Automatic Construction of Accurate Bioacoustics Workflows Under Time Constraints Using a Surrogate Model

Alexander Brown[a], James Montgomery[a], Saurabh Garg[a]

[a]*School of Information and Communication Technology, University of Tasmania, Hobart, Australia*

Abstract

Automated bioacoustics analysis is being increasingly used to describe environmental phenomena such as species abundance and biodiversity. Within this research area, many algorithms have been proposed. These achieve different sub-objectives within bioacoustics processes and can be combined to form workflows. However, these algorithms are typically evaluated in a limited number of scenarios and are rarely evaluated with different combinations of other tasks. This can result in workflows that are not well optimised to serve a given scenario, particularly under resource and time constraints, which ultimately leads to inaccurate bioacoustics analyses. This work examines the problem of bioacoustics workflow construction by searching and ordering combinations of tasks to determine which produce the most accurate output while remaining under user-defined time constraints. Workflow construction is investigated within a scenario where species need to be classified within synthetically generated soundscapes with different numbers of species, noise levels, and densities of species. A search algorithm is created that applies Particle Swarm Optimisation (PSO) to a neural network-based surrogate model. This algorithm is used to efficiently search for candidate workflow structures. This is compared to a random search, a genetic algorithm, and a PSO algorithm without the surrogate model, as well as exist-

Email addresses: alexander.brown@utas.edu.au (Alexander Brown), james.montgomery@utas.edu.au (James Montgomery), saurabh.garg@utas.edu.au (Saurabh Garg)

ing workflows based on previous research. It is found that for all scenarios, the surrogate model-based search method can quickly find effective workflows in a low number of searches. Furthermore, it is found that workflow effectiveness varies depending on the scenarios and recordings used.
Keywords: bioacoustics, surrogate model, scientific workflow, particle swarm optimisation, combinatorial optimisation

---

1. Introduction

Bioacoustics analysis provides a cost-effective, unobtrusive way to monitor the environment [17]. In typical analyses, sensors are placed within an environment of interest to record audio. When animals, such as birds, make calls, these sensors can be used to establish their presence and abundance within an area of interest. A key challenge for performing this form of analysis is to determine the sources of sounds within recordings [5]. An obvious approach is to have experts manually listen to and annotate recordings, but the time and cost required to do this become unsustainable at larger scales [17]. As such, computational based approaches have arisen to perform bioacoustics analysis automatically [52]. These typically aim to perform automatic annotation, such as to identify sounds of interest and match these to species [54], or to quantify biodiversity across a region of interest [18].

There is a clear workflow structure for performing automated bioacoustics classification, made up of many tasks which achieve different sub-objectives. However, there are a huge number of ways within this structure from which bioacoustics analysis can be carried out using any combination of different tasks to achieve processing goals. The best approach for carrying out any given processing task is scenario-specific. To best suit researchers' specific needs, a more general method is needed.

Researchers are constrained by time and cost when it comes to analysing their data, and these constraints are variable depending on the research in question. Selecting workflows that are ill-suited to any given scenario can result in

wasted computational resources and time. Such losses in computational efficiency and accuracy are particularly significant, given that some analyses can use several months' worth of recordings and large amounts of computation and time resources to complete [17]. This also means that spending significant computation time trying to find the best workflow is potentially worth the benefit of improved efficiency in the long term.

Currently, research into automatic bioacoustics processes typically examines individual components of a workflow, often studying a task that contributes only partially to an analysis process, and targets one specific scenario. This means that methods are potentially limited to evaluated scenarios without any understanding of their effect on other workflow components. There is currently no research, to the best of our knowledge, that has looked at how to construct accurate and time-efficient bioacoustics workflows to target a wide range of specific processing scenarios. This is problematic because some tasks do not work well together and workflows constructed that combine these might be inaccurate and inefficient. Additionally, some tasks might be unsuitable for some applications. Research in related scientific domains has looked at optimising efficiency and not effectiveness, and only consider a limited number of complete workflow structures (e.g. [40, 2]).

Bioacoustics workflow construction is a challenging problem because there are a huge range of possible applications and scenarios for bioacoustics processing, and a huge range of alternative combinations possible to generate workflows from. Simply brute-forcing all combinations of tasks is not viable for anything outside of simple cases, because the number of possible combinations becomes exponentially higher as more tasks are added to select from. As such, a more intelligent approach is required to determine effective task combinations in a timely manner.

This work examines how to efficiently determine effective bioacoustics workflows. A prototype bioacoustics workflow system is developed containing many bioacoustics processes from a large number of research works. From here, search algorithms are used to select workflows to generate accurate models, going

through several types of processes, to select accurate workflows for a given scenario. This work compares a genetic algorithm, a particle swarm optimsation algorithm, and a surrogate model-based algorithm to determine which is the best for searching workflows.

In summary, this work addresses existing research gaps by contributing the following:

- A mechanism to represent and evaluate bioacoustics workflows which enables similar tasks to be swapped and added into workflows to evaluate tasks' effectiveness in combination with each other rather than individually like in previous works.
- Processes to intelligently search for and select tasks to construct strong bioacoustics workflows quickly, informed by results of previous evaluations.
- An evaluation on realistic data which compares the effectiveness of these processes to each other across different datasets.
- An exploration of whether the workflows constructed that target some scenarios are useful in other scenarios.

Section 2 describes the problem being addressed in more detail. Section 4 describes the mechanism used to represent workflows such that individual tasks can be swapped in and out in order to evaluate a huge number of task combinations. Section 5 then describes the search algorithms used to find strong workflows. Section 6 then describes the testing methodology. Section 7 then summarises the findings of the evaluation. The research is then concluded in Section 8.

2. Problem Description

The core aim of this research is to determine how to find the most accurate combinations of bioacoustics tasks under user-specified constraints. This section defines what a request is for the purposes of this analysis, and, based on this definition, the problem is defined in more specific terms.

This section uses several symbols to describe different properties of the problem and the proposed solution. Table 1 provides a reference to keep track of these key terms.

To contextualise the types of problems that the proposed workflow selection algorithms are targeted at, a motivating example is considered. From there, user requests and the workflow model are mathematically represented within the context of this problem.

2.1. Motivating Example

In late 2019-early 2020, southern Australia experienced a series of devastating bushfires that destroyed animal habitats and had a severely detrimental effect on biodiversity [16]. Ecosystems are currently early in the process of recovering from these bushfires. To help track this recovery, and hence, to help make effective conservation decisions, monitoring of the affected environments is required. However, the region affected was very large, which makes it difficult and expensive for researchers to use traditional in-situ monitoring methods to cover the entire region. Bioacoustics could provide a cost-effective approach for performing analyses at this scale.

The question then is, what exactly do researchers want to analyse. Bioacoustics has a broad use of applications including monitoring specific species and determining biodiversity at a broad level. Perhaps one analysis might look at the distribution of glossy black cockatoos who lost a significant amount of their food source in the fires [13], while another might want to track a broader range of species to calculate changes in biodiversity as the forests recover. Each analysis is somewhat unique, with different species making different sounds, different applications requiring different tools, and differences in background noise masking sounds of interest in different ways. In the case of cockatoos, their calls are noisy, compared to other animals whose calls are more tonal, and this could influence which methods are best suited for detecting them. Researchers need to find what the right tools are for the analysis they are doing. However, current works are limited in that they only target specific applications in specific

Table 1: Key terms used to define requests, workflows, etc. in this work

Term	Meaning
R	User Request
D_a	Annotated (i.e. Training) Dataset
b_a	Annotated audio recording
A	Annotations
D_u	Unannotated Dataset
b_u	Unannotated audio recording
G	Request goal
c_1	Time constraint for processing workflows
c_2	Time constraint for selecting workflows
W	Workflow used to fulfil R
τ_i	Task in the workflow
F	The function of a given task
β	The type of function performed by a given task
$\iota(n, m)$	The quantity (n) and type (m) of input used by a given task
$o(n, m)$	The quantity and type of output used by a given task
σ	A stage within a workflow template. All tasks in a stage have the same processing type β
ζ	A workflow template made up of multiple stages (σs)
P	Pool of tasks which can be selected to construct workflows

scenarios. This work aims to generalise workflow representations and find the best workflow for any goal.

2.2. User Request

A user request is of the form $R(D_a(b_a, A), D_u(b_u), G, c_1, c_2)$. This contains:

- An annotated dataset D_a with recording data b_a and annotations A
- An unannotated dataset D_u with recording data b_u
- An overall goal G
- Time constraints c_1 for workflow processing and c_2 for searching for the best workflow.

This is summarised in Figure 1. For the glossy black cockatoo example, there might be a recording dataset where the cockatoo sounds have already been identified, and another dataset which is to be annotated automatically. This dataset could be made up of recordings in different locations. The goal could be to find all the glossy black cockatoo sounds in a set of forest recordings, given an annotated recording for example. The output can be a list of times and locations where the bird's call was detected, which could be used to track their distribution. While this work primarily focuses on classification problems, there is no reason why annotated data could contain different types of information, such as biodiversity indices that describe phenomena more broadly [46].

The goal G of a request is to in some way annotate the raw bioacoustic dataset D_u which contains bioacoustic recordings b_u to correlate with ground truth annotations from D_a in less time than the constraint c_1. The core goal G can be to annotate D_u directly using the same format as A, or calculating a statistic, such as a biodiversity estimate (e.g. the number of species in a time window) that correlates as closely as possible to A.

