Bird-Species Audio Identification, Ensembling of EfficientNet-B0 and Pre-trained EfficientNet-B1 model

Notebook for the <Cornell Lab of Ornithology> Lab at CLEF 2024

Aaditya Porwal[1]

[1]*Indian Institute of Technology, Dhanbad, India*

Abstract

In this study, I present a novel approach to audio classification, specifically for the BirdCLEF 2024 challenge, by employing an ensemble of EfficientNet models. My methodology integrates EfficientNet-B0, trained exclusively on the current competition's data, and EfficientNet-B1, pre-trained on datasets from previous BirdCLEF competitions. The EfficientNet-B0 model leverages heavy augmentation techniques to enhance generalization and robustness. Data preprocessing involves transforming audio signals into Mel spectrograms, optimized through feature engineering and augmentation methods. The ensemble strategy, combining predictions from both models, achieves superior performance compared to individual models. My results demonstrate the efficacy of this approach, with significant improvements in classification accuracy and robustness, exemplified by achieving the 25th rank out of 975 competitors on the BirdCLEF 2024 leaderboard.

Keywords

Deep Learning, Bird Species Classification, Transfer Learning, Attention Mechanism, Sound Detection, Audio Source Detection, EfficientNet, Ensembling, Audio Classification, BirdCLEF, Ensemble Learning, Data Augmentation, Mel Spectrogram, Convolutional Neural Network, Feature Engineering, ROC-AUC

1. Introduction

There are about 10,000 different bird species in this world, and they all play an important role in the natural world. Birds are excellent indicators of biodiversity change since they are highly mobile and have diverse habitat requirements. BirdCLEF 2024 [1] is a Kaggle competition organized by The Cornell Lab of Ornithology in collaboration with LifeCLEF 2024 [2], whose challenge is to identify which birds are calling in long recordings, given training data generated in meaningfully different contexts. The BirdCLEF 2024 competition focuses on identifying bird calls in long recordings, particularly from the sky-islands of the Western Ghats. This competition presents significant challenges, including imbalanced training data per species, domain shifts between training and test data, and a strict two-hour time limit for analyzing extensive recordings.

This paper is structured to first provide details of the competition and the given data to ensure a clear understanding of the challenges posed by the train and test data. Additionally, I will provide a detailed solution to the approaches used for this challenge, including data preparation, approach, augmentations, model building, training procedures, post-processing techniques, and conclusion. If successful, this effort will advance ongoing initiatives to protect avian biodiversity in the Western Ghats, India.

2. Data

2.1. Training Data

- **Train audio:** The bulk of the training data consists of short recordings of individual bird calls from xeno-canto.org. These files have been downsampled to 32 kHz where applicable to match the test set audio and converted to the ogg format. Information of 182 unique species has been given.

CLEF 2024: Conference and Labs of the Evaluation Forum, September 09–12, 2024, Grenoble, France

aadityaporwal234@gmail.com (A. Porwal)

https://github.com/AADI-234 (A. Porwal)

- **Train metadata:** Along with audio files, metadata is also provided which consists of primary label, secondary labels, type, latitude, longitude, scientific name, common name, author, filename, license, rating, and URL.
- **Unlabeled soundscapes:** Unlabeled audio from the same locations as the test soundscapes is provided.
- **eBird_Taxonomy_v2021.csv:** Contains data on species relationships.

2.2. Test Data

Test_soundscapes: The test_soundscapes directory will be populated with approximately 1,100 recordings to be used for scoring. They are 4 minutes long and in ogg audio format.

3. My Approach

My final approach to this dataset was to ensemble two different EfficientNets (B0 and B1) [3]. The B1 model [4] was conditioned on data from previous years BirdCLEF competitions during its pretraining stage, while the B0 model [5] was trained using only this competition's data.

