Classic Approaches to Bird Song Classification

Mihai-Dimitrie Minuț, Cristian Simionescu and Adrian Iftene

University "Alexandru Ioan Cuza" of Iasi - Faculty of Computer Science, Street General Henri Mathias Berthelot 16, Iași 700483, Iași, Romania

Abstract

Processing bird audio data and building classifier algorithms lead to different insights about the bird species, in turn improving what is known about avian biodiversity. The research for audio processing and audio classification has been ongoing for some time and with major advances. Paired together with the many existing species and sound types of birds it makes the task of bird song classification complex. This work explores popular audio methods to preprocess and augment two types of extracted features, audio and spectrogram images, by making use of deep learning techniques such as pretraining, fine-tuning, and transfer learning. Results feature averagely good models while showcasing the effect of certain data modifications or classification training techniques.

Keywords

Sound classification, Mel Spectrograms, 1/2D Convolutions, EfficientNetV2, Noise, Augmentation

1. Introduction

This work represents the working note and the details from participating in the BirdClef2023 competition, which is taking place as part of the LifeClef2023 branch [1]. The competition goal is to construct classification algorithms that will be used to recognize and attribute the bird species for a given audio signal as input [2]. The main focus of this working note is the deep learning experiments based on both types of data: audio time series and image spectrogram data; paired together with different preprocessing and augmentation techniques for both image and signal data. The best results feature models based on 1D CNN architectures trained and fine-tuned on audio signal data that make use of background noise as augmentation; achieving 0.63186 and 0.74384 private and public scores respectively. The rest of the paper includes the details related to the activity submitted by us regarding sending the runs in this competition.

2. Solution Objectives

The large amounts of audio data and the BirdClef2023 competition requirements for the model evaluation, under 2 hours and CPU only runs, imposed quite strict restrictions on what an individual can submit, making very deep and complex models impractical [2]. By taking the restrictions and the development environment, along with the team size of one person and the late registration to the competition (last month), the goals are not set to be on the overpromising side. As such, the proposed objectives of this work are to:

• establish a preprocessing and augmentation pipeline for the dataset;

• experiment on simpler model architectures trained from scratch;

• experiment with finetuning and transfer learning on pre-trained architectures, such as the EfficientNetv2 [3];

[1]CLEF 2023: Conference and Labs of the Evaluation Forum, September 18–21, 2023, Thessaloniki, Greece
EMAIL: aelchim.tunim@gmail.com; cristian@nexusmedia.ro; adiftene@info.uaic.ro
ORCID: 0009-0005-9200-1776; 0000-0003-1274-0914; 0000-0003-3564-8440

• cover experiments on both signal time series data and spectrogram image data;
• explore with Gaussian and Normally distributed background noise [4] [5];
• construct a final model pipeline with the best-found insights;

Some of the selected experiments were chosen to test and experiment with past popular techniques applied to audio time series and spectrogram data. A good example would be the time series and spectrogram image augmentation results presented in [6], which present Time Warping and Scaling as being among the best augmentation methods for audio time series.

3. Data Exploration

The dataset provided by the competition after registration contains 264 folders representing each class through an ID, with distinct amounts containing audio data files with varying durations. Each directory contains audio recordings saved in the OGG digital format. By exploring the dataset, some issues are identified: the audio file class imbalance (Fig. 1), different noise levels between audio files (Fig. 3), and empty segments longer than 5 seconds (Fig. 2).

