Machine Learning in Practice Report Template

Alessandro Vecchi (#s1158778), Lazar Đoković (#s1158770), Mauro Dìaz Lupone (#s1158767)

June 18, 2025

1 Introduction

Monitoring biodiversity in tropical ecosystems is extremely important for evaluating conservation efforts.

This project focuses on automated species identification from acoustic recordings in El Silencio Natural Reserve, using labeled and unlabeled audio from 206 species across birds, amphibians, mammals, and insects. The goal is to accurately predict species presence in 5-second segments of 1-minute test recordings.

We use spectrogram-based CNNs with state-of-the-art sound event detection methods, trained on both labeled and pseudo-labeled data. We use multiple strategies to improve generalization, including data augmentation, secondary label handling, semi-supervised learning via pseudo-labeling, curriculum training, and hard negative mining.

2 Method

Our primary objective was to develop a high-performing, interpretable model capable of detecting species from audio spectrograms under limited supervision and high class imbalance. We focused on building a strong single-model baseline to validate our preprocessing, feature extraction, and learning strategies before exploring ensembling and semi-supervised extensions.

This section outlines our full pipeline, covering audio preprocessing, model architecture, training strategies, augmentations, semi-supervised learning, and post-processing.

2.1 Audio Preprocessing and Spectrogram Extraction

The labeled data came from three sources (XC, iNat, CSA) with varying quality, so we first cleaned and standardized the audio while preserving diversity. XC samples were typically clean, while CSA often included human voices and noise. To support generalization to a different test distribution, we avoided over-cleaning.

Audio was resampled to 32 kHz and split into 5-second segments with 50% overlap. Each segment was converted into a log-mel normalized spectrogram using 128 mel bands, a 1024-sample STFT window, which spans $1024/32000\text{s} = 32\text{ms}$, and a 512-sample hop (16 ms overlap). These parameters were chosen after manual inspection of species vocalizations, which typically lasted between 10 ms and several hundred milliseconds. The chosen settings provided sufficient temporal and spectral resolution to capture harmonics without exceeding memory constraints.

To reduce noise and irrelevant information, we applied:

- **Bandpass filtering** (50–16000 Hz): To remove microphone noise and retain high-frequency insect sounds, while skipping high-frequency artifacts.
- **Voice Activity Detection (VAD)**: Based on the Silero VAD model, with adaptive thresholds per collection (stricter for CSA, lighter for XC) to filter human speech and silence.
- **Silence filtering with RMS**: Only segments above the first quartile in root mean square (RMS) energy were retained, improving signal quality without excessive data loss.
- **Cycling padding**: Applied to clips shorter than 5 seconds to maintain temporal consistency.

A different sampling strategy was applied for each audio file based on the collection weight, the class weight, the number of samples of the given species present in the dataset and the audio quality. The resulting spectrograms were computed in parallel and stored as `.npy` arrays for efficient disk-based training.

For implementation details, refer to the preprocessing notebook in Section 6.

2.2 Base Model: EfficientNet with SED Head

We use a convolutional architecture based on EfficientNet-B0 as our primary model, chosen for its balance between performance and computational efficiency in a CPU-only setting. In our ensemble, we include two attention-based variants where the original classifier is replaced with a custom Sound Event Detection head. One model applies frequency-wise attention to emphasize important mel bands, while the other uses time-wise attention to retain temporal information. These are followed by a convolutional projection to 206 output classes, adaptive max pooling, and flattening to produce multi-label logits.

What's the idea behind it? The CNN feature extractor processes log-mel spectrograms into 4D feature maps (batch size, channels, frequency, time), preserving both time and frequency structure. Aggregating only along the frequency or time axis allows us to retain the corresponding information in that feature map, either **when** a species vocalizes or **at what frequency**. A standard CNN classifier is also included in the ensemble. We highlight the subtle yet complementary differences between the time and frequency attention models. Their visual behavior is illustrated in Figure 4.

