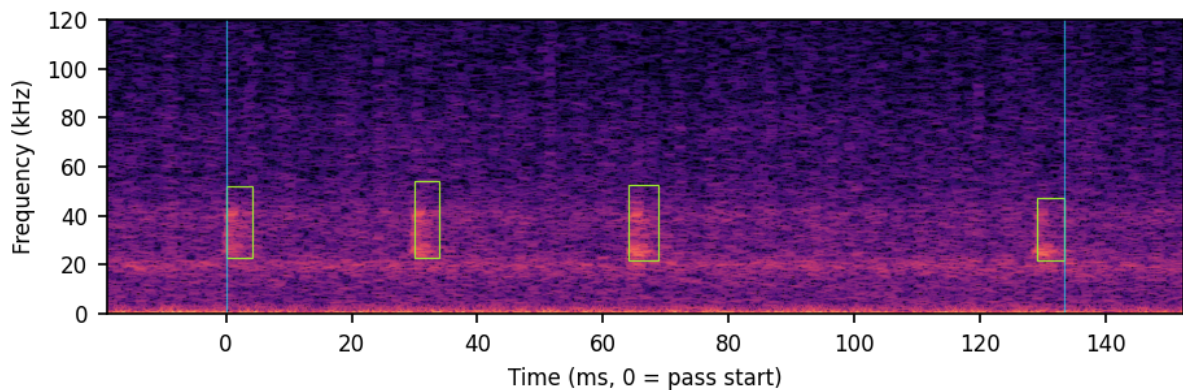
File	Audiomoths/20251007/Audiomoth1/20251004/243B1F026487A7CB_20251004_181500.WAV
Time	01:34.535 - 01:34.982 (447 ms)
Species	cneser - Serotine bat / Breitflügelfledermaus (<i>Eptesicus serotinus</i>)
Confidence	Classification 45% / Detection 68%
Frequency band	22.9 - 40.9 kHz
Swiss Red List	VU (Vulnerable)



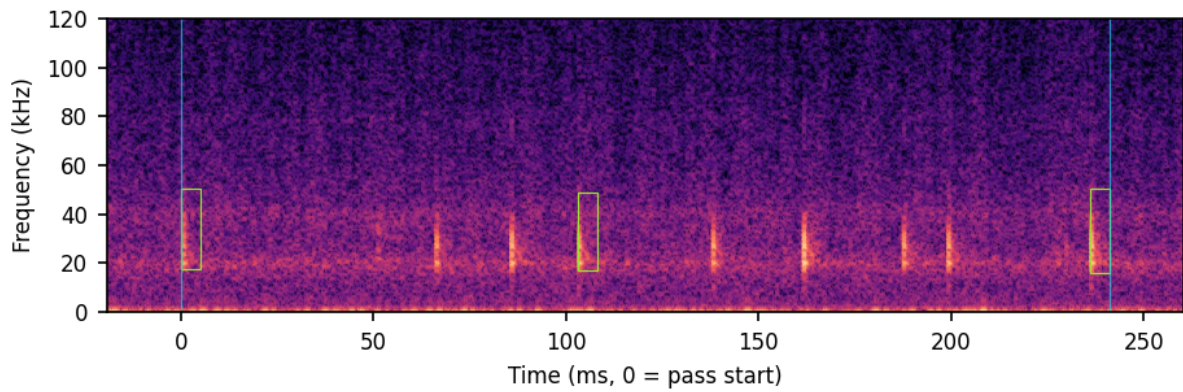
File	Audiomoths/20251015/Audiomoth3/20251014/243B1F026487A7FE_20251014_224500.WAV
Time	00:14.505 - 00:14.697 (192 ms)
Species	cneser - Serotine bat / Breitflügelfledermaus (<i>Eptesicus serotinus</i>)
Confidence	Classification 43% / Detection 58%
Frequency band	28.0 - 43.6 kHz
Swiss Red List	VU (Vulnerable)

F.4.7 Natterer's bat / Fransenfledermaus (myonat)