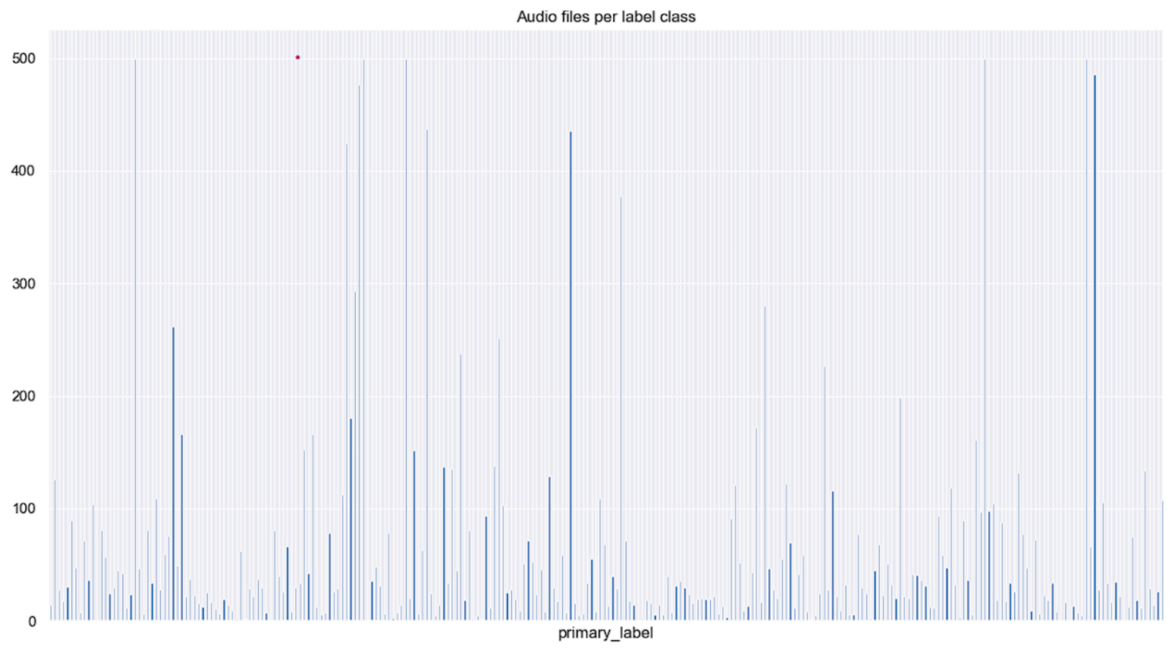

Figure 1. Audio files per bird species contained within the competition dataset

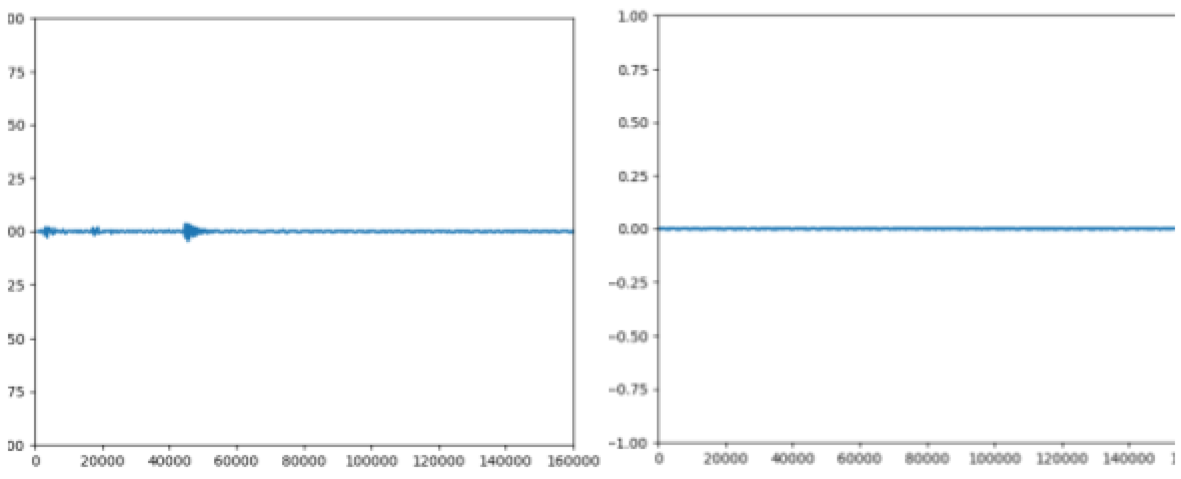

Figure 2. Audio 5 second frames from bird species malkin1; left contains bird data and right does not

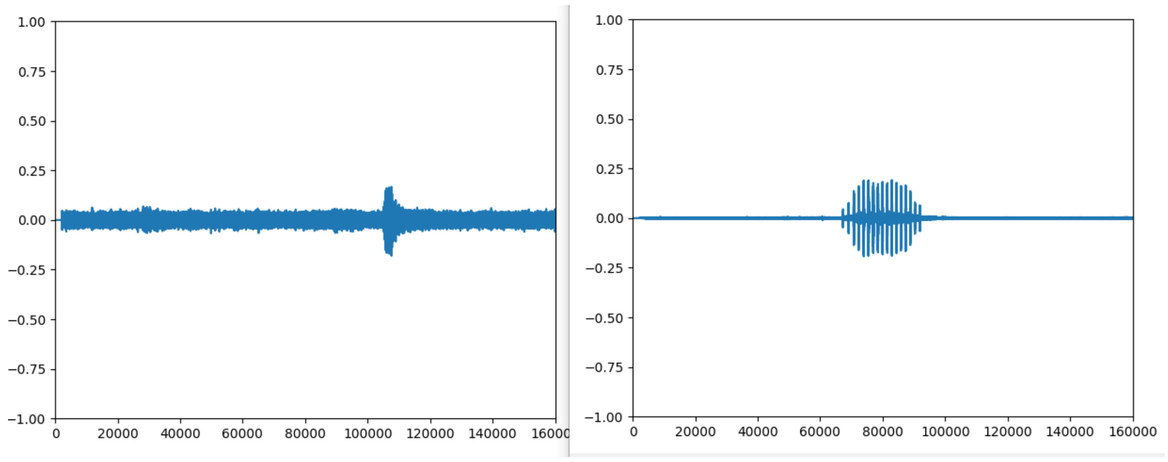
Figure 3. Audio frames from different recordings showcasing the background noise discrepancy

4. Data Preprocessing

4.1.Audio Data Frame Cutting

The initial audio file data is not in the correct format that allows for easy processing, so it needs to be cut into 5 seconds pieces with a sample rate of 32,000, implying a resulting dataset of samples of shape (160,000). Among the considered techniques for cutting up data from Figure 4, the one that would bring the most quality would be using bird sound detection [7]. The compromise between time and quality was to take the simple window shift without overlap because it would contain windows with the bird calls and a smaller quantity of empty fragments compared to the window shift with the overlap method. By applying the aforementioned method, the balance of data has suffered even further, with some classes having as little as 1 window of 5 seconds while others have over 1,000 inputs worth of data.

