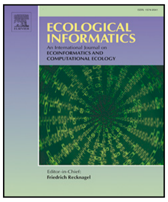
Contents lists available at ScienceDirect

Ecological Informatics

journal homepage: www.elsevier.com/locate/ecolinf



Enhancing sound-based classification of birds and anurans with spectrogram representations and acoustic indices in neural network architectures

Fábio Felix Dias [a,b], Moacir Antonelli Ponti [b], Rosane Minghim [c,*]

[a] Tandon School of Engineering, New York University, Brooklyn, United States of America
[b] Instituto de Ciencias Matematicas e de Computacao, University of Sao Paulo, Sao Carlos, Sao Paulo, Brazil
[c] School of Computer Science and Information Technology, University College Cork, Cork, Ireland

ARTICLE INFO

Keywords:
Sound event detection
Feature fusion
Convolutional neural networks

ABSTRACT

Research on habitat monitoring via passive acoustics has generated vast audio resources for soundscape ecology, calling for automated methods to aid data analysis. While Deep Neural Networks excel in classification tasks, their application to audio collected in the wild presents several challenges compared to other audio sources. Nature recordings present ambient noise, sparsity of targeted events, various vocalizations attributed to the same species, and fine-grained sound variance. In addition to sound characterization, we lack annotated datasets of suitable size to train networks accurately for detecting and identifying animal species. To leverage the best from these models, this work investigates different audio input representations, particularly spectrogram-based and acoustic indices, which are pre-processed features extracted from audio sources. We evaluate the impact of combining both input categories, often treated separately, in various architectures, employing quantification in the training process as well as transfer learning. With that, we propose guidelines for using neural networks to classify species based on their sound patterns, even for a small dataset. We have evaluated these guidelines with a dataset collected in Brazil under different environmental conditions and a dataset for detecting and classifying acoustic scenes and events. The empirical results ratify that the pre-trained network learns better (accuracy up to 0.91); that using acoustic features can improve the results marginally (up to 13 percentage points of difference) depending on the time-frequency input and main architecture; and that combining spectrogram representations with acoustic features yields the best results (accuracy up to 0.91).

1. Introduction

Sound is a rich source of information related to animal and landscape changes and the impact of human activities in natural areas, thus supporting applications such as measuring environmental health and biodiversity (Servick, 2014). The combination of sounds in a landscape is known as soundscape and the study of relations between biological, geophysical, and anthropological sounds is referred to as Soundscape Ecology (Servick, 2014; Krause, 1987; Pijanowski et al., 2011; Scarpelli et al., 2020), bioacoustics (Laiolo, 2010; Penar et al., 2020), or ecoacoustics (Sueur and Farina, 2015; Farina and Gage, 2017). This area has taken advantage of large databases collected with passive acoustic monitoring (PAM) devices, which record sounds during long periods and produce several terabytes of data (Thomas et al., 2019). PAM reduces observer bias, creates a permanent base for further analysis, generates negligible disturbance on species and habitats, and increases the probability of detecting rare species (Pieretti et al., 2017; Znidersic et al., 2020). Such data can be used in the context of Soundscape Ecology, for example, to classify target sounds of bird and anuran species, because those are considered bio-indicators, i.e., their presence and behavior are proxy measures that reflect the state of environment (Mitchell et al., 2020; Strout et al., 2017). However, such an amount of data brings great challenges when exploring, analyzing, and extracting meaningful ecological information.

In such a challenging case, data science and analytics tools based on handcrafted features, so-called Acoustic Indices, and neural networks are widely used in environmental sound analysis. Acoustic Indices have been largely employed to assess sound dynamics and biodiversity attributes (Bradfer-Lawrence et al., 2019; Dröge et al., 2021; Scarpelli et al., 2021; Sueur et al., 2014). Furthermore, Convolutional Neural Networks (CNN) are a type of network largely employed for audio classification models, having capabilities of recognizing patterns in spectrograms related to frequency modulation, as well as identifying time-frequency patterns related to different natural sounds (Salamon and Bello, 2017).

[*] Corresponding author.
E-mail address: rosane.minghim@ucc.ie (R. Minghim).

https://doi.org/10.1016/j.ecoinf.2025.103232
Received 17 April 2024; Received in revised form 18 April 2025; Accepted 24 May 2025
Available online 11 June 2025

In that context, different CNN flavors were explored for species detection due to differences in data patterns, data availability, and model assumptions (Cakir et al., 2017; Kirsebom et al., 2020; Shiu et al., 2020). For instance, BirdVoxDetect (Lostanlen et al., 2019) is a pre-trained model to detect the presence of avian flight calls that yielded an area under the precision–recall curve (AUPRC) greater than 76%. The architecture is a variation of context-adaptive network (Delcroix et al., 2015) that contains two branches: the main branch derived from the model proposed by Salamon and Bello (2017) and an auxiliary branch with one convolutional layer followed by one dense (also known as fully connected) layer. They fed the main branch with either mel-spectrogram or per-channel energy normalization (PCEN) and the auxiliary branch with statistical measures of the power spectral density. With their results, a combination of PCEN, context-adaptive network with adaptive threshold, and data augmentation improved the results of architectures trained with mel-spectrograms. BirdNET (Kahl et al., 2021b) was trained to classify 984 different bird species from Europe and North America that generated a mean average precision (mAP) of 0.79. Authors used mel-spectrograms as input to a model based on ResNet-50 with architecture and modifications inspired by Zagoruyko and Komodakis (2016), He et al. (2019) and Schlüter (2018). In another example, Strout et al. (2017) used CNN architectures to generate features that feed a Support Vector Machine (SVM) to classify 15 anuran species, obtaining accuracy up to 77%. They used models like R-CNN (Girshick et al., 2014), AlexNet (Krizhevsky et al., 2012), and CaffeNet (Jia et al., 2014), pre-trained with ImageNet (Deng et al., 2009) as feature extractors of spectrogram images. Xie et al. (2022) created a lightweight CNN model to detect the presence of a specific frog species. They performed a frequency selection, fed the network with a multi-view spectrogram (3 dimensions) based on mel-spectrogram variations, and created a loss function that combines binary cross-entropy and focal loss (Lin et al., 2017). The model achieved an F1-score of 96.4 ± 2.0 with fewer parameters than a VGGish model. LeBien et al. (2020) used RestNet-50 pre-trained on the ImageNet database and added a pooling layer and two dense layers with dropout on the architecture top. The model can classify mel-spectrograms of 24 bird and frog species, yielding a mAP equal to 0.893 and a total average precision of 0.975. Following these and many other references, CNNs remain more accessible due to their reduced parameter sizes, allowing training even with under low computational resources, although we acknowledge that Vision Transformers are also gaining traction in this field (Tang et al., 2023). CNNs also have a lower volume requirement for training. In recent challenges, CNNs demonstrated effectiveness in achieving higher results than transformer-based ones (Kahl et al., 2021a).

Such previous studies have shown high metric values, e.g., accuracy and precision above 70%, when identifying sound sources. However, they differ in using architectures, input representation, and used metrics. Some given architecture and training strategy that works for a specific scenario and dataset may not generalize to other data due to inherent optimization problems and lack of learning guarantees (Ponti et al., 2021). Moreover, proper detection of animal sounds can be affected by other animal vocalizations, geophysical noise, sounds from human activities, and electronic recorder noise (Kahl et al., 2021b). This is relevant since PAM is recording “in the wild”, an unconstrained environment. To mitigate issues related to small labeled datasets and improve model generalization, researchers use techniques such as data augmentation (Salamon and Bello, 2017; Lostanlen et al., 2019; Parascandolo et al., 2016). Furthermore, transfer learning (Strout et al., 2017; LeBien et al., 2020; Kong et al., 2020) is also applied to improve models’ generalization by leveraging the learned knowledge of different datasets and tasks.

Even with large applicability in species detection, there is no consistent knowledge about the best inputs to feed a neural network identifier to handle a specific or more generic range of sounds. For example, while mel-spectrogram is known to be largely employed in sound classification, we do not know whether it is always the best representation (Purwins et al., 2019). A combination of both signal and other feature representations could also be investigated, for instance (Lostanlen et al., 2019) combined time-frequency representations with statistical features from the power spectral density, Aytar et al. (2016) combined audio signal with video frames to create enhanced semantically rich representations, and Jeantet and Dufourq (2023) combined metadata such as time and location with spectrograms to improve detection of birds and primates. Even with studies presenting methods for network weights initialization (Dufourq et al., 2022), building a road-map to apply neural networks to natural sound classification (Stowell, 2022), or even analyzing different input frontends (Ghaffari and Devos, 2024), there is space for better evaluation of input types, pre-processing steps, network architectures, training process, so that on can adapt the models to the best input representations.

In the present study, we investigate a pipeline to evaluate important choices to improve the learned representations even for a difficult problem, such as the classification of sound captured in the wild and the code and best models are available on GitHub.[1]

In summary, we investigate:

- different time-frequency (spectrogram, mel-spectrogram, and per-channel energy normalization) as input to CNNs and different combination strategies of such representations,
- combination (fusion) of time-frequency and handcrafted features,
- different architectures, pre-training, and normalization strategies,
- and a custom loss function combining classification and quantification (see Section 2.6.4) to compare with previous results reported by Dias et al. (2021b).

Over these points, the main empirical evidence suggests that mel-spectrogram is a proper representation for the tested sound patterns and that feature combination (or fusion) can be more effective on small architectures. Smaller models require less data to train Mello and Ponti (2018), as well as reduce energy requirements for inference for a more sustainable setup (Ferro et al., 2023). Therefore, it is a step towards allowing such systems to be deployed for local processing in the wild.

The remainder of this text is organized as follows. Section 2 presents the steps followed and the materials used in our experiments. Section 3 reports the experimental results obtained with the experiments. Section 4 discusses the experimental results. Finally, Section 5 provides the conclusions and directions for future work.

2. Method

This section describes the blocks in Fig. 1, which represent the steps for training and evaluating the models and their combinations with different inputs. In the first step, the training subset presented in Table 1 was balanced by applying data augmentation approaches for the sound signal (Fig. 1 left blocks). We extracted acoustic features from the dataset during the second step (Fig. 1 bottom-center blocks) and trained and evaluated a linear classifier to create a baseline. We also extracted different time-frequency representations of the data (Fig. 1 top center blocks) and trained the neural networks following Fig. 1 rightmost blocks. The next sections detail each of these steps.

