Received: 28 April 2023 | Accepted: 20 July 2023

DOI: 10.1111/2041-210X.14196

Methods in Ecology and Evolution

APPLICATION

OpenSoundscape: An open-source bioacoustics analysis package for Python

Sam Lapp | Tessa Rhinehart | Louis Freeland-Haynes | Jatin Khilnani | Alexandra Syunkova | Justin Kitzes

Department of Biological Sciences, University of Pittsburgh, Pittsburgh, Pennsylvania, USA

Correspondence
Sam Lapp
Email: sam.lapp@pitt.edu

Funding information
Department of Biological Sciences at the University of Pittsburgh; Gordon and Betty Moore Foundation, Grant/Award Number: 72173; Mascaro Center for Sustainable Innovation, University of Pittsburgh; National Fish and Wildlife Foundation, Grant/Award Number: 70153, 72173 and 73615; National Geographic Society, Grant/Award Number: NGS-55651T-18; National Science Foundation, Grant/Award Number: 1935507, 2120084 and ACI-1548562; University of Pittsburgh Center for Research Computing; Bridges2

Handling Editor: Aaron Ellison

Abstract

1. Landscape-scale bioacoustic projects have become a popular approach to biodiversity monitoring. Combining passive acoustic monitoring recordings and automated detection provides an effective means of monitoring sound-producing species' occupancy and phenology and can lend insight into unobserved behaviours and patterns. The availability of low-cost recording hardware has lowered barriers to large-scale data collection, but technological barriers in data analysis remain a bottleneck for extracting biological insight from bioacoustic datasets.
2. We provide a robust and open-source Python toolkit for detecting and localizing biological sounds in acoustic data.
3. OpenSoundscape provides access to automated acoustic detection, classification and localization methods through a simple and easy-to-use set of tools. Extensive documentation and tutorials provide step-by-step instructions and examples of end-to-end analysis of bioacoustic data. Here, we describe the functionality of this package and provide concise examples of bioacoustic analyses with OpenSoundscape.
4. By providing an interface for bioacoustic data and methods, we hope this package will lead to increased adoption of bioacoustics methods and ultimately to enhanced insights for ecology and conservation.

KEYWORDS
acoustic monitoring, automated detection, bioacoustics, localization, machine learning, OpenSoundscape, Python

1 | INTRODUCTION

The sounds of the natural world provide us with a unique opportunity to spy on ecological happenings that are otherwise hidden from observation. For centuries, naturalists have relied on keen ears for biological sounds as a way to identify and study sound-producing organisms such as birds and frogs. More recently, technologies for capturing and recognizing natural sounds have transformed the study of bioacoustics from a small-scale endeavour to a large-scale, data-driven discipline powered by remote sensing—much in the way that satellite imagery transformed cartography from an intimately local and experience-based practice to a massive-scale data-driven one.

Large-scale bioacoustic monitoring projects have become a popular approach to biodiversity monitoring. Effective and affordable

automated recording unit (ARU) hardware enables researchers to collect landscape-scale acoustic data. These are complemented by machine learning methods that provide efficient and accurate means of extracting species detections from the resulting audio data (Stowell, 2022). For instance, deep learning image recognition models have been used to recognize species-specific vocalizations of birds (e.g. Knight & Bayne, 2018; Ruff et al., 2020) and cetaceans (e.g. Bermant et al., 2019; Madhusudhana et al., 2021). Combining these two new technologies—automated recording hardware and automated sound detection software—provides an effective means of monitoring species across space and time and can provide insight into unobserved behaviours and patterns.

Tools for analysing bioacoustic monitoring data are still catching up with the rapid expansion of data collection that has been enabled by new recording hardware (Ulloa et al., 2021). Various existing packages and software, summarized in the following section, provide interfaces to some aspects of data management, exploration, annotation or automated species detection. However, employing state-of-the-art methods to extract species detections from large-scale acoustic datasets still requires an advanced understanding of machine learning and computer programming, which has limited the adoption of these methods. In practice, many recent acoustic monitoring studies have relied on techniques such as template-based cross-correlation or energy detection, or have alternatively employed generic pre-trained classifiers (e.g. Cole et al., 2022; Toenies & Rich, 2021) even though training or fine-tuning automated classifiers using local data generally improves model performance (Lauha et al., 2022). When research groups have developed domain-specific deep learning classifiers, automated classifier performance can be excellent and can lead to novel biological insights (e.g. Bermant et al., 2019; Nolan et al., 2023; Wightman et al., 2022; Zhong et al., 2021). Making such analysis methods more accessible to researchers could broadly improve the quality of bioacoustic data analyses and enhance insights into ecological processes (Stowell, 2022).

