Applied Intelligence
https://doi.org/10.1007/s10489-023-04486-8

Identifying bird species by their calls in Soundscapes

Kyle Maclean[3] · **Isaac Triguero**[1,2,3]

Accepted: 23 January 2023

Abstract
In many real data science problems, it is common to encounter a domain mismatch between the training and testing datasets, which means that solutions designed for one may not transfer well to the other due to their differences. An example of such was in the BirdCLEF2021 Kaggle competition, where participants had to identify all bird species that could be heard in audio recordings. Thus, multi-label classifiers, capable of coping with domain mismatch, were required. In addition, classifiers needed to be resilient to a long-tailed (imbalanced) class distribution and weak labels. Throughout the competition, a diverse range of solutions based on convolutional neural networks were proposed. However, it is unclear how different solution components contribute to overall performance. In this work, we contextualise the problem with respect to the previously existing literature, analysing and discussing the choices made by the different participants. We also propose a modular solution architecture to empirically quantify the effects of different architectures. The results of this study provide insights into which components worked well for this challenge.

Keywords Multi-label classification · Signal processing · Domain mismatch · Convolutional neural networks

1 Introduction

Ecologists monitor bird influences in ecosystems. Historically, people have manually conducted bird monitoring processes after learning to recognise birds by their vocalisations (calls) since visibility is often low in dense forests. The use of machine learning (ML) methods to automate this kind of process can save on human resources, reduce bias in measurement, reach massive scale and target previously inaccessible areas, such as reed ecosystems [31, 37].

The recent BirdCLEF2021 Kaggle competition[1] provided a great platform to deal with bird species identification in audio data. In particular, the goal was to correctly list all bird species that can be heard calling in a five-second segment of audio. Thus, this falls into the category of multi-label classification [20], with many quirks that make it difficult to apply existing ML methods effectively, providing in that way opportunities to create novel designs.

The data collected for this competition is comprised of *soundscapes* and *focals*. *Soundscapes* contains up-to-10-minute long audio files in which different bird calls and background noises can be heard. However, *focals* contain seconds-to-minutes long audio files in which one species can be heard calling primarily, with minimal background noise and potentially a list of secondary species that can also be heard, but less loudly/often. Thus, the calls are often easier to distinguish in *focals* compared to *soundscapes*, as illustrated by their audio waveform patterns in Fig. 1.

To mimic the exact problem faced by scientists that try to automate the monitoring of bird populations from audio recordings, *focals* are provided as training data, and *soundscapes* are used for test/validation. This becomes a key challenge of this competition, as there is a *domain mismatch* [27] between the training and validation/testing datasets, which means that solutions developed for one may

✉ Isaac Triguero
triguero@decsai.ugr.es

Kyle Maclean
kyle@maclean.co.za

[1] Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain

[2] DaSCI Andalusian Institute in Data Science and Computational Intelligence, Granada, Spain

[3] School of Computer Science, University of Nottingham, Nottingham, UK

[1] https://www.kaggle.com/c/birdclef-2021/overview

Published online: 20 March 2023

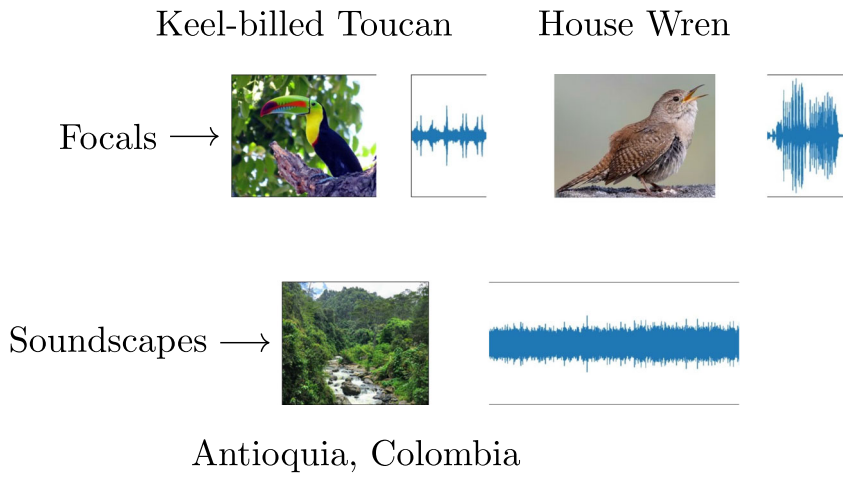


Fig. 1 Illustrating the mismatch between the *focals* training data and the *soundscapes* validation/testing data. Image credit: `ebird.org`, `seecolombia.travel`

not transfer well to the other due to differences in their characteristics. We had access to 62,875 training instances, but only twenty validation instances. This made it difficult to use the validation dataset during training, even though it is in the same domain as the testing dataset. Another big issue for ML techniques is related to the *class imbalance* [10, 30]. The class distribution is referred as long-tailed, meaning that head/majority classes have many samples (e.g. common bird species) while tail classes have very few samples [35]. In addition, there is inconsistent label distribution among audio segments. Last but not least, the labelling of the data is considered weak, as this labelling comes from a Citizen Science project. Those are likely to be inexact and inaccurate (e.g. a sample may not have all bird calls audible in the recording), resulting in weak labels that highly complicate the learning process [29, 38].