2.1. Datasets

The main dataset in our experiments (Fig. 1 leftmost side) consists of recordings collected on natural landscapes and provided by professor Mílton C. Ribeiro from the Spatial Ecology and Conservation Lab (LEEC),[2] at the São Paulo State University, Rio Claro, Brazil, as pre-

[1] github.com/fabiofelix/CNN-Input-Combination.

[2] github.com/LEEClab.

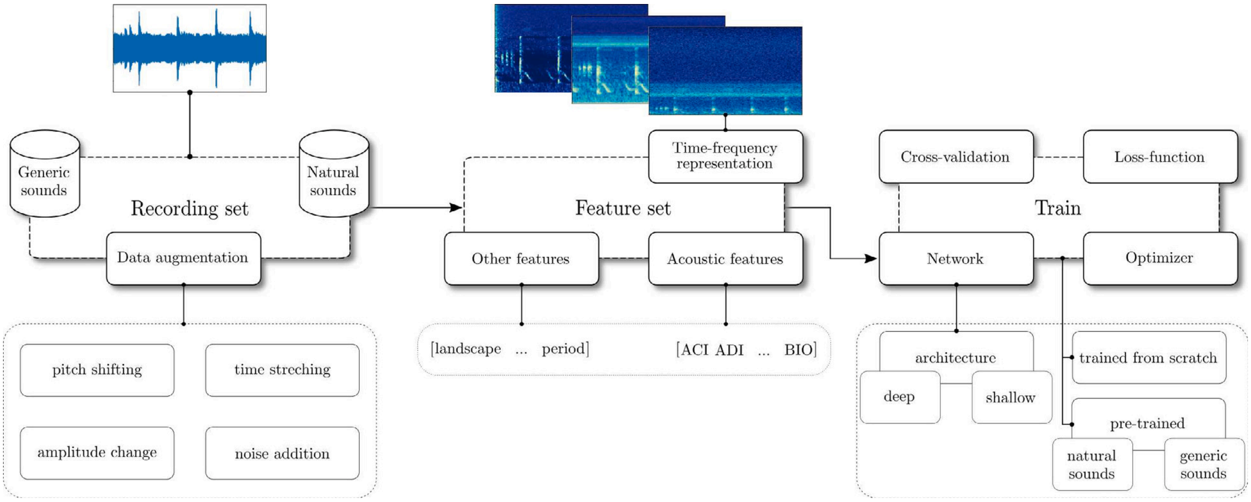


Fig. 1. Steps of the training pipeline investigated and the specific choices made. The description focuses only on the training process, and their details and evaluation are described in this text.

viously explored with other techniques (Scarpelli et al., 2021; Hilasaca et al., 2021a,b). The sounds were recorded in the Ecological Corridor of *Cantareira-Mantiqueira*, São Paulo state, Brazil, between October 2016 and January 2017, We employed audio files recorded from 5 a.m. to 8:25 a.m. (to capture bird sounds) and from 6:30 p.m. to 10:45 p.m. (to capture anuran sounds). As reported in previous work cited above, sounds were recorded in different places, named as open (mainly agricultural and pasture areas), forest (Atlantic Forest remnants), and stream (forest fragments near water bodies). All recordings are mono in WAVEform (WAV) audio format, recorded at a sampling rate of 44,100 Hz, 16-bit depth, and Pulse Code Modulation (PCM).

Moreover, following Kahl et al. (2021b), we added instances from AudioSet (Gemmeke et al., 2017) to improve representation learning by allowing models to differentiate sounds of interest from other sounds. Using the available ontology[3] of the AudioSet, we downloaded sounds from different classes and saved them with the same format, sampling rate, bit-depth, and modulation of the other recordings. AudioSet recordings have approximately 10 s and were split into clips of three seconds. To download AudioSet recordings, we employed Python and youtube-dl (v2021.4.26) library.

Bird and anuran experts labeled 12 species of interest, detailed in Table 1, following the methodology described by Gaspar (Gaspar et al., 2023), using tools like Raven Pro,[4] and labeling only one species. They annotated one vocalization per species per 1 min audio file, even when the animal called more than once. We extracted only the annotated parts from each recording, splitting them into 3-s clips and padding the clips with adjacent parts when they did not fit three seconds. Therefore, our main dataset contains only labeled parts of the recording related to events of interest. Although the audio clips may contain other sound sources, such as insects and background noise, those were not annotated. For AudioSet, we considered sounds of animals such as dogs, birds, insects, etc.; natural sounds, such as windows, thunder, rain, etc.; and human sounds, such as engines, piano, singing, etc., generating three classes. Hence, the full task considered 15 classes in total.

The resulting dataset was split using a stratified method of the classes in training (90%) and test (10%), totaling 5000 clips of tree seconds (250 min. in total), as presented in Table 1. These subsets contain data from the three recorded places (open, forest, and stream) and AudioSet sounds, ensuring the models have access to many variations, and the evaluation also covers them. Following Fig. 1 top-right blocks, we applied k-fold cross-validation with $k = 5$ in the training subset and for each iteration, one partition was used as a validation subset.

Table 1
Quantities of 3-s audio clips grouped by species and named with a short label. The column **original** communicates the number of original recordings per class. There could be file overlaps, where a single recording has multiple classes annotated, and a single animal call may generate several 3-s clips.

	Specie	Label	#original	#train	#test	Total
Bird	*Basileuterus culicivorus*	basi_culi	533	483	54	537
	Cyclarhis gujanensis	cycl_guja	428	390	43	433
	Myiothlypis leucoblephara	myio_leuc	266	411	46	457
	Pitangus sulphuratus	pita_sulp	368	352	39	391
	Vireo chivi	vire_chiv	778	724	81	805
	Zonotrichia capensis	zono_cape	631	574	64	638
				2934	327	3261
Anuran	*Adenomera marmorata*	aden_marm	94	116	13	129
	Aplastodiscus leucopigyus	apla_leuc	175	186	21	207
	Boana albopunctata	boan_albo	267	283	32	315
	Dendropsophus minutus	dend_minu	169	229	26	255
	Ischnocnema guenteri	isch_guen	114	136	15	151
	Physalaemus cuvieri	phys_cuvi	257	290	32	322
				1240	139	1379
Other		Animal	51	108	12	120
		Human	51	109	11	120
		Natural	51	109	11	120
				326	34	360
Total				4500	500	5000

We conducted additional tests to understand whether the proposed approach can be applied in other contexts. In those, we used an **additional dataset**, consisting of part of the available dataset of Task 5 from the Detection and Classification of Acoustic Scenes and Events (DCASE2024) (Liang et al., 2024). The WMW subset consists of recordings from the Western Mediterranean Wetlands Bird dataset, with 161 recordings of different lengths annotated for 26 different classes of 20 bird species. We removed classes with less than 10 samples and from the remaining 22 classes, we extracted 3604 clips of three seconds (180 min in total).

2.2. Balancing data classes

The dataset in Table 1 is slightly unbalanced and we used some data augmentation techniques (Salamon and Bello, 2017) (see Fig. 1 left blocks) to reduce possible problems, such as model poor generalization and improper predictions for samples of minority classes (Johnson and Khoshgoftaar, 2019; Wang et al., 2017b).

[3] research.google.com/audioset/ontology/index.html.

[4] ravensoundsoftware.com/software/raven-pro/.

Our augmentation process generates $m = \lceil (\max_c - \#class)/\#class \rceil$ modified copies of all 3-s training recordings and adds them to the original training subset to extract features and generate spectrograms. When cross-validation splits training files, it divides the set into 3600 files ($k - 1$ partitions) for training and 900 for validation. After that, it takes augmentations until each class in the training partitions reaches $\max_c = 580$ audio clips, generating a set with 8700 clips. The choice of 580 is to approach the majority class (*Vireo chivi*) in the training partitions. Hence, for the **main dataset**, each cross-validation iteration contains 8700 clips for training, 900 clips for validation, and 500 clips for testing. Following the same process for the **additional dataset**, with $max_c = 565$, each cross-validation iteration contains 10,282 clips for training, 649 clips for validation, and 361 for testing.

We considered *pitch shifting*, *time stretching*, *noise addition*, and *amplitude change* as proposed by Salamon and Bello (2017) to increase sound variability, and followed the same implementations and parameters used in Dias et al. (2021b).

2.3. Handcrafted features

To create the feature set depicted in the bottom-center of Fig. 1, we extracted, for each 3-s clip, 30 well-known acoustic features and measures. They were Bioacoustic Index (Bio) (Boelman et al., 2007), a set composed of Temporal Entropy (H_t), Frequency Entropy (H_f), and Acoustic Entropy Index (H) (Sueur et al., 2008b), Acoustic Complexity Index (ACI) (Pieretti et al., 2011), Acoustic Evenness Index (AEI) (Villanueva-Rivera et al., 2011), Median of Amplitude Envelope (M) and Acoustic Richness (AR) (Depraetere et al., 2012), Normalized Difference Soundscape Index (NDSI) (Kasten et al., 2012), Acoustic Diversity Index (ADI) (Pekin et al., 2012), Number of Peaks (NP) (Gasc et al., 2013), Background noise index (BGN) (Dias et al., 2022), Sound Pressure Level (SPL) (Sánchez-Gendriz and Padovese, 2016), functions that describe signal variations, such as Roughness (Ramsay, 2006) and Rugosity (Mezquida and Martínez, 2009), Root Mean Square (RMS) (Eldridge et al., 2018), the mean of the Power Spectral Density (PSD) (Welch, 1967), Signal-to-noise ratio (SNR) (Bedoya et al., 2017), and twelve Mel-frequency Cepstrum Coefficients (MFCC) (Logan, 2000). Moreover, following approaches such as (Jeantet and Dufourq, 2023), information about the place (open, stream, and forest) and period (am and pm) of the recording was incorporated. We coded these place and period features with one-hot encoding and added five new features, generating a feature vector with 35 dimensions. These features can improve the training process by giving valuable information about the environment and variations of sound patterns (Sueur et al., 2014).