A goal G is achieved through processing combinations of tasks τs in the form of a workflow $W(\tau_1, \tau_2, ..., \tau_N)$. Different combinations of tasks will take different amounts of time to execute and have differing levels of effectiveness,

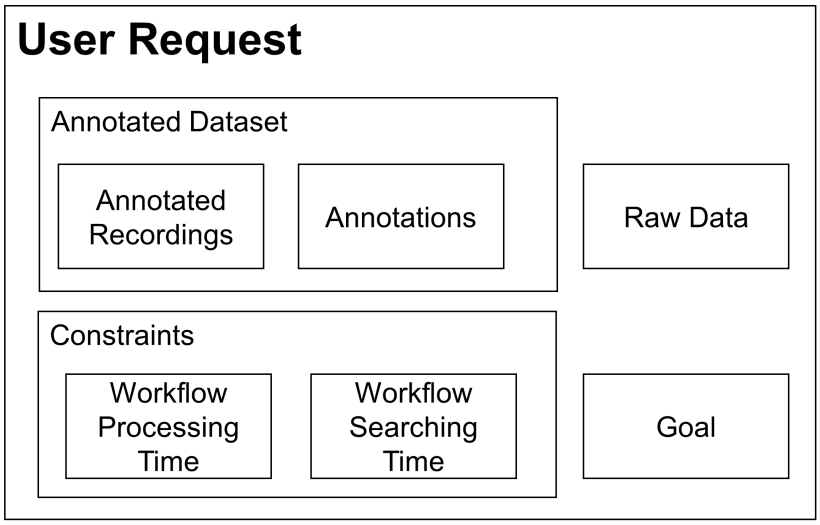


Figure 1: Graphical Summary of User Request

and these will depend on several scenario factors such as the amount of audio being processed and background noise.

2.3. Problem Objective

This research is aiming to address problems where researchers want to find, given a pool of tasks $P\{\tau_1, \tau_2, ..., \tau_n\}$ which workflow W most accurately fulfils a goal G given time constraints c_1 and c_2. Because of the time constraint c_2 and the potentially huge number of possible permutations of W, this cannot be done by simply brute forcing every possible workflow. As such, a more intelligent approach for evaluating workflow structure is needed.

3. Related Works

Workflow construction from smaller tasks is an uncommon problem not explored in existing scientific literature. Related problems have been explored in the literature such as selecting which workflows to processes to run in scenarios where not all deadlines can be reliably met (e.g.[40, 2]). While problems such as these do contain workflows and some form of selection, these are very different

problems. Complete workflow structures are known and the only variables of interest are the workflows' total execution times, which are approximately known. The challenge explored in this work involves building workflows from smaller components and the number of possible workflows is well into the billions. Furthermore, this challenge also targets processing accuracy, which is difficult to estimate. Even execution times are not known prior to measuring. As such, methods used in these related workflow selection problems are not applicable to this problem.

However, this problem can be mapped to other mathematical optimisation problems. In particular, combinatorial optimisation problems are a family of problems where the goal is to select from a finite set of discrete objects to maximise or minimise a function. A classic problem that is analogous to this problem is the Knapsack Problem. The goal of this problem is to select the highest value elements that fit within a constraint (often called 'profit' and 'weight'). This problem is more complicated than a typical knapsack problem, most notably because the values and weights of items (workflow tasks in the case of the research problem here) vary depending on what other items are being used and on the specific scenario being evaluated. It is also more difficult and time-consuming to evaluate solutions, and there are extra constraints as to which items are allowed. For example, only one activity detector can ever be used. In short, this is a knapsack problem with unknown, highly non-linear functions for calculating both the profit and weight that are slow to evaluate, subject to an irregular set of constraints.

Some generalisations of the knapsack problem are closer to the problem being examined. The quadratic knapsack problem (QKP) for example gives additional values for having specific combinations of items. This can be relaxed further, such as in the Quadratic Knapsack Problem with Conflict Graphs (QKPCG), which adds an additional constraint of having items that are incompatible with each other. This problem is the subject of a work by Dahmani et al. [11]. They use a binary Particle Swam Optimisation (PSO)-based approach, noting its generality and simplicity. They apply an additional repairing procedure to

ensure incompatible solutions are not selected. This however assumes it takes a short amount of time to evaluate a solution. Additionally, workflow structures dictate which combinations of tasks are not permitted, and this can be used to create an alternative representation that can ensure every PSO solution is valid. There are also constraints regarding some options that need to be selected from, unlike the QKPCG problem.

Another relaxation is the Generalized Quadratic Multiple Knapsack Problem (G-QMKP). This uses multiple 'knapsacks' which can hold items from different sets, each with their own constraints. Items can belong to different classes, which can be subject to different restrictions, and these items can have different profits depending on which knapsack they are selected for. Avci and Topaloglu [4] use a multi-start iterated local search (MS-ILS) approach to solve this problem which is based on searching around a local neighbourhood for a local optimum before applying perturbations to move the search to other neighbourhoods. A challenge with the problem being addressed here is that it is very difficult to determine what a small change is and what a large change is. Adding a redundant acoustic feature calculation task might make no change to results except for adding execution time, while adding a very useful task of the same type could dramatically improve results, and it is difficult to know which one will do what. The function represented in the workflow selection problem is so complex that it is difficult to know what a small change is compared to a large change without trying them.

A key challenge in this work is that the time needed to evaluate workflows is very long. This challenge has been addressed in other domains by the use of surrogate models. The idea behind these models is to replace the real-world evaluation with an approximate model that is much faster to evaluate. Optimization can then be aided with this model. Examples include Bisbo and Hammer for global structure optimisation [8], Liu et al. for microwave filter optimisation [30], and Song et al. for coastal aquifier management [42]. Such a model is normally not needed in knapsack-like problems as the functions involved are known and simple prior to the optimisation process beginning. However, this

problem has interactions with components which are unknown and difficult to evaluate and could benefit from such a model.

To summarise, while similar problems exists and have been explored in literature, these problems are simpler than the bioacoustics workflow construction problem being examined here. One way to categorise this problem is a knapsack problem with highly non-linear black-box functions for profit and weight where item values depend on each other, and even the order in which the items are placed. While some problems share some characteristics with this problem, this has not been explored to the best of our knowledge in the literature. Furthermore, there are additional characteristics to this problem which make it even more difficult, such as a slow search time, and constraints as to which items can be used at once.

4. Workflow Construction Model

Bioacoustics processes are often similar in structure, with analyses being essentially made up of workflows of smaller subtasks that are used for different sub-objectives. While the structure is often similar between bioacoustics analyses, there are a large number of algorithms proposed in the literature to perform a variety of different bioacoustics analyses under different scenarios and conditions. If bioacoustics workflow structures can be generalised, then it becomes possible to interchange and compare combinations of tasks to select more effective and efficient workflows.

The workflow construction process aims to build a workflow W that achieves user goals as effectively as possible. The user goal G itself informs how processing will be done, particularly in terms of workflow structure. These goals require workflows with different structures which inform what tasks are selected for creating workflows. In order to examine these structures, processing tasks themselves need to be formally defined. A task is represented in the form $\tau_i(F(b_j), \beta, \iota(n, m), o(n, m))$ where $F(b_j)$ is the operation the task performs on bioacoustic data bj, β is the task's type, ι is the task's type of input, and o is its type of output, both of which have a quantity n and type m. This is essentially

a general representation of workflow task execution that would be used in many scientific workflow applications.

The *type* of processing, β, defines what each task actually does. This loosely informs what order to performs tasks in and what tasks are competing with each other. There are six types of processing to consider:

1. **Audio Pre-Processing**: Modifies the audio prior to any analysis. Includes noise reduction.
2. **Activity Detection**: Isolates sounds of interest from raw recording data.
3. **Audio Representation**: There are several ways to transform audio signals to give information not available by merely examining their waveforms, for example, Short-Time Fourier Transforms (STFTs). These can be used to further calculate acoustic features.
4. **Acoustic Feature Calculation**: Calculates statistics of audio samples to be used for classification.
5. **Feature Pre-Processing**: Alters features into a format that is easier for classifiers and regressors to work with.
6. **Classification/Regression**: Takes acoustic features and fits sounds to models based on data annotations.

To enable bioacoustics workflows to be easily generated and compared, workflow structures are defined using templates. A template is made up of several workflow stages (σ). These can accept different tasks depending on what type of processing (β) they perform and are limited by the number of tasks they can perform (usually 0-many, 1-many, or exactly 1). For example, in a linear classification task, σ_1 might only permit audio pre-processing, while σ_2 might permit activity detection or additional denoising, etc.

From here, a template can be defined using $\zeta = \{\sigma_1, \sigma_2, ..., \sigma_n\}$, from which a workflow is defined as

$W = \{\sigma_1(\tau_1, \tau_2, ..., \tau_i), \sigma_2(\tau_{i+1}, \tau_{i+2}, ..., \tau_j),$
$..., \sigma_m(\tau_k, \tau_{k+1}, ..., \tau_n)\}.$

In other words, workflows are selected based on selecting combinations of tasks that fit the template. Within these templates, one task's input (ζ) also needs to be compatible with outputs (o) from previous tasks. In some cases, multiple os from one task might be processed by several instances of a subsequent task that take in one input each ζ. Intermediate tasks might need to be executed to ensure that tasks can receive inputs of the correct type. Templates are based on a user's processing goals. For example, a biodiversity calculation might be only concerned with long term trends and compute indices for long durations of audio while a population monitoring task for a specific species might want to uniquely identify individual calls or even individual syllables of a call.

Figure 2 shows an example of a potential workflow structure. Tasks τ_a and τ_b are in the first phase of the workflow. They could be of different types, as long as they are permitted by the template ζ. The type of input from τ_b must match the output of τ_a. In other words $m_{o_{\tau_a}} = m_{\iota_{\tau_b}}$. However, τ_b has multiple outputs which are sent to τ_c which only accepts one input. As such, multiple instances of τ_c are created accepting one output each. Then, τ_d accepts every output from all instances of τ_c.

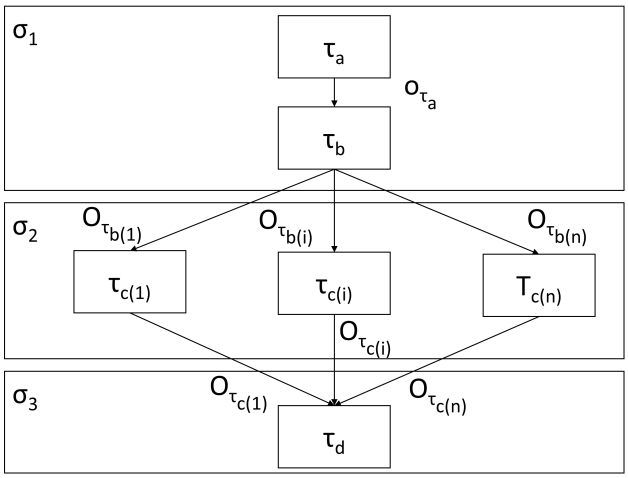


Figure 2: Example of bioacoustics workflow

This proposed workflow structure, encapsulating workflow stages, processing

types, and templates, enables tasks to be swapped with each other or executed in different orders and different combinations. As a consequence, it becomes possible to evaluate different bioacoustics workflows easily. Based on this workflow structure, a process for executing workflows, including constructing, evaluating, selecting, and executing accurate and efficient workflows is described in Figure 3. The system will first map a user request to a template. In this work, templates are pre-defined prior to searching during testing. Upon mapping to a template, a workflow searcher will test the effectiveness of individual workflows, one after another, until a stopping criterion is reached. Specific selection mechanisms are described in Section 4 and evaluated in Section 7. These workflow evaluations can inform which workflows are selected in subsequent evaluations.