4. EfficientNet-B0 using heavy augmentation

4.1. Overview

The task of audio classification involves identifying and categorizing audio signals into predefined classes. In this project, I employed EfficientNet-B0, a highly efficient convolutional neural network [6], to classify audio signals. EfficientNet-B0 is chosen because of its balance of performance and computational efficiency.

The audio data is preprocessed and transformed into Mel spectrograms, which are fed into the EfficientNet-B0 model. The model is trained using a variety of data augmentation techniques to enhance generalization and robustness. The classification performance is optimized through advanced feature engineering, cross-validation, and careful selection of hyperparameters.

4.2. Data Preparation

The audio data used in this project is sampled at a rate of 32,000 samples per second (sr = 32000). Each audio clip for training has a duration of 30 seconds. To handle the diverse length of audio files, a fixed length of 30 seconds is set for all training samples. From these 30-second clips, random 5-second segments are extracted for Short-Time Fourier Transform (STFT) processing [7]. For testing, a uniform duration of 5 seconds is used. More details are provided in 1.

Table 1
Audio data parameters and spectrogram configuration

Parameter	Value
Sample Rate	32000
Clip Duration	30 sec
STFT Segment	5 sec
Frequency Range	20 Hz - 15,000 Hz
Mel Bands	128
FFT Components	1024
Spectrogram Time Axis	512

4.3. Feature Engineering

- Feature engineering focuses on transforming raw audio data (acoustic signal) into a format suitable for model training. Mel spectrograms are generated from the audio signals, converting them into a visual representation of frequency content over time. This transformation is achieved using Short-Time Fourier Transform (STFT) [8], where each 5-second audio slice undergoes Fourier transformation to capture the spectral properties. An illustration of this process can be seen in Figure 1.
- Additionally, various augmentation techniques are employed to enhance the training data. Spectrogram-specific augmentations like masking and coarse dropout [9] are used, further diversifying the training data. Mixup augmentation [10], both in waveform and spectrogram forms, combines multiple examples to create synthetic training samples, enhancing the model's robustness and generalization capabilities.

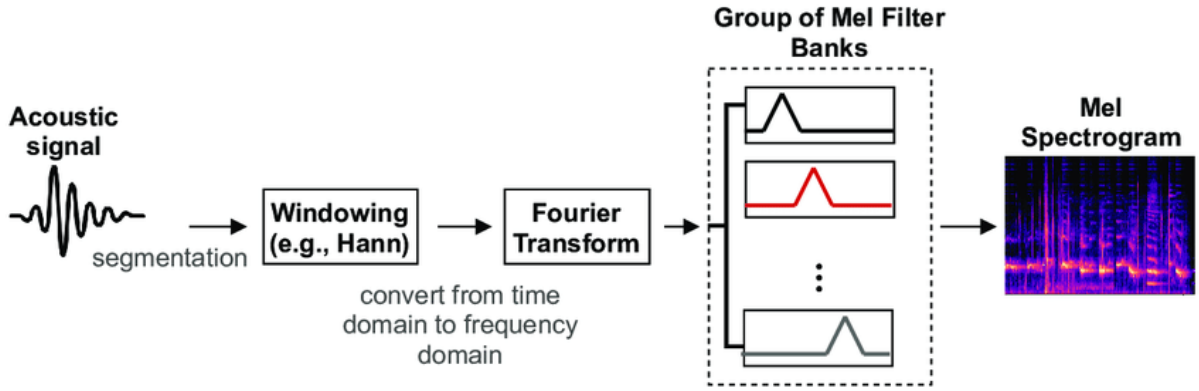

Figure 1: The process of extracting the Mel spectrogram from an acoustic signal. Source: ResearchGate