We have extracted acoustic features with R packages Seewave (Sueur et al., 2008a) (v2.1.3), Soundecology (v1.3.3), and tuneR (v1.3.3). For all routines with a max frequency parameter, we set it to 22.050 Hz (ADI, AEI, BIO, and NDSI biomax param) because this is the highest frequency captured by the recordings' sampling rate. For all routines that depend on the Fourier Transform, we used a Hanning window with 1024-length and 10% of overlap (PSD, SPL, SNR, and H) to avoid elevating the processing time and memory usage. SPL and SNR values are calculated based on the PSD results. We reimplemented the seewave H index to fix memory usage problems, return H_t, H_f, Hilbert envelope (used as input for RMS and BGN), return results of the internal *meanspec* function (used as input for Roughness, NP, and M) and configure it with aforementioned Fourier parameters. AR index was calculated with the aforesaid values of H_t and M indices to reduce processing time. We also set the cluster size = 1 of soundecology ACI, allowing the calculation of 3-s files. The tuneR routine, which calculates MFCC, returns a matrix with coefficients (columns) and components (rows). Consequently, we used the column means for representing coefficients, and the first twelve of them were considered (Dias et al., 2021a). We used the default parameters of this MFCC routine. The code that extracts all features is available on github.[5]

2.4. Baseline definition

We used the Support Vector Machine (SVM) with linear kernel, $cost = 1$, $iterations = 10^6$, and the handcrafted features presented in Section 2.3 as input. In our experiments, normalization of the input values did not improve the results of SVM, thus, we did not consider normalizing the features. SVM was used owing to its lower bounds in terms of learning guarantees and its capacity to generate a proxy measure for the separability of the classes in a feature space (Mello and Ponti, 2018). We also performed cross-validation with $k = 5$ in the training subset, trained the model with the training partitions, discarded validation partitions because we did not evaluate classifier parameters, and tested with the test subset of Table 1. All routines are available in scikit-learn (v0.22.1) (Pedregosa et al., 2011) with Python programming language.

2.5. Time-frequency representations

This section describes the time-frequency representation of Fig. 1 top-center. This representation is a common approach to feed classifiers and visually analyze sound signal patterns, associating frequency spectrum and time and giving different views of the sound. We considered the spectrogram (Thomas et al., 2019; Strout et al., 2017; Casanova et al., 2022) and its variations, such as mel-spectrogram (LeBien et al., 2020; Parascandolo et al., 2016; Cakır et al., 2017) and per-channel energy normalization (PCEN) (Lostanlen et al., 2019; Cramer et al., 2020; Harvey, 2018) that is not only a time-frequency representation but also a denoising algorithm.

For all audio clips, gray-scale spectrograms (spec), mel-spectrograms (mel), and PCEN (pcen) images (256 × 256) were created with librosa (v0.7.2) routines, using a Hanning window with a length of 2048, and an overlap of 75% (or hop length of 25%). Length and overlap contribute to building a representation with suitable frequency and time resolutions to represent a large pattern variation. In addition, mel-spectrogram was configured to return 128 mel bands and PCEN used $\delta = 2.0$, $r = 0.5$, and $\alpha = 0.98$, as the original paper (Wang et al., 2017a).

2.6. Model architectures and their variations

We employed four network architectures to fulfill the train block of the Fig. 1: a simple CNN2D trained from scratch (Dias et al., 2021b); a non-hierarchical multitask model (named here as BirdVox) pre-trained on American Northeast Avian Flight Call Classification dataset (ANAFCC) (Cramer et al., 2020); ResNet-50 (He et al., 2016) and Inception-V3 (Szegedy et al., 2016) both pre-trained on ImageNet (Deng et al., 2009). ResNet-50 is a common tool to classify animal species, based on their sounds (Thomas et al., 2019; LeBien et al., 2020; Harvey, 2018) and Inception has parallel filters with different sizes that vary the architecture width and can facilitate the learning of patterns with different sizes. BirdVox achieved well-suited results to classify birds and with CNN2D is possible to compare the other models with a small and trained from scratch model. Using models pre-trained on ImageNet is a common approach for transfer learning, as reported in some references in the introduction and the review of Dufourq et al. (2022). All models were trained and evaluated with spec, mel, pcen, and combinations (fusion) of inputs, and those images were normalized with *max-norm*, dividing pixel values by 255. In cases in which we did not perform input combinations, we attempted to change the models as little as possible to maintain their original characteristics.

To define the optimizer and learning rate (lr) of the pre-trained models, we tested three optimizers with a smaller learning rate than the programming API default values. That was default divided by 10, following the guidelines of Becher and Ponti (2021): Adam ($lr = 10^{-4}$), SGD ($lr = 10^{-3}$), and RMSprop ($lr = 10^{-4}$). RMSprop was the only one that presented lower accuracy with a lower learning rate than the

[5] github.com/fabiofelix/AudioTools.

default value, therefore, we maintained the default $lr = 10^{-3}$. For the model trained from scratch, we conducted a grid search with SGD, fixing the $momentum = 0.9$ and varying the $lr \in [10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}]$ in addition to the tests with the other two optimizers. The optimizer configurations with the best average accuracy in the validation subsets were used for the tests. CNN2D has two convolutional layers followed by pooling and three dense layers (we did not use dropout). Stochastic gradient descent (SGD) optimizer (Bottou, 1998) was employed to minimize the loss function with a learning rate of 10^{-2}, momentum 0.9, 100 epochs, and batch size equal to 80. The optimizer, learning rate, and momentum increased the model results, and the batch size was limited by the memory of the video card.

We used similar preparations for both ResNet-50 and Inception-V3. Our images are gray-scale (1-channel), thus we concatenated three copies of the input (3-channel) to use as inputs for ResNet and Inception. Furthermore, we added other top layers, a global average pooling, and a dense layer to classify the specific class number used. ResNet used Adaptive moment estimation (Adam) optimizer (Kingma and Ba, 2014) with a learning rate of 10^{-4}, 100 epochs, and batch size equal to 30. This configuration enhances model results, as presented in the following sections, and the batch size also depends on the memory of the video card. Inception used RMSProp optimizer (Tieleman et al., 2012) with a learning rate of 10^{-3}, 100 epochs, and batch size equal to 80. This configuration also results in the best empirical results and the batch size is also defined by the video card limits.

With BirdVox, we replaced the input layer with our input configuration (256 × 256), removed the top 4 dense layers (64, 1, 5, and 15 units), and added another dense layer with 64 units and the same configurations of the original (Cramer et al., 2020) (He Normal initializer (He et al., 2015), L2 regularizer with factor 10^{-3}, and not using bias) and a dense layer to classify the specific class number used. Adam was employed to train the model with a learning rate of 10^{-4}, 100 epochs, and a batch size equal to 80.

Models were implemented with Python (v3.5.2) associated with Keras (v2.2.5) and TensorFlow (v1.10) libraries. To load BirdVox weights, we used the model available on birdvoxclassify (v0.2.0).

2.6.1. *Combining time-frequency representations with handcrafted features*

Inspired by an example of context-adaptive neural network in Lostanlen et al. (2019) and a combination of time-frequency representation with recording metadata (Jeantet and Dufourq, 2023), an auxiliary branch processes handcrafted features, as depicted in Fig. 2a. We analyzed three configurations of this branch: one dense layer with 128 units and ReLU activation, one batch normalization layer, or a combination of batch normalization followed by a dense layer.

We did not normalize handcrafted features before training networks. In the networks' main branch, after the flattening layer (CNN2D and BirdVox) or global average pooling layer (ResNet-50 and Inception-V3), we removed the original classification layers and placed a dense one with 128 units and ReLU activation. Additionally, to add the same layer to BirdVox, we used the model configurations (initializer, regularizer, and bias) as presented in the previous subsection. The result of the two branches' concatenation passes through one dense layer with 128 units and ReLU activation, and lastly, the dense classification layer.

2.6.2. *Combining different time-frequency representations*

Beyond individual training with spec, mel, and pcen, we conducted experiments with combinations of these representations, as presented in Fig. 2. We hypothesized that a combination of different representations could improve the learning process, similar to Xie et al. (2022). For instance, spec turns more visible higher sound harmonics, mel presents lower patterns (up to 8 kHz), and pcen shows a filtered version of the signal. In one experiment (see Fig. 2b), the representations were combined in a 3-channel input and passed to models. Thus, we changed the CNN2D input shape to deal with 3-channel. In BirdVox, after the input, we had to add a lambda layer[6] that returns the channels' mean to pass a 1D input to the model. BirdVox also used, after the flatten layer, a dense layer with 128 units and original BirdVox configurations (initializer, regularizer, and bias) before the classification layer. ResNet-50 and Inception-V3 were originally trained with 3-channel input, thus we just changed their top layers as described in Section 2.6.1.

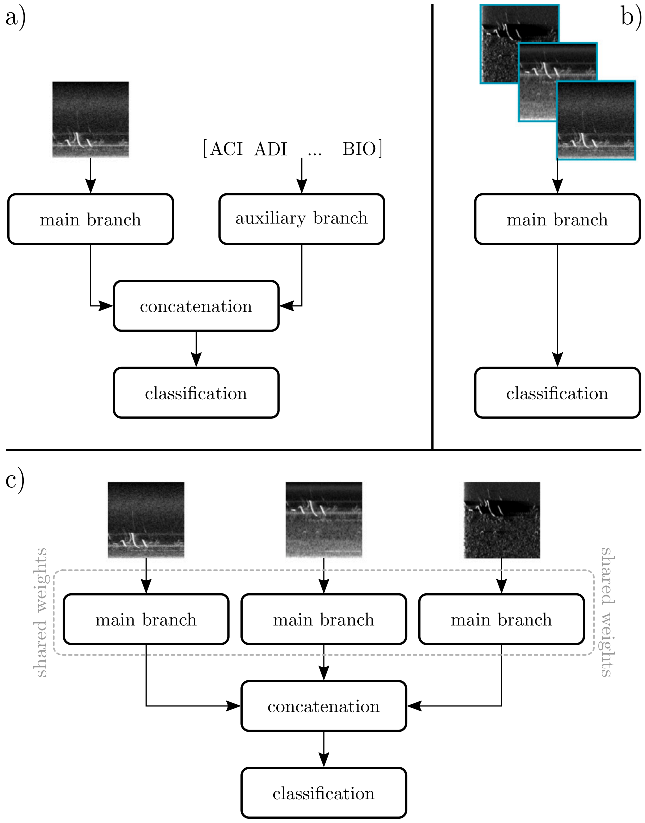


Fig. 2. Sketch of the architectures with a combination of features. (*a*) Time-frequency representations with handcrafted features, (*b*) different time-frequency representations in 3-channel, and (*c*) different time-frequency representations in 3-input. Blue borders on (*b*) only facilitate the visualization in this figure, and network inputs do not consider them.