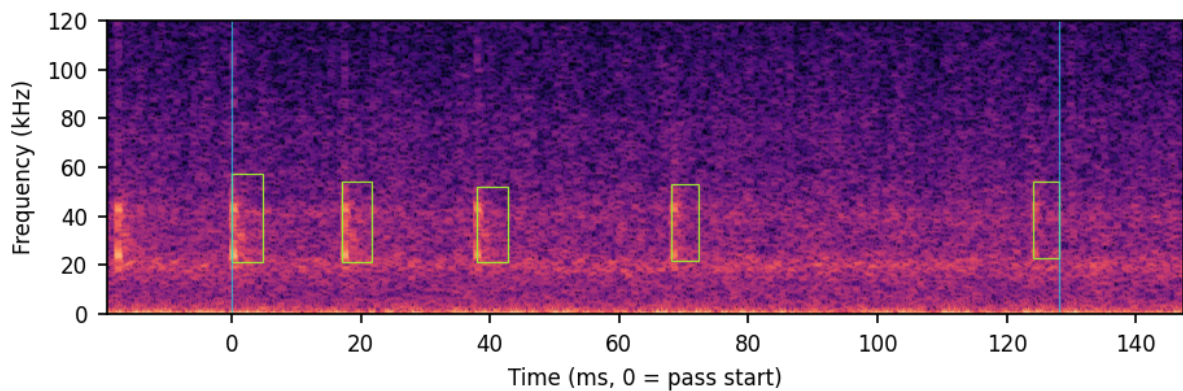
Myotis nattereri · Swiss Red List: NT · detected 43 times



File	Audiomoths/20251112/Audiomoth3/20251101/243B1F026487A7FE_20251101_014500.WAV
Time	04:42.654 - 04:42.787 (133 ms)
Species	myonat - Natterer's bat / Fransenfledermaus (<i>Myotis nattereri</i>)
Confidence	Classification 46% / Detection 51%
Frequency band	22.0 - 54.0 kHz
Swiss Red List	NT (Near Threatened)



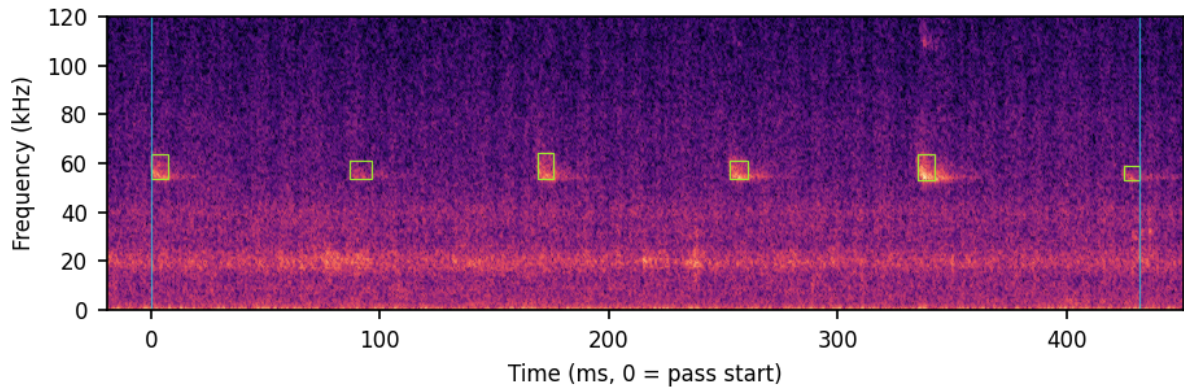
File	Audiomoths/20251015/Audiomoth1/20251013/243B1F026487A7CB_20251013_044500.WAV
Time	02:36.233 - 02:36.474 (241 ms)
Species	myonat - Natterer's bat / Fransenfledermaus (Myotis nattereri)
Confidence	Classification 41% / Detection 51%
Frequency band	16.0 - 50.4 kHz
Swiss Red List	NT (Near Threatened)



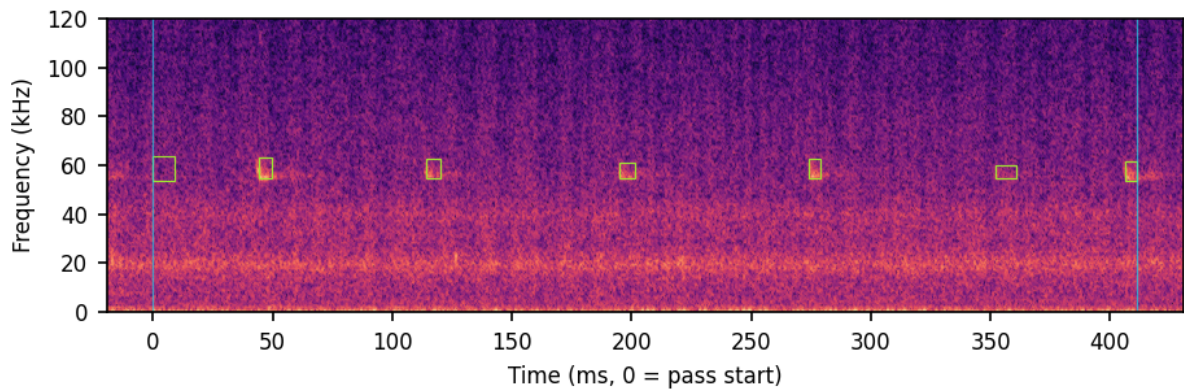
File	Audiomoths/20251015/Audiomoth3/20251014/243B1F026487A7FE_20251014_023000.WAV
Time	03:30.667 - 03:30.795 (128 ms)
Species	myonat - Natterer's bat / Fransenfledermaus (Myotis nattereri)
Confidence	Classification 40% / Detection 44%
Frequency band	21.2 - 57.6 kHz
Swiss Red List	NT (Near Threatened)