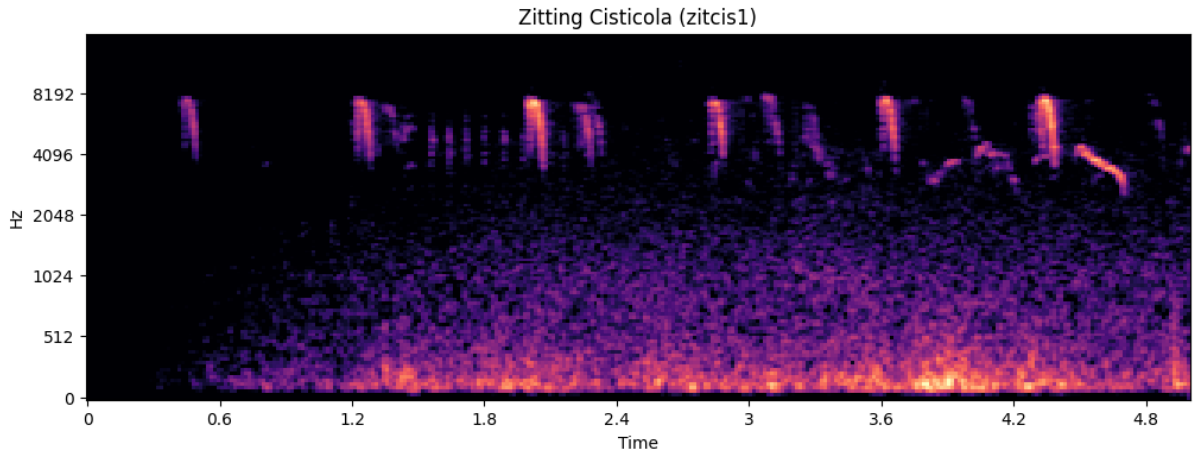

Figure 2: Logarithmic magnitude spectrogram showing frequency content over time using STFT. Source: Kaggle

4.4. Model Building

The model architecture is based on EfficientNet-B0. It incorporates average pooling (pool_type = 'avg') and utilizes Binary Cross-Entropy with Focal Loss (loss_type = "BCEFocalLoss") to address class imbalance. More details are provided in Table 2.

Table 2
EfficientNet-B0 model configurations and training parameters

Component	Configuration
Model	EfficientNet-B0
Pooling	Average Pooling
Loss Function	Binary Cross-Entropy with Focal Loss
Optimizer	adan
Learning Rate	1.0e-03
Weight Decay	1.0e-02
Early Stopping	Patience = 5 epochs
Mixed Precision	Automatic Mixed Precision (AMP) enabled

4.5. Augmentation

Data augmentation plays a crucial role in improving the model's generalization. Various augmentation techniques are employed to introduce variability in the training data. These include:

- **Waveform Augmentation**:
  - **Random Noise Addition**: This technique involves adding random noise to the audio signal to simulate real-world audio conditions and improve the model's robustness to background noise.
  - **Gain Adjustments**: Adjusting the volume levels to handle recordings with different gain levels. This helps the model generalize better across recordings with varying loudness.
  - **Pitch Shifting**: This technique was considered but ultimately set to 0 because it disrupted the distinct frequency patterns critical for bird call classification.
  - **Time Shifting**: Similarly, time shifting was considered but set to 0 as it disrupted the temporal patterns essential for accurate classification.
- **Spectrogram Augmentation**:
  - **Masking Parts of the Spectrogram**: Techniques like spectrogram masking were employed to hide parts of the spectrogram, making the model more robust to missing data. However, this did not show significant improvements, due to the distinct and critical nature of the bird call frequency patterns.
  - **Randomly Dropping Coarse Regions**: This technique, known as coarse dropout, was used to randomly drop regions of the spectrogram. It aimed to make the model more robust, but preliminary experiments showed limited effectiveness.
  - **Horizontal Flipping**: Not used because flipping the time axis of a spectrogram is not meaningful for audio data and would disrupt the temporal sequence of the sound.
- **Mixup Augmentation**: Both waveform and spectrogram mixup techniques were employed. This involves linearly combining two examples to create a new synthetic example, enhancing the model's robustness and generalization capabilities. This was controlled by parameters such as:
  - **Waveform Mixup (aug_wave_mixup)**: Set to 1.0. This parameter indicates that waveform mixup was applied to all audio samples during training. Waveform mixup involves combining the waveforms of two different audio samples by taking a weighted average of their amplitudes.
  - **Spectrogram Mixup (aug_spec_mixup)**: Set to 0.0. This parameter indicates that spectrogram mixup was not applied to the spectrograms during training. Spectrogram mixup would involve combining the spectrogram representations of two different audio samples in a similar manner to waveform mixup.