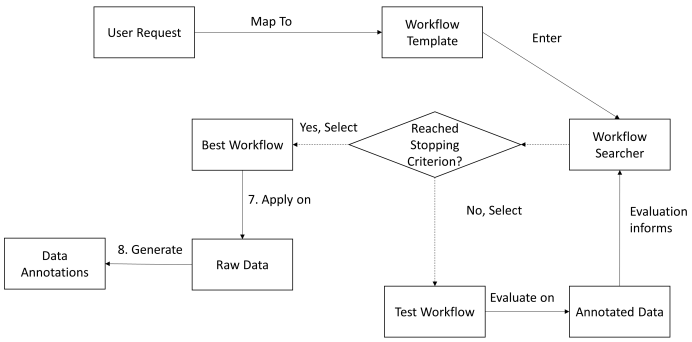


Figure 3: Summary of Workflow Processing

5. Investigated Search Algorithms

As previously established, workflow searching cannot be achieved efficiently using brute force methods in anything other than very simple templates with very few options because the search space is potentially huge. As such, it is important to search workflows that are likely going to improve and previous searches so that a good workflow can be selected in a reasonable timespan. As such, this work examines the potential for evolutionary algorithms to construct and evaluate workflows that are similar to those that were previously found to be

accurate. Specifically, a genetic algorithm, a PSO algorithm, and a surruogate model-based algorithm are compared to a random workflow search to determine the potential of such strategies.

5.1. Genetic Algorithm

A Genetic Algorithm (GA) is a population-based algorithm where a population of solutions compete to maximise the fitness functions [21]. At the end of each iteration, a selection mechanism decides which members of the population will be used to determine a new population for the next round. These selected solutions generate new solutions using a crossover step, where parts of two solutions are swapped with each other, and a mutation step, where some values within the solutions are changed or swapped randomly with each other.

For the workflow construction process, each solution represents a permutation, similar to other existing works [7]. Most elements of the solution represent permutations for each workflow stage. Each element represents a unique bioacoustic processing task. These are preceded by elements representing the number of tasks to be selected during template stages where the number of tasks is not fixed. The permutations are truncated to the length given by these length elements.

For example, say there is a task pool consisting of 3 denoisers, 3 activity detectors, 3 feature set calculators, and 3 classifiers. The template defines that there can only be one activity detector and one classifier per workflow. As such, these workflow stages do not require any elements representing how many tasks to choose (i.e. length elements). A potential solution is represented in Figure 4.

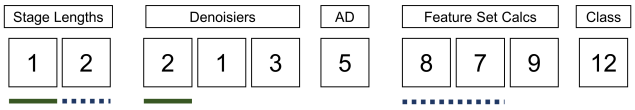


Figure 4: Solution structure for GA representation

This would correspond to a workflow with 1 denoiser and 2 feature set calculators. Values 3–5 in the solution represent denoiser tasks, however, only one

denoiser is specified by the workflow, so only the first value (2) is used. This corresponds to a denoiser task with ID 2. Then, the 6^{th} value corresponds to the activity detector with ID 5. Values 7–9 are feature set calculators. As the second value is 2, the first 2 values are selected, corresponding to task IDs 8 and 7. Finally, the task with ID 12 is selected as the classifier. The final constructed workflow is then [DN2, AD5, FS8, FS7, CL12], where DN, AD, FS, and CL refer to the types of tasks. This will be the order of execution, although some tasks might be able to be executed in parallel if they do not affect each other.

Tournament selection is used to determine which solutions will be parents for the next generation. Here, a subset of random solutions is nominated and the one with the highest fitness is selected to be a parent. This process repeats until all parents are selected. In the evaluations, these subsets include four solutions. To generate child solutions, permutation crossover and mutation approaches are applied. For the permutation crossover, two solutions are combined, swapping between the two at a crossover point. Any stages finishing earlier than the cross point are copied in their entirety from one solution, and any starting after the cross point are copied in their entirety from the other. For the stage in the cross point, tasks from the first solution are copied in, and then, from the beginning of the stage, tasks are copied from the second solution, skipping any duplicated from the first one.

For elements representing the number of tasks in a given stage, Gaussian mutation is applied, whereas for other elements, swap mutation is applied with a fixed probability of any given task being swapped with another task from the same stage (or replaced in the case a stage only has one task).

Upon completing crossover and mutation, fitness values are recalculated and the cycle repeats until a set number of searches are carried out.

5.2. Particle Swarm Optimisation

Particle Swarm Optimisation (PSO) [23] is a swarm-based search algorithm. Here, the solution space is represented as particles that move semi-randomly, guided by the local and global maximum particle positions of previous itera-

tions. The first dimensions of the particle search space are the same as for the GA, representing the number of tasks for each stage within a workflow. The other dimensions represent how preferred each task is. The tasks of each type, whose associated dimensions have the highest values out of those permitted by a template stage are added to the workflow. The same task can only appear in one stage where there are multiple workflow stages allowing tasks of the same type.

This implementation uses a ring topology where particles only know about particles that are logically (i.e. arbitrarily labelled as) adjacent to them. The number of adjacent particles to be considered is set to 2 for the purposes of testing.

After each round, the particles move as follows for each dimension [45]:

$$v_{i,j}(t+1) = v_{i,j}(t)\alpha + R_1\beta(p_{i,j}(t) - x_{i,j})(t) + R_2\beta(p_{l,j} - x_{i,j}) \tag{1}$$

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1) \tag{2}$$

Where $v_{i,j}(t)$ is the velocity of particle i in dimension j at time (i.e. generation) t, $p_{i,j}$ is the best particle's j, l is the particle in the local neighbourhood of i with the best score, $x_{i,j}(t)$ is the particle's position, α and β, and R_1 and R_2 are random numbers between 0 and 1.

Particle boundaries are set to the range 0–1 for tasks and the minimum and maximum number of tasks for values indicating the number of tasks in a stage. If these boundaries are exceeded, a variation of damping [22] is used where particles are reflected with half of their initial velocity. This damping is done because standard reflection was found in early testing to sometimes result in diverging velocities, rapidly alternating between positive and negative.

Additionally, the particles' positions and velocities, with the exception of the first dimensions indicating the number of tasks for types where this is applicable, is limited to the range (0,1). If a particle's dimension exceeds this bound, it is treated as if it has hit a wall and reflects in the opposite direction with half its

existing velocity. In other words, for any given dimension:

$$\text{Adjusted Pos} = \begin{cases} \frac{-\text{Initial Pos}}{2} & \text{Initial Pos} < 0 \\ \text{Initial Pos} & 0 \leq \text{Initial Pos} \leq 1 \\ 1 - (\frac{\text{Initial Pos} - 1}{2}) & \text{Initial Pos} > 1 \end{cases} \tag{3}$$

$$\text{Adjusted Vel} = \begin{cases} \frac{-\text{Initial Vel}}{2} & \text{Initial Pos} < 0 \\ \text{Initial Vel} & 0 \leq \text{Initial Pos} \leq 1 \\ \frac{-\text{Initial Vel}}{2} & \text{Initial Pos} > 1 \end{cases} \tag{4}$$

Where pos is the position, and vel is the velocity. Dividing by two helps to prevent divergence, where particle velocities flip between increasingly extreme values in either direction.

The dimensions representing the number of tasks for a given stage work the same way, except they are bound by the number of tasks allowed for that stage.

5.3. Surrogate Model

Evaluating workflows even with a low amount of training data is time-consuming, and this, combined with the number of possible combinations, means that metaheuristic models cannot feasibly search enough combinations to optimise for this problem. However, the problem can be mapped to a model that can be evaluated quickly. This is the central idea of surrogate models [3]. These have been used in other applications, often related to engineering optimisation problems (e.g. [39, 32]), but this form of modelling has not been used to select bioacoustics workflows to the best of our knowledge.

The choice of surrogate model in this problem is an Artificial Neural Network (ANN). This is trained to predict the fitness score of any solution. Metaheuristic search algorithms can then go through many iterations quickly to find solutions with high fitness, and the best workflow found can then be evaluated on the real model.

This method works as follows:

1. Evaluate a bioacoustics workflow (i.e. using real data)

2. Train a neural network to estimate the fitness of workflows based on previous evaluations
3. Use an algorithm to search workflows using the ANN model to predict fitness values
4. Select the workflow with the highest fitness based on this search that has not been previously evaluated.
5. Return to step 1, evaluating the workflow selected.

Early in the search, the neural network will be wildly inaccurate in estimating scores for workflows because there will not be enough information for it to find any meaningful patterns, but this is fine as the resulting selected workflows would be near random using any other search algorithm anyway.

To encourage this search algorithm to select diverse workflows so that it does not prematurely converge, an extra function called 'novelty' is added to the estimated score. This is defined as:

$$\text{Novelty} = \frac{\sum_{k=1}^{n} \max(0, \alpha - \beta \times s(t_k))}{|\text{Tasks in new WF}|} \tag{5}$$

Where t_k is a task, $s(t_k)$ is the number of workflows featuring a given task t_k have been evaluated in the real-world model, and α and β are parameters that affect the sensitivity and weight of the function. For this evaluation, α and β are set to 1 and 0.05 respectively (meaning after a task has been used in 20 workflows, it has no effect on the function), but these could be changed to encourage searching for more or less diverse workflows. For reference, the score of a workflow is always between 0–1 (although the surrogate model is allowed to exceed these bounds).