4.2.Dataset Splitting

The setup for evaluation is done by splitting the training data by a ratio of 70:30 between train and evaluation, as a swifter alternative to the better-performing K-folds cross-validation technique [9].

4.3.Feature extraction

From the initial raw audio frames, by applying the Short-time Fourier Transform [9] and the Mel Scale transformation the log-mel features can be extracted [10], which can be further transformed into the Mel spectrogram image. To obtain a square image shape with size (128, 128), the following parameters for the algorithm were used: 128 Number of Mels, the competition sample rate of 32,000/second, and a hop length equal to $\lceil 160000 / 128 \rceil$.

4.4.Dataset Augmentation

The augmentation was done both on the preprocessing side for the spectrogram data but only dynamically on batch load during training. The three used techniques for augmentation, which are shown to work well on audio data [6], are Time Warping, Scaling, and Shifting (a subclass of permutation with one cut). Data scaling is a form of augmentation that reduces or decreases the values of a random time window of the time series [6]. The magnitude of the scaling was uniformly chosen between -0.1 and 0.1. Time Warping dilates or contracts random windows of time that are predominantly used as a technique for time series augmentation [6]. As can be seen in Fig.5 the effect on the spectrogram data is pretty damaging which is why it was omitted from the spectrogram preprocessing pipeline; this augmentation technique is still safe to use in the case of working directly with the time series data.

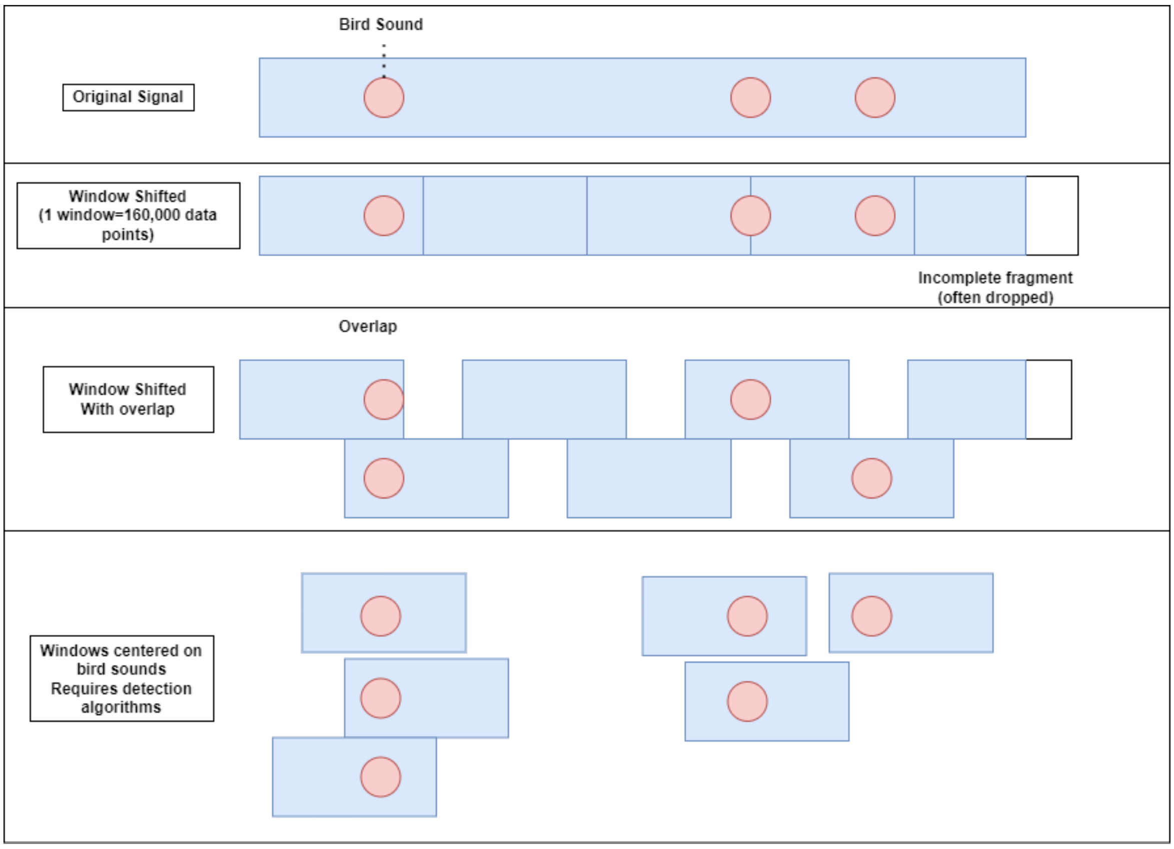
Bird Sound
Original Signal
Window Shifted (1 window=160,000 data points)
Incomplete fragment (often dropped)
Overlap
Window Shifted With overlap
Windows centered on bird sounds Requires detection algorithms

Figure 4. Possible solutions for extracting audio frame samples from the initial audio recordings