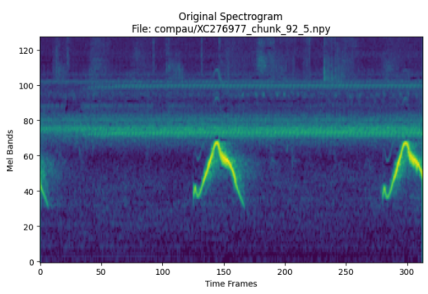

Figure 1: *
(a) Original Spectrogram

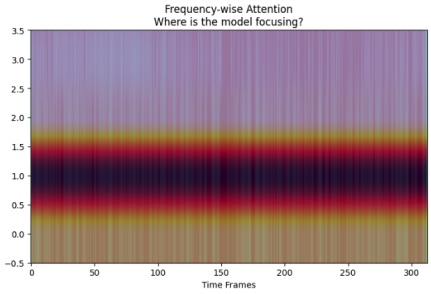

Figure 2: *
(c) Frequency-wise Attention

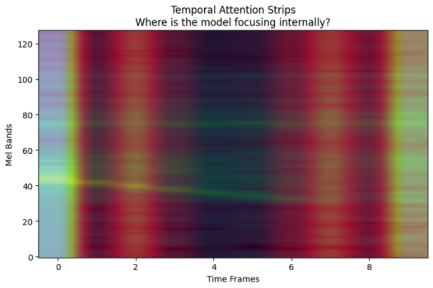

Figure 3: *
(b) Time-wise Attention

Figure 4: Comparison between original spectrogram and the learned attention maps in the time-wise and frequency-wise SED models. These illustrate how the model focuses either on when a vocalization occurs or at what frequency, depending on the aggregation strategy. See Section 6 for architectural details.

2.3 Training Setup and Augmentation

To handle the highly imbalanced, multi-label species data, we trained models using a custom hybrid loss combining Binary Cross-Entropy (BCE) and Focal Loss with $\gamma = 2.0$, which helped mitigate the effect of easy negatives and class imbalance. Secondary species labels were down-weighted (0.5) since they were consistently less present than the primary species. The final loss is a linear combination of all.

Training used the `AdamW` optimizer with a two-stage learning rate schedule: a warm-up phase with a modified `GradualWarmupScheduler`, followed by cosine annealing. This stabilized early training and enabled efficient learning rate decay. To improve robustness and adapt to domain shift, we applied spectrogram-based data augmentations:

- **1D Audio Augmentations (pre-spectrogram)**: **Mixup**: Combines two samples and interpolates both features and labels using a Beta distribution ($\alpha = 0.4$), implemented within a custom `collate_fn` to preserve batch-wise consistency.
- **2D Spectrogram Augmentations (post-spectrogram)**:
  - **Time masking**: Randomly masks vertical slices (time) in the spectrogram.
  - **Frequency masking**: Randomly masks horizontal slices (mel bands).
  - **Random erasing**: Removes rectangular regions to simulate local occlusions or clipping.

These augmentations are applied conditionally by class (e.g., time masking is emphasized for amphibians, frequency masking for insects), as defined in a custom `SpectrogramAugmentor` class (see `training_BirdCLEF2025.ipynb` 6). This strategy helped target augmentation toward known signal characteristics of different taxonomic groups. We employed `torch.amp.GradScaler` for mixed-precision training.

To utilize the unlabeled soundscapes, we implemented a pseudo-labeling pipeline: soft predictions from a trained model were thresholded and used as training targets for spectrograms of unlabeled audio. This augmented dataset improved generalization in subsequent training rounds.

Curriculum learning was employed by starting with high-confidence samples and gradually introducing noisier data, easing optimization. Hard negative mining was used to over-sample examples often misclassified as positives, helping reduce false positives.

Final predictions are made by ensembling models with weighted averages. A simple but effective post-processing idea is introduced: Each test file is divided into 12 non-overlapping 5-second segments, and we need to predict a label for each 5-second chunk. Since it's likely that a single species will dominate the whole soundscape file, we compute the average predicted probability for each species across all 12 segments of a soundscape. Each individual segment prediction is then reweighted by multiplying it with the corresponding mean species probability over the full file. Formally, let $p_{i,j}$ be the predicted probability of species j in segment i of soundscape s. We compute:

$$\hat{p}_{i,j} = p_{i,j} \cdot \frac{1}{12} \sum_{k=1}^{12} p_{k,j}$$

The idea behind this is that the correct species should appear more than once in the recording, and this pumping up of the correct species should correct predictions of the model on the noisy chunks of the test file.

3 Results

Model performance was evaluated primarily via the public leaderboard, as traditional cross-validation was unreliable due to extreme class imbalance and underrepresented species. Over 50 submissions were made during development, each testing specific pipeline modifications, with an initial focus on optimizing a single architecture before progressing to ensembling.

Single-Model Performance: SED architectures outperformed standard CNN classifiers by over 7% in macro-averaged ROC-AUC. Among SED variants, the frequency-wise attention model consistently performed best, achieving our top single-model public score of **0.814**.

Data augmentation was key to performance gains. Mixup notably improved generalization by simulating multi-species conditions, which appaerently were more representative of the test soundscapes, while CutMix showed no benefit and was eventually excluded.