– **Probability of applying Spectrogram Mixup (aug_spec_mixup_prob)**: Set to 0.5. This parameter defines the probability with which spectrogram mixup would be applied to an audio sample during training. Even though spectrogram mixup was set to 0.0 in this case, this parameter would control the likelihood of its application if it were used.

The mix ratio for combining the samples was determined by a Beta distribution with $\alpha = 0.95$. The Beta distribution is commonly used in mixup techniques to control the interpolation between two samples. A Beta distribution with $\alpha = 0.95$ produces mix ratios that are generally close to 0 or 1, meaning that the synthetic samples are predominantly composed of one of the original samples, with only a small contribution from the other. This helps maintain the distinct features of each sample while still providing the benefits of data augmentation.

4.6. Training Procedure

The training procedure involves cross-validation, early stopping, and augmentation strategies. The training configurations are provided in Table 3.

Table 3
Training procedure parameters and configurations

Parameter	Configuration
Cross-Validation	5 folds
Training Epochs	Max 9 epochs, Augmentation for first 6 epochs
Batch Size	Training: 32, Validation: 1
Oversampling	Threshold: 60
Mixup Function	Enabled
Spectral Mixup	Enabled

The mixup function combines two examples in the dataset to create a new example, enhancing generalization. Spectral mixup further diversifies training data by combining spectrograms from different audio samples. These strategies help the model learn robust and generalized representations, improving classification performance.

5. EfficientNet-B1 with Pre-Training

5.1. Overview

The task of audio classification involves categorizing audio signals into predefined classes. In this project, I developed a model for identifying bird calls using TensorFlow and the EfficientNet-B1 architecture, drawing inspiration from previous work on pre-training by Awsaf (Md Awsafur Rahman) [11].The model was pre-trained on BirdCLEF datasets from 2021-2023 and Xeno-Canto Extend, and fine-tuned on BirdCLEF 2024 data to enhance transfer learning. Advanced audio processing and feature extraction techniques were employed to optimize performance on TPU devices, addressing challenges such as spectrogram augmentation and effective transfer learning.

5.2. Data Preparation

Data Sources: BirdCLEF datasets from 2021-2024 and Xeno-Canto Extend [12] [13] [14] [15] [16] [17]. The raw audio data, stored in .ogg format, is efficiently handled using the ‘tf.data‘ API.

Each audio clip is sampled at 32,000 Hz and has a uniform duration of 10 seconds. To effectively capture the audio features, the spectrogram parameters are carefully chosen. The frequency range spans from 20 Hz to 16,000 Hz. The Short-Time Fourier Transform (STFT) parameters include an FFT window size of 2028 and a spectrogram window size of 2048. These configurations ensure that the critical audio characteristics are well-represented for model training. More details are provided in Table 4.

Table 4
Audio data parameters and spectrogram configurations for EfficientNet-B1

Parameter	Value
Sample Rate	32000
Clip Duration	10 sec
Frequency Range	20 Hz - 16,000 Hz
FFT Window Size	2028
Spectrogram Window Size	2048

5.3. Feature Engineering

Feature engineering focuses on transforming raw audio data into a format suitable for model training. Mel spectrograms are generated from the audio signals, converting them into a visual representation of frequency content over time. This transformation is achieved using the 'MelSpectrogram' layer, where each 10-second audio slice undergoes Fourier transformation to capture the spectral properties.