Here, we present OpenSoundscape, an open-source Python package for detecting biological sounds of interest in acoustic monitoring data. OpenSoundscape provides access to powerful acoustic detection, classification, and localization methods including both machine learning and signal processing algorithms (Figure 1). The package represents a synthesis of more than 4 years of development motivated by the authors' direct applications of these tools to ecology and conservation research. Extensive package documentation and tutorials for OpenSoundscape provide step-by-step instructions and examples for manipulating audio and automating the recognition of biological sounds. OpenSoundscape has already been used in a variety of bioacoustics applications, from exploratory data analysis to automated recognition of frogs (Lapp et al., 2021), birds (Lapp, Larkin, et al., 2023; Malamut, 2022) and gunshots (Katsis et al., 2022) through various signal processing and deep learning methods.

The remainder of this manuscript is organized into five sections. First, we review related work and place OpenSoundscape in the context of existing software for automated species detection. Second, we outline the key functionalities of OpenSoundscape (version 0.9.1). Third, we briefly discuss the development practices and design principles of the package. Fourth, we provide four concise examples of bioacoustic workflows with OpenSoundscape to demonstrate the utility of this package. Finally, we conclude by outlining future directions for the package.

2 | EXISTING SOFTWARE FOR AUTOMATED SPECIES DETECTION

OpenSoundscape is designed to tackle the steps of bioacoustic monitoring data analyses that concern detecting, measuring and localizing sounds of interest in audio recordings, especially those captured by ARUs. A rapidly growing body of software (reviewed in Rhinehart, 2022/2023) supports other steps in the bioacoustics workflow, including data management (e.g. PAMguard, Gillespie et al., 2009), exploration (e.g. Audacity, Audacity Team, 2021) and annotation (e.g. Raven Pro, Bioacoustics Research Program, 2019).

Automating the detection of species in audio is a key step in bioacoustic monitoring, and is the focus of several existing software projects (e.g. Kaleidoscope Pro, Wildlife Acoustics, 2023; BirdNET,

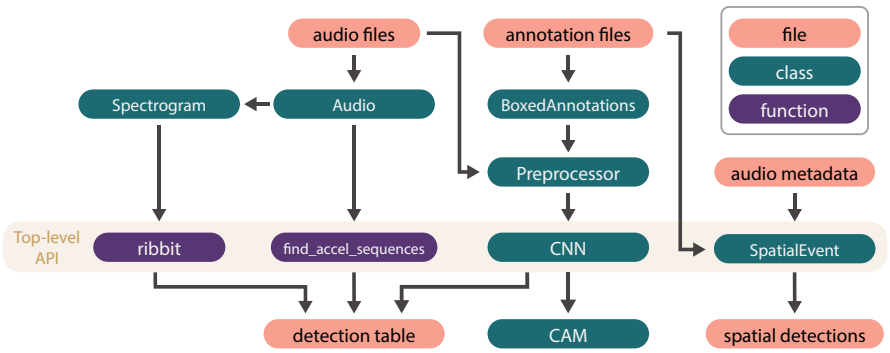

FIGURE 1 The classes and functions of OpenSoundscape produce detections of biological sounds in time and space. Arrows in the diagram represent the flow of information from inputs to outputs. The classes and methods in the top-level API (beige box) can be called directly to use the core functionality of OpenSoundscape without knowledge of other parts of the API. API, application programming interface; CAM, class activation maps; CNN, convolutional neural network.

Kahl, 2019/2022). At present, the gold standard method for most classification tasks is considered to be deep learning, especially using convolutional neural networks (CNNs; Kahl et al., 2020; Stowell, 2022). The most powerful and widely adopted machine learning tools for deep learning are TensorFlow and PyTorch, both of which require advanced knowledge of Python and machine learning to use effectively. While there are several bioacoustics software tools that aim to provide high-level APIs that simplify the process of training deep learning algorithms for bioacoustic detection (Ketos, Kirsebom et al., 2021; Koogu, Madhusudhana, 2022; aviaNZ, Marsland et al., 2019; ANIMAL-SPOT, Bergler et al., 2022; gibbonfindR, Clink & Klinck, 2019; soundClass, Silva et al., 2022), we believe that the flexibility provided by OpenSoundscape will allow the package to be applied to a wider range of bioacoustics problems than is possible with existing software. For example, in the two Python packages in the list above (Ketos and Koogu), customizing data augmentation or machine learning model architecture requires the user to understand the underlying package TensorFlow. Furthermore, some classification packages are geared towards the classification of particular taxa or use cases (e.g. underwater acoustics, Ketos; birds of New Zealand, aviaNZ), limiting their more general adoption by bioacoustics researchers, while others (including soundClass) are not optimized for the large-scale parallel computing required to efficiently run deep learning algorithms at scale. Finally, a variety of packages focus on alternatives to deep learning methods (e.g. template matching in ARBIMON, Rainforest Connection, 2023 and monitoR, Katz et al., 2016). Compared to existing software for the detection and classification of biological sounds, OpenSoundscape is distinguished by simultaneously (1) providing default workflows that are streamlined and customized for bioacoustic data and (2) providing flexibility to adapt each aspect of this workflow to new projects and new domains.