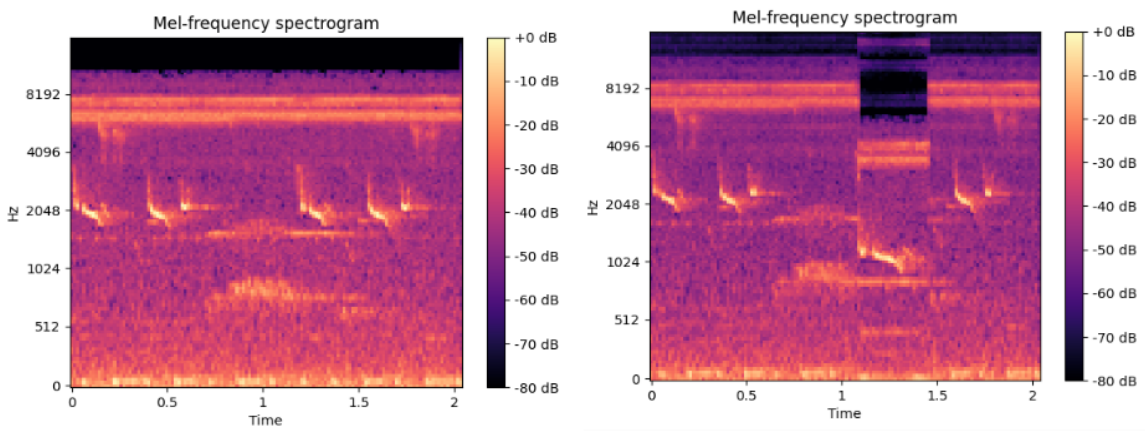

Figure 5. Mel Spectrogram of a plain bird signal audio (left) and augmented with Time Warp (right)

Shifting, a subclass of permutation, is an augmentation technique that cuts the time series in two and rearranges the order resulting in pieces [6]. Two types of shifting were applied: shifting the time series by a factor between -0.1 and 0.1, and cutting directly in half, and reordering the two pieces. The impact of the shifting on the spectrogram image data was minimal in most cases due to the bird song not being constant during the audio. Still, on the signal time series data counterpart, the discontinuity of values among the time axis was pretty visible and clear to affect the data quality (see Fig. 6).

The algorithm used to augment the dataset in the preprocessing phase is the one described in Algorithm 1. The goal was to obtain at least 20 distinct input samples per class and favor species with less than 264 input audio frames and generate more data for them.

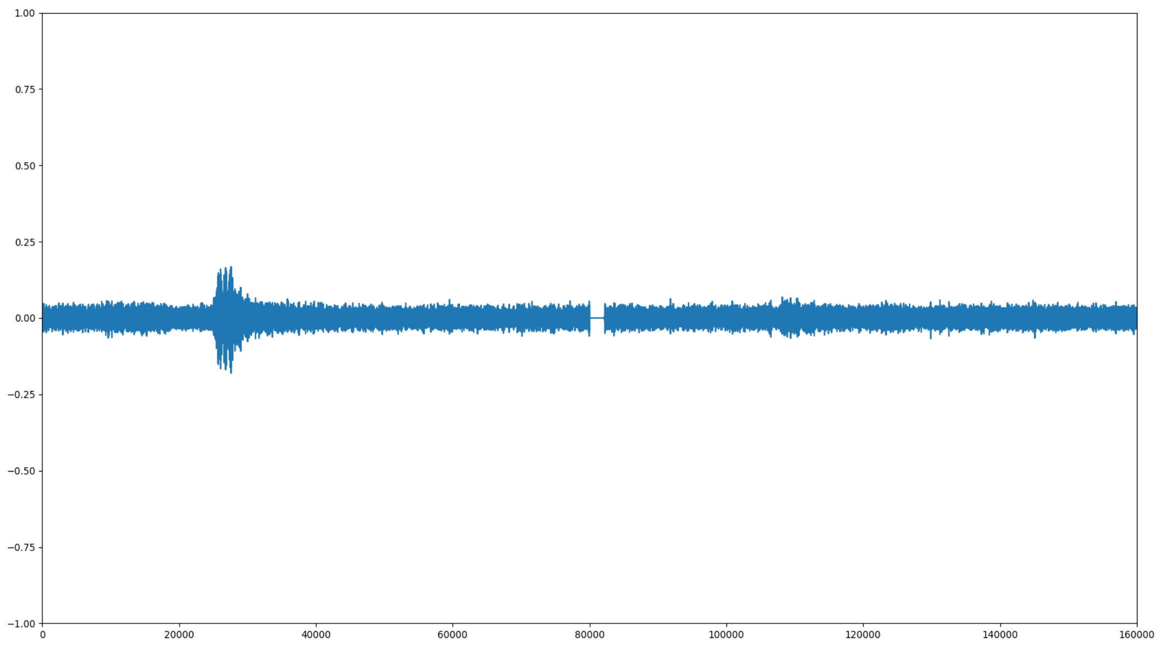

Figure 6. Audio signal shifted circularly at the middle point and leaving a big discontinuity in data making the data less usable as plain waveform

```
Algorithm 1 Augmentation process
Require: $class\_samples$
  if $len(class\_samples) > 264$ then
      return $class\_samples$
  $class\_samples\_shifted\_05 \leftarrow shift\_and\_reorder(class\_samples, min\_shift \leftarrow 0.5, max\_shift \leftarrow 0.5)$
  $class\_samples\_scaled\_01 \leftarrow scale(class\_samples, min\_scale \leftarrow -0.1, max\_scale \leftarrow 0.1)$
  $total\_class\_samples = class\_samples + class\_samples\_shifted\_05 + class\_samples\_scaled\_01$
  if $length(total\_class\_samples) > 20$ then
      return $total\_class\_samples$
  while $len(total\_class\_samples) < 20$ do
      $random\_index = random(len(class\_samples))$
      $length(total\_class\_samples) += shift\_and\_reorder(class\_samples[random\_index], min\_shift \leftarrow -0.1, max\_shift \leftarrow 0.1)$
  return $total\_class\_samples$
```