To improve the model's robustness, I apply various augmentation techniques on the spectrograms:

- **Time and Frequency Masking:** Randomly masks parts of the spectrogram in both time and frequency dimensions.
- **Normalization:** Standardizes the data using mean and standard deviation, followed by rescaling to the [0, 1] range.

5.4. Model Building

The model architecture is based on EfficientNet-B1, a convolutional neural network known for its efficiency and performance. Key configurations include:

- **Pretraining:** Initialized with ImageNet weights to leverage transfer learning. The Final Activation Function used is Softmax for multi-class classification. Filter Stride Reduction (FSR) used for reducing the stride in the stem block.

The model incorporates several custom layers using TensorFlow library to handle specific tasks. This can also be implemented using PyTorch.

- **MelSpectrogram Layer:** Converts audio signals into Mel spectrograms.
- **TimeFreqMask Layer:** Applies time and frequency masking for spectrogram augmentation.
- **ZScoreMinMax Layer:** Standardizes and rescales the spectrogram data.
- **MixUp and CutMix Layers:** Augment the training data by mixing audio samples.

5.5. Augmentation

Data augmentation plays a crucial role in improving the model's generalization. Various augmentation techniques are employed to introduce variability in the training data:

- **Audio Augmentation**:
  - **Gaussian Noise Addition**: This technique was chosen to simulate different environmental noise conditions and make the model robust to noise. Applied with a probability of 0.5, it adds random noise to the audio signal, improving the model's ability to generalize to noisy data.
  - **Time Shifting**: Shifting the audio signal in time to introduce variability, but set to 0 as it disrupted the temporal patterns essential for accurate classification.

  - **MixUp**: This technique involves mixing two audio signals to create a synthetic example, applied with a probability of 0.65. This helps the model generalize better by providing varied training examples.
  - **CutMix**: Similar to MixUp, CutMix combines two audio signals but by cutting and pasting parts of each, also applied with a probability of 0.65.
- **Spectrogram Augmentation**:
  - **Time and Frequency Masking**: This technique was chosen to make the model more robust to missing data by randomly masking parts of the spectrogram in both time and frequency dimensions, applied with a probability of 0.5. It helps the model learn to handle occlusions and missing parts in the data.
  - **Normalization**: The 'ZScoreMinMax' layer standardizes the spectrogram data using mean and standard deviation, followed by rescaling to the [0, 1] range. This ensures that the data fed into the model is on a consistent scale, improving learning stability.

Effectiveness of Augmentation Techniques: The pre-trained EfficientNet-B1 model benefited more from time and frequency masking techniques compared to the EfficientNet-B0 model. This is likely because the EfficientNet-B1 model had already learned general audio features during its pre-training phase, making it more adaptable to variations introduced by augmentations. The EfficientNet-B0 model, lacking this pre-trained knowledge, struggled with the same augmentations as it was still learning fundamental patterns from the current dataset.

5.6. Training Procedure

The training procedure involves a carefully designed pipeline to ensure effective learning and generalization. The dataset is stratified into five folds for cross-validation, with classes with very few samples always included in the training set to address class imbalance. Upsampling is employed to ensure that minority classes are adequately represented in the training data. The training configurations are provided in Table 5.

Table 5
Training procedure parameters and configurations

Parameter	Configuration
Cross-Validation	5 folds
Batch Size	32
Learning Rate	1e-3 (cosine scheduler)
Optimizer	Adam
Loss Function	Categorical Cross Entropy with label smoothing (0.05)
Early Stopping	Patience: 5 epochs

The model is trained on a TPU device, utilizing the TPU-VM for automatic device selection and training acceleration. Early stopping is implemented to prevent overfitting, with a patience of 5 epochs. The model's performance is evaluated using the padded cmAP (macro-averaged average precision) score, which accounts for class imbalance and zero true positive labels for certain species. Additionally, the Precision Recall (PR) curve is used as the primary metric for AUC evaluation.

6. Post Processing

Post-processing is integral to refining model predictions and enhancing overall classification performance. The ensemble method combines the predictions of two distinct models to leverage their individual strengths, thereby enhancing robustness and accuracy.