3 | LIBRARY OVERVIEW

OpenSoundscape is a bioacoustics toolkit for Python that provides a set of tools for detecting, classifying and localizing biological sounds in audio data. OpenSoundscape aims to achieve both simple interfaces accessible to non-programmers and powerful flexibility for addressing the complexity of diverse bioacoustic monitoring analysis tasks. It is designed to support scaling analyses across distributed computing systems. The package is publicly available on PyPI (pypi.org/project/opensoundscape) and GitHub (github.com/kitzeslab/opensoundscape) under an MIT licence which allows unrestricted use and modification.

The primary functionality of OpenSoundscape is the development and application of automated algorithms for locating biological sounds of interest in space and time. In bioacoustics, some automated recognition tasks are well suited to machine learning while others are best solved with signal processing (Lapp et al., 2021), and OpenSoundscape provides functionality for both approaches. OpenSoundscape interfaces with the popular PyTorch library (Paszke et al., 2019) to provide machine learning tools. Dozens of options for CNN model architectures and training parameters (e.g. learning rate, loss function) allow for easy and flexible model customization without requiring the user to program in PyTorch or other lower level packages. Furthermore, OpenSoundscape includes integration with the Weights and Biases platform (Biewald, 2020) to provide real-time monitoring of preprocessed samples, classification performance metrics and computational metering while training and predicting with machine learning models. OpenSoundscape supports several flavours of class activation mapping (Selvaraju et al., 2020), which increases model interpretability by highlighting the regions of input samples that influence a classifier's score outputs.

OpenSoundscape also provides intuitive and robust Audio and Spectrogram classes for interacting with audio data. The ability to retain, manipulate and inspect attributes and metadata of audio files alongside the raw sample data increases interpretability and reproducibility during acoustic data analyses, but to our knowledge, all other available Python tools lack this functionality. In OpenSoundscape, the Audio and Spectrogram classes provide a streamlined and featured API for inspecting and manipulating audio and spectrogram data and their associated metadata.

The top-level API of OpenSoundscape (beige box in Figure 1) consists of classes and functions that users can call directly to generate species detections directly from data files. Directly using these classes and methods provides the core functionality of OpenSoundscape without requiring the user to access any other parts of the API. Intermediate- and low-level classes and functions enable more flexibility and control for more advanced users. Here, we describe the key top-level and intermediate-level classes and functions shown in Figure 1.

Top-level API components:

- CNN class: Train CNNs with custom parameters and flexible architecture; generate predictions on audio data; save and load trained models; monitor training and inference progress and metrics through integration with the Weights and Biases platform (Biewald, 2020).
- signal_processing module: Access a set of signal processing tools, including the find_accel_sequences function used to detect Ruffed Grouse drumming in (Lapp, Larkin, et al., 2023).
- localization module: Spatially localize audio events from time-synchronized recordings using the SynchronizedRecorderArray or SpatialEvent classes.
- ribbit function: Detect sounds with periodic amplitude modulation using the method described in (Lapp et al., 2021).

Selected intermediate-level API components:

- Audio class: Load, manipulate and save audio files with the Audio class; read and update audio file metadata; retrieve and calculate parameters (e.g. sample rate, duration, and decibels full scale (dBFS)); trim, extend, normalize or loop audio; split audio into clips of equal length; extract audio segments from longer files.

- Spectrogram class: Calculate, plot and save the spectrogram of an audio object with custom parameters.
- BoxedAnnotations class: View and manipulate audio annotations; prepare annotated data for training and evaluation of automated classification algorithms; load and save annotation files; filter, aggregate, manipulate and correct labels across a dataset.
- Preprocessor class: Customize the preprocessing and augmentation of training data for machine learning models.
- CAM class: Produce class activation maps such as GradCAM (Selvaraju et al., 2020) for visualizing what regions of a sample cause a CNN to predict a particular class.

OpenSoundscape interfaces with other software and Python packages to streamline bioacoustic monitoring workflows. In particular, Pandas (McKinney, 2010) and PyTorch (Paszke et al., 2019) classes are created and manipulated by various parts of the code base. Key machine learning classes subclass the base classes provided by PyTorch. Integration with Raven Pro and Raven Lite annotation software (Bioacoustics Research Program, 2019) is provided through the import and export of Raven-formatted files in the BoxedAnnotations class. Integration with the Python package Crowsetta is a work in progress and will enable users to use various audio annotation file formats in the future. Key dependencies of OpenSoundscape include the machine learning libraries torch (Paszke et al., 2019), torchvision (Marcel & Rodriguez, 2010) and scikit-learn (Pedregosa et al., 2011); the audio libraries Librosa (McFee et al., 2015) and SoundFile (Bechtold, 2023); the scientific computing libraries Numba (Lam et al., 2015), NumPy (van der Walt et al., 2011) and Pandas (McKinney, 2010); the logging platform Weights and Biases (Biewald, 2020); the wavelet analysis package pywavelets (Lee et al., 2019); and the plotting library matplotlib (Hunter, 2007).

4 | EXAMPLES

Four Python notebooks demonstrating common workflows in OpenSoundscape are included in the Supporting Information and are hosted on a public GitHub repository (github.com/kitzeslab/demos-for-opso). Each notebook is described briefly below. Detailed documentation and tutorials on the entirety of the OpenSoundscape package are available at opensoundscape.org.