Algorithm 1. Augmentation process for each of the 5 second input audio

4.5. Class Weights

After the augmentation of the spectrograms, the least represented classes had at least 20 samples but still nothing in comparison to 8,000 others; the time series data being unaffected. The problem with weights being too high or low is that during training this will cause the loss to overshoot and explode or vanish certain errors, making the learning process much harder. The solution applied was to smooth out the weights for both the spectrogram and the signal data samples, the result is shown in Fig. 7. The process for smoothing the data is described in Algorithm 2.

Algorithm 2 Smoothing the weights

```
$smooth\_factor \leftarrow 0.3$
$mean\_w \leftarrow mean(weights)$
$ideal\_weight \leftarrow 1.0/264.0$
$weights\_diff \leftarrow mean\_w - weights$
$smooth\_weights \leftarrow weights + weights\_diff * smooth\_factor$
```

Algorithm 2. Method used to directly scale and smooth the training weights for the model training

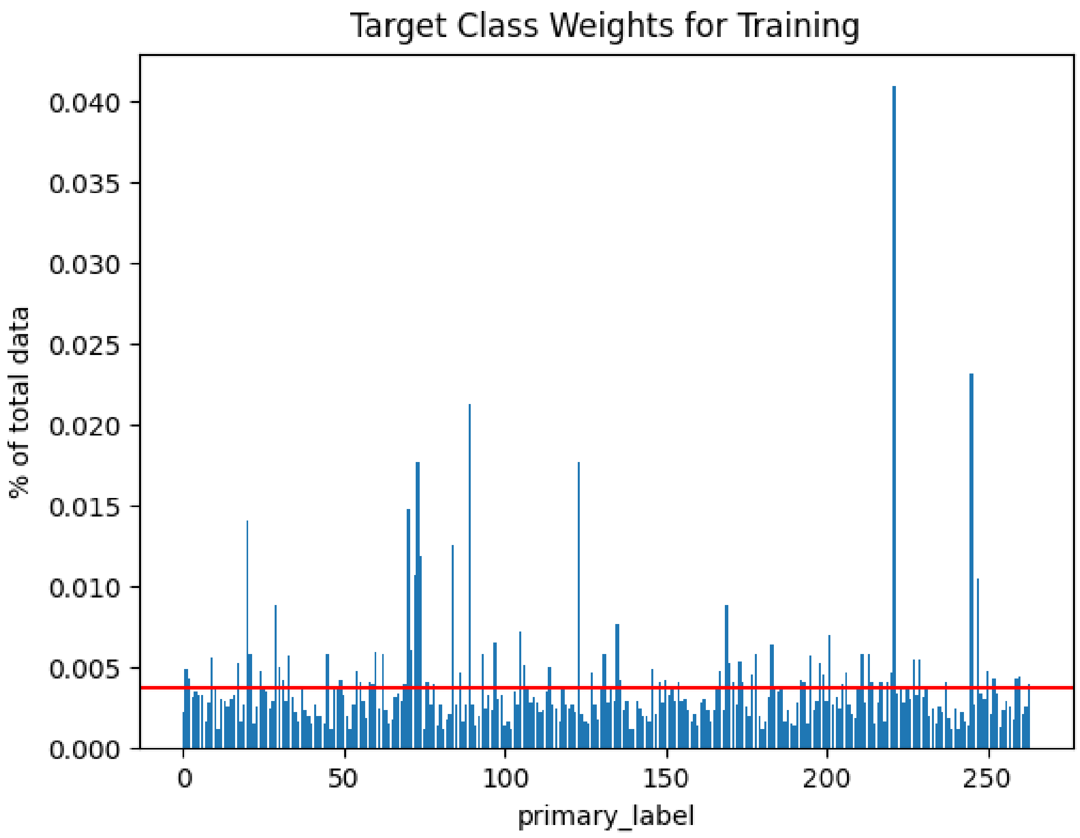

Figure 7. Resulting class weights used for to increase loss during training for misrepresented species

4.6.Mixup

The mixup operation was done directly on the audio signals, and then the spectrogram feature extraction was applied to the final combined signal to obtain instead of the original, the mixed-up spectrogram. Usually, Mixup is done during training between random samples [9], but given the circumstances of time and resources available, it was decided to build a dataset generator using the mixup method. Additionally, to address the remaining imbalance within the dataset the mixup dataset generator also got provided with the initial class weights and target class weights for the resulting dataset which further balanced the data amount per bird species.

The target mixup generated dataset has a size of 100,000 and minimum label items of 264. The current label weights is the one presented in Figure 40, the target label weights are the ones in Figure 7 and the data input is the augmented data, as previously mentioned in the augmentation section for spectrograms. The total amount of data needed per label was given by the target class weights multiplied by the dataset size and finally clipped by the minimum requirement of 264 items.

4.7. Partitioning

The final dataset data was separated into multiple partition files, which contained the respective training and validation data in order. One last step applied was to also shuffle the data in order to break the order between the dataset species.