6.1. Ensemble Strategy

To achieve optimal classification results, I implemented an ensemble method where the final prediction for each audio clip is computed as a weighted average of predictions from EfficientNet-B0 and EfficientNet-B1. The ensemble weights were empirically determined based on cross-validation performance: 0.6 for EfficientNet-B0 and 0.4 for EfficientNet-B1. This weighting scheme balances the unique capabilities of each model effectively.

$$\text{Final Prediction} = 0.6 \times \text{Predictions}_{\text{EfficientNet-B0}} + 0.4 \times \text{Predictions}_{\text{EfficientNet-B1}}$$

This weighted average helps in smoothing out the variances and combining the high-confidence predictions from each model. This approach effectively leverages the complementary strengths of EfficientNet-B0 and EfficientNet-B1, providing a comprehensive solution for bird classification in the BirdCLEF24 challenge.

7. Results

The following table summarizes the performance of various models and strategies evaluated in this study, measured by their private and public scores on the BirdCLEF 2024 competition leaderboard. The scores represent the macro-averaged ROC-AUC [18], accounting for class imbalance and providing a robust measure of model performance.

Table 6
Model Descriptions and Performance Scores

Model Descriptions	Private Score	Public Score
EfficientNet-B0	0.649998	0.654307
EfficientNet-B1 with Pretraining	0.596740	0.632268
Models Ensemble	0.652743	0.663388

7.1. Analysis

- **EfficientNet-B0:** The EfficientNet-B0 model [5] demonstrated consistent performance, achieving a private score of 0.649998 and a public score of 0.654307. This balanced performance across both datasets indicates its robustness and generalization capabilities in handling the BirdCLEF dataset.
- **EfficientNet-B1 with Pretraining:** The EfficientNet-B1 model [4], which was enhanced with pretraining, showed a slight dip in the private score (0.596740) compared to its public score (0.632268). This suggests that while pretraining improved its performance on the public dataset, it may have led to some overfitting or less effective generalization on the private dataset.
- **Models Ensemble:** The ensemble of models [19] achieved the highest scores, with a private score of 0.652743 and a public score of 0.663388. This approach effectively leveraged the strengths of multiple models, leading to better overall performance and indicating the effectiveness of model ensembling in complex tasks such as bird species identification from audio recordings.

Overall, the results were very stable, with a correlation of 0.96 between public and private scores [20]. However, there were significant changes visible in the public and private leaderboards.

8. Conclusion

The ensemble of EfficientNet-B0 and EfficientNet-B1 models proved to be an effective strategy for the BirdCLEF 2024 challenge, outperforming individual models and enhancing overall classification performance. EfficientNet-B0, trained with heavy data augmentation, and EfficientNet-B1, pre-trained

on historical BirdCLEF datasets, complemented each other well. The ensemble approach leveraged the strengths of both models, achieving a balance between computational efficiency and classification accuracy. My findings underscore the importance of combining diverse data sources and robust augmentation techniques in building resilient audio classification systems. Future work could explore further optimization of ensemble weights and the incorporation of additional data augmentation methods to continue improving performance in audio classification tasks.

9. Acknowledgments

I would like to thank Stefan Kahl, Willem-Pier Vellinga, Tom Denton, Holger Klinck, and Hervé Glotin for their exceptional leadership and expertise throughout the BirdCLEF24 competition. I am also immensely grateful to the collaborating institutions—Kaggle, Chemnitz University of Technology, Google Research, the K. Lisa Yang Center for Conservation Bioacoustics at the Cornell Lab of Ornithology, the Indian Institute of Science Education and Research (IISER) Tirupati, LifeCLEF, and Xeno-canto—for providing invaluable resources, data, and support. Their collective contributions have been crucial to the success of this project.