4.1 | Notebook 1: Exploring the acoustic structure of *Atelopus varius* vocalizations

Because of the great diversity of biological sounds present in soundscapes, gaining a qualitative and quantitative understanding of audio data is an essential first step in any bioacoustic analysis. This process can identify potential issues or pitfalls and provide insight for optimizing data quality and clarity (for instance, through spectrogram parameter selection and gain adjustments, or the removal of noisy or invalid audio files). Unlike previously published Python tools, OpenSoundscape's Audio class retains and allows modification of audio metadata alongside the raw sample data, which increases interpretability and reproducibility during acoustic data analysis. Similarly, the Spectrogram class retains information about parameters and frequency and time bins alongside the raw spectrogram data. Notebook 1 uses the Audio and Spectrogram classes to inspect the characteristics of a vocalization of *Atelopus varius* (the variable harlequin frog), demonstrating the power of changing spectrogram parameters for revealing acoustic structure (Figure 2). Inspecting the temporal and harmonic characteristics of the call leads to the insight that the call's rapid amplitude modulation is produced by constructive and destructive interference of tones differing by 130 Hz.

4.2 | Notebooks 2 and 3: Training a CNN to classify bird songs

Deep learning models provide a powerful means of automating the detection and classification of complex biological sounds. When trained and used appropriately, these methods can provide accurate automation of acoustic detection that scales to analyses of thousands

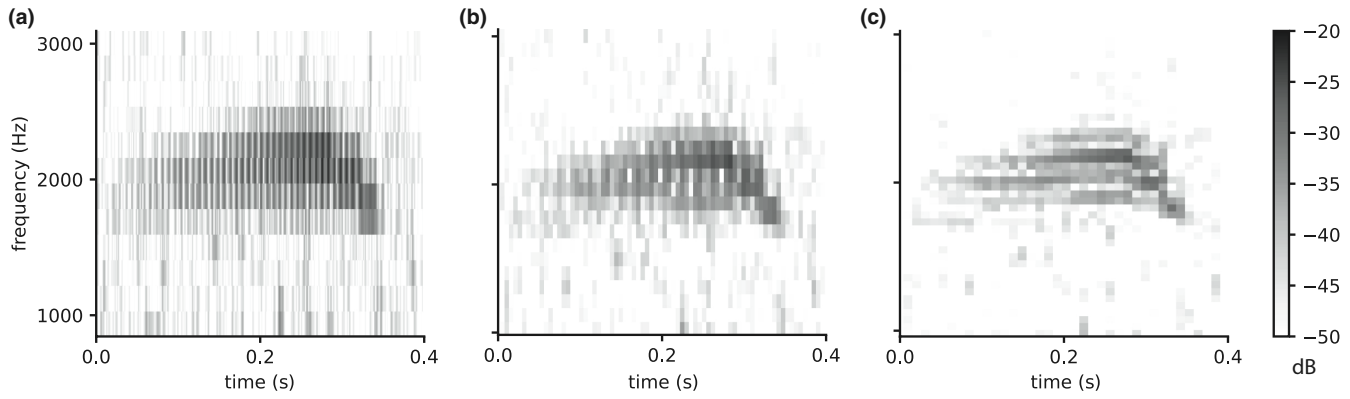

FIGURE 2 Using Spectrogram settings to inspect the acoustic structure of a recording of *Atelopus varius* (the variable harlequin frog). (a) window_samples = 256 and overlap_fraction = 0.9; (b) default parameters, window_samples = 512 and overlap_fraction = 0.5; (c) window_samples = 1024 and overlap_fraction = 0.5. See Notebook 1 in the S1 for further details and code.

of hours of audio (Stowell, 2022). Although the process of training a machine learning model involves many decisions about preprocessing, augmentation and hyperparameters, OpenSoundscape simplifies this process by providing functionality and default parameter values tailored to bioacoustics. Notebook 2 first prepares training and test data from a public dataset of Raven-annotated audio (Chronister et al., 2021), then trains a CNN to recognize the vocalizations of seven bird species. Notebook 3 evaluates the performance of the CNN and demonstrates that it effectively recognizes bird songs such as that of the Eastern Towhee (*Pipilo erythrophthalmus*) in spectrograms (Figure 3).

4.3 | Notebook 4: Detecting repeated element sounds with signal processing

Although deep learning models are popular and effective approaches to many automated detection problems, they can be difficult to apply when little to no training data is available. In these scenarios, which may be common for bioacoustics tasks, signal processing methods may be preferable. Unlike deep learning approaches, these methods have interpretable parameters that can be tuned to biologically relevant values by the user. They can be especially effective in detecting songs with stereotyped temporal structure, a feature of many anuran vocalizations and invertebrate stridulations (Lapp et al., 2021). This notebook demonstrates the use of two detection methods from the signal_processing module, using each to detect the song of the Northern Flicker (*Colaptes auratus*).