In another experiment (see Fig. 2c), we took inspiration from the concept of Siamese networks (Bromley et al., 1994; Chopra et al., 2005) and built architectures with three equal branches (3-input) that share weights and receive different views (spec, mel, or pcen) of the same data, but did not use contrastive loss function. As branches, we considered CNN2D and BirdVox until the first dense layer after their flatten (128 units), and ResNet-50 and Inception-V3 until their global average pooling layer. The concatenation of the three branches of each model also passes through two dense layers (one with 128 units and the other with #class units) to classify the data. In the 3-input tests, we changed the batch size of the training due to the limits of the video card: CNN2D and Inception-V3 equal to 30, BirdVox equal to 70, and ResNet-50 equal to 15.

In Fig. 2b, a network learns a feature space from the early combination of the three input channels, meanwhile, in Fig. 2c, a network learns three feature spaces that represent different pattern views, and later combines them to perform the classification.

2.6.3. *Different time-frequency representations and handcrafted features*

Furthermore, we combined the two sections above. In the experiment with 3-channel input (Fig. 2b), we configured inputs as described

[6] It adds an arbitrary function or expression as a network layer.

in Section 2.6.2 and followed Section 2.6.1 to add the auxiliary branch (Fig. 2a). In the 3-input experiment (Fig. 2c), the configuration is similar to Section 2.6.2, but we configured main branches to output 128 units and combined them with the auxiliary branch as in Section 2.6.1.

2.6.4. Quantification

Finally, we also tested the custom loss function (see Fig. 1 right-top block) that combines cross-entropy and quantification, a task related to estimating class distribution in a dataset (Bella et al., 2010; Maletzke et al., 2017). The loss function was defined by Dias et al. (2021b) as

$$\ell_{CQ}(X) = \lambda_1 CCE(X) + \lambda_2 CC_{err}(f(X)), \tag{1}$$

where X is a subset (batch) of instances, CCE is the categorical cross-entropy, CC_{err} is the absolute error (absolute value of the difference between the estimated and real class distributions) of the classify and count quantification method (Beijbom et al., 2015; Gao and Sebastiani, 2016). $f(.)$ is the output of a classifier that provides the predicted class distribution, and λ_i are the respective weights. The goal is to regularize the loss function to guide the training process and compare the current results with quantification and evaluate the impact of the modifications in the dataset and training process with our prior results. To compare with our previous paper (Dias et al., 2021b), we added the loss to all models without the changes of Section 2.6 and to the changed ResNets that reached the best results. In all cases, we used as input solely the mel-spectrogram. The weights of Eq. (1) were initialized following the original paper: $\lambda_1 = \lambda_2 = 1.0$ (C1Q1), $\lambda_1 = 1.0$ and $\lambda_2 = 0.5$ (C2Q1), and $\lambda_1 = 0.5$ and $\lambda_2 = 1.0$ (C1Q2).

2.7. Evaluation

We evaluated the results with balanced accuracy score,[7] learning curves (the loss function and categorical accuracy Sokolova and Lapalme, 2009), and the recall (sensitivity) measure,[8] all available in scikit-learn. We trained models to classify 15 classes (see Section 2.1), but we assessed the balanced accuracy of the 12 classes related to the specific animal species of interest (see Table 1), to facilitate the comparison with our previous works. We also employed a two-tailed Student's t-test to compare results using paired tests with a significance level equal to 0.05.

To generate reproducible results when training SVM and neural networks, we used Python seed values (1030), following the Keras FAQ.[9] As aforementioned, we used cross-validation with $k = 5$, the model trained in each iteration is applied to the test subset, and we computed balanced accuracy mean and standard deviation to perform evaluations. The general use of the cross-validation technique does not consider intermediary models for the final evaluation. However, in our analysis, we averaged the metrics of each intermediate model applied to the test subset to determine how the models' behavior changes with slight variations of the training dataset.

Finally, the baseline was processed by an Intel Core i7-6850K CPU, 3.60 GHz, 6 cores, and 124 GB RAM. Neural network training and testing were performed with an NVidia Titan XP video card, with driver v387.26, Cuda v9.0, and cuDNN v7.0.5.15.

3. Results

This section reports the experimental results of our tests with different combinations (fusion) of CNN inputs, considering the datasets presented in Section 2.1. The first sections are related to the **main dataset** and the last section to the **additional dataset**. Such results were compared with SVM trained with handcrafted features and evaluated with balanced accuracy $\in [0, 1]$. We trained our models to classify 15 classes (species of interest + AudioSet) of sounds and used 12 of them (species of interest) to evaluate the results and compare them with our previous works. AudioSet did not change baseline accuracy but improved its recall average by 1.25 percentage points. Adding the place and period features increased our baseline accuracy in $\approx$48% (from 0.29 to 0.43). We tested two types of inputs for neural networks: images (spec, mel, and pcen) and handcrafted features. Following Fig. 2, we combined images with handcrafted features using three different auxiliary branches: dense layer (dense128), batch normalization layer (bnorm), and batch normalization with dense layer (bn+d128). We combined the images, creating a 3-dimensional tensor (3-channel) and a shared weighted architecture with three branches that receive different time-frequency representations (3-input). We also concatenated 3-channel and 3-input with the branches of handcrafted features aforesaid. Finally, we tested a weighted loss function with three weight cases, named C1Q1, C2Q1, and C1Q2 (see Section 2.6.4), to compare with current results and our prior ones. All models were trained with 5-fold cross-validation and intermediate models' results on the test subset are reported.

3.1. Combining time-frequency representations and handcrafted features

Table 2 presents results that combine images with handcrafted features. In the first value column, most mel results are greater than the results of other representations, although Student's t-tests did not reject the null hypothesis (p-value ≥ 0.05) when comparing mel/spec and mel/pcen for CNN2D and ResNet50, respectively. InceptionV3 presented great differences in any result comparison, e.g., mel yielded a result 15 percentage points greater than spec. Nevertheless, BirdVox obtained higher results with pcen than with spec (up to 7 percentage points) or than mel (up to 4 percentage points).

The addition of handcrafted features processed by *dense128* always generated means less than 0.36, while results without combinations are greater than 0.47. The random scenario has a balanced accuracy of 0.08, and some results were random, indicating the difficulty in convergence under such conditions. This also repeats in other experiments as described later.

In the sequel, columns *bnorm* and *bn+d128* have values greater than or equal to the column without such combinations, e.g., for BirdVox results, pcen with *bn+d128* yielded 0.62 against 0.56 with a smaller deviation. This appears to solve the issue of convergence. ResNet50 with spec and InceptionV3 with mel (both combined with *bn+d128*) are the only exceptions that decreased results, generating 0.60 against 0.66 in the ResNet50 case. These columns (*bnorm* and *bn+d128*) present similar results with range $[0.51, 0.77]$ and with no significant differences between their values, e.g., when comparing mel results of InceptionV3.

Combinations with mel also presented results greater than or equal to other spectral representations, except for BirdVox whose pcen results are greater than 0.60. Even with feature combinations, CNN2D and BirdVox obtained results up to 0.62, meanwhile, ResNet50 and InceptionV3 generated results up to 0.77. However, the combinations influenced more CNN2D and BirdVox (results up to 7 percentage points higher) than the deeper models. Moreover, except in column *dense128*, all results are greater than the SVM baseline. In Fig. 3, it is possible to visually verify these differences between the best results of each model and the gap between CNN2D/BirdVox and ResNet50/InceptionV3.

[7] scikit-learn.org/stable/modules/generated/sklearn.metrics.balanced_accuracy_score.html.

[8] scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html.

[9] keras.io/getting_started/faq/.

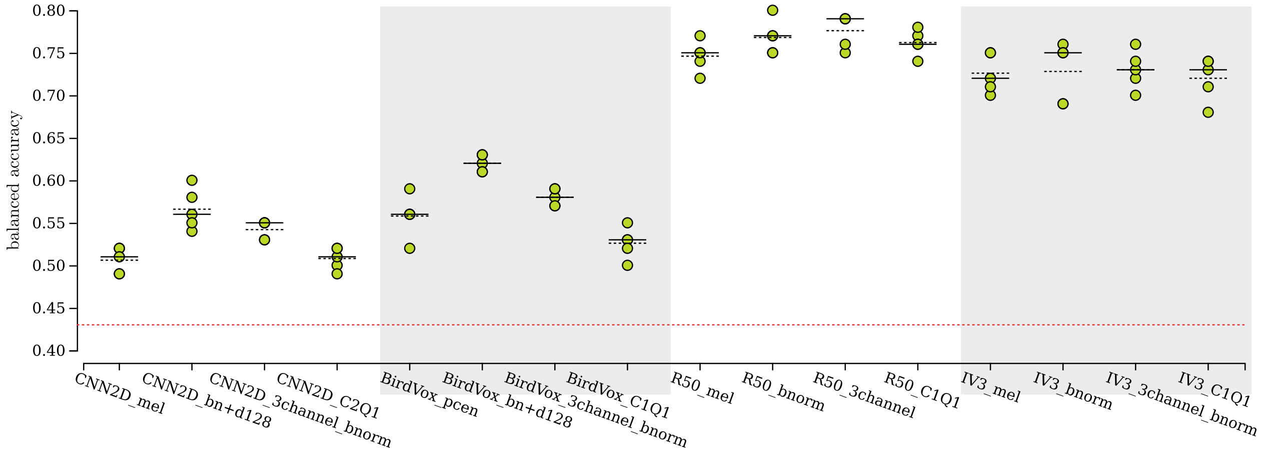


Fig. 3. Individual value plots of balanced accuracy of neural networks applied to test subset. Each plot area (white and shaded gray) shows the best results of a specific model (from left to right: CNN2D, BirdVox, ResNet50, and InceptionV3). There is an overlap between points that generates model results with a different number of points. Solid and dotted lines in each group communicate median and mean values, respectively. The long dotted red line on the bottom is the SVM balanced accuracy mean.

Table 2
Mean and standard deviation of balanced accuracy of the models applied to the test subset. The first result column presents results for models with images as input and the remaining columns combine these representations with handcrafted features processed by different branch layers. Bold values highlight the best results of each model (greatest mean and lowest standard deviation).

			dense128	bnorm	bn+d128
CNN2D	spec	0.49±0.02	0.08±0.00	0.56±0.01	0.56±0.02
	mel	0.51±0.02	0.08±0.00	0.56±0.02	**0.57 ±0.02**
	pcen	0.48±0.02	0.08±0.00	0.54±0.02	0.55±0.01
BirdVox	spec	0.49±0.02	0.32±0.11	0.51±0.01	0.51±0.02
	mel	0.52±0.04	0.34±0.15	0.52±0.02	0.54±0.02
	pcen	0.56±0.02	0.35±0.13	0.61±0.03	**0.62±0.01**
ResNet50	spec	0.66±0.04	0.24±0.16	0.68±0.02	0.60±0.13
	mel	0.75±0.02	0.25± 0.17	**0.77 ±0.02**	0.77±0.02
	pcen	0.70±0.08	0.25±0.17	0.75±0.02	0.73±0.02
InceptionV3	spec	0.58±0.04	0.08±0.00	0.63±0.02	0.62±0.03
	mel	**0.73±0.02**	0.20±0.26	0.73±0.03	0.71±0.02
	pcen	0.66±0.04	0.09±0.01	0.68±0.03	0.69±0.02

SVM (baseline) 0.43 ± 0.02.