References

[1] S. Kahl, T. Denton, H. Klinck, V. Ramesh, V. Joshi, M. Srivathsa, A. Anand, C. Arvind, H. CP, S. Sawant, V. V. Robin, H. Glotin, H. Goëau, W.-P. Vellinga, R. Planqué, A. Joly, Overview of BirdCLEF 2024: Acoustic identification of under-studied bird species in the western ghats, Working Notes of CLEF 2024 - Conference and Labs of the Evaluation Forum (2024).

[2] A. Joly, L. Picek, S. Kahl, H. Goëau, V. Espitalier, C. Botella, B. Deneu, D. Marcos, J. Estopinan, C. Leblanc, T. Larcher, M. Šulc, M. Hrúz, M. Servajean, et al., Overview of lifeclef 2024: Challenges on species distribution prediction and identification, in: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, 2024.

[3] M. Tan, Q. V. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, arXiv preprint arXiv:1905.11946 (2020). URL: https://arxiv.org/abs/1905.11946.

[4] A. Aikhmelnytskyy, Birdclef24 pretraining is all you need - infer, 2024. URL: https://www.kaggle.com/code/aikhmelnytskyy/birdclef24-pretraining-is-all-you-need-infer.

[5] TC0000, Birdclef starter notebook, 2024. URL: https://www.kaggle.com/code/tc0000/birdclef-starter-notebook.

[6] K. O'Shea, R. Nash, An introduction to convolutional neural networks, arXiv preprint arXiv:1511.08458 (2015). URL: https://arxiv.org/abs/1511.08458.

[7] Nicholson, Birdclef 2024: Spectrograms imagenet run, 2024. URL: https://www.kaggle.com/code/richolson/birdclef-2024-spectrograms-imagenet-run#Initialize-submit-DF-with-correct-columns.

[8] G. Zhou, Y. Zhang, J. Pan, Z. Han, Short-time fourier transform with the window size fixed in the frequency domain (2017). URL: https://www.researchgate.net/publication/321043608_Short-Time_Fourier_Transform_with_the_Window_Size_Fixed_in_the_Frequency_Domain.

[9] C. Muljana, T.-P. M. Luo, A review of online course dropout research: Implications for practice and future research, ResearchGate (2019). URL: https://www.researchgate.net/publication/227246914_A_review_of_online_course_dropout_research_Implications_for_practice_and_future_research.

[10] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, arXiv preprint arXiv:1411.1792 (2014). URL: https://arxiv.org/pdf/1411.1792.

[11] Awsaf49, Birdclef23 pretraining is all you need - train, 2023. URL: https://www.kaggle.com/code/awsaf49/birdclef23-pretraining-is-all-you-need-train.

[12] Kaggle, Birdclef 2021 competition, 2021. URL: https://www.kaggle.com/competitions/birdclef-2021.

[13] Kaggle, Birdclef 2022 competition, 2022. URL: https://www.kaggle.com/competitions/birdclef-2022.

[14] Kaggle, Birdclef 2023 competition, 2023. URL: https://www.kaggle.com/competitions/birdclef-2023.

[15] Kaggle, Birdclef 2024 competition, 2024. URL: https://www.kaggle.com/competitions/birdclef-2024.
[16] R. Rao, Xeno-canto bird recordings extended a-m, 2024. URL: https://www.kaggle.com/datasets/rohanrao/xeno-canto-bird-recordings-extended-a-m.
[17] R. Rao, Xeno-canto bird recordings extended n-z, 2024. URL: https://www.kaggle.com/datasets/rohanrao/xeno-canto-bird-recordings-extended-n-z.
[18] Metric, Birdclef roc auc, 2024. URL: https://www.kaggle.com/code/metric/birdclef-roc-auc.
[19] A. Porwal, Silver medal solution - 25th place, 2024. URL: https://www.kaggle.com/code/aadityaporwal/silver-medal-solution-25th-place.
[20] Correlation between public and private scores, 2024. URL: https://www.kaggle.com/competitions/birdclef-2024/discussion/512197.