F.4.8 Soprano pipistrelle / Mückenfledermaus (pippyg)

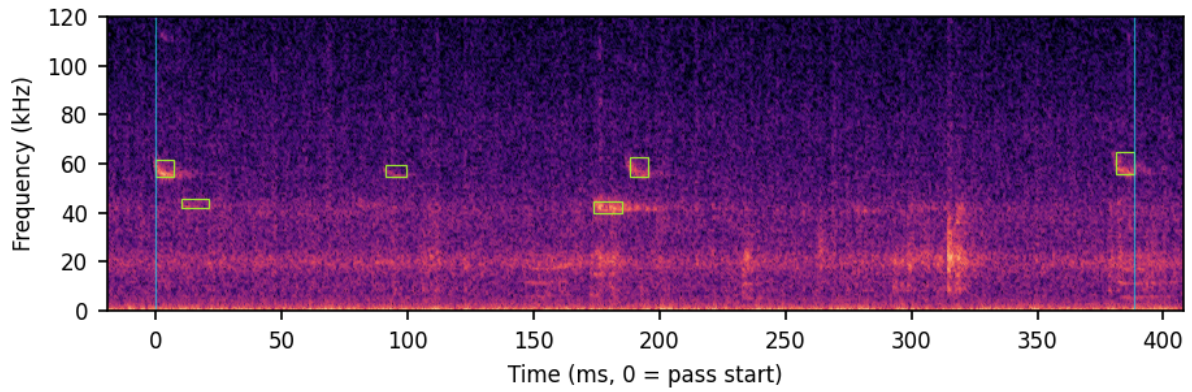
Pipistrellus pygmaeus · Swiss Red List: NT · detected 34 times



File	Audiomoths/20251007/Audiomoth1/20251005/243B1F026487A7CB_20251005_040000.WAV
Time	03:40.506 - 03:40.938 (432 ms)
Species	pippyg - Soprano pipistrelle / Mückenfledermaus (Pipistrellus pygmaeus)
Confidence	Classification 73% / Detection 75%
Frequency band	53.0 - 64.5 kHz
Swiss Red List	NT (Near Threatened)



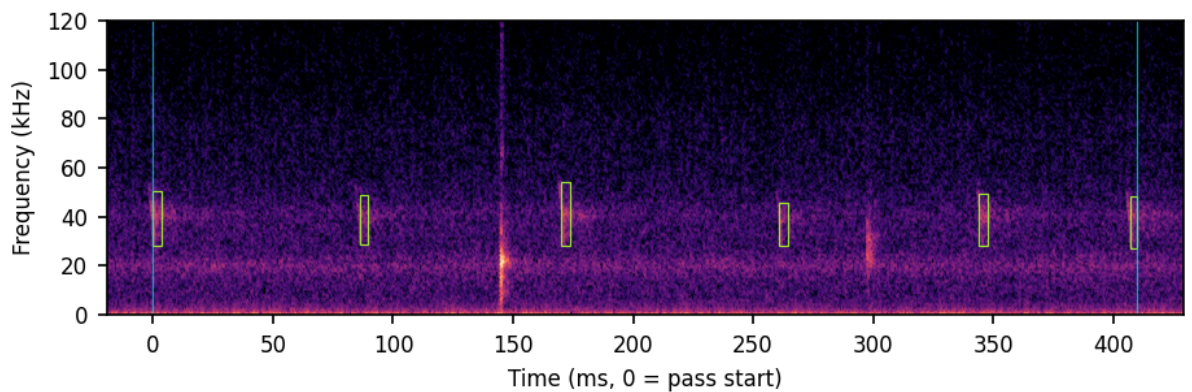
File	Audiomoths/20251007/Audiomoth1/20250930/243B1F026487A7CB_20250930_210000.WAV
Time	02:55.000 - 02:55.411 (411 ms)
Species	pippyg - Soprano pipistrelle / Mückenfledermaus (Pipistrellus pygmaeus)
Confidence	Classification 71% / Detection 72%
Frequency band	53.8 - 63.6 kHz
Swiss Red List	NT (Near Threatened)