Table 3
Mean and standard deviation of balanced accuracy of the models applied to the test subset. First two result columns present model results with combinations of image inputs and the remaining columns combine these representations with handcrafted features processed by different branch layers. Bold values highlight the best results of each model (greatest mean and lowest standard deviation).

			dense128	
	3-input	3-channel	3-input	3-channel
CNN2D	0.49±0.01	0.49±0.02	0.08±0.00	0.08±0.01
BirdVox	0.31±0.21	0.57±0.02	0.21±0.18	0.43±0.13
ResNet50	0.10± 0.04	**0.78±0.02**	0.09±0.01	0.25±0.18
InceptionV3	0.08±0.00	0.73±0.03	0.10±0.04	0.19±0.24
	bnorm		bn+d128	
	3-input	3-channel	3-input	3-channel
CNN2D	0.53±0.02	**0.54±0.01**	0.53±0.03	0.54±0.02
BirdVox	0.21±0.19	**0.58±0.01**	0.35±0.20	0.57±0.02
ResNet50	0.15±0.09	0.77±0.02	0.24±0.03	0.76±0.01
InceptionV3	0.31±0.02	**0.73±0.02**	0.34±0.03	0.68±0.04

SVM (baseline) 0.43 ± 0.02.

3.2. Combining different time-frequency representations and handcrafted features

Table 3 reports results that combine three time-frequency image representations and handcrafted features. In the first two columns, the results of 3-channel are greater than or equal to the results of 3-input. For instance, ResNet50 yielded a result with 3-channel 68 percentage points greater than the result with 3-input.

Equally to Table 2, when we used a *dense128* to combine other features, the results decreased mainly for 3-channel, e.g., from 0.73 to 0.19 for InceptionV3. In the following, the majority of comparisons between *bnorm* (or *bn+dense128*) and the column without it presented similar results (null hypothesis was not rejected with p-value ≥ 0.05). However, all CNN2D results, ResNet50 (3-input and *bn+d128*) and InceptionV3 (3-input and *bnorm* or *bn+d128*) presented significant increments. For example, for InceptionV3, while the first 3-input column presents 0.08, the combination with *bn+dense128* shows 0.34.

Both additions, with *bnorm* and *bn+dense128* yielded similar results, except for InceptionV3 and 3-channel, in which *bn+dense128* decreased results of *bnorm* by 5 percentage points. In all combinations, 3-channel obtained results greater than or equal to 3-input, e.g., ResNet50 with *bnorm*, 0.77 against 0.15. CNN2D and BirdVox yielded results up to 0.58, meanwhile, ResNet50 and InceptionV3 obtained results up to 0.78.

All 3-channel columns, except the ones with *dense128* and InceptionV3 with *bn+d128*, presented results greater than or equal to Table 2 first result column. An example is BirdVox with 3-channel and *bnorm* and BirdVox with pcen input obtained, respectively, 0.58 and 0.56 balanced accuracy mean. In all 3-channel cases, except with *dense128*, results are greater than the SVM baseline.

A comparison between combinations in Tables 2 and 3 highlights that CNN2D and BirdVox presented higher results with handcrafted features addition than using 3-channel, e.g, BirdVox with pcen and *bn+d128* obtained 0.62, with 3-channel input yielded 0.57, and with 3-channel and *bnorm* generated an intermediate result of 0.58 (see Fig. 3). However, comparisons of ResNet50 results or InceptionV3 results did not reject the null hypothesis of the statistical test (p-value ≥ 0.05), presenting similarity between results. For instance, ResNet50 with mel (or 3-channel) and *bnorm* generated 0.77, with 3-channel input obtained 0.78.

3.3. Quantification

Table 4 describes results with the quantification loss function. We considered only mel as input because this representation presented the best results in Table 2 (column without combinations) and to compare with our previous works. There are no significant differences (null hypothesis was not rejected with p-value ≥ 0.05) between balanced

Table 4
Mean and standard deviation of balanced accuracy of the models applied to the test subset. Results were generated with mel-spectrogram as input. Bold values highlight the best results of each model (greatest mean and lowest standard deviation). Columns communicate the weights of the loss function: $\lambda_1 = \lambda_2 = 1.0$ (C1Q1), $\lambda_1 = 1.0$ and $\lambda_2 = 0.5$ (C2Q1), and $\lambda_1 = 0.5$ and $\lambda_2 = 1.0$ (C1Q2).

	mel	C1Q1	C2Q1	C1Q2
CNN2D	0.51±0.02	0.51±0.02	**0.51±0.01**	0.50±0.01
BirdVox	0.52±0.04	**0.53±0.02**	0.53±0.02	0.53±0.02
ResNet50	0.75±0.02	**0.76±0.01**	0.75±0.04	0.76±0.01
InceptionV3	**0.73±0.02**	0.72±0.03	0.71±0.03	0.71±0.03

SVM (baseline) 0.43 ± 0.02.

Table 5
ResNet50 best results with test subset. First columns describe the principal configurations of the models and the first two rows communicate our earlier research. Rows 6 and 8 were calculated to show the impact of the quantification on the current best results. Highlighted row presents the highest balanced accuracy and the other bold values emphasize results that overlap, considering the confidence interval.

Input	Optimizer	Quantif.	B.Acc.
mel Dias et al. (2021b)	SGD	–	0.52±0.03
mel Dias et al. (2021b)	SGD	C2Q1	0.52±0.01
mel	Adam	–	0.75±0.02
mel	Adam	C1Q1	**0.76±0.01**
mel+bnorm feats	Adam	–	**0.77± 0.02**
mel+bnorm feats	Adam	C2Q1	**0.77±0.02**
3-channel	**Adam**	**–**	**0.78±0.02**
3-channel	Adam	C1Q1	**0.77±0.02**

SVM (baseline) 0.43 ± 0.02.

accuracy or between recall of models with and without quantification. Furthermore, in Fig. 3, we can verify a comparison of balanced accuracy between models with quantification and the combinations previously described. To perform one more test beyond the basic evaluation, we calculated the silhouette coefficient of the training subset (4500 samples) using the features extracted by the penultimate layer of each model. In all cases, quantification increased silhouette values by 0.02 points (for example, ResNet50 from 0.34 to 0.36), except for InceptionV3, where the silhouette decreased from 0.40 to 0.36. Even then, the quantification did not have power to change the learned space.

3.4. The best results comparison

Table 5 presents a comparison of the best results of Tables 2–4, and Dias et al. (2021b). ResNet50 with mel, combination processed with *bnorm*, and *3-channel* input attained proper balanced accuracy, as aforementioned. To compare with quantification results, we used the custom loss function to train the best models that combine inputs and present their results. In this comparison, there are high differences between the results in Dias et al. (2021b) and the current results, mainly because we changed the optimizer algorithm and related configurations based on the description of Section 2.6. Moreover, the null hypothesis of the Student's t-test was not rejected (p-value ≥ 0.05) when comparing results with and without quantification, showing significant similarity between them.

3.5. Evaluation of learning curves

Fig. 4 presents learning curves of the best results for each network model aforesaid. In general, training reached valleys where loss and accuracy remain monotonic. CNN2D started decreasing loss (training and validation) and increasing accuracy (training and validation), and around the 20th epoch, the model overfitted, increasing the difference between training and validation curves. ResNet50 presented the lowest differences between training and validation curves: ≈1.10 (loss) and ≈0.19 (accuracy), and InceptionV3 presents the highest variation of validation curves.

Table 6
Mean and standard deviation of balanced accuracy of the models trained and applied to **additional data**. Highlighted rows present the higher balanced accuracy of each model.

Model	Input	B.Acc.
BirdVox	pcen	0.83±0.02
	pcen+bn+dense128	0.85±0.03
	3-input	0.88±0.02
	3-input+bn+dense128	0.89±0.03
	3-channel	0.87±0.07
	3-channel+bn+dense128	**0.91±0.01**
ResNet50	**mel**	**0.90±0.02**
	mel+bn+dense128	0.89±0.02
	3-input	0.49±0.15
	3-input+bnorm	0.62±0.10
	3-channel	0.89±0.02
	3-channel+bnorm	0.86±0.04

SVM (baseline) 0.76 ± 0.04.