5 | DEVELOPMENT PRACTICES

The OpenSoundscape source code is hosted on GitHub and published under the MIT licence, which allows unrestricted use and modification as well as public contributions to package development. On the GitHub web page, issues track bugs and feature requests for the code base, while discussions provide a public forum for user support and more general conversations. Continuous integration provided by GitHub ensures code quality through testing with PyTest (Krekel et al., 2004) and PEP8 (Van Rossum et al., 2001) code style compliance checks with Black (Langa, 2022). Major and minor releases are published to the Python Package Index (PyPI, 2023) under the name opensoundscape.

6 | FUTURE DIRECTIONS

As nascent fields, bioacoustics and the broader machine learning community have yet to adopt shared protocols and formats. In particular, cross-platform machine learning model operability and standardized audio metadata formats are areas of active development and represent important features for future OpenSoundscape development. Cross-platform interoperability refers to the ability of machine learning models to work across different computing environments and software. For example, it would enable a model to be trained on one platform and seamlessly deployed on another. Achieving cross-platform interoperability requires the standardization of data and model formats, model parameters and APIs. Current efforts include the ONNX format for representing models and toolkits such as OpenVINO, ONNX Runtime and TVM for deploying models across different hardware architectures. In the short term, OpenSoundscape will add integration with an online model repository for loading and using trained machine learning models. In future development, OpenSoundscape will support the import and export of cross-platform model formats.

Standardizing metadata for audio and audio annotations also represents an important area of development (Stowell, 2022). The capability to store, modify and retrieve metadata associated with an audio file is important for data analysis and reproducibility in bioacoustics workflows, but the community has yet to adopt

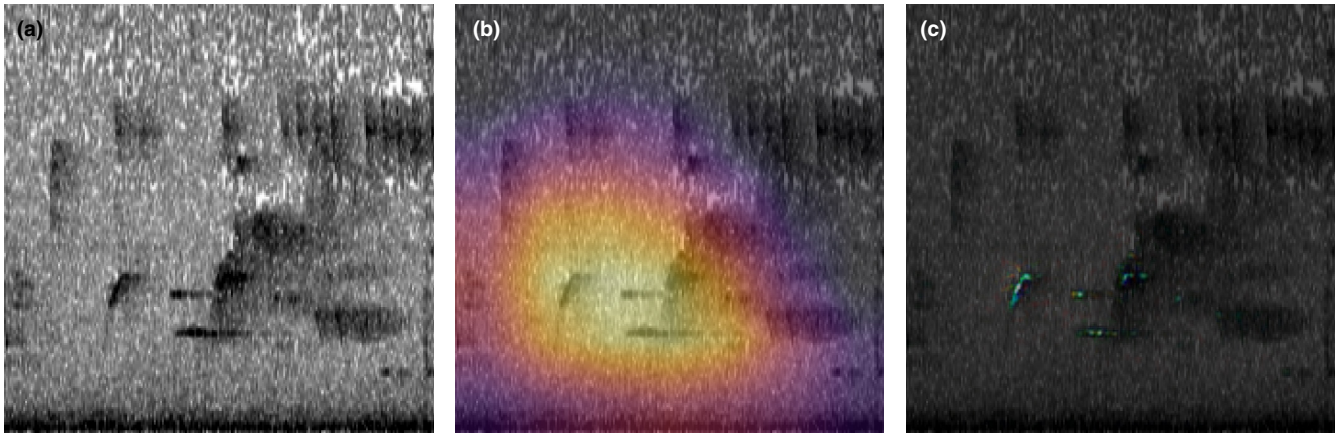

FIGURE 3 A CNN trained with OpenSoundscape recognizes the song of an Eastern Towhee in a sample. (a) Sample created by the CNN, a spectrogram representing 3 s of audio. (b) gradCAM activation heatmap (c) guided backpropagation visualization highlighting the pixels that caused the network to recognize it as an Eastern Towhee. See Notebook 3 in the SI for CNN training, evaluation and visualization. CAM, class activation maps; CNN, convolutional neural network.

standardized formats for audio data and annotations. At present, the Raven software annotation format serves as a de facto standard, and OpenSoundscape supports loading and saving Raven formatted annotations. Additionally, the recently developed Python package Crowsetta (Nicholson, 2023) provides a much-needed common interface for several audio annotation formats, and support for integration with Crowsetta is a short-term priority for OpenSoundscape development. OpenSoundscape will add support for additional audio metadata formats as they become adopted by the community.

Finally, future development of OpenSoundscape will work towards the release of a static and cohesive application programming interface (API). To achieve this, some modules currently included in the package may be ported to separate packages or repositories. Limiting the scope of the package and providing a static API will enhance the reliability of OpenSoundscape as a tool for reproducible research.