A variety of solutions were proposed within this competition. Although the data forms a time series because microphones sample audio sequentially through time, most of the solutions submitted split the training data into five-second segments and classify them independently of each other. This is useful because it matches the format required in the validation/testing phase, and because a lot of ML algorithms are designed for fixed-size input vectors. A further popular processing step is to convert each segment from an audio file into an image file. It is achieved by applying a Short-Time Fourier transformation (STFT) [28] and the result is called a *spectrogram*, which visualises the intensity of different audio frequencies in a two-dimensional colour graph. Overall, this strategy allows discrete classification techniques from the field of Computer Vision field to be applied to time series audio data and demonstrates state-of-the-art performance [22]. A simplified illustration is shown in Fig. 2.

Competitors implemented this strategy in various ways, but it is unclear how different solution components contribute to overall performance. Some teams created complex architectures, while others achieved decent performance with simple ones. In this work, we quantify the effects of different architectures through empirical testing. It is important to note that we do not aim to present a new solution that performs significantly better than existing solutions, but make further study of the techniques used in this domain easier. The key contributions of this work are as follows:

- We empirically investigate the importance of the different solution components, such as external datasets, class distribution statistics, machine learning models, etc., in the existing solutions.
- We propose a modular solution which makes it easy to evaluate the performance contributions of individual components, as well as a simple-yet-effective baseline solution.
- It was challenging to compare the solutions which were publicly posted for this competition due to inconsistent programming and documentation among competitors. This paper may therefore be advantageous to readers

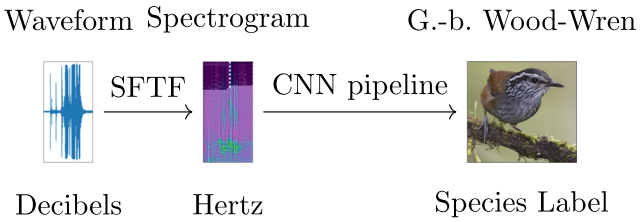


Fig. 2 A common technique to process bird call audio for automatic recognition. The *CNN pipeline* includes data augmentation and model ensembling. Note that the bird image is a representative individual of the species for illustrative purposes (predictions are made at the species-level, not the individual level). Image credit: `birdphotos.com`

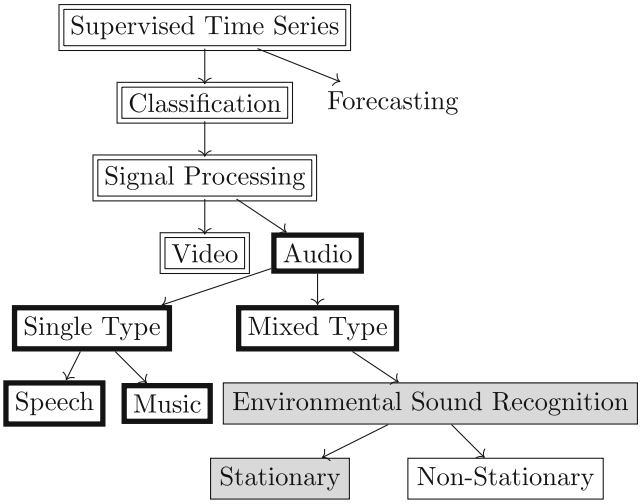


Fig. 3 Categories of Data Science techniques in relation to target data types and tasks. Each rectangle type is associated with a specific section: double border with Section 2.1, bold border with Section 2.2, grey fill with Section 2.3, and single thin border with Section 2.4

who want a broader review and comparison of high-performing techniques for this task.

The source code of the proposed solution, as well as all the results obtained in this paper, are available on GitHub[2].

This paper is organised as follows. Section 2 introduces the related work and discusses approaches followed by the participants of the competition. Section 3 analyses the data provided in this competition. Section 4 presents the proposed solution and how we evaluate its performance. Section 5 discusses the results obtained. Finally, Section 6 draws the main conclusions and insights learned from this work.