The ANN-based surrogate model is depicted in Figure 5. This uses a sparse representation of the bioacoustics task pool as input, with zero indicating a task is not in a workflow, and one indicating it is used. This representation does not consider task order, meaning any ordering decisions in selected workflows are essentially random, but this does simplify the input representation. This is connected to three dense layers, each containing 32 nodes. Finally, a predicted

fitness score is output in the final layer. Each dense layer is L2-regularized ($\lambda = 0.001$). An Adam optimiser [24] is used to optimise the network weights with a mean squared error loss function. The network undergoes 1000 epochs, which can be performed very quickly as the network is very simple and has a low amount of training data for the evaluations being performed.

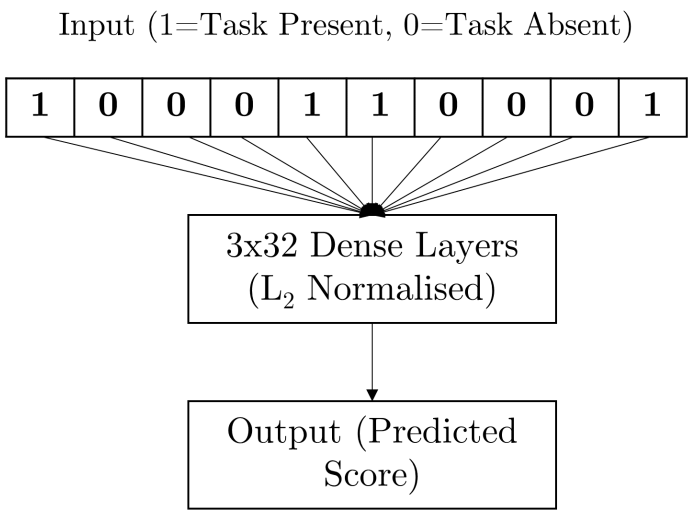


Figure 5: Neural Network Surrogate Model

In this evaluation, the PSO search algorithm described in Section 5.2 is applied to the surrogate model to find candidate workflows for evaluation on the real model. As the surrogate model accepts the same type of input and output as the real model, excluding the transformation of the workflow to a sparse representation, the PSO search operates the same way as it would do on the real model. This work makes no attempt to optimise the fit of the surrogate model to real evaluation data or to the search algorithm and only intends to evaluate the viability of the use of surrogate models for this problem. The use of PSO with a surrogate model has been previously found to be effective in other optimisation problems (e.g. [14, 33]).

5.4. Random Search

As a baseline, workflows are searched randomly using the GA representation as a basis. This works by randomly generating as many 'solutions' as desired searches with no selection and mutation phases. This approach will determine if intelligent workflow selection is effective for this problem.

The use of the GA representation does mean that this search is not truly random in the sense that not each possible combination is selected with equal probability because the GA representation selects length with uniform probabilities, while distribution of lengths of possible workflows is not uniform. This is not necessarily a bad thing however because it encourages searching for workflows of varying complexities, whereas a random selection where each combination is equally likely would strongly favour moderately complex workflows, which might not necessarily be optimal.

6. Testing Methodology

To evaluate the workflow search mechanism, the following questions are examined:

- Which workflow searches find accurate and efficient bioacoustic workflows the quickest? How do these search algorithms progress over time?
- Which workflows are selected by these search mechanisms? Are there any preferences towards specific tasks and combinations?
- Do the workflow search algorithms select better workflows than pre-selected workflows based on existing research?
- Are the selected workflows scenario specific? How well do selected workflows perform when moved to different audio?

To address these questions, an implementation of the workflow search algorithm, based on the previously developed AcoustiCloud [10] architecture, is developed. The aim is to develop a set of tasks of each of the different types and select tasks based on workflow templates.

6.1. Evaluation Processes

The workflow selection algorithms are evaluated using real bioacoustics processes on realistic datasets. The *AcoustiCloud* prototype enables workflows to be generated by dividing tasks by type and selecting these from workflow templates. The selection algorithms can be used in these systems to find strong workflows. Three different evaluations are performed on four separate datasets of different complexities. These datasets represent different scenarios, perhaps indicating separate recording locations with varying amounts and types of activity, which could require different types of processing.

- Evaluation of the ability of the search algorithms to select strong workflows
- Comparison of the best workflows selected by the search algorithms to workflows selected based on existing literature
- Cross-examination of the best workflows found on each dataset against each other to determine if workflow performance change based on the datasets involved (and hence, different recording scenarios)

Five hundred workflows are evaluated for each algorithm per test. For the GA and PSO, this is done in the form of 50 workflows per generation for 10 generations. For the surrogate model, only one workflow is selected at a time before the model is re-executed. To more directly compare with the GA and PSO, results of these evaluations are split into groups of 50, (by order of evaluation) similar to generations in the GA and PSO (labelled '**group numbers**' in subsequent figures), although these are not 'true' generations. During the evaluation of the surrogate model search, 20 particles are evaluated through 10 iterations on the surrogate model (for a total of 200 workflows searched) and the discovered workflow with the highest fitness is evaluated on the real data. This means that 100,000 workflows are evaluated on the surrogate model and 500 on the real data. The time taken to build the model and perform the PSO search after each real-world workflow test is less than 30 seconds per real-world workflow searched, which is proportionally very small compared to searching

one workflow on the real datasets. Only workflows evaluated in the real world are shown for the surrogate model evaluation. For all search algorithms, the vast majority of the time is spent evaluating solutions on the real world data and the time taken by the search algorithms to discover new workflows to evaluate is proportionally very small.

Mean Average Precision (MAP) is used to determine the fitness of each workflow solution. As opposed to raw accuracy, MAP considers classification probabilities as opposed to binary true/false classifications. As such, this metric is more revealing about how well a workflow is classifying the sounds of interest, particularly in scenarios where researchers might want to change the sensitivity of the classifier. This has been used in previous bioacoustics research to evaluate similar classifiers (e.g. [25, 19, 28]). This is evaluated using

$$MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q} \tag{6}$$

Where there are Q classes and

$$AveP(q) = \frac{\sum_{k=1}^{n}(P(k)rel(k))}{\text{Number of documents of class } Q} \tag{7}$$

where $P(k)$ is the precision at rank k and $rel(k)$ is 1 if a document contains class q or otherwise 0. Here, the rank is the sorted probability output by the classifier of a sample containing q. When computing the MAP, ground truth annotations and classifier outputs are snapped to the nearest 5-second interval, under the assumption that sounds are to be identified to this level of accuracy. If there are multiple classifications for audio segments (e.g. if an activity detector task identifies multiple distinct sounds in the same 5-second interval) within the same 5-second interval, the one with the maximum probability of containing the class is used. In other words, if there is a classification of 0.5 between 0 and 2 seconds into the recording, and one of 0.25 between 2 and 5 seconds, a probability of 0.5 for the range of 0–5 seconds for the purposes of calculating the MAP. This is done, as opposed to an average, because any bird is likely to not be calling for the entire five-second duration of the sample.

When searching, workflows cannot exceed a specified time deadline. This deadline is set because in a realistic scenario, time and cost budgets will restrict the types of processing available to use on very large datasets, and the workflow searcher would take a prohibitively long time if it was able to consistently select very long workflows. If workflow deadlines are exceeded, fitness scores are subtracted by the amount of overtime divided by 10.

Prior to searching through the main audio, workflows must complete processing of the two-minute audio dataset in 60 seconds for the synthetically generated audio and isolate at least one segment. This is done to remove workflows early that take an exceptionally long time to complete or discard all audio to avoid wasting processing time on larger datasets. Workflows failing these constraints have a further 100 subtracted from their fitness scores for the GA and PSO search algorithms to strongly discourage their selection in subsequent workflows.

For the surrogate model search, and for showing evaluation results, any scores which are less than zero are set to zero. For the surrogate model search, this makes it easier to generate a more accurate model as the distribution of scores is kept to a narrower bound and is less skewed, while for evaluation, mean workflow scores are less biased towards outlier workflows that take an exceptionally long time to complete. On the other hand, this is not done during the selection phases of the GA and PSO search algorithms to give more pressure against selecting very long workflows.

To speed up workflow searching, an approach used in the previously developed AcoustiCloud prototype [10] is to intermediate data from previous searches. While this does speed up searching, this could result in workflows that might have otherwise exceeded the time deadline being passed through. As such, after testing, workflows with high scoring MAPs are processed once without reuse to determine if they actually process under the deadline constraint, or if they only got under because of speed-up due to data reuse.

6.1.1. Bioacoustics Datasets

There are four different datasets used in this evaluation. Three of these are synthetically generated soundscapes and one is an existing dataset used in a bioacoustics competition. The synthetic Soundscapes are artificially generated using Scaper [41]. This tool enables users to generate bioacoustics soundscapes with different levels of background noise, and inserts recordings of different foreground sounds and random, often overlapping times, which can be time-stretched, pitch-shifted, etc. and outputs annotations indicating where each sound was inserted. Multiple recordings of several bird species were taken from Xeno-Canto (2–4 each) and these were used to generate realistic soundscapes. Some Xeno-Canto recordings had multiple calls from these birds, and these were each manually segmented and split from recordings, with any silence removed. Pink noise is used as background noise for these soundscapes. Soundscapes are generated with different numbers of species (2, 4, and 8), densities of calls (one call every 15 seconds, 5 seconds, and 3 seconds for 2, 4, and 8 species respectively on average, randomly placed in the recording using a uniform distribution), and noise levels (more noise for higher species tests, although it is difficult to quantify). The Sampled species are shown in Table 2 (the numbers of calls for each species are approximately equal regardless of the number of unique samples available):

Table 2: Bird Sounds used in synthetically generated recording

Species	# Samples	In Tests
Indian Peafowl	3	2, 4, 8
Torresian Crow	12	2, 4, 8
Eastern Yellow Robin	15	4, 8
Silvereye	12	4, 8
Brush Cockoo	4	8
Laughing Kookuburra	6	8
Masked Lapwing	10	8
Variegated Fairywren	11	8

Alongside these, a real dataset used in a competition, the MLSP 2013 bird detection challenge is also tested. This dataset is selected because of its similarities to the problem being solved here, its relatively small size (approximately 1.5 hours), and easy comparability to existing methods. This dataset includes recordings from 13 locations and 19 species. The dataset contains supplementary data, such as outputs from supervised segmentation and some precomputed features. The location information can also be used to assist with classification. Essentially, the evaluation is done under the rules of the competition. Any supplementary information is essentially added to the system as extra 'tasks' that can be selected to improve processing.