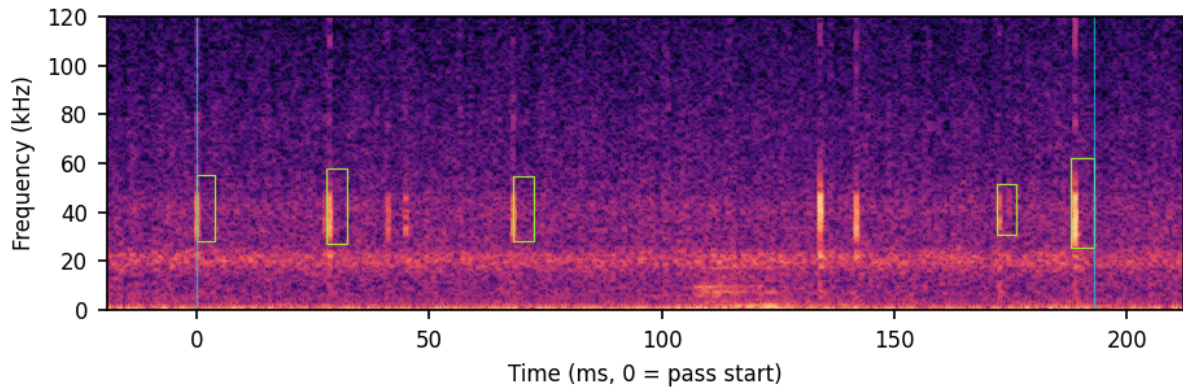
File	Audiomoths/20251007/Audiomoth3/20251005/243B1F026487A7FE_20251005_181500.WAV
Time	03:13.559 - 03:13.948 (389 ms)
Species	pippyg - Soprano pipistrelle / Mückenfledermaus (Pipistrellus pygmaeus)
Confidence	Classification 68% / Detection 71%
Frequency band	54.7 - 64.9 kHz
Swiss Red List	NT (Near Threatened)

F.4.9 Daubenton's bat / Wasserfledermaus (myodau)

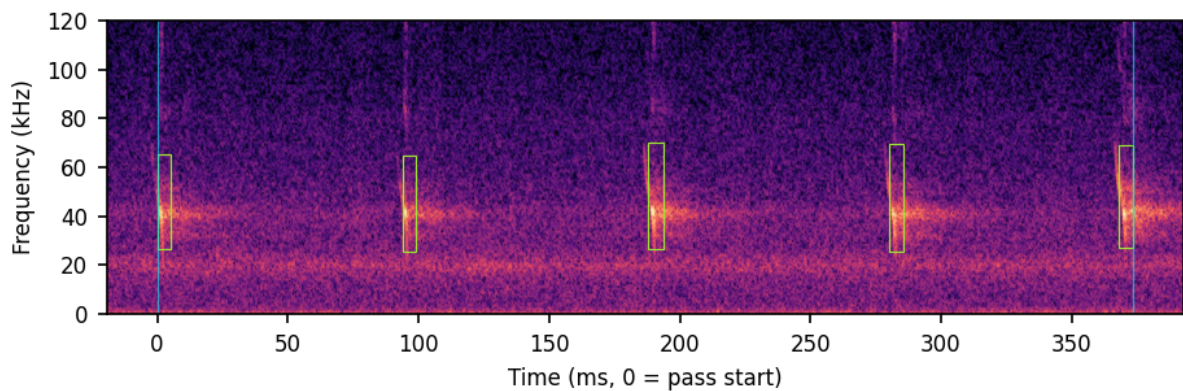
Myotis daubentonii · Swiss Red List: NT · detected 34 times



File	Audiomoths/20251007/Audiomoth3/20251005/243B1F026487A7FE_20251005_020000.WAV
Time	00:29.567 - 00:29.977 (410 ms)
Species	myodau - Daubenton's bat / Wasserfledermaus (Myotis daubentonii)
Confidence	Classification 50% / Detection 56%
Frequency band	27.2 - 54.0 kHz
Swiss Red List	NT (Near Threatened)