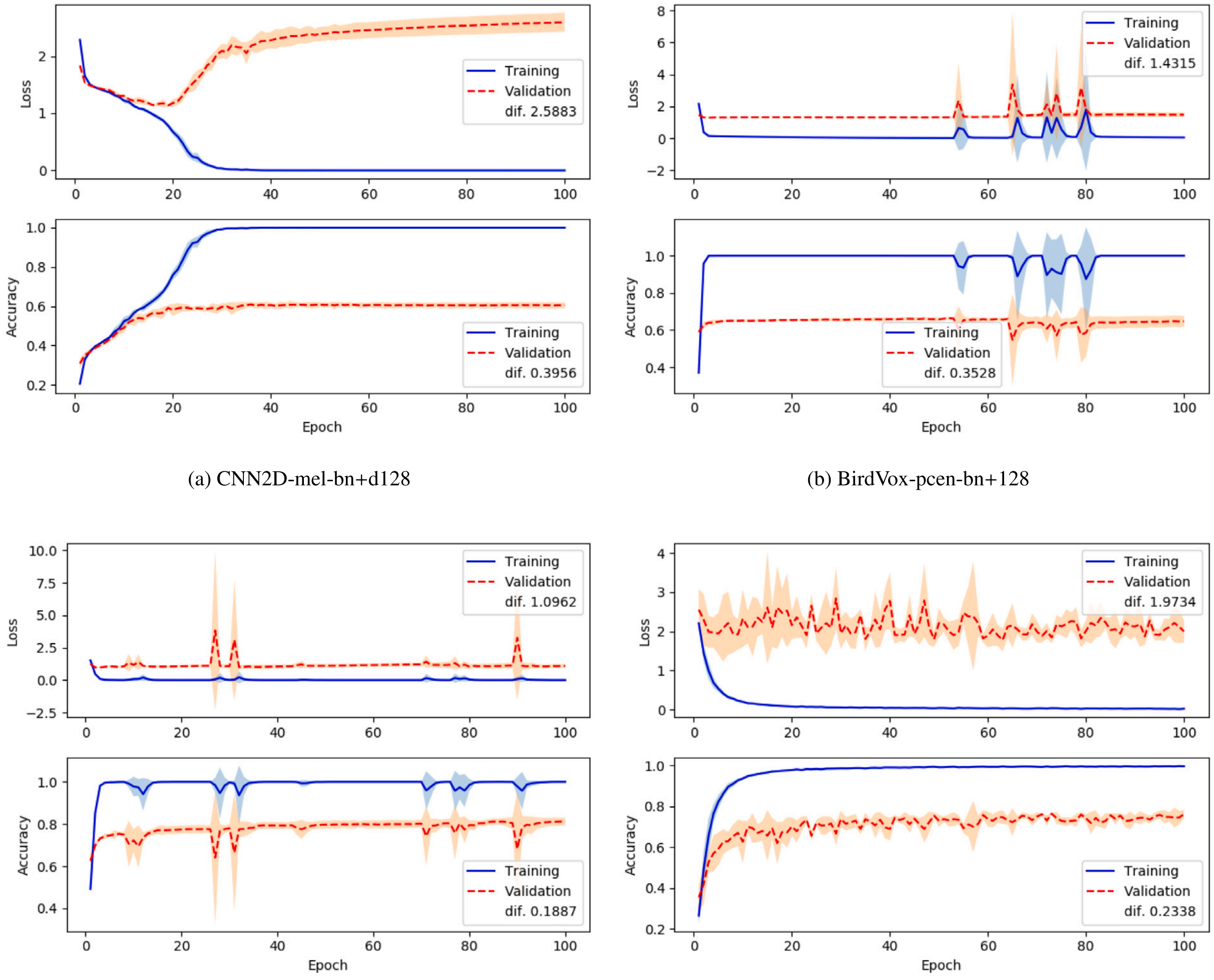
3.6. Additional results

Considering an additional dataset, we followed the same procedure presented in Section 2. However, we did not add AudioSet or apply quantification because they did not influence the accuracy. Besides, we did not consider information about the place and period because they were unavailable in the dataset, using a handcrafted feature vector with 30 dimensions. Additionally, for tests such as presented in Table 2, we only report mel (CNN2D, ResNet50, and InceptionV3) and pcen (BirdVox) because they returned the best results.

Table 6 reports the models and combinations with the best results, which presented values above the SVM baseline. Models with different depths, BirdVox with three convolutional layers and ResNet50 with more than 50 convolutional layers presented similar results. Independent of the time-frequency representation (pcen, 3-input, or 3-channel), all BirdVox tests with handcrafted features did not present significant differences (null hypothesis was not rejected with p-value ≥ 0.05) to the tests without these additions, e.g., 3-input and its counterpart with *bn+d128*, balanced accuracy respectively 0.88 and 0.89. Moreover, 3-input and 3-channel tests did not present differences between them but outperformed pcen results in most cases. Results of pcen are in the interval $[0.78, 0.85]$, 3-input results in $[0.86, 0.89]$, and 3-channel results in $[0.86, 0.91]$.

In ResNet50 tests, most time-frequency representations combined with handcrafted features did not present significant differences from their versions without combinations. One exception is 3-input with *dense128* (not in Table 6) with balanced accuracy 0.63 ± 0.11. Tests with 3-input dropped the results, for example, in 3-input versus mel, respectively, 0.49 and 0.90. Meanwhile, 3-channel maintained equivalent results to mel tests (null hypothesis was not rejected with p-value ≥ 0.05). Results of mel are in the interval $[0.63, 0.90]$, 3-input results in $[0.34, 0.62]$, and 3-channel results in $[0.63, 0.89]$.

In addition to the results in Table 6, CNN2D presented results close to the random scenario with a balanced accuracy of 0.05 in all time-frequency inputs and their combinations using *dense128*, but the other combinations reached results up to 0.57. Lastly, InceptionV3 achieved results in a random scenario only for 3-input without combinations, and in the other tests, it generated balanced accuracy up to 0.40.



(a) CNN2D-mel-bn+d128

(b) BirdVox-pcen-bn+128

(c) ResNet50-3channel

(d) InceptionV3-mel

Fig. 4. Learning curves of the best models for each architecture. Central curves are the mean of the cross-validation steps and the shaded area around them is the deviation. Dif. values are the difference between the training and validation in the last epoch. Accuracy curves are related to categorical accuracy.

4. Discussion

Over this discussion, the main tests are related to the first sections of Section 3 and the additional tests are related to Section 3.6. In general, with the configurations used, mel-spectrogram yielded the best results, both with and without combining (fusing) other features. For example, for ResNet50 without combinations in the main datasets, mel generated results up to 14% greater than other representations, and for InceptionV3 using batch normalization to process handcrafted features, mel obtained results up to 16% greater than the other image inputs. We suggest the expansion performed by mel-spectrogram in low-frequencies highlighted important patterns that are not properly recognizable in the regular spectrogram and PCEN attenuated important information to discriminate our specific sound patterns. However, mel-scale representations were designed for sounds perceivable by human hearing (Logan, 2000) and may not accurately represent some animals' vocalization. Generally speaking, the choice of representation depends on the characteristics of the event of interest or the application, e.g., source separation and sound synthesis also depend on the signal phase, which is discarded by this representation (Purwins et al., 2019). When checking the additional results, BirdVox with PCEN achieved results close to ResNet50 with mel-spectrogram, a behavior that reinforces the need to evaluate the best representation for each scenario (Stowell, 2022) and that input compression and normalization, as in PCEN approach, can aid models in noisy environments (Ghaffari and Devos, 2024).

The branch that processes handcrafted features solely with dense layers attained poor results (less than 0.36 in the main tests) because we did not pre-process the input features with normalization. In the SVM baseline, normalization did not improve results, hence we also used features in neural networks without pre-processing steps to guarantee a fair comparison, and because part of the tests has normalization layers. Notwithstanding, the other combinations generated results greater than or equal to architecture without combinations, e.g., in the main tests, BirdVox with PCEN and combined with batch normalization auxiliary branch obtained results 5 percentage points greater than the architecture with the same input but without combinations. Moreover, the branch with batch normalization generated results with significant similarity (null hypothesis was not rejected with p-value $\geq$ 0.05) to the branch with batch normalization and dense layer, nonetheless, with fewer trainable parameters added to the main branch. It is also important to pay attention to the context because, in the additional tests, BirdVox with PCEN and a dense layer obtained results comparable to other combinations and any version without handcrafted features. These results highlight the possibility of normalizing the inputs based on the input needs, a common practice with neural networks. In addition, as in our tests, it is possible to use batch normalization or layer normalization inside the model structure to normalize features or samples over the mini-batches.

Unlike the other models, BirdVox presented better results with PCEN and combinations, instead of mel-spectrogram input. Furthermore, in the additional tests, the same association of model and input reached results similar to a deep model like ResNet50. The model was pre-trained with PCEN and we can also tune these representation parameters manually or learn them with a specific network architecture (Wang et al., 2017a) to achieve higher results. In addition, applying noise-reduction such as PCEN can improve the results of neural networks detecting animal sounds (Ghaffari and Devos, 2024; Allen et al., 2021).

In the sequel, 3-channel input obtained results greater than or equal to 3-input regardless of combining or not with handcrafted features. For instance, for ResNet50 in the main tests, 3-channel obtained results almost 8× (without other combinations) and 5× (combined with batch normalization) greater than 3-input. We hypothesize that the gradient was not sufficient to update the shared weights and the origin of ResNet50 and InceptionV3, trained with 3-dimensional inputs, also contributed to 3-channel higher results. Again, the combination of handcrafted features processed by a dense layer decreased network results (less than 0.45 in the main tests). All in all, in the main tests, combinations with handcrafted features did not change results significantly (null hypothesis was not rejected with p-value ≥ 0.05). However, these combinations yielded proper results in CNN2D, in all combinations of InceptionV3 and 3-input, and in ResNet50 with 3-input combined with batch normalization and a dense layer. In the additional tests, 3-channel and 3-input presented similar results in BirdVox and followed the same previous patterns for ResNet50, with 3-input showing lower results.

Also in the main tests, the input combinations impacted more on CNN2D and BirdVox than on ResNet50 and InceptionV3 (see Fig. 3) but in the additional tests it did not impact BirdVox results because the results are higher even without the combinations. Similar to our main tests, Jeantet and Dufourq (2023) also incorporated extra features into small architectures and achieved result improvements. We suggest the auxiliary branch of the network, with few layers and limited output size, compared with the main branch, could not influence the results of deep architectures, such as ResNet50 (more than 50 convolutional layers) and InceptionV3 (more than 90 convolutional layers).

The use of quantification to regularize the loss function, following (Dias et al., 2021b), provided no relevant changes of balanced accuracy (±1 percentage point) or class recall. Silhouette coefficients were also similar with/without quantification, showing increments of 0.02 point, except for InceptionV3, which yielded smaller silhouette values (−0.04 point), which is related to the decrease in the model accuracy. These results are similar to the prior work that showed that the quantification loss function did not influence accuracy but generated a subtle increase in silhouette values.

A general comparison with our previous paper (Dias et al., 2021b) reinforced the importance of refining the optimization setup based on the data and the task. With suitable configurations, the ResNet yielded results up to 50% greater than our prior results. Generally speaking, Adam adapts its rates during the training process (Kingma and Ba, 2014) and is appropriate to fine-tune models, and SGD can achieve better results using momentum (Becher and Ponti, 2021). More empirical information about choosing and tuning optimizers can be found in Becher and Ponti (2021).

With the evaluation of learning curves, one can notice that independent of architecture depth, width, or training approach (pre-trained or not), it was difficult to properly generalize the models (training loss around zero and validation loss > 1), attaining balanced accuracy up to 0.78. In a scenario with a small number of labeled samples (less than 10K) to train a deep learning model, we suggest, for instance, inspecting variations of architecture parameters, regularization, early stopping, and training techniques to improve convergence and model generalization.