For each scenario, three different recordings are used:

- A very short (2 minute) sample used to determine if a selected workflow will keep some audio or is likely to exceed the deadline. This is the same over all evaluations and has similar characteristics to the 2 species audio.
- An annotated audio sample used to train the machine learning models (24 minutes for the 2 species recording, 20 minutes for the 4–8 species recordings, 53 minutes, 40 seconds for the MLSP dataset)
- An annotated audio sample used to evaluate the trained models (6 minutes for 2 species recording, 5 minutes for 4–8 species recordings, 53 minutes, 40 seconds for the MLSP dataset)

The 2 species recordings are longer, but also contain a significant amount of near-silence and are overall the fastest to process with effective activity detection, while the shorter 4–8 species recordings are more densely populated with bird sounds with far more overlap. Time deadlines are set to 10 minutes for the synthetic datasets and 15 minutes, 40 seconds (i.e. 1000 seconds) for the MLSP dataset. While the time deadline is longer for the MLSP dataset, this is overall much more strict, considering how long the length of the recording data is compared to the synthetic datasets.

6.1.2. Baseline Workflows

The evaluated algorithms are compared against three existing workflows:

- A two-stage approach to automatically detect and classify woodpecker (Fam. Picidae) sounds [50].
- Spherical k-Means clustering-based classification algorithm [43].
- Wavelet-conditioned Convolutional Neural Network [25]

On top of the three methods, two other methods were added to evaluate against the MLSP dataset. These methods use the supplemental data provided by the competition organisers. The first one uses location information alongside provided segment histogram information in which segment data is clustered using k-means++ to get a feature set for each 10-second interval. The second one uses the segmentation information provided and the provided feature set, alongside location information. Both of these are not pre-processed beyond anything that was pre-provided and a random forest classifier is used to determine bird species.

6.2. Bioacoustics Task Suite

There are several tasks of each stage used for this processing. These are shown in Table 3.

Libraries used for the implementation include Scipy [51], Numpy [34, 48], PyWavelets [29], Tensorflow [1], Scikit-Image [49], Scikit-Learn [36], Librosa [31], and Sidekit [26].

6.2.1. Spectrogram-Based Activity Detector

As part of testing, a known good algorithm by Lasseck [27] used an activity detector that drew bounding boxes around spectrograms. However, the article only describes the activity detector as selecting 'all connected pixels exceeding a certain spatial extension' after several denoising processes, including binarizing the spectrogram (i.e. setting pixel values to 1 or 0). Based on this, a similar algorithm is replicated here.

Table 3: Pool of tasks used to select workflows from

Processing Type	Process Name
Pre-Processing	Splitting audio (to either 2, 5, 10, or 30 seconds) Downsample to 22.05 kHz Convert stereo signal to mono
Denoising	High-Pass Filter Minimum Mean Square Error Short Time Spectral Amplitude (MMSE STSA) filter [15] Median Clipping (uses include Bedoya et al. [6]) Binary Dilation [27] Binary Closing [27] Peak Normalization (this is a pre-processing task that is considered a denoising task within the system)
Activity Detection	Hilbert Follower [38] Spectrogram-Based detection (custom-made, based on [27]) Classifier-Based silence detector [50] (baseline evaluation only)
Acoustic Features	Mel Frequency Cepstral Coefficients (MFCCs) [12] Linear Predictive Coding (LPC) coefficients [35] Linear Prediction Cepstral Coefficients (LPCCs) Perceptual Linear Prediction (PLP) Coefficients [20] Spectrogram-based features (maximum, average, and standard deviation for frequency bins, size reduced and flattened spectrogram) Spectral Entropy[44] Acoustic Cover (CVR) [47] Acoustic Complexity Index (ACI) [37] Temporal Entropy [44] Segment Length Root Mean Square (RMS) Zero-Crossing Rate (ZCR)
Classification Pre-Processing	Normalisation and ZCA Whitening [43] Spherical K-Means Feature Learning [43] Principal Components Analysis (PCA) [53]
Classifier	Random Forest [9] K-Nearest Neighbour Convolutional Neural Network (CNN) (combines spectrograms and acoustic features) Artificial Neural Network (ANN)

Firstly, the spectrogram is normalized per frequency band. Specifically, the spectrogram is divided into $x \times n$ squares. If the sum of the spectrogram intensities is greater than the median plus 3 standard deviations of the intensity for a given frequency band, the square is given a value of 1, or otherwise 0. Then, for each square, the algorithm looks at adjacent squares (first to the right, then to the left) to see if they are shaded. If a certain proportion is shaded, then a bounding box is extended to the adjacent points. This repeats until there are not enough shaded squares in either direction to extend the box. If the resulting box is large enough, it is accepted as a new segment. The core idea is shown in Figure 6.

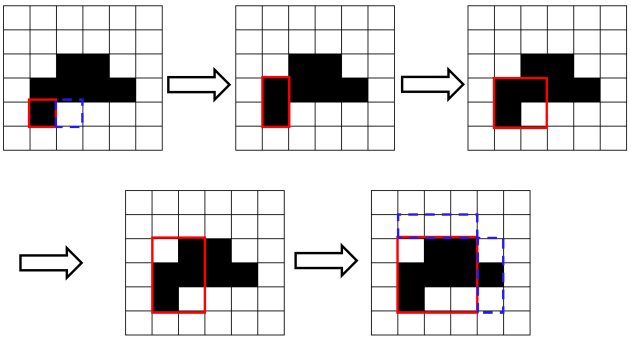


Figure 6: Spectrogram-based intensity detector demonstration. This assumes that the box is extended if at least 50% of adjacent cells are shaded. The red box illustrates the selected segment, and blue dotted boxes represent rejected extensions of the box.

A typical output is shown in 7 showing the algorithm accurately identifies regions where there are sounds, but tends to subdivide complex bird calls into too many parts. However, also note that the low frequency (Torresian Crow) calls have been correctly separated from the other calls occurring at the same time. This algorithm is not intended as a core contribution of the paper (it is mostly an attempt to replicate the work of Lasseck [27]), but can be used to determine the feasibility of this approach against amplitude detection. Segments are band-pass filtered within the identified frequency ranges as well as split into their time segments in future steps.

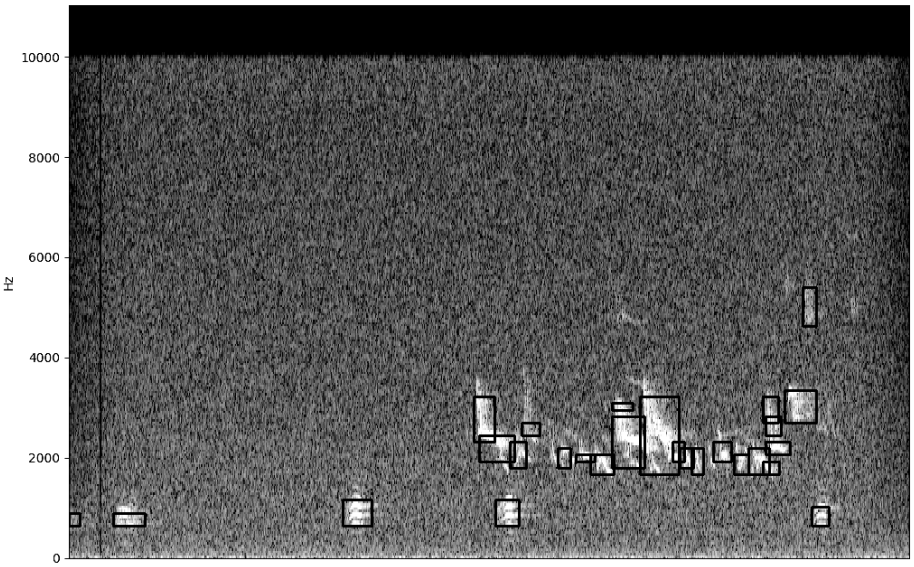


Figure 7: Output of a typical spectrogram intensity detector process. The timespan of this spectrogram is 10 seconds, Identified segments in boxes. The contrast and brightness of the raw spectrogram have been altered for clarity

6.2.2. CNN and ANN Architectures

The CNN and ANN architectures are designed similar to previous works that used them [50, 25]. In the CNN spectrogram images are resized to a certain resolution (given by parameters). For short segments caught isolated by an activity detector, the spectrogram is padded by repeating the segment as many times as needed to fill the image. This is done as opposed to stretching, which would cause inconsistent temporal resolutions. The spectrogram is also converted to a dB scale and normalised, such that the loudest pixel has a value of 1, and the quietest has a value of zero.

The CNN begins with an input layer, followed by a 3×3 convolutional layer with 32 filters, and then a 2×2 max pooling layer. Following that are a series of 3×3 convolutional layers with 64 filters and 2×2 max pooling layer, the number of which is determined by a parameter. This is then flattened. Meanwhile, another layer featuring calculated acoustic features connects to a 32 value dense layer which connects for a 4 value dense layer. Then, this is combined with

the flattened spectrogram layer. This flattened layer is connected to a 128 value dense layer, then to a $4 \times$ number of classes dense layer, and finally to an output of size equal to the number of classes. The ANN architecture is very simple, with the inputs being sent to a number of dense layers of a given size, before being sent to an output layer. The architectures are summarised in Figure 8.