7 | CONCLUSION

The rapid uptake of acoustic monitoring as a surveying tool demonstrates that ecologists appreciate the potential value of this technology in ecology and conservation research. At the same time, the fields of artificial intelligence and machine learning have generated methods capable of synthesizing insights from unstructured data. However, applying such methods to bioacoustic analysis currently requires expertise in the development of automated recognition methods and extensive project-specific code for adapting methods to the audio domain. By embedding these methods within an ecosystem of data pipelines and other bioacoustics tools, we hope that OpenSoundscape will allow users to connect their powerful data with powerful methods, ultimately leading to rich biological insights.

AUTHOR CONTRIBUTIONS

Sam Lapp, Tessa Rhinehart and Justin Kitzes conceived the ideas and structure of the Python package and of this manuscript; Sam Lapp, Louis Freeland-Haynes, Alexandra Syunkova and Jatin Khilnani developed the example notebooks; Sam Lapp led the writing of the manuscript and edited the notebooks. All authors contributed critically to the development of the code base and the drafts of this manuscript. All authors gave final approval for publication.

ACKNOWLEDGEMENTS

This work was financially supported by the Mascaro Center for Sustainable Innovation and the Department of Biological Sciences at the University of Pittsburgh. This work was also financially supported by the Gordon and Betty Moore Foundation under Grant 72173. This material is based on work supported by the National Science Foundation under Grants 1935507 and 2120084. This work was also supported by the National Fish and Wildlife Foundation's Central Appalachia Habitat Stewardship Program and Delaware River Program under grants 70153, 73615 and 72173, and by National Geographic Society under grant NGS-55651T-18. This research was also supported in part by the University of Pittsburgh Center for Research Computing through the resources provided. This work used the Extreme Science and Engineering Discovery Environment (XSEDE) which is supported by National Science Foundation grant number ACI-1548562, and the Bridges2 supercomputing cluster. We thank B. Moore, A. Watts and J. Jia for contributions to the OpenSoundscape codebase.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

PEER REVIEW

The peer review history for this article is available at https://www.webofscience.com/api/gateway/wos/peer-review/10.1111/2041-210X.14196.

DATA AVAILABILITY STATEMENT

OpenSoundscape is available at https://github.com/kitzeslab/opensoundscape and on PyPI, and version 0.9.1 is published on Zenodo with the DOI https://doi.org/10.5281/zenodo.8170077 (Lapp, Rhinehart, et al., 2023). Documentation is hosted at https://opensoundscape.org. The demonstration notebooks are available in the Supporting Information and also at https://github.com/kitzeslab/demos-for-opso and version 1.0.0 is published on Zenodo with the DOI https://doi.org/10.5281/zenodo.8170062 (Lapp, Freeland-Haynes, et al., 2023).

ORCID

Sam Lapp https://orcid.org/0000-0003-1637-6822
Tessa Rhinehart https://orcid.org/0000-0002-4352-3464

REFERENCES

Audacity Team. (2021). *Audacity(R): Free audio editor and recorder* (3.0.0) [computer software]. https://audacityteam.org

Bechtold, B. (2023). *Python-soundfile* [computer software]. https://github.com/bastibe/python-soundfile/

Bergler, C., Smeele, S. Q., Tyndel, S. A., Barnhill, A., Ortiz, S. T., Kalan, A. K., Cheng, R. X., Brinkløv, S., Osiecka, A. N., Tougaard, J., Jakobsen, F., Wahlberg, M., Nöth, E., Maier, A., & Klump, B. C. (2022). ANIMAL-SPOT enables animal-independent signal detection and classification using deep learning. *Scientific Reports*, *12*(1), 21966. https://doi.org/10.1038/s41598-022-26429-y

Bermant, P. C., Bronstein, M. M., Wood, R. J., Gero, S., & Gruber, D. F. (2019). Deep machine learning techniques for the detection and classification of sperm whale bioacoustics. *Scientific Reports*, *9*(1), 12588. https://doi.org/10.1038/s41598-019-48909-4

Biewald, L. (2020). *Experiment tracking with weights and biases*. https://www.wandb.com/

Bioacoustics Research Program. (2019). *Raven Pro: Interactive sound analysis software (1.6) [computer software]*. Center for Conservation Bioacoustics, Cornell Lab of Ornithology. https://ravensoundsoftware.com/software/raven-pro/

Chronister, L. M., Rhinehart, T. A., Place, A., & Kitzes, J. (2021). An annotated set of audio recordings of eastern north American birds containing frequency, time, and species information. *Ecology*, *102*, e03329. https://doi.org/10.1002/ecy.3329

Clink, D. J., & Klinck, H. (2019). GIBBONFINDR: An R package for the detection and classification of acoustic signals. *arXiv preprint arXiv:1906.02572*. https://doi.org/10.48550/arXiv.1906.02572

Cole, J. S., Michel, N. L., Emerson, S. A., & Siegel, R. B. (2022). Automated bird sound classifications of long-duration recordings produce occupancy model outputs similar to manually annotated data. *Ornithological Applications*, *124*(2), duac003. https://doi.org/10.1093/ornithapp/duac003

Gillespie, D., Mellinger, D. K., Gordon, J., McLaren, D., Redmond, P., McHugh, R., Trinder, P., Deng, X., & Thode, A. (2009). PAMGUARD: Semiautomated, open source software for real-time acoustic detection and localization of cetaceans. *The Journal of the Acoustical Society of America*, *125*(4), 2547. https://doi.org/10.1121/1.4808713

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95.