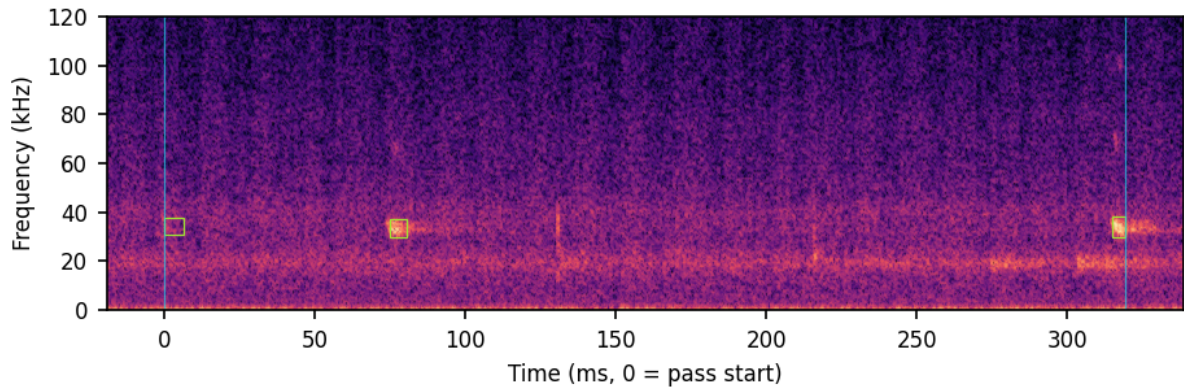
File	Audiomoths/20250923/Audiomoth3/20250921/243B1F026487A7FE_20250921_080000.WAV
Time	03:12.621 - 03:12.814 (193 ms)
Species	myodau - Daubenton's bat / Wasserfledermaus (Myotis daubentonii)
Confidence	Classification 46% / Detection 57%
Frequency band	25.5 - 62.3 kHz
Swiss Red List	NT (Near Threatened)



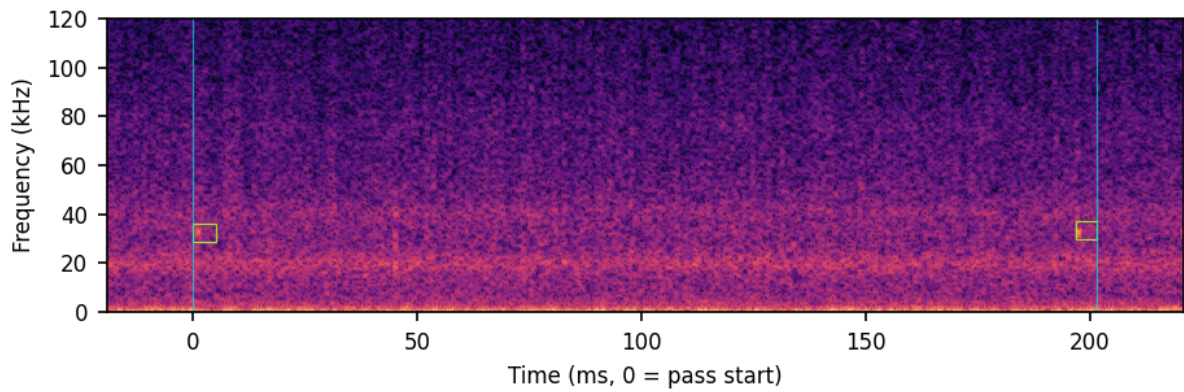
File	Audiomoths/20251015/Audiomoth2/20251014/248D9B016487A731_20251014_181500.WAV
Time	01:58.537 - 01:58.911 (374 ms)
Species	myodau - Daubenton's bat / Wasserfledermaus (Myotis daubentonii)
Confidence	Classification 46% / Detection 60%
Frequency band	25.5 - 70.1 kHz
Swiss Red List	NT (Near Threatened)

F.4.10 Western barbastelle / Mopsfledermaus (barbar)