5. Conclusion

This paper addressed a series of tests with combinations (fusions) of inputs to improve neural network representation of natural sound patterns. We have tested four architectures, one trained from scratch and the others pre-trained with images from natural sounds and a general image dataset. We also performed tests in two different datasets, one with bird and anuran sounds and the second one with only birds but with more species. The empirical evidence suggests mel-spectrogram is a proper representation of our datasets, except for BirdVox in which PCEN is the best choice. A combination of image inputs with handcrafted features can be implemented with the addition of a simple branch that contains a batch normalization layer. These combinations are suitable for small architectures, for example, with two or three convolutional layers, as BirdVox, but generated subtle improvements in ResNet50 and InceptionV3, even with a 3-dimensional representation of spectrogram variations. Lastly, the quantification loss function presented similar results to our earlier work, but future tests can be considered to improve quantification predictions along with searching for suitable weight values.

Overall, using a larger model transfers better, which may indicate there is no need to design specific architectures, but leveraging pre-trained models of different depths as best as possible corroborates with (Dufourq et al., 2022). Indeed, combination techniques are much more effective on small networks when we compare the results of networks with and without input combinations.

Handcrafted features and combined inputs improved or maintained the performance, which may be good practice for general applications of Soundscape Ecology. Also, generalization issues are still present, which may generate poor classification with errors greater than 20%, and it is still a matter for future investigation. Thus, future work is demanded to assess other inputs and their parameters, test other architectures such as transformers, evaluate regularization approaches to improve model generalization, improve of the combination (fusion) of the features as in Dai et al. (2021), Jatavallabhula et al. (2023), and investigate the applicability of other approaches, such as (Kong et al., 2020; Cramer et al., 2019; Guzhov et al., 2022). Besides, it is important to conduct a deeper evaluation of optimizers and their parameters in sound identification.

CRediT authorship contribution statement

Fábio Felix Dias: Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Investigation, Formal analysis, Data curation. **Moacir Antonelli Ponti:** Writing – review & editing, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization. **Rosane Minghim:** Writing – review & editing, Visualization, Supervision, Investigation, Funding acquisition, Conceptualization.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work, the author(s) used Grammarly to refine the text language and readability. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, FAPESP (grant 2019/07316-0 and 2021/08322-3), and CNPq (National Council of Technological and Scientific Development) grant 304266/2020-5. The authors would like to thank Professor Mílton C. Ribeiro from the São Paulo State University, Rio Claro, Brazil, for his data and useful feedback.

Data availability

The code for this work is published on GitHub at the link at the end of the introduction. The same repository provides links to the best models' weights. The main dataset, managed by another research group, will soon be available on their website, with a link in the dataset section. Moreover, the additional dataset (DCASE2024) and Google AudioSet are accessible online on their citations and respective web pages, also presented in the dataset section.

References

Allen, A.N., Harvey, M., Harrell, L., Jansen, A., Merkens, K.P., Wall, C.C., Cattiau, J., Oleson, E.M., 2021. A convolutional neural network for automated detection of humpback whale song in a diverse, long-term passive acoustic dataset. Front. Mar. Sci. 8.

Aytar, Y., Vondrick, C., Torralba, A., 2016. Soundnet: Learning sound representations from unlabeled video. In: Advances in Neural Information Processing Systems. pp. 892–900.

Becher, A.R., Ponti, M.A., 2021. Optimization Matters: Guidelines to improve representation learning with deep networks. In: Anais do XVIII Encontro Nacional de Inteligência Artificial e Computacional, SBC. pp. 595–606.

Bedoya, C., Isaza, C., Daza, J.M., López, J.D., 2017. Automatic identification of rainfall in acoustic recordings. Ecol. Indic. 75, 95–100.

Beijbom, O., Hoffman, J., Yao, E., Darrell, T., Rodriguez-Ramirez, A., Gonzalez-Rivero, M., Guldberg, O.H., 2015. Quantification in-the-wild: data-sets and baselines. arXiv preprint arXiv:1510.04811.

Bella, A., Ferri, C., Hernández-Orallo, J., Ramirez-Quintana, M.J., 2010. Quantification via probability estimators. In: 2010 IEEE International Conference on Data Mining. IEEE, pp. 737–742.

Boelman, N.T., Asner, G.P., Hart, P.J., Martin, R.E., 2007. Multi-trophic invasion resistance in hawaii: bioacoustics, field surveys, and airborne remote sensing. Ecol. Appl. 17, 2137–2144.

Bottou, L., 1998. Online algorithms and stochastic approximations. In: Saad, D. (Ed.), Online Learning and Neural Networks. Cambridge University Press, Cambridge, UK, URL: http://leon.bottou.org/papers/bottou-98x. (Revised October 2012).

Bradfer-Lawrence, T., Gardner, N., Bunnefeld, L., Bunnefeld, N., Willis, S.G., Dent, D.H., 2019. Guidelines for the use of acoustic indices in environmental research. Methods Ecol. Evol. 10, 1796–1807.

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R., 1994. Signature verification using a siamese time delay neural network. Adv. Neural Inf. Process. Syst. 6.

Cakir, E., Adavanne, S., Parascandolo, G., Drossos, K., Virtanen, T., 2017. Convolutional recurrent neural networks for bird audio detection. In: 2017 25th European Signal Processing Conference. EUSIPCO, IEEE, pp. 1744–1748.

Cakır, E., Parascandolo, G., Heittola, T., Huttunen, H., Virtanen, T., 2017. Convolutional recurrent neural networks for polyphonic sound event detection. IEEE/ ACM Trans. Audio Speech Lang. Process. 25, 1291–1303.

Casanova, E., Weber, J., Shulby, C.D., Junior, A.C., Gölge, E., Ponti, M.A., 2022. YourTTS: Towards zero-shot multi-speaker TTS and zero-shot voice conversion for everyone. In: International Conference on Machine Learning. PMLR, pp. 2709–2720.

Chopra, S., Hadsell, R., LeCun, Y., 2005. Learning a similarity metric discriminatively, with application to face verification. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. (CVPR'05), Vol. 1, IEEE, pp. 539–546.

Cramer, J., Lostanlen, V., Farnsworth, A., Salamon, J., Bello, J.P., 2020. Chirping up the right tree: Incorporating biological taxonomies into deep bioacoustic classifiers. In: ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, IEEE, pp. 901–905.

Cramer, A.L., Wu, H.-H., Salamon, J., Bello, J.P., 2019. Look, listen, and learn more: Design choices for deep audio embeddings. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 3852–3856.

Dai, Y., Gieseke, F., Oehmcke, S., Wu, Y., Barnard, K., 2021. Attentional feature fusion. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. WACV, pp. 3560–3569.

Delcroix, M., Kinoshita, K., Hori, T., Nakatani, T., 2015. Context adaptive deep neural networks for fast acoustic model adaptation. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, IEEE, pp. 4535–4539.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. Ieee, pp. 248–255.

Depraetere, M., Pavoine, S., Jiguet, F., Gasc, A., Duvail, S., Sueur, J., 2012. Monitoring animal diversity using acoustic indices: Implementation in a temperate woodland. Ecol. Indic. 13, 46–54.

Dias, F.F., Pedrini, H., Minghim, R., 2021a. Soundscape segregation based on visual analysis and discriminating features. Ecol. Inform. 61, 101184.

Dias, F.F., Ponti, M.A., Minghim, R., 2021b. A classification and quantification approach to generate features in soundscape ecology using neural networks. Neural Comput. Appl. 34, 1923–1937.

Dias, F.F., Ponti, M.A., Minghim, R., 2022. Implementing simple spectral denoising for environmental audio recordings. arXiv preprint arXiv:2201.02099.

Dröge, S., Martin, D.A., Andriafanomezantsoa, R., Burivalova, Z., Fulgence, T.R., Osen, K., Rakotomalala, E., Schwab, D., Wurz, A., Richter, T., et al., 2021. Listening to a changing landscape: Acoustic indices reflect bird species richness and plot-scale vegetation structure across different land-use types in north-eastern Madagascar. Ecol. Indic. 120, 106929.

Dufourq, E., Batist, C., Foquet, R., Durbach, I., 2022. Passive acoustic monitoring of animal populations with transfer learning. Ecol. Inform. 70, 101688.

Eldridge, A., Guyot, P., Moscoso, P., Johnston, A., Eyre-Walker, Y., Peck, M., 2018. Sounding out ecoacoustic metrics: Avian species richness is predicted by acoustic indices in temperate but not tropical habitats. Ecol. Indic. 95, 939–952.

Farina, A., Gage, S.H., 2017. Ecoacoustics: a new science. Ecoacoustics: Ecol. Role Sounds 1–11.

Ferro, M., Silva, G.D., de Paula, F.B., Vieira, V., Schulze, B., 2023. Towards a sustainable artificial intelligence: A case study of energy efficiency in decision tree algorithms. Concurr. Comput.: Pr. Exp. 35, e6815.

Gao, W., Sebastiani, F., 2016. From classification to quantification in tweet sentiment analysis. Soc. Netw. Anal. Min. 6, 19.

Gasc, A., Sueur, J., Pavoine, S., Pellens, R., Grandcolas, P., 2013. Biodiversity sampling using a global acoustic approach: contrasting sites with microendemics in New Caledonia. PLoS One 8, e65311.

Gaspar, L.P., Scarpelli, M.D.A., Oliveira, E.G., Alves, R.S.-C., Gomes, A.M., Wolf, R., Ferneda, R.V., Kamazuka, S.H., Gussoni, C.O., Ribeiro, M.C., 2023. Predicting bird diversity through acoustic indices within the atlantic forest biodiversity hotspot. Front. Remote. Sens. 4, 1283719.

Gemmeke, J.F., Ellis, D.P.W., Freedman, D., Jansen, A., Lawrence, W., Moore, R.C., Plakal, M., Ritter, M., 2017. Audio set: An ontology and human-labeled dataset for audio events. In: Proc. IEEE ICASSP 2017. New Orleans, LA.

Ghaffari, H., Devos, P., 2024. On the role of audio frontends in bird species recognition. Ecol. Inform. 81, 102573.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 580–587.

Guzhov, A., Raue, F., Hees, J., Dengel, A., 2022. Audioclip: Extending clip to image, text and audio. In: ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, pp. 976–980.

Harvey, M., 2018. Acoustic detection of humpback whales using a convolutional neural network. URL: https://ai.googleblog.com/2018/10/acoustic-detection-of-humpback-whales.html.

He, K., Zhang, X., Ren, S., Sun, J., 2015. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In: Proceedings of the IEEE International Conference on Computer Vision. ICCV.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

He, T., Zhang, Z., Zhang, H., Zhang, Z., Xie, J., Li, M., 2019. Bag of tricks for image classification with convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 558–567.

Hilasaca, L.M.H., Gaspar, L.P., Ribeiro, M.C., Minghim, R., 2021a. Visualization and categorization of ecological acoustic events based on discriminant features. Ecol. Indic. 126, 107316.