4.8. Preprocessing Pipeline

There were two final configurable preprocessing pipelines for building the datasets used to train the models submitted to the competition, which can be seen in Fig. 8: one for images and another one for signals. In Fig. 8, all of the experimental preprocessing steps were included: with green the kept ones and red the discarded ones. Validation data is represented by the orange color while training by the violet one. The final product of the pipelines is the variations of the signal dataset, variations of the spectrogram dataset, and the final training class weights. The final datasets containing spectrograms were generated both with and without any augmentation to cover all cases when experimenting. A considerable observation would be that the augmentation that will happen during the training process is not shown in the below schema, and it will be included in the Deep Learning Experiments chapter.

5. Deep Learning Experiments

Common denominators among all the experiments are the Adam optimizer with a starting learning rate of 0.00001-0.000001, the Binary Crossentropy Loss function, and the ROC AUC metric used to evaluate the model on training and validation data. The PR AUC metric is also used as a measurement in some of the experiments. One common feature is the method of loading the data partitions which was done by prefetching and loading at least 2 partitions ahead, which improved the training times for both the image and time series datasets. Initial testing was done with the batch size with values from {16, 32, 64, 128, 256} and 64 items per batch achieved the shortest train epoch duration in both scenarios: image and time series data. All of the models were scheduled to train up to 100 epochs and used a method to save and keep track of the best validation-performing model at each epoch. If no improvements to the validation metrics were made for more than 20 epochs the training would be stopped. The experiments tables are made to highlight the most significant changes in the training process and one entry from those tables might be in reality a series of tests concluding into that result. There were 3 model types with which were experimented: 2D convolutional models trained from scratch on spectrogram data, 1D convolutional models for signal data trained from scratch, and experiments with transfer learning and fine-tuning the EfficientNet V2 B0.

The final submission was made by taking the best three already submitted models (Table 4, indices 6, 7, and 8, focus on public score) and fine-tuning them in 3 steps of 3 epochs each, on the validation data. Fine-tuning was done only on the last dense layers with a much lower learning rate. After the 3 fine tuning iterations, the best model was the one using the base 1D convolutional architecture paired together with some uniform background noise (indices 8-11 from Table 4).