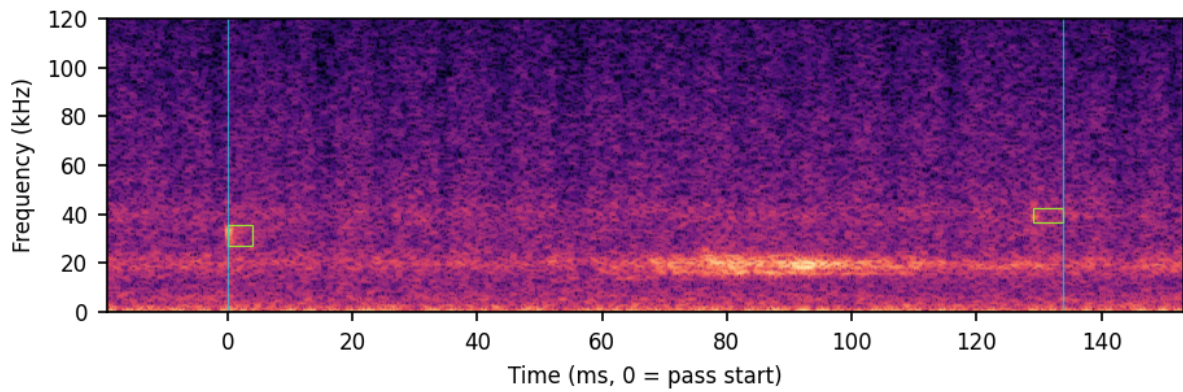
Barbastella barbastellus · Swiss Red List: EN · detected 14 times



File	Audiomoths/20251015/Audiomoth1/20251010/243B1F026487A7CB_20251010_190000.WAV
Time	02:43.131 - 02:43.451 (320 ms)
Species	barbar - Western barbastelle / Mopsfledermaus (<i>Barbastella barbastellus</i>)
Confidence	Classification 61% / Detection 69%
Frequency band	29.8 - 38.2 kHz
Swiss Red List	EN (Endangered)



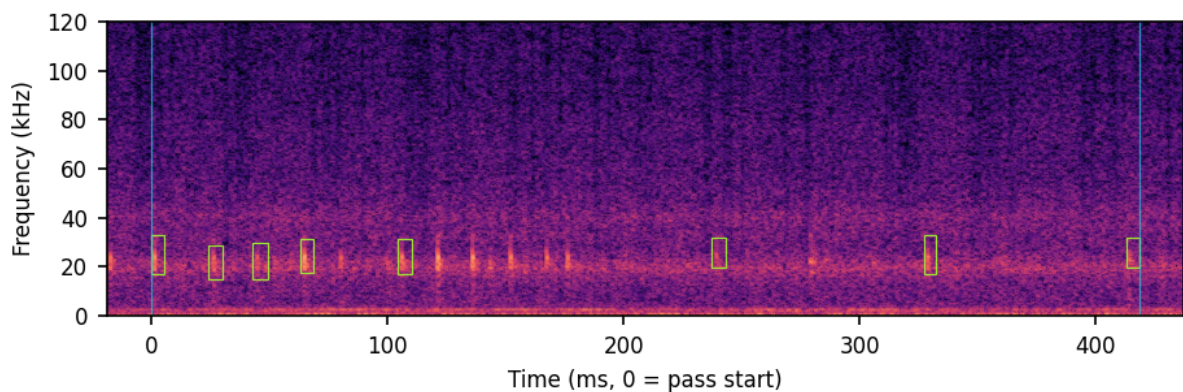
File	Audiomoths/20251007/Audiomoth3/20251007/243B1F026487A7FE_20251007_063000.WAV
Time	03:50.613 - 03:50.815 (202 ms)
Species	barbar - Western barbastelle / Mopsfledermaus (<i>Barbastella barbastellus</i>)
Confidence	Classification 54% / Detection 62%
Frequency band	28.9 - 37.1 kHz
Swiss Red List	EN (Endangered)



File	Audiomoths/20250923/Audiomoth4/20250922/242A260564877FAB_20250922_031500.WAV
Time	01:18.564 - 01:18.698 (134 ms)
Species	barbar - Western barbastelle / Mopsfledermaus (Barbastella barbastellus)
Confidence	Classification 49% / Detection 67%
Frequency band	27.2 - 42.6 kHz
Swiss Red List	EN (Endangered)

F.4.11 Brown long-eared bat / Braunes Langohr (pleaur)