Hilasaca, L.H., Ribeiro, M.C., Minghim, R., 2021b. Visual active learning for labeling: A case for soundscape ecology data. Information 12, 265.

Jatavallabhula, K., Kuwajerwala, A., Gu, Q., Omama, M., Chen, T., Li, S., Iyer, G., Saryazdi, S., Keetha, N., Tewari, A., Tenenbaum, J., de Melo, C., Krishna, M., Paull, L., Shkurti, F., Torralba, A., 2023. Conceptfusion: Open-set multimodal 3d mapping. Robot.: Sci. Syst. (RSS).

Jeantet, L., Dufourq, E., 2023. Improving deep learning acoustic classifiers with contextual information for wildlife monitoring. Ecol. Inform. 77, 102256.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T., 2014. Caffe: Convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM International Conference on Multimedia. pp. 675–678.

Johnson, J.M., Khoshgoftaar, T.M., 2019. Survey on deep learning with class imbalance. J. Big Data 6, 1–54.

Kahl, S., Denton, T., Klinck, H., Glotin, H., Goëau, H., Vellinga, W.-P., Planqué, R., Joly, A., 2021a. Overview of birdclef 2021: Bird call identification in soundscape recordings. In: CLEF (Working Notes). pp. 1437–1450.

Kahl, S., Wood, C.M., Eibl, M., Klinck, H., 2021b. BirdNET: A deep learning solution for avian diversity monitoring. Ecol. Inform. 61, 101236.

Kasten, E.P., Gage, S.H., Fox, J., Joo, W., 2012. The remote environmental assessment laboratory's acoustic library: An archive for studying soundscape ecology. Ecol. Inform. 12, 50–67.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kirsebom, O.S., Frazao, F., Simard, Y., Roy, N., Matwin, S., Giard, S., 2020. Performance of a deep neural network at detecting North Atlantic right whale upcalls. J. Acoust. Soc. Am. 147, 2636–2646.

Kong, Q., Cao, Y., Iqbal, T., Wang, Y., Wang, W., Plumbley, M.D., 2020. PANNs: Large-scale pretrained audio neural networks for audio pattern recognition. IEEE/ ACM Trans. Audio Speech Lang. Process. 28, 2880–2894.

Krause, B., 1987. Bioacoustics, habitat ambience in ecological balance. Whole Earth Rev. 57, 14–18.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. 25.

Laiolo, P., 2010. The emerging significance of bioacoustics in animal species conservation. Biol. Cons. 143, 1635–1645.

LeBien, J., Zhong, M., Campos-Cerqueira, M., Velev, J.P., Dodhia, R., Ferres, J.L., Aide, T.M., 2020. A pipeline for identification of bird and frog species in tropical soundscape recordings using a convolutional neural network. Ecol. Inform. 101113.

Liang, J., Nolasco, I., Ghani, B., Phan, H., Benetos, E., Stowell, D., 2024. Mind the domain gap: a systematic analysis on bioacoustic sound event detection. URL: https://arxiv.org/abs/2403.18638 arXiv:2403.18638.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., Dollar, P., 2017. Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision. ICCV.

Logan, B., 2000. Mel frequency cepstral coefficients for music modeling. In: In International Symposium on Music Information Retrieval.

Lostanlen, V., Salamon, J., Farnsworth, A., Kelling, S., Bello, J.P., 2019. Robust sound event detection in bioacoustic sensor networks. PLoS One 14, e0214168.

Maletzke, A.G., dos Reis, D.M., Batista, G.E., 2017. Quantification in data streams: Initial results. In: 2017 Brazilian Conference on Intelligent Systems. BRACIS, IEEE, pp. 43–48.

Mello, R.F., Ponti, M.A., 2018. Machine Learning: A Practical Approach on the Statistical Learning Theory. Springer.

Mezquida, D.A., Martínez, J.L., 2009. Platform for bee-hives monitoring based on sound analysis, a perpetual warehouse for swarm's daily activity. Span. J. Agric. Res. 7, 824–828.

Mitchell, S.L., Bicknell, J.E., Edwards, D.P., Deere, N.J., Bernard, H., Davies, Z.G., Struebig, M.J., 2020. Spatial replication and habitat context matters for assessments of tropical biodiversity using acoustic indices. Ecol. Indic. 119, 106717.

Parascandolo, G., Huttunen, H., Virtanen, T., 2016. Recurrent neural networks for polyphonic sound event detection in real life recordings. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, IEEE, pp. 6440–6444.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.

Pekin, B., Jung, J., Villanueva-Rivera, L., Pijanowski, B., Ahumada, J., 2012. Modeling acoustic diversity using soundscape recordings and LIDAR-derived metrics of vertical forest structure in aneotropical rainforest. Landsc. Ecol. 27, 1513–1522.

Penar, W., Magiera, A., Klocek, C., 2020. Applications of bioacoustics in animal ecology. Ecol. Complex. 43, 100847.

Pieretti, N., Farina, A., Morri, D., 2011. A new methodology to infer the singing activity of an avian community: The acoustic complexity index (ACI). Ecol. Indic. 11, 868–873.

Pieretti, N., Martire, M.L., Farina, A., Danovaro, R., 2017. Marine soundscape as an additional biodiversity monitoring tool: A case study from the Adriatic Sea (Mediterranean Sea). Ecol. Indic. 83, 13–20.

Pijanowski, B.C., Farina, A., Gage, S.H., Dumyahn, S.L., Krause, B.L., 2011. What is soundscape ecology? An introduction and overview of an emerging new science. Landsc. Ecol. 26, 1213–1232.

Ponti, M.A., d. Santos, F.P., Ribeiro, L.S.F., Cavallari, G.B., 2021. Training deep networks from zero to hero: avoiding pitfalls and going beyond. In: 2021 34th SIBGRAPI Conference on Graphics, Patterns and Images. SIBGRAPI, IEEE, pp. 9–16.

Purwins, H., Li, B., Virtanen, T., Schlüter, J., Chang, S.-Y., Sainath, T., 2019. Deep learning for audio signal processing. IEEE J. Sel. Top. Signal Process. 13, 206–219.

Ramsay, J.O., 2006. Functional Data Analysis. Wiley Online Library.

Salamon, J., Bello, J.P., 2017. Deep convolutional neural networks and data augmentation for environmental sound classification. IEEE Signal Process. Lett. 24, 279–283.

Sánchez-Gendriz, I., Padovese, L., 2016. Underwater soundscape of marine protected areas in the south Brazilian coast. Marine Poll. Bull. 105, 65–72.

Scarpelli, M.D., Ribeiro, M.C., Teixeira, C.P., 2021. What does Atlantic Forest soundscapes can tell us about landscape? Ecol. Indic. 121, 107050.

Scarpelli, M.D., Ribeiro, M.C., Teixeira, F.Z., Young, R.J., Teixeira, C.P., 2020. Gaps in terrestrial soundscape research: it's time to focus on tropical wildlife. Sci. Total Environ. 707, 135403.

Schlüter, J., 2018. Bird identification from timestamped, geotagged audio recordings. In: CLEF (Working Notes).

Servick, K., 2014. Eavesdropping on ecosystems. Sci. (N. Y. N. Y.) 343, 834–837.

Shiu, Y., Palmer, K., Roch, M.A., Fleishman, E., Liu, X., Nosal, E.-M., Helble, T., Cholewiak, D., Gillespie, D., Klinck, H., 2020. Deep neural networks for automated detection of marine mammal species. Sci. Rep. 10, 1–12.

Sokolova, M., Lapalme, G., 2009. A systematic analysis of performance measures for classification tasks. Inf. Process. Manage. 45, 427–437.

Stowell, D., 2022. Computational bioacoustics with deep learning: a review and roadmap. PeerJ 10, e13152.

Strout, J., Rogan, B., Seyednezhad, S.M., Smart, K., Bush, M., Ribeiro, E., 2017. Anuran call classification with deep learning. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, IEEE, pp. 2662–2665.

Sueur, J., Aubin, T., Simonis, C., 2008a. Seewave, a free modular tool for sound analysis and synthesis. Bioacoustics 18, 213–226.

Sueur, J., Farina, A., 2015. Ecoacoustics: the ecological investigation and interpretation of environmental sound. Biosemiotics 8, 493–502.

Sueur, J., Farina, A., Gasc, A., Pieretti, N., Pavoine, S., 2014. Acoustic indices for biodiversity assessment and landscape investigation. Acta Acust. United Acust. 100, 772–781.

Sueur, J., Pavoine, S., Hamerlynck, O., Duvail, S., 2008b. Rapid acoustic survey for biodiversity appraisal. PLoS One 3, e4065.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2016. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2818–2826.

Tang, Q., Xu, L., Zheng, B., He, C., 2023. Transound: Hyper-head attention transformer for birds sound recognition. Ecol. Inform. 75, 102001.

Thomas, M., Martin, B., Kowarski, K., Gaudet, B., Matwin, S., 2019. Marine mammal species classification using convolutional neural networks and a novel acoustic representation. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, pp. 290–305.

Tieleman, T., Hinton, G., et al., 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Netw. Mach. Learn. 4, 26–31.

Villanueva-Rivera, L., Pijanowski, B., Doucette, j., Pekin, B., 2011. A primer of acoustic analysis for landscape ecologists. Landsc. Ecol. 26, 1233–1246.

Wang, Y., Getreuer, P., Hughes, T., Lyon, R.F., Saurous, R.A., 2017a. Trainable frontend for robust and far-field keyword spotting. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, IEEE, pp. 5670–5674.

Wang, J., Perez, L., et al., 2017b. The effectiveness of data augmentation in image classification using deep learning. Convolutional Neural Netw. Vis. Recognit. 11, 1–8.

Welch, P., 1967. The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. IEEE Trans. Audio Electroacoust. 15, 70–73.

Xie, J., Zhu, M., Hu, K., Zhang, J., Hines, H., Guo, Y., 2022. Frog calling activity detection using lightweight CNN with multi-view spectrogram: a case study on kroombit tinker frog. Mach. Learn. Appl. 7, 100202.

Zagoruyko, S., Komodakis, N., 2016. Wide residual networks. arXiv preprint arXiv:1605.07146.

Znidersic, E., Towsey, M., Roy, W.K., Darling, S.E., Truskinger, A., Roe, P., Watson, D.M., 2020. Using visualization and machine learning methods to monitor low detectability species - The Least Bittern as a case study. Ecol. Inform. 55, 101014.