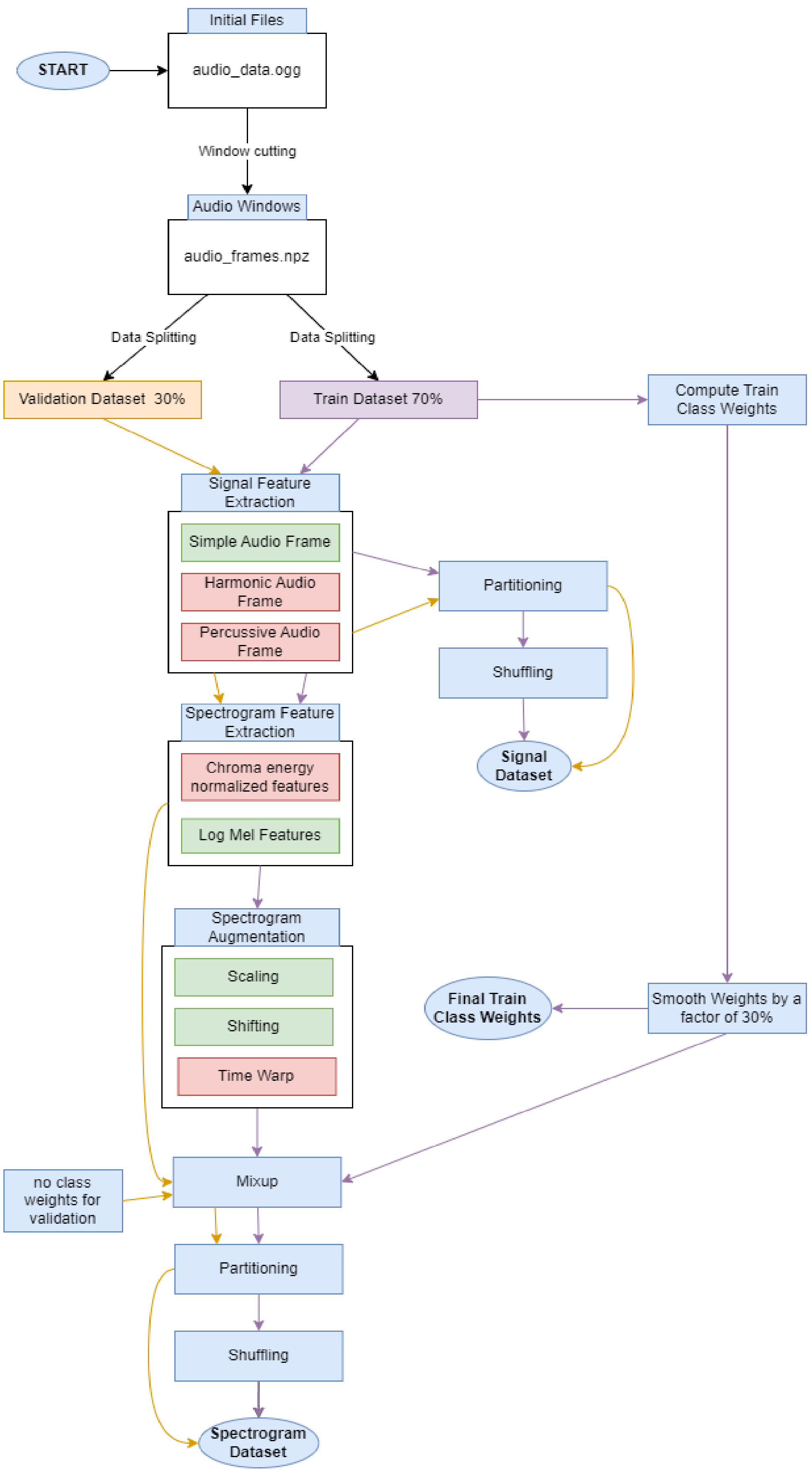

Figure 8. Data preprocessing pipeline which includes the steps used to obtain the Image Spectrogram, Audio Waveform Datasets and training class weights (red marks unperformant methods while green marks the ones used within final pipelines)

Table 1. Experiments with self made and trained 2D convolutional models

Index	Experiment Name/Observations	ROC AUC	VAL ROC AUC	Loss	Val Loss	Best Validation Epoch	Epoch Train Duration
1	No modification+ 10e-7lr	0.76	0.528	0.43	0.028	48	68-72s
2	Relu->LeakyRelu + 10e-7lr	0.74	0.65	0.47	0.034	64	68-72s
3	Relu->Sigmoid + 10e-7lr	0.745	0.655	0.39	0.031	72	68-72s
4	Rescaling + 10e-5lr	0.755	0.648	0.455	0.024	19	70-74s
5	Rescaling+GaussNoise (0.01)+10e-5lr	0.68	0.649	0.61	0.025	6	80-90s
6	Rescaling+GaussNoise (0.1)+10e-5lr	0.752	0.649	0.455	0.023	17	80-90s
7	Rescaling+UniformNoise (0.05)+10e-5lr	0.753	0.648	0.455	0.24	18	90-100s
8	No training weights+10e-5lr	0.72	0.47	0.8	0.071	37	66-71s

Table 2. Transfer Learning and Fine-tuning Experiments on EfficientNetv2B0 backbone

Index	Experiment Name/Observations	ROC AUC	VAL ROC AUC	Loss	Val Loss	Best Validation Epoch	Epoch Train Duration
1	EffNet frozen + GaussNoise(0.05) + Rescaling	0.739	0.673	0.617	0.0254	9	195-200s
2	Frozen + Dropout 0.2->0.3 + GaussNoise(0.05) + Rescaling	0.753	0.669	0.4489	0.023	15	195-200s
3	Frozen + Dropout 0.2->0.3	0.775	0.6437	0.0205	0.024	20	160-170s
4	Frozen + Dropout 0.2->0.3 + Fine Tune Last 30	0.82	0.686	0.019	0.021	10	420s
5	Frozen + Dropout 0.2->0.3 + Fine Tune Last 45	0.717	0.709	0.768	0.0294	6	600s

Table 3. Experiments with self made and trained 1D convolutional models

Index	Experiment Name/Observations	ROC AUC	VAL ROC AUC	PR AUC	VAL PR AUC	Loss	Val Loss	Best Validation Epoch	Epoch Train Duration
1	No modification+ 10e-4lr	0.936	0.9153	0.542	0.481	0.1503	0.0123	44	500s
2	UniformNoise(5e-4) + 10e-4lr	0.942	0.913	0.557	0.4838	0.117	0.0124	64	430-480s
3	Scaling Augmentation 50% + 10e-4lr	0.8203	0.828	0.061	0.072	0.238	0.0205	34	500-520s
4	Scaling 10% + Time Warp 10% Augmentations+ 10e-4lr	0.858	0.866	0.15	0.183	0.216	0.0181	61	580-600s

The model submissions were made using the Kaggle web platform[2] for uploading the models and running the notebooks. An important note is that only one author kaggle account was used in preparing and submitting competition runs.

6. Experiment Results

The augmentation methods that were used, Time Warping, Scaling, and Shifting as described in [6], did not seem to impact the training that much, sometimes even being detrimental; with no strong evidence in the case of spectrograms due to the case of faulty data. Among all of the training experiments, multiple activation functions have been tried and the ones with the most consistent good results were Sigmoid, Relu, and LeakyRelu. By consulting all the experiments, the fine-tuning techniques always helped and provided models with an easier way to improve. Another, not as highlighted, technique is the class weights applied to the loss, which helped combat the data imbalance, especially during the 1D convolutional training sessions. The best submission made (index 9 from Table 4) was a convolutional 1D model built and trained from scratch by using a variety of deep learning techniques combined with added background uniform noise; which was also fine-tuned on the validation dataset as a last successful step at improving the quality.