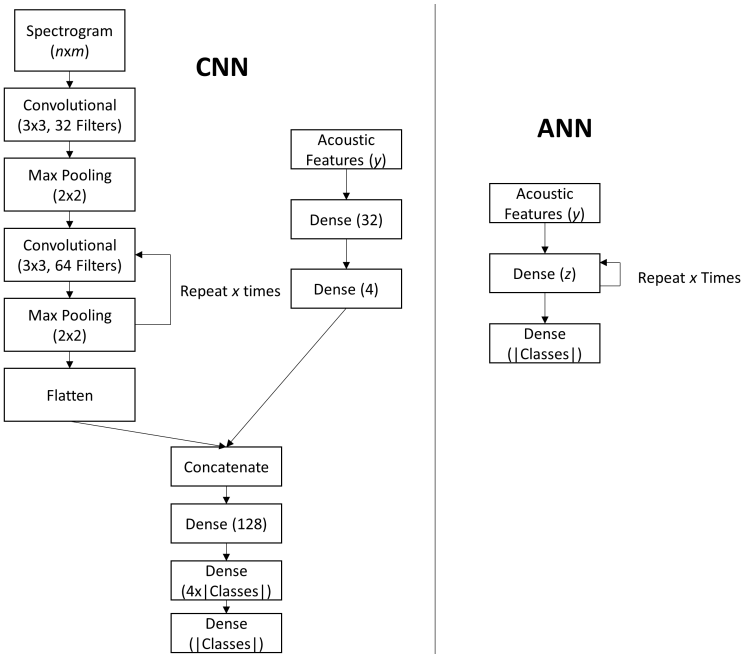


Figure 8: CNN and ANN architectures used in workflow evaluation

For both networks, all dense layers use Rectified Linear Unit (ReLU) activation except the final output layer which uses sigmoid activation. This is used instead of softmax as segments can have sounds from multiple species simultaneously. The output values are set to 1 (indicating the presence of a class) if it is greater than 0.5, or otherwise 0. All convolutional and dense layers are L2-regularised ($\lambda = 0.001$) to help avoid overfitting. In both cases, the loss function is categorical cross-entropy and are fit using the Adam optimizer (learning rate=0.001) [24]. The number of epochs is determined as an input parameter.

6.2.3. Parameters

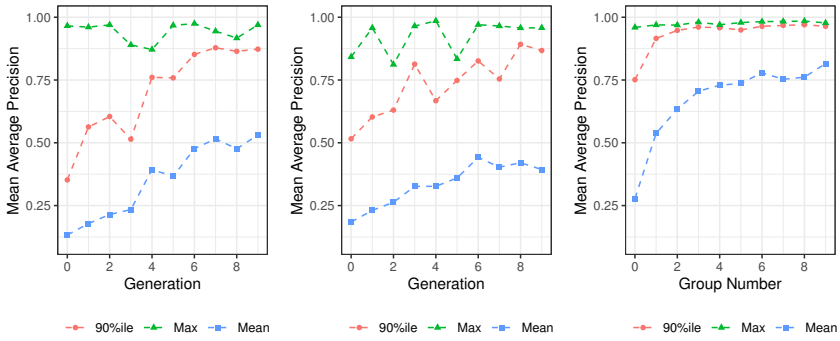
A key component of optimising any classification algorithm is to tune parameters. This idea extends to this problem, except now there are parameters for many other tasks outside of the classifier itself, such as denoising, activity detection, and acoustic feature calculation. This work considers the tasks with different parameter sets to be independent of each other, regardless of whether they are doing the same processing, effectively being considered by the system as unique tasks. Parameter sets are carefully chosen for each task in the workflow system.

7. Results

7.1. Workflow search effectiveness

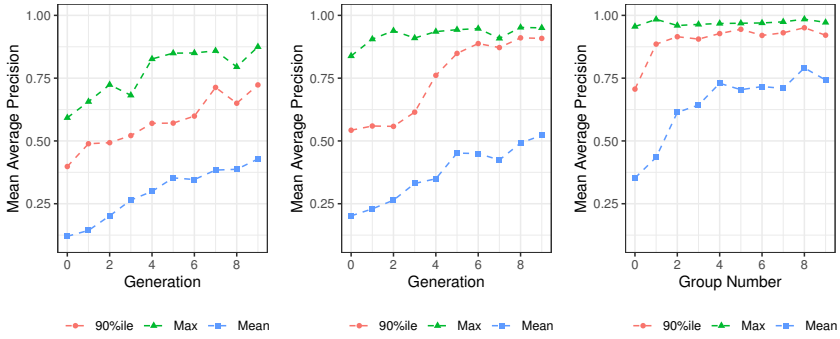
Figures 9–12 show the ability of the workflow search algorithms to find workflows across four different data sets of increasing complexity. This shows that, throughout all tests, while each search algorithm generally improves in effectiveness as they generate more workflows, the surrogate model-based workflow search consistently finds strong workflows quickly and more reliably than the GA and PSO search algorithms, with consistently higher means, 90th percentiles, and best workflows per groups of 50 workflows. The GA and PSO searches occasionally find workflows equally as effective or even slightly exceeding those found by the surrogate model, although they find these less consistently and often need more searches to find good workflows. When comparing the GA to the PSO, it is difficult to definitively say one is better than the other. The GA is a long way behind the PSO on the 4 species audio, but the GA easily beats PSO on the MLSP audio. This might be due to more favourable initial generations. This might indicate the number of workflows per generation is too small to capture the entire search space, but increasing this number is difficult to justify given the number of generations is already very small.

Aside from direct comparisons between search algorithms, while each algorithm finds better workflows on average over time, there is often no clear improvement in the best workflow over time, with the exceptions being the GA



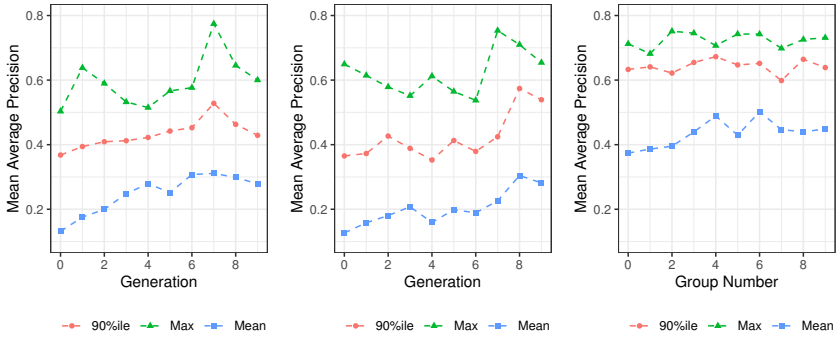
(a) Genetic Algorithm (b) PSO (c) Surrogate Model

Figure 9: Performance of each search algorithm for 2 species audio



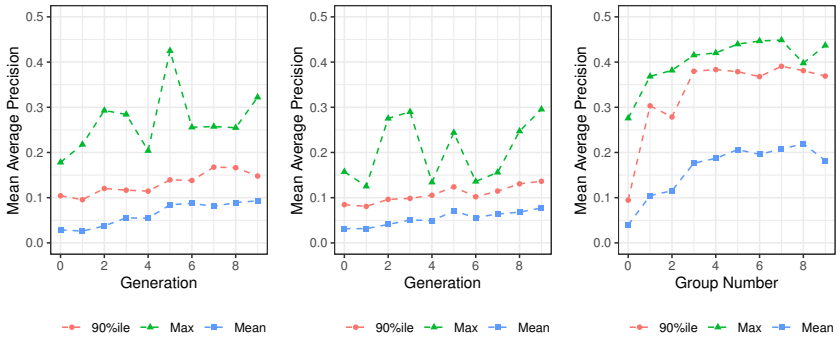
(a) Genetic Algorithm (b) PSO (c) Surrogate Model

Figure 10: Performance of each search algorithm for 4 species audio



(a) Genetic Algorithm (b) PSO (c) Surrogate Model

Figure 11: Performance of each search algorithm for 8 species audio



(a) Genetic Algorithm (b) PSO (c) Surrogate Model

Figure 12: Performance of each search algorithm for MLSP audio

on the 4 species audio (Figure 10a) and the surrogate model on the MLSP audio (Figure 12c), although the surrogate model is often already improving before finishing searching 50 workflows, which is not clearly represented in the figures. This seems to indicate that there are clear patterns that are easy for search algorithms to identify that enable 'good' workflows (e.g. running 10 noise reduction tasks is bad), although the patterns that enable 'great' workflows are more difficult to identify.

Tables 4–5 show how the scores of workflows of a specific rank for each search algorithm for the two most difficult datasets, including random search. This shows that the random search is significantly worse than any of the more selective search algorithms ran on both datasets. Even though it did find one workflow that was much better than any PSO workflows on the MLSP audio, this is a large outlier and, looking at how the GA found a similar outlier (see Figure 12a), it still seems like the PSO could find a similar outlier in a repeated evaluation, and is likely better overall, given other workflows found. A Wilcoxon Signed-Rank test of the top 20 workflows found by the PSO and Random search algorithms gives a p-value of 0.002, with the PSO having the higher median. This table also clearly shows the surrogate model far exceeding the ability of the other search algorithms.

Table 4: Comparison of search algorithms over the 8 species audio. Rank refers to the n^{th} best workflow found by the search

Rank	Random	GA	PSO	Surrogate
1	0.672	0.774	0.753	0.751
2	0.632	0.673	0.710	0.745
5	0.549	0.637	0.614	0.735
10	0.488	0.577	0.590	0.731
20	0.406	0.528	0.543	0.695

Table 5: Comparison of search algorithms over the MLSP audio. Rank refers to the n^{th} best workflow found by the search

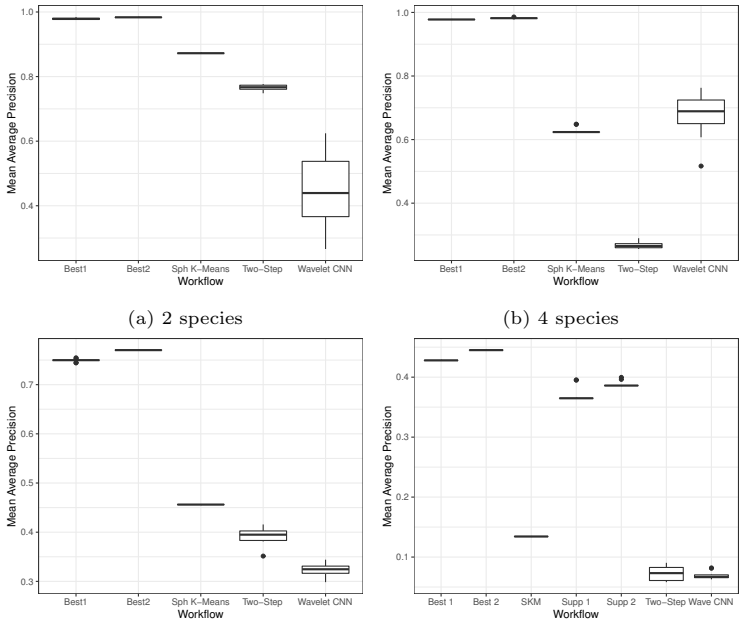
Rank	Random	GA	PSO	Surrogate
1	0.385	0.425	0.295	0.449
2	0.313	0.322	0.290	0.447
5	0.190	0.285	0.244	0.437
10	0.139	0.251	0.182	0.412
20	0.113	0.190	0.135	0.399

7.2. Comparison to baseline workflows

Figure 13 shows that workflows selected by the surrogate model method consistently outperform those selected based on existing literature, with the gap widening as workflows become more complex. The selected workflows also beat the author-provided base method specifically for the MLSP dataset despite choosing to ignore location data, which should have been advantageous.