Loss experiments showed our weighted Focal Loss outperformed BCE, boosting leaderboard score by over **0.5 points**. VAD didn't improve scores but enhanced CNN feature localization per Grad-CAM analysis (see Nb 6).

Architectural and Hyperparameter Exploration: EfficientNet-B0 and B3-pruned variants performed best (we only tried EfficientNet models). Spectrogram hyperparameters—especially mel bands, FFT size, and hop length—greatly affected results; altering them caused sharp drops, highlighting the need for careful audio front-end design. Oversampling rare species or changing sampling distributions reduced the performance but cut training time significantly.

Semi-Supervised Learning and Curriculum Training: Curriculum training and hard negative mining underperformed, lowering the public score to **0.807**. This was due to low model confidence on many pseudo-labeled samples—especially rare classes—limiting usable data per phase (see Nb 6). Only 11,000 pseudo-labels passed thresholds (0.25 hard, 0.05 soft), with skewed class coverage. Overall, 223,698 pseudo-labeled samples averaged 21.94 labels each; 509 were hard, 10,459 soft, and 92,032 had all-zero activations.

Manual review showed that many of these recordings were too ambiguous for reliable pseudo-labeling. Birds were the most frequent pseudo-labeled species, while underrepresented species remained missing due to the inability of the model to discern them properly. To mitigate this problem, hard negative mining was used, but it did not yield the expected results, likely because improved species were not present enough at test time or their distribution was completely differed a lot from the train audios data.

Ensembling and Postprocessing: While ensembling multiple models didn't improve public leaderboard score, it improved the private scores and thus robustness. By leveraging temporal coherence, the post-processing step reduced noisy predictions and consistently improved performance.

4 Discussion

A deeper analysis of model predictions was performed in `predictions_understanding.ipynb` 6, using metrics like accuracy, precision, and recall alongside macro-averaged AUC. Notably, some species had near-perfect AUC but

very low precision and recall, highlighting a key limitation of relying solely on AUC: models may rank samples well without making confident, correct predictions—harmful in imbalanced multi-label settings.

Model Explainability and Grad-CAM Analysis: To better understand model decisions, we used Grad-CAM to visualize attention in CNN layers. Early layers highlighted relevant acoustic features, such as harmonics and modulation patternsHowever, by the final convolutional layer, the model often lost any track of what we humans would define relevant in that spectrogram to detect the species. This "unlearning" effect suggests that deeper layers might have overfitted to spurious cues, especially in underrepresented and noisy recordings (see Fig. 5 for examples).

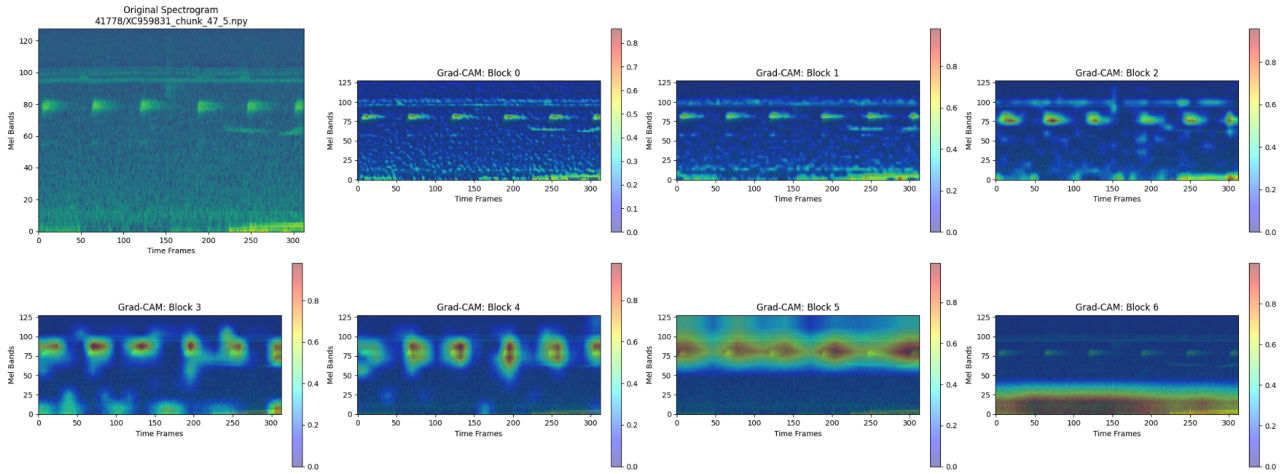

Figure 5: **Grad-CAM visualization across convolutional blocks of the EfficientNet-B0 model.** Top-left panel shows the original log-mel spectrogram of a 5-second audio chunk. Subsequent panels visualize Grad-CAM heatmaps from each convolutional block (Blocks 0–6). Early layers (Blocks 0–2) highlight fine-grained spectral structures such as harmonics and localized modulations. Middle layers (Blocks 3–4-5) show focused activation on salient species cues. However, the last layer (Blocks 6) focus on a completely different portion of the spectrogram.