Kahl, S. (2022). *BirdNET [Python]*. https://github.com/kahst/BirdNET (Original work published 2019)

Kahl, S., Clapp, M., Hopping, W., Goëau, H., Planqué, R., Vellinga, W.-P., & Joly, A. (2020). *Overview of BirdCLEF 2020: Bird sound recognition in complex acoustic environments*. CLEF: Conference and Labs of the Evaluation.

Katsis, L. K. D., Hill, A. P., Piña-Covarrubias, E., Prince, P., Rogers, A., Patrick Doncaster, C., & Snaddon, J. L. (2022). Automated detection of gunshots in tropical forests using convolutional neural networks. *Ecological Indicators*, *141*, 109128. https://doi.org/10.1016/j.ecolind.2022.109128

Katz, J., Hafner, S. D., & Donovan, T. (2016). Tools for automated acoustic monitoring within the R package monitoR. *Bioacoustics*, *25*(2), 197–210. https://doi.org/10.1080/09524622.2016.1138415

Kirsebom, O. S., Frazao, F., Padovese, B., Sakib, S., & Matwin, S. (2021). Ketos—A deep learning package for creating acoustic detectors and classifiers. *The Journal of the Acoustical Society of America*, *150*(4), A164. https://doi.org/10.1121/10.0007998

Knight, E. C., & Bayne, E. M. (2018). Classification threshold and training data affect the quality and utility of focal species data processed with automated audio-recognition software. *Bioacoustics*, *28*, 539–554. https://doi.org/10.1080/09524622.2018.1503971

Krekel, H., Oliveira, B., Pfannschmidt, R., Bruynooghe, F., Laugher, B., & Bruhin, F. (2004). *Pytest 7.3*. https://github.com/pytest-dev/pytest

Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: A llvm-based python jit compiler. *Proceedings of the second workshop on the LLVM compiler infrastructure in HPC*, 1–6.

Langa, Ł. (2022). *Black code formatter* (22.12.0) [computer software]. https://github.com/psf/black

Lapp, S., Freeland-Haynes, L., Khilnani, J., Syunkova, A., Rhinehart, T., & Kitzes, J. (2023). *Kitzeslab/demos-for-opso* (v1.0.0) [computer software]. https://doi.org/10.5281/ZENODO.8170062

Lapp, S., Larkin, J. L., Parker, H. A., Larkin, J. T., Shaffer, D. R., Tett, C., McNeil, D. J., Fiss, C. J., & Kitzes, J. (2023). Automated recognition of ruffed grouse drumming in field recordings. *Wildlife Society Bulletin*, *47*(1), e1395.

Lapp, S., Rhinehart, T., Freeland-Haynes, L., Khilnani, J., Syunkova, A., & Kitzes, J. (2023). *OpenSoundscape v0.9.1* [computer software]. https://doi.org/10.5281/ZENODO.8170077

Lapp, S., Wu, T., Richards-Zawacki, C., Voyles, J., Rodriguez, K. M., Shamon, H., & Kitzes, J. (2021). Automated detection of frog calls and choruses by pulse repetition rate. *Conservation Biology*, *35*(5), 1659–1668. https://doi.org/10.1111/cobi.13718

Lauha, P., Somervuo, P., Lehikoinen, P., Geres, L., Richter, T., Seibold, S., & Ovaskainen, O. (2022). Domain-specific neural networks improve automated bird sound recognition already with small amount of local data. *Methods in Ecology and Evolution*, *13*(12), 2799–2810. https://doi.org/10.1111/2041-210X.14003

Lee, G. R., Gommers, R., Waselewski, F., Wohlfahrt, K., & O'Leary, A. (2019). PyWavelets: A python package for wavelet analysis. *Journal of Open Source Software*, *4*(36), 1237. https://doi.org/10.21105/joss.01237

Madhusudhana, S. (2022). Shyamblast/Koogu: V0.7.1 [computer software]. *Zenodo*. https://doi.org/10.5281/zenodo.7275319

Madhusudhana, S., Shiu, Y., Klinck, H., Fleishman, E., Liu, X., Nosal, E.-M., Helble, T., Cholewiak, D., Gillespie, D., Širović, A., & Roch, M. A. (2021). Improve automatic detection of animal call sequences with temporal context. *Journal of the Royal Society Interface*, *18*(180), 20210297. https://doi.org/10.1098/rsif.2021.0297

Malamut, E. J. (2022). *Using autonomous recording units and image processing to investigate patterns in avian singing activity and nesting phenology* [M.S., University of California, Los Angeles]. https://www.proquest.com/docview/2676140480/abstract/3448026094149C1PQ/1

Marcel, S., & Rodriguez, Y. (2010). Torchvision the machine-vision package of torch. *Proceedings of the 18th ACM international conference on multimedia*, 1485–1488. https://doi.org/10.1145/1873951.1874254