The poor results of the baseline workflows might be due to suboptimal implementations, with a relatively low number of epochs for the neural networks and a restrictively low amount of training data, among other things. Nonetheless, these are realistic constraints as one might not have the training data to train these neural networks and might not be able to cope with the high computational cost of these methods.

These two baselines, designed specifically for the competition still do not perform as well as the best workflow as selected by the surrogate model. The



(a) 2 species

(b) 4 species

(c) 8 species

(d) MLSP

Figure 13: Workflow comparison to baselines under each dataset

three baseline methods compared throughout all datasets do not perform well with this dataset.

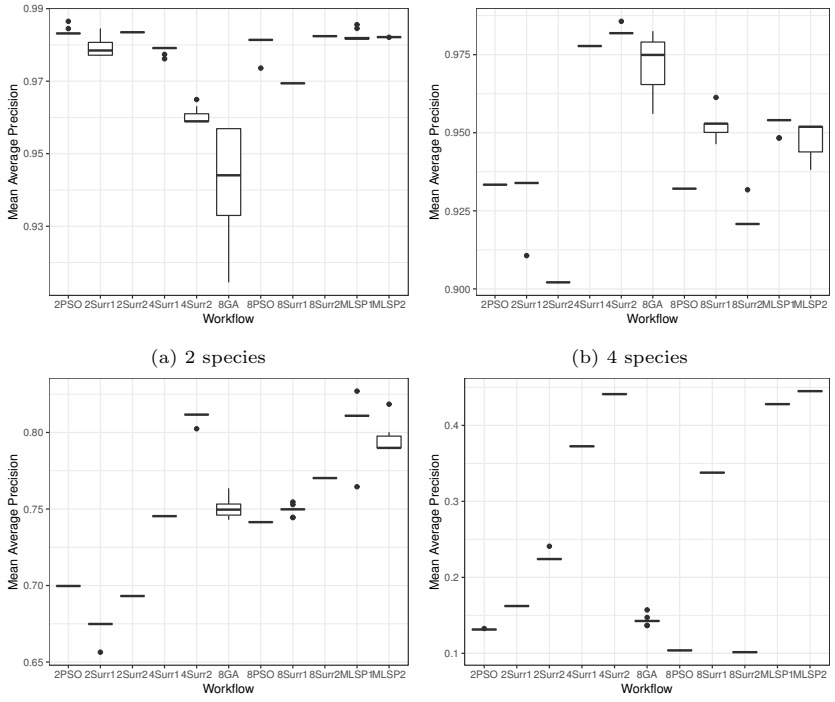
However, these are all significantly behind the competition winner, which scores a MAP of 0.616 (not tested inside the system), but this used a computationally intensive feature learning/template matching approach taking a claimed 10–12 hours to complete in serial which would not be able to meet the tight deadline constraints that the search algorithms were evaluating against (although this could be parallelised and is likely faster on more modern CPUs, given the competition was ran in 2013). All MAP values for this dataset might be under-representing their predictive ability as there are some species with very few labels in the dataset which cannot be predicted with any accuracy likely skew the results downwards (MAP averages precision scores over all species, treating each with equal weighting).

7.3. Workflow Cross-Testing

If workflow performance is scenario-specific, then the workflows selected by the search algorithms for a specific scenario should fit better than a workflow selected for another scenario. With this in mind, the best workflows for each of the audio sources are evaluated against each other. The top two highest scoring workflows selected by the surrogate model and any workflows with higher scores using the other selection methods are compared across different audio sets. Results are shown in Figure 14. Overall, workflow performance appears to vary significantly with the chosen scenario. While, in general, the workflows selected for a specific scenario do perform better than those selected for other scenarios, this is not the case for the 8 species audio, where MLSP workflows, as well as one selected for the 4 species audio, significantly outperform 8 species workflows. The same 4 species workflow also performs well on the MLSP dataset. Workflows selected by the GA and PSO do not appear to be any more or less flexible to scenarios compared to the surrogate model workflows.

The workflow selected by the GA on the 8 species audio (GA8) exceeds deadline constraints for 8 species and the MLSP audio, while two other workflows, the first 4 species surrogate model workflow (4Surr1) and the 2 species PSO workflow (2PSO) also exceed deadline constraints for the MLSP dataset, meaning they would have scored zero during the selection phase. GA8 likely benefited from data reuse from a previous workflow during the 8 species search and hence, 'unfairly' fell under the time constraint, and probably should have scored zero and not been selected.

The most versatile workflows appear to be the two selected for the MLSP dataset and the second workflow selected by the surrogate model for the 4 species audio (4Surr2). In general, workflows selected do not necessarily translate well into other scenarios and it is difficult to predict if they can. Workflows selected for more difficult scenarios might be more useful in simpler scenarios than vice versa, although more testing would need to be done to be confident in that assertion. In general, selecting workflows targeting one scenario specifically is more likely to deliver strong workflows for that scenario.



(a) 2 species

(b) 4 species

(c) 8 species

(d) MLSP

Figure 14: Classification performance the best workflows selected from each scenario

7.4. Selected workflow analysis

Now that it is known which workflow search algorithms are the best and how they compare over different datasets, it is important to consider what workflows these algorithms are actually selecting. This can inform if search algorithms are prematurely converging, but also could inform which tasks are best suited in different scenarios.

The best workflows selected in all scenarios are generally very simple. It is unclear if this is because the best workflows are inherently the most simple, or that these are easier to find for search algorithms. The most common trends are the absence of any noise reduction tasks in most workflows, and the consistent use of random forest classifiers. Among acoustic features, MFCCs are very well represented above all other acoustic features, often being the only features in many top workflows. This indicates, as has been well established in previous research, that they are well suited to bioacoustic classification. Feature pre-processing features in some workflows, but is often also skipped.

More specifically within different searches, the main differences between top workflows selected by the surrogate model comes in the choice of activity detectors. The 2 species audio favours use of the Hilbert Follower, the 4 species audio prefers the spectrogram-based activity detector, the MLSP audio prefers no activity detector (with splitting almost exclusively to 2 seconds, the shortest option allowed), and the 8 species audio is comparatively diverse, changing between the Hilbert Follower and the spectrogram-based approach. The absence of activity detector use on the MLSP dataset is particularly interesting given there is the option to use segments manually annotated and provided by the competition organisers available to the surrogate model algorithm, but is left unused. This might be because it increases computation time too much, as this increases the number of samples to classify (and hence, the number of tasks to perform) significantly.

The GA and PSO tend to select more diverse workflows compared to the surrogate model search, albeit with worse results. Most notably, workflows selected by these selection algorithms often have complex feature pre-processing

chains, with multiple steps of PCA, normalization, ZCA whitening, and spherical k-means clustering, often repeating tasks with different parameter sets. For example, in the 8 species search, the GA averaged 6 feature pre-processing tasks per workflow, and the surrogate model averaged 0.3 feature pre-processing tasks among the top 10 best workflows selected by each model . The GA is also more likely to use denoising tasks (0.9 on average vs. 0.1 for the surrogate model for the same top 10 workflows). It is difficult to conclude if these differences are caused by selection biases by the selection models or because the surrogate model is truly selecting fewer feature pre-processing and denoising tasks because these approaches are inherently better.

7.5. Results Discussion

Overall, from testing, it is clear that:

- There is merit to searching for workflows rather than blindly using workflows from literature. In almost all cases, workflows selected by the search algorithms (even random search, given enough iterations) outperformed those based on existing literature.
- The PSO and GA search algorithms show a significant improvement over random search, but searching with a surrogate model easily outperforms these simple metaheuristic search algorithms.
- For the audio tested, workflows selected for a specific scenario are normally well-suited to it compared to alternatives, although some workflows selected can work well across a variety of scenarios outside of the ones they were selected for.

8. Conclusion and Future Work

This work examined the potential of using search algorithms to find effective bioacoustics workflows suited for specific scenarios. It found that using a surrogate model which estimates workflow scores based on previous results can greatly improve search effectiveness over random and metaheuristic searches for

this problem. It also finds that selecting workflows that are similar to those in existing literature does not necessarily give the best results. However, the effectivenesses of some workflows found do carry over to other scenarios, even sometimes outperforming those specifically selected for the given scenario. In general, it is likely that workflows found through searching more complex scenarios are more versatile than those optimised for simple scenarios. Nonetheless, this method does mostly reliably select workflows that are well suited for their target scenario without any prior knowledge and it is likely general enough to work on different recordings in different locations. Ultimately, searching through workflows might improve the effectiveness of bioacoustics processing for many different scenarios.

In the future, the surrogate model could be investigated more deeply. The choice of model here and the method used to search through the model are not explored and it is likely further improvements can be made in this respect. The model could be made more general, moving beyond species recognition tasks into similar modelling scenarios, optimising for different types of problems with different goal functions and constraints. The pool of bioacoustics tasks could also be improved by identifying and removing poor methods, adding more known good methods, and optimising parameters, particularly as far as neural network classifiers are concerned, as these processes underperformed compared to what was expected based on the literature.

The search system could also be expanded to dynamically build and select workflows for different scenarios without needing to search from scratch. The surrogate model could add scenario parameters as inputs and could use data from previous searches to help optimise for other scenarios without needing to search from scratch.

Acknowledgements

Thanks the recorders whose recordings uploaded to Xeno-Canto were used to generate synthetic bioacoustics soundscapes. These were Fernand Deroussen,

Frank Lambert, Greg McLachlan, Mandar Bhagat, Marc Anderson, Nick Talbot, Oswaldo Cortes, Pradnyavant Mane, Ramit Singal, and Sander Lagerveld.

The lead author is supported by the Australian Government's Research Training Program (RTP).

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.