Architectural Insights: Attention visualizations and prediction patterns suggest the final convolutional block of EfficientNet may cause overfitting or feature degradation. Removing or replacing it could preserve better features and improve generalization, likely explaining why SED models outperformed standard CNNs.

Error Analysis by Species: Manual review of misclassified species revealed specific failure modes. For instance, `turvul` (Turkey Vulture) showed low precision (0.2) and accuracy (0.05) despite ample training data. Upon listening, we observed that the call types associated with this bird vary widely (e.g., wing beats, nestling sounds, alarm calls), with very different acoustic profiles. These intra-class variabilities were not distinguishable through spectrograms alone, likely leading to confusion. Including the call type as metadata input may have mitigated this effect. We did not discover it early enough to try it, though. Another key finding was that some correctly predicted rare species resulted from memorization rather than genuine learning. In pseudo-labeled data, predictions for underrepresented species matched overfit training patterns instead of consistent acoustic features.

Class Confusion and Bird Misclassifications: Many misclassifications involved birds, despite their dataset dominance. This likely stems from birds often co-occurring and having similar spectrogram features. MixUp helped by exposing the model to mixed-species spectrograms, but additional methods must be applied for better disambiguation.

5 Conclusion

This project demonstrated a comprehensive pipeline for species classification from audio in a real-world biodiversity monitoring context. By combining spectrogram-based CNNs, attention-based SED models, and targeted augmentations, we achieved strong single-model performance (AUC 0.814) under challenging conditions of label imbalance and acoustic noise. While ensembling and post-processing improved robustness, semi-supervised methods like curriculum training and hard negative mining were less effective due to low model confidence on ambiguous samples. Grad-CAM visualizations and manual analysis provided valuable insights into model behavior and failure modes. Our approach prioritized interpretability and scientific understanding over brute-force ensembling. The insights and tools developed here can support future eco-acoustic research and monitoring systems at scale.

6 Code Notebooks

The project was implemented using a modular set of well-documented Jupyter notebooks, each focused on a specific pipeline component. Below is a summary of the notebooks and their roles, each identified with a label for reference throughout the report:

- `EDA-BirdCLEF2025.ipynb` : Contains the exploratory data analysis. Examines label distributions, quality differences across the three data collections (XC, iNat, CSA), and audio and signal characteristics.
- `precomputing-spectrograms.ipynb` : Implements the preprocessing pipeline, including 5-second chunking, log-mel spectrogram generation, voice activity detection (VAD), silence and RMS-based filtering, normalization, and storage.
- `training_BirdCLEF2025.ipynb` : Contains the full training pipeline. Defines dataset and dataloader classes, integrates multiple augmentation strategies, and supports various model architectures with custom loss functions (e.g., Focal Loss with secondary label handling).
- `predictions-understanding.ipynb` : Analyzes trained model predictions in depth. Includes accuracy, precision, recall, and AUC metrics, false positive/negative tracking, Grad-CAM visualizations across model layers, attention maps, and prediction behavior across time and frequency.
- `pseudo-labeling-birdclef2025.ipynb` : Automates the pseudo-labeling process. Applies a trained model to the unlabeled soundscapes to generate soft targets, extracts spectrograms from confident predictions, and saves them for use in semi-supervised training.
- `curriculum-training_hnm_BirdCLEF2025.ipynb` : Implements curriculum learning and hard negative mining. Sorts training samples by prediction entropy or signal quality and gradually introduces more ambiguous examples while emphasizing frequently misclassified negatives.
- `inference-BirdCLEF.ipynb` : Handles final model inference. Applies ensembling across multiple model architectures, post-processing and prepares the output submission file.

7 Generative A.I. Acknowledgement

ChatGPT4 was used in the writing of the report. The main reason behind its use, was the page constraint. The first draft of the report was more than 6 pages long(figures excluded), and we had to find a way to reduce the text without losing too much substance. Being non-native speakers, we feel like it is a fair use of the instrument in question. A good way to learn how to be rephrase sentences while keeping the meaning the same. However, for a project this big, we honestly feel like 4 pages are not enough, and even with the help of GPT, we had to cut off several interesting figures and remove a couple of thought processes that explained why we thought of implementing some of the features. There was simply not enough space. Please, next time move the figures outside of the page limit.