Marsland, S., Priyadarshani, N., Juodakis, J., & Castro, I. (2019). AviaNZ: A future-proofed program for annotation and recognition of animal sounds in long-time field recordings. *Methods in Ecology and Evolution*, *10*(8), 1189–1195. https://doi.org/10.1111/2041-210X.13213

McFee, B., Raffel, C., Liang, D., Ellis, D., McVicar, M., Battenberg, E., & Nieto, O. (2015). *Librosa: Audio and music signal analysis in python*. https://doi.org/10.25080/Majora-7b98e3ed-003

McKinney, W. (2010). Data structures for statistical computing in python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th python in science conference* (pp. 56–61). https://doi.org/10.25080/Majora-92bf1922-00a

Nicholson, D. (2023). Crowsetta: A python tool to work with any format forannotating animal vocalizations and bioacoustics data. *Journal of Open Source Software*, *8*(84), 5338. https://doi.org/10.21105/joss.05338

Nolan, V., Scott, C., Yeiser, J. M., Wilhite, N., Howell, P. E., Ingram, D., & Martin, J. A. (2023). The development of a convolutional neural network for the automatic detection of northern bobwhite Colinus virginianus covey calls. *Remote Sensing in Ecology and Conservation*, *9*(1), 46–61. https://doi.org/10.1002/rse2.294

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library (arXiv:1912.01703). *arXiv*. https://doi.org/10.48550/arXiv.1912.01703

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. (2011). Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research*, *12*, 2825–2830.

Python Package Index (PyPI). (2023). Python Software Foundation. https://pypi.org/

Rainforest Connection. (2023). *Arbimon*. https://arbimon.rfcx.org/

Rhinehart, T. (2023). *Bioacoustics software* [computer software]. https://github.com/rhine3/bioacoustics-software (Original work published 2022)

Ruff, Z. J., Lesmeister, D. B., Duchac, L. S., Padmaraju, B. K., & Sullivan, C. M. (2020). Automated identification of avian vocalizations with deep convolutional neural networks. *Remote Sensing in Ecology and Conservation*, *6*(1), 79–92. https://doi.org/10.1002/rse2.125

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2020). Grad-CAM: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, *128*(2), 336–359. https://doi.org/10.1007/s11263-019-01228-7

Silva, B., Mestre, F., Barreiro, S., Alves, P. J., & Herrera, J. M. (2022). soundClass: An automatic sound classification tool for biodiversity monitoring using machine learning. *Methods in Ecology and Evolution*, *13*(11), 2356–2362. https://doi.org/10.1111/2041-210X.13964

Stowell, D. (2022). Computational bioacoustics with deep learning: A review and roadmap. *PeerJ*, *10*, e13152. https://doi.org/10.7717/peerj.13152

Toenies, M., & Rich, L. (2021). Advancing bird survey efforts through novel recorder technology and automated species identification. *California Fish and Wildlife Journal*, *107*(2), 56–70. https://doi.org/10.51492/cfwj.107.5

Ulloa, J. S., Haupert, S., Latorre, J. F., Aubin, T., & Sueur, J. (2021). Scikit-maad: An open-source and modular toolbox for quantitative soundscape analysis in python. *Methods in Ecology and Evolution*, *12*(12), 2334–2340. https://doi.org/10.1111/2041-210X.13711

van der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). The NumPy Array: A structure for efficient numerical computation. *Computing in Science & Engineering*, *13*(2), 22–30. https://doi.org/10.1109/MCSE.2011.37

Van Rossum, G., Warsaw, B., & Coghlan, N. (2001). *Style guide for python code (PEP 8)*. https://www.python.org/dev/peps/pep-0008/

Wightman, P. H., Henrichs, D. W., Collier, B. A., & Chamberlain, M. J. (2022). Comparison of methods for automated identification of wild Turkey gobbles. *Wildlife Society Bulletin*, *46*(1), e1246. https://doi.org/10.1002/wsb.1246

Wildlife Acoustics. (2023). *Kaleidoscope pro analysis software* [computer software]. https://www.wildlifeacoustics.com/products/kaleidoscope-pro

Zhong, M., Taylor, R., Bates, N., Christey, D., Basnet, H., Flippin, J., Palkovitz, S., Dodhia, R., & Lavista Ferres, J. (2021). Acoustic detection of regionally rare bird species through deep convolutional neural networks. *Ecological Informatics*, *64*, 101333. https://doi.org/10.1016/j.ecoinf.2021.101333

SUPPORTING INFORMATION

Additional supporting information can be found online in the Supporting Information section at the end of this article.

Supporting Information S1: Python notebooks demonstrating common use cases of OpenSoundscape.

How to cite this article: Lapp, S., Rhinehart, T., Freeland-Haynes, L., Khilnani, J., Syunkova, A., & Kitzes, J. (2023). OpenSoundscape: An open-source bioacoustics analysis package for Python. *Methods in Ecology and Evolution*, *14*, 2321–2328. https://doi.org/10.1111/2041-210X.14196