[2] D. Arellanes and K.-K. Lau, "Workflow variability for autonomic iot systems," in *2019 IEEE International Conference on Autonomic Computing (ICAC)*. IEEE, 2019, pp. 24–30.

[3] C. Audet, J. Denni, D. Moore, A. Booker, and P. Frank, "A surrogate-model-based method for constrained optimization," in *8th symposium on multidisciplinary analysis and optimization*, 2000, p. 4891.

[4] M. Avci and S. Topaloglu, "A multi-start iterated local search algorithm for the generalized quadratic multiple knapsack problem," *Computers & Operations Research*, vol. 83, pp. 54–65, 2017.

[5] R. Bardeli, D. Wolff, F. Kurth, M. Koch, K.-H. Tauchert, and K.-H. Frommolt, "Detecting bird sounds in a complex acoustic environment and application to bioacoustic monitoring," *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1524–1534, 2010.

[6] C. Bedoya, C. Isaza, J. M. Daza, and J. D. López, "Automatic recognition of anuran species based on syllable identification," *Ecological Informatics*, vol. 24, pp. 200–209, 2014.

[7] C. Bierwirth, "A generalized permutation approach to job shop scheduling with genetic algorithms," *Operations-Research-Spektrum*, vol. 17, no. 2-3, pp. 87–92, 1995.

[8] M. K. Bisbo and B. Hammer, "Efficient global structure optimization with a machine-learned surrogate model," *Physical review letters*, vol. 124, no. 8, p. 086102, 2020.

[9] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[10] A. Brown, S. Garg, and J. Montgomery, "AcoustiCloud: A cloud-based system for managing large-scale bioacoustics processing," *Environmental Modelling & Software*, p. 104778, 2020.

[11] I. Dahmani, M. Hifi, T. Saadi, and L. Yousef, "A swarm optimization-based search algorithm for the quadratic knapsack problem with conflict graphs," *Expert Systems with Applications*, vol. 148, p. 113224, 2020.

[12] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE transactions on acoustics, speech, and signal processing*, vol. 28, no. 4, pp. 357–366, 1980.

[13] N. A. Delzoppo, K. Berris, D. Teixeira, and B. van Rensburg, "The impact of fire on the quality of drooping sheoak (allocasuarina verticillata) cones for the endangered kangaroo island glossy black-cockatoo (calyptorhynchus lathami halmaturinus)," *Global Ecology and Conservation*, vol. 28, p. e01645, 2021.

[14] J. Dong, Y. Li, and M. Wang, "Fast multi-objective antenna optimization based on rbf neural network surrogate model optimized by improved pso algorithm," *Applied Sciences*, vol. 9, no. 13, p. 2589, 2019.

[15] Y. Ephraim and D. Malah, "Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 6, pp. 1109–1121, 1984.

[16] A. I. Filkov, T. Ngo, S. Matthews, S. Telfer, and T. D. Penman, "Impact of australia's catastrophic 2019/20 bushfire season on communities and environment. retrospective analysis and current trends," *Journal of Safety Science and Resilience*, vol. 1, no. 1, pp. 44–56, 2020.

[17] T. Ganchev, *Computational Bioacoustics: Biodiversity Monitoring and Assessment*, ser. Speech Technology and Text Mining in Medicine and Healthcare. Walter de Gruyter, 2015. [Online]. Available: https://books.google.com.au/books?id=KmCeswEACAAJ

[18] A. Gasc, J. Sueur, F. Jiguet, V. Devictor, P. Grandcolas, C. Burrow, M. Depraetere, and S. Pavoine, "Assessing biodiversity with sound: Do acoustic diversity indices reflect phylogenetic and functional diversities of bird communities?" *Ecological Indicators*, vol. 25, pp. 279–287, 2013.

[19] H. Glotin, J. Ricard, and R. Balestriero, "Fast chirplet transform injects priors in deep learning of animal calls and speech," 2017.

[20] H. Hermansky, "Perceptual linear predictive (plp) analysis of speech," *the Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[21] J. H. Holland *et al.*, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[22] T. Huang and A. S. Mohan, "A hybrid boundary condition for robust particle swarm optimization," *IEEE antennas and wireless propagation letters*, vol. 4, pp. 112–117, 2005.

[23] J. Kennedy, "Particle swarm optimization," *Encyclopedia of machine learning*, pp. 760–766, 2010.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[25] I. Kiskin, D. Zilli, Y. Li, M. Sinka, K. Willis, and S. Roberts, "Bioacoustic detection with wavelet-conditioned convolutional neural networks," *Neural Computing and Applications*, vol. 32, no. 4, pp. 915–927, 2020.

[26] A. Larcher, K. A. Lee, and S. Meignier, "An extensible speaker identification sidekit in python," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5095–5099.

[27] M. Lasseck, "Bird song classification in field recordings: winning solution for nips4b 2013 competition," in *Proc. of int. symp. Neural Information Scaled for Bioacoustics, sabiod. org/nips4b, joint to NIPS, Nevada*, 2013, pp. 176–181.

[28] J. LeBien, M. Zhong, M. Campos-Cerqueira, J. P. Velev, R. Dodhia, J. L. Ferres, and T. M. Aide, "A pipeline for identification of bird and frog species in tropical soundscape recordings using a convolutional neural network," *Ecological Informatics*, vol. 59, p. 101113, 2020.

[29] G. R. Lee, R. Gommers, F. Waselewski, K. Wohlfahrt, and A. O'Leary, "Pywavelets: A python package for wavelet analysis," *Journal of Open Source Software*, vol. 4, no. 36, p. 1237, 2019.

[30] B. Liu, H. Yang, and M. J. Lancaster, "Global optimization of microwave filters based on a surrogate model-assisted evolutionary algorithm," *IEEE Transactions on Microwave Theory and Techniques*, vol. 65, no. 6, pp. 1976–1985, 2017.

[31] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, vol. 8, 2015, pp. 18–25.

[32] D. Meng, S. Yang, Y. Zhang, and S.-P. Zhu, "Structural reliability analysis and uncertainties-based collaborative design and optimization of turbine

blades using surrogate model," *Fatigue & Fracture of Engineering Materials & Structures*, vol. 42, no. 6, pp. 1219–1227, 2019.

[33] H. B. Nguyen, B. Xue, and P. Andreae, "Pso with surrogate models for feature selection: static and dynamic clustering-based methods," *Memetic Computing*, vol. 10, no. 3, pp. 291–300, 2018.

[34] T. E. Oliphant, *A guide to NumPy*. Trelgol Publishing USA, 2006, vol. 1.

[35] D. O'Shaughnessy, "Linear predictive coding," *IEEE potentials*, vol. 7, no. 1, pp. 29–32, 1988.

[36] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[37] N. Pieretti, A. Farina, and D. Morri, "A new methodology to infer the singing activity of an avian community: the acoustic complexity index (aci)," *Ecological Indicators*, vol. 11, no. 3, pp. 868–873, 2011.

[38] I. Potamitis, S. Ntalampiras, O. Jahn, and K. Riede, "Automatic bird sound detection in long real-field recordings: Applications and tools," *Applied Acoustics*, vol. 80, pp. 1–9, 2014.

[39] J. Qian, J. Yi, Y. Cheng, J. Liu, and Q. Zhou, "A sequential constraints updating approach for kriging surrogate model-assisted engineering optimization design problem," *Engineering with Computers*, pp. 1–17, 2019.

[40] A. Rezaeian, M. Naghibzadeh, and D. H. Epema, "Fair multiple-workflow scheduling with different quality-of-service goals," *The Journal of Supercomputing*, vol. 75, no. 2, pp. 746–769, 2019.

[41] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *2017 IEEE*

Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA). IEEE, 2017, pp. 344–348.

[42] J. Song, Y. Yang, J. Wu, J. Wu, X. Sun, and J. Lin, "Adaptive surrogate model based multiobjective optimization for coastal aquifer management," *Journal of hydrology*, vol. 561, pp. 98–111, 2018.

[43] D. Stowell and M. D. Plumbley, "Automatic large-scale classification of bird sounds is strongly improved by unsupervised feature learning," *PeerJ*, vol. 2, p. e488, 2014.

[44] J. Sueur, S. Pavoine, O. Hamerlynck, and S. Duvail, "Rapid acoustic survey for biodiversity appraisal," *PloS one*, vol. 3, no. 12, p. e4065, 2008.

[45] D. Tian and Z. Shi, "Mpso: Modified particle swarm optimization and its applications," *Swarm and evolutionary computation*, vol. 41, pp. 49–68, 2018.

[46] M. Towsey, J. Wimmer, I. Williamson, and P. Roe, "The use of acoustic indices to determine avian species richness in audio-recordings of the environment," *Ecological Informatics*, vol. 21, pp. 110–119, 2014.

[47] M. Towsey, L. Zhang, M. Cottman-Fields, J. Wimmer, J. Zhang, and P. Roe, "Visualization of long-duration acoustic recordings of the environment," *Procedia Computer Science*, vol. 29, pp. 703–712, 2014.

[48] S. Van Der Walt, S. C. Colbert, and G. Varoquaux, "The numpy array: a structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, no. 2, p. 22, 2011.

[49] S. Van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu, "scikit-image: image processing in python," *PeerJ*, vol. 2, p. e453, 2014.

[50] E. Vidaña-Vila, J. Navarro, R. M. Alsina-Pagès, and Álvaro Ramírez, "A two-stage approach to automatically de-

tect and classify woodpecker (fam. picidae) sounds," *Applied Acoustics*, vol. 166, p. 107312, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0003682X19305080

[51] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright *et al.*, "Scipy 1.0: fundamental algorithms for scientific computing in python," *Nature methods*, vol. 17, no. 3, pp. 261–272, 2020.

[52] J. Wimmer, M. Towsey, P. Roe, and I. Williamson, "Sampling environmental acoustic recordings to determine bird species richness," *Ecological Applications*, vol. 23, no. 6, pp. 1419–1428, 2013. [Online]. Available: https://esajournals.onlinelibrary.wiley.com/doi/abs/10.1890/12-2088.1

[53] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[54] J. Xie, M. Towsey, A. Truskinger, P. Eichinski, J. Zhang, and P. Roe, "Acoustic classification of australian anurans using syllable features," in *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2015 IEEE Tenth International Conference on*. IEEE, 2015, pp. 1–6.