[2] https://www.kaggle.com/competitions/birdclef-2023

Table 4. Model variants submitted to the competition through Kaggle and their final results

Index	Model Submitted	Private Score	Public Score
1	2D CNN base +No modification+ 10e-7lr	0.60233	0.71746
2	2D CNN base +No modification+ 10e-7lr + outputs through softmax	0.60366	0.71956
3	2D CNN base + Relu->LeakyRelu + 10e-7lr	0.60335	0.71817
4	EffNetV2 base + Frozen + Dropout 0.2->0.3 + Fine Tune Last 30	0.60308	0.71861
5	EffNetV2 base + Frozen + Dropout 0.2->0.3 + Fine Tune Last 45	0.60392	0.71872
6	1D CNN base	0.61395	0.73069
7	Scaling 10% + Time Warp 10% Augmentations+ 10e-4lr	0.62781	0.74172
8	1D CNN +UniformNoise(5e-4) + 10e-4lr	0.6048	0.7205
9	1D CNN + UniformNoise(5e-4) + 10e-4lr + Final fine-tune on the validation dataset (3 epochs)	0.63115	0.74384
10	1D CNN + UniformNoise(5e-4) + 10e-4lr + Final fine-tune on the validation dataset (6 epochs)	0.63066	0.74369
11	1D CNN + UniformNoise(5e-4) + 10e-4lr + Final fine-tune on the validation dataset (9 epochs)	0.63186	0.7438

Training and experimenting with both the waveform of audio data and spectrogram image data helps in gathering (Tables 1-3) a variety of helpful insights regarding the portability and consistency of the used training and pretraining methods. It also provides a solid ground for training more complex ensemble models that make use of both types of data.

7. Future Work

There are many possible directions for further explorations, coming from both explored and unexplored ideas. An interesting idea that some participants used during the competition was to complement the existing dataset with external bird audio data. This proved to be one of the key points in obtaining higher-quality classification models. A future iteration of the dataset could potentially fix the problem either by applying transformations to entire data files or by finding a way to exclude empty audio frames altogether. There are many possible directions for further explorations, coming from both explored and unexplored ideas. An interesting idea that some participants used during the competition 24 was to complement the existing dataset with external bird audio data. This proved to be one of the key points in obtaining higher-quality classification models. Because of the data issue, the spectrograms' convolutional models were compromised. A future iteration of the dataset could potentially fix the problem either by applying transformations to entire data files or by finding a way to exclude empty audio frames altogether.

8. Conclusion

Looking back to the original goals and comparing them to the current state, the dataset-building pipeline has been finalized and, besides the problem in the mel spectrogram image pipeline, it helped in building training-ready datasets. The performed experiments added value through the insights they offered and allowed in the end to build better valid submissions. Most techniques did not manage to achieve a good performance, while others seemed to always improve the results; a case being the added background noise. The classic and simple model architectures turned out to be able to learn to distinguish among the bird species and even showed much more potential.

9. References

[1] A. Joly, C. Botella, L. Picek, S. Kahl, H. Goëau, B. Deneu, D. Marcos, J. Estopinan, C. Leblanc, T. Larcher, R. Chamidullin, M. Šulc, M. Hrúz, M. Servajean, H. Glotin, R. Planqué,

W.-P. Vellinga, H. Klinck, T. Denton, I. Eggel, P. Bonnet, H. Müller, Overview of LifeCLEF 2023: evaluation of ai models for the identification and prediction of birds, plants, snakes and fungi, in: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, 2023.

[2] Kahl, S., Denton, T., Klinck, H., Reers, H., Cherutich, F., Glotin, H., Goëau, H., Vellinga, W.P., Planqué, R., Joly, A.: Overview of BirdCLEF 2023: Automated bird species identification in Eastern Africa. Working Notes of CLEF 2023 – Conference and Labs of the Evaluation Forum (2023)

[3] M. Tan and Q. V. Le, "Efficientnetv2: Smaller models and faster training," CoRR, vol. abs/2104.00298, 2021.

[4] M. V. Shugaev, N. Tanahashi, P. Dhingra, and U. Patel, "Birdclef 2021: building a birdcall segmentation model based on weak labels," in Conference and Labs of the Evaluation Forum, 2021.

[5] C. Zhang, H. Zhan, Z. Hao, and X. Gao, "Classification of complicated urban forest acoustic scenes with deep learning models," Forests, vol. 14, p. 206, Jan 2023

[6] B. K. Iwana and S. Uchida, "An empirical survey of data augmentation for time series classification with neural networks," PLOS ONE, vol. 16, pp. 1–32, 07 2021.

[7] A. S. Kumar and D. Kowerko, "Tuc media computing at birdclef 2021: Noise augmentation strategies in bird sound classification in combination with densenets and resnets," in Proceedings of the Working Notes of CLEF 2021 - Conference and Labs of the Evaluation Forum, Bucharest, Romania, September 21st - to - 24th, 2021 (G. Faggioli, N. F. 0001, A. Joly, M. Maistro, and F. Piroi, eds.), vol. 2936 of CEUR Workshop Proceedings, pp. 1617–1626, CEUR-WS.org, 2021.

[8] Korjus, M. N. Hebart, and R. Vicente, "An efficient data partitioning to improve classification performance while keeping parameters interpretable," PLOS ONE, vol. 11, pp. 1– 16, 08 2016.

[9] E. Sejdic, I. Djurovi ´ c, and J. Jiang, "Time–frequency feature representation using energy ´ concentration: An overview of recent advances," Digital Signal Processing, vol. 19, no. 1, pp. 153–183, 2009.

[10] B. Logan et al., "Mel frequency cepstral coefficients for music modeling.," in Ismir, vol. 270, p. 11, Plymouth, MA, 2000

[11]H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk ´ minimization," CoRR, vol. abs/1710.09412, 2017.