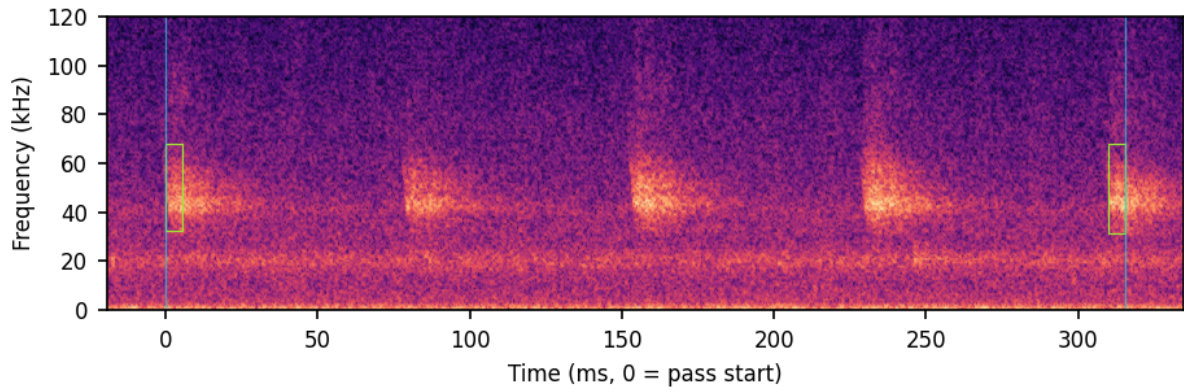
Plecotus auritus · Swiss Red List: VU · detected 13 times



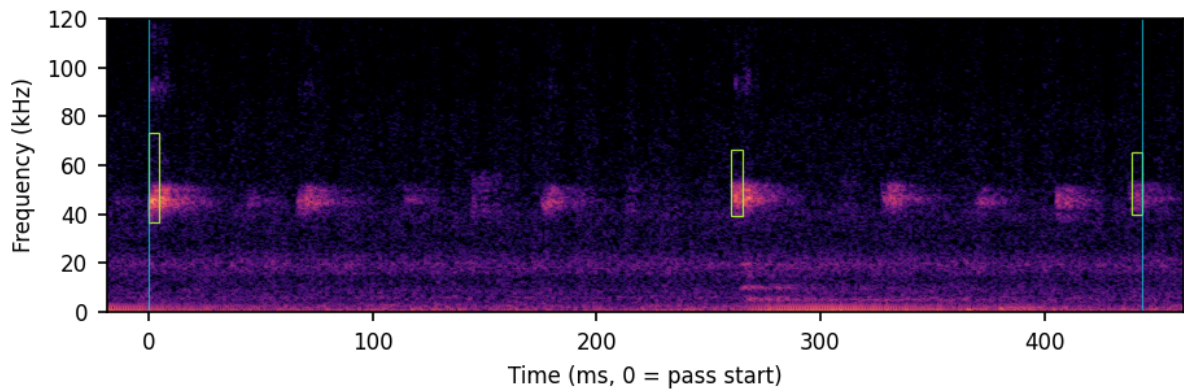
File	Audiomoths/20251112/Audiomoth4/20251103/242A260564877FAB_20251103_041500.WAV
Time	04:49.527 - 04:49.945 (418 ms)
Species	pleaur - Brown long-eared bat / Braunes Langohr (Plecotus auritus)
Confidence	Classification 32% / Detection 47%
Frequency band	15.2 - 33.1 kHz
Swiss Red List	VU (Vulnerable)

F.4.12 Brandt's bat / Grosse Bartfledermaus (myobra)

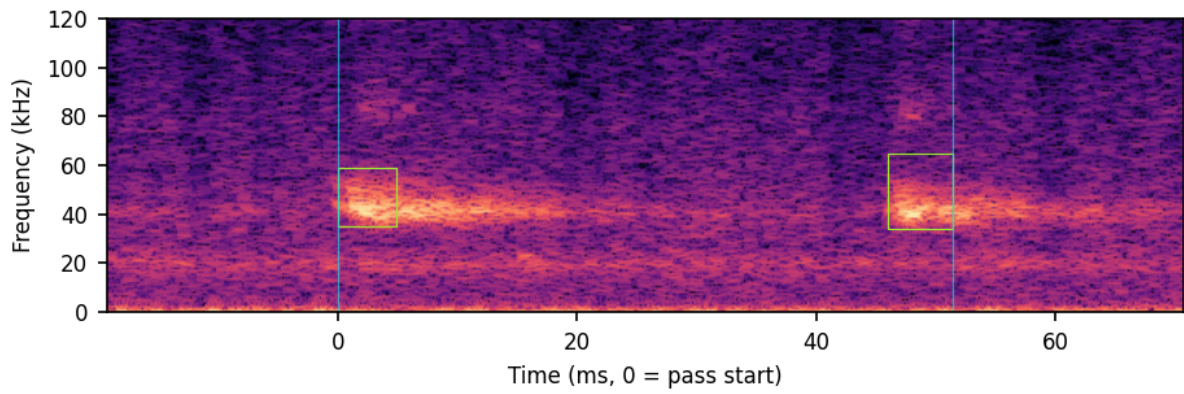
Myotis brandtii · Swiss Red List: VU · detected 7 times