8 Author Contributions

This project was a collaborative effort, with all team members contributing to discussions, literature review, and problem-solving. The following summarizes the primary areas each member focused on:

- **Alessandro Vecchi:** Focused on audio preprocessing, designed the custom loss function and the spectrogram augmentation framework. Conducted output analysis and error interpretation, and authored the inference notebook.
- **Lazar Đoković :** Led exploratory data analysis (EDA), implemented the model architectures, and developed the dataset and data loader modules. Also implemented the pseudo-labeling pipeline.
- **Mauro Dìaz Lupone:** Built and managed the training pipeline, and implemented curriculum learning and hard negative mining. Contributed significantly to reviewing related literature and papers.

Throughout the project, all members actively participated in reading academic papers, exploring the Kaggle discussion forums, and collectively refining the pipeline through regular meetings and joint debugging sessions.

9 Evaluation of the Process

Reflecting on the overall process, we are very happy with how the project unfolded and how we managed to approach each challenge. Above all, we had fun working on it (much better than the previous one, in all honesty). It was an exciting project with real-world impact, and it was our first experience working with audio data. Being able to visually represent audio and build a classifier that could extract meaningful features from it to predict species was mindblowing at first. Throughout the project, we made sure to dive deep into the data and understand the reasoning behind the model's predictions. While we conducted numerous intriguing experiments and discussions, we unfortunately couldn't include all of them here in the report due to space limitations.

In terms of leaderboard performance, we don't really mind not being in a top position. As far as we saw, a very easy way to get to a good score was by ensembling A LOT of models and average in some way their predictions. Some colleagues got a public score near 0.85 even with models with individual scores around 0.6 or 0.7. Also, we read about TTA everywhere in the discussion forum, but deemed it to be an unoriginal idea and chose not to pursue it. However, apparently it was an easy improvement for any pipeline.

After the private leaderboard was updated, we reviewed implementations from top performers and saw that their success came largely from massive ensembles of models trained on multiple seeds, different cross-validation folds, extensive hyperparameter optimization, and even manual audio cleaning. Some even used data from previous competitions. By combining models with individual scores between 0.8 and 0.84, they were able to achieve scores above 0.9 on the public leaderboard. The more models in the ensemble, the more stable and robust the score became.

However, after this analysis, we feel even more content with how we approached the project. Our goal was never to top the leaderboard, but rather to learn as much as possible while enjoying the process. Training 50 models with different seeds and cross-validation folds doesn't sound like a fun way of spending the not-much time we could dedicate to the project. Instead, we focused on understanding why the model was behaving in a certain way and discussed how we could improve that behaviour. We focused on key elements such as the importance of preprocessing, the impact of augmentations, and the development of innovative techniques like our custom focal loss and post-processing strategy. We also explored semi-supervised learning strategies, including curriculum training and hard negative mining, which offered valuable insights into the model's actions.

Moreover, we took the time to explore the Sound Event Detection (SED) architecture, implementing from scratch the simplest SED models to understand how it learned differently from traditional CNN classifiers. This exploration provided us with deeper insights into the unique challenges and strengths of audio classification. Ultimately, we are proud of the balance we struck between innovation and practical problem-solving, and we believe that this project has been a rewarding learning experience.

10 Evaluation of the Supervision

Due to overlapping deadlines with other major projects, we were only able to attend a limited number of tutorial sessions. The early meetings were somewhat less helpful—not due to a lack of guidance, but because we were still in the exploratory phase and lacked a clear sense of direction. At that stage, it was difficult for both us and the instructors to assess whether our ideas were promising and why the model was not working as expected.

However, the supervision we received later in the project, particularly during the final weeks of May, proved far more impactful. By that point, we had concrete results and well-formed ideas, which made the feedback from the teaching staff significantly more actionable. Seeing genuine interest and enthusiasm from the supervisor during these sessions also gave us additional motivation and confidence in our work.

One particularly insightful suggestion in this last meeting—integrating metadata such as call types into the training process—was discussed during supervision but initially set aside, as we didn't yet have a compelling reason to implement it. Ironically, that reason emerged on the very last day, during the bird misclassification analysis. In hindsight, we should have trusted the suggestion earlier. Lesson learned!

Overall, while we could have benefited from more structured feedback earlier on, the support we received later in the project was highly valuable and helped shape our final approach.