File	Audiomoths/20251007/Audiomoth2/20251005/248D9B016487A731_20251005_000001.WAV
Time	00:23.149 - 00:23.464 (315 ms)
Species	myobra - Brandt's bat / Grosse Bartfledermaus (Myotis brandtii)
Confidence	Classification 26% / Detection 38%
Frequency band	31.5 - 68.1 kHz
Swiss Red List	VU (Vulnerable)



File	Audiomoths/20251028/Audiomoth4/20251017/242A260564877FAB_20251017_060000.WAV
Time	03:59.028 - 03:59.471 (443 ms)
Species	myobra - Brandt's bat / Grosse Bartfledermaus (Myotis brandtii)
Confidence	Classification 26% / Detection 34%
Frequency band	36.6 - 73.2 kHz
Swiss Red List	VU (Vulnerable)



File	Audiomoths/20251007/Audiomoth4/20251004/242A260564877FAB_20251004_210000.WAV
Time	04:17.509 - 04:17.560 (51 ms)
Species	myobra - Brandt's bat / Grosse Bartfledermaus (<i>Myotis brandtii</i>)
Confidence	Classification 24% / Detection 39%
Frequency band	34.1 - 64.7 kHz
Swiss Red List	VU (Vulnerable)

F.5 Original Assignment

Title	Automatic recognition of birds and bats in audio samples
Students	Studierende 1 und 2
Supervisor	Hannes Badertscher
Project partner	Paul Haverkamp, ILF Institut für Landschaft und Freiraum, paul.haverkamp@ost.ch
Topic	Artificial Intelligence

F.5.1 Description

On different locations on our campus in Rapperswil, Paul has installed audio-recorders. These audio-recorders regularly record audio samples of 5-minute length. If automatic detection and classification of bird or bat species were possible, an overview of campus wildlife could be obtained.

The audio-based survey is in addition to image based recognition of ground-based wildlife using camera traps.

The goal of this bachelor thesis is to build this audio processing software which should be able to:

1. Detect if birds or bats are present in a 5-minute audio sample
2. Distinguish between bats and birds and state which are present in an audio sample
3. Classify different bird species
4. (Optional) Classify different bat species
5. Give statistical overview of results

F.5.2 Tasks

- Research in audio processing based on spectrogram or raw audio. Gather information about existing bird-song classifiers (like BirdNet).
- Implement algorithms to detect if birds or bats are present in a 5-minute audio sample (and in which part of the 5-minutes)
- Extract the bird or bat songs, remove as much noise as possible
- In collaboration with the ILF, come up with a method to create a small ground-truth test set (e.g. labeling, download an existing, ...)
- Try existing bird-song classifiers
- Train or fine-tune machine-learning model to classify bird species. Use open source datasets (e.g. xeno-canto) to get around labeling a training dataset.
- (Optional) Train a machine-learning model to classify bat species
- Build an efficient audio processing pipeline to process large amounts of files in a suitable amount of time.
- Implement a demo software to visualize the results, consisting of a fast and reliable processing pipeline, user friendly enough to allow users to process new files on their own. Additionally, it should be possible to extract statistical properties from the results.

F.5.3 Expected Results

- Ground truth test dataset to validate the results
- Demo-software with the ability to
 - filter large amounts of audio files by searching for birds and bats
 - extract regions with birds or bats presence
 - identifying different species based on audio files

- visualize the results
- extract statistical properties (e.g. how often is a certain bird heard, ...)
- be used by non-technical users
- Documentation of the theory, studied approaches, chosen solution, and all developed hardware and software.
- A personal lab notebook (on paper or as PDF).

F.5.4 Additional Information

- You must write a lab notebook (on paper or digitally using a tool of your choice) where you write down, what you are working on, and what the current problems and results are. In the lab notebook, you also must keep a record of your working hours.
- A coach (experienced engineer from the ICAI team) will supervise your work and will help you with support and guidance.
- In weekly meetings with your coach, you discuss the current progress of your work, find solutions to problems you are facing, and define the next steps and tasks to perform in the next week.
- Every 4 weeks (or whenever required, e. g. for pre-defined milestones, or if important decisions must be made) we will set up a milestone meeting with me, your coach and, if required, the industry partner.
- The day before each meeting (deadline: 24:00), you send an email to your coach and me, containing a short summary of what you have been working on this week, what problems you are facing, and what the results are.