2 Related work

This section contextualises our problem domain within the larger taxonomy of Data Science. The subsections traverse down the hierarchy shown in Fig. 3, leading from broader to narrower levels of relevancy. Section 2.1 starts generally with Time Series Data and Signal Processing. Section 2.2 drills into audio signals specifically. Section 2.3 deals with a particular category of mixed type audio called Environmental Sound Recognition. Section 2.4 covers the most immediately relevant category of bird call recognition in *soundscapes*. Finally, Section 2.5 explains some of the techniques used to address the peculiarities and conditions which the competitors of BirdCLEF2021 faced.

[2]https://github.com/KyleMaclean/Bird-Calls-Soundscapes

2.1 Time series data and signal processing

Time Series Forecasting is very popular, but recently, Time Series Classification (TSC) has increased in popularity significantly [15, 18] in various real-world applications ranging from human activity recognition [14] to medical diagnosis [19]. In this area, the use of ensemble learning has been very common due to their high performance [1].

However, researchers moved away from large ensembles for efficiency reasons and away from hand-crafted feature extractors for performance reasons. The success of deep CNNs in computer vision [12] inspired their utilisation in other time series signal processing domains such as natural language processing and speech recognition [24]. Bird call recognition is similar in principle because previous sound samples greatly influence the meaning of subsequent samples.

CNNs do well for signal-processing tasks, because the convolutional layers can be specialised to compress information into what is relevant for discriminating particular signals. Without the convolutions reducing the number of parameters, it would usually be computationally infeasible to learn weights for the amount of parameters that an image (whether a photograph, diagram, spectrogram, etc.) would require [26]. The convolutions are also capable of picking out features independently of where they are in the space of the image [23]. The earlier layers in a CNN learn more "primitive" image features, such as edges, and the later layers learn more sophisticated shapes. We expect diminishing returns after fifty layers for bird call recognition because the details recognisable with that many layers are not particularly relevant.

2.2 Understanding audio with CNNs

The two major techniques for analysing audio are with amplitude over time or amplitude over frequency. Combining those three units into one measure yields the spectrogram. These measures grant an understanding of audio at three different levels [7]:

1. Using the average frequency of a segment to understand *low-level acoustics*.
2. By the short audio signature of an event, like a glass shattering, to understand *mid-level sound objects*.
3. Analysing longer segments of audio to look for patterns, like a crowd of people talking, of *high-level scene classes*.

The time, amplitude and frequency measurements can be used to develop audio features to describe the two major types of audio, *single type*, which is linearly separable in its feature space, and *mixed type*, which is not. It is difficult to use some audio features, like zero crossing rate

(ZCR), to differentiate mixed type audio due to its non-linearity [6]. ZCR is usually only applied to speech/music recognition/understanding because it measures the rate that the signal crosses negative to positive magnitudes [11]. Some of the "cleanest" *focal* recordings in our dataset can be considered examples of this type, because different bird species can exhibit different patterns in the ZCR statistic. However, during preliminary experiments, we found no distinguishable pattern when sampling different audio clips of the *Keel-billed Toucan* and classifying them with the ZCR statistic. Therefore, our data can generally be considered *mixed type audio*.

Convolutional recurrent neural networks (CRNNs) have shown high performance for Sound Event Detection (SED) [4], but there is a difference between usual SED tasks and ours. A usual SED example is: given a twenty second audio clip, determine that *footsteps* were heard between seconds four and fifteen and *thunder* was heard between seconds twelve and seventeen. It is difficult to use these techniques for our bird call recognition task because we have 397 classes of "sound events" (bird species), whereas these models are often only shown to perform well with less than ten. Also, many of these bird calls sound similar, whereas the common SED classes are often quite different, as in the example. Most mixed type audio fits into the category of environmental sound recognition (ESR).

2.3 Environmental sound recognition

Broadly, *environmental sounds* are artificial or quotidian "background sounds". A survey [5] described solutions to classify them as *frame-based* (where a frame is the same as what we have referred to as a segment), in which one classification is made per frame, or *sub-frame-based*, where classifications of frames-of-frames (with or without overlap) are combined by some means, or *sequential*, in which very small segments are classified. Environmental sounds can be *stationary*, where the "phenomena responsible for signal production do not vary with time". Researchers have generally found spectral features to work very well to classify these sounds, although at the time the survey was published (2014), matching pursuit-based techniques, which use sparse signal representations with overcomplete dictionaries, were better. The justification given was that spectra are cursed by high dimensionality. However, it did not account for later advancements in deep learning (starting in 2015 [13]), which allowed efficient utilisation of high-dimensionality data through convolutional layer compression. These advancements led to an interest in developing the *bird call recognition* subcategory of ESR, due to the new potential for high performance.

2.4 Bird call recognition in soundscapes

As an alternative to CNNs, vision transformers (ViTs), which have been largely used for natural language processing have recently been applied to image recognition [8]. In ViT, a *vanilla*/*pure* transformer yields classifications after direct application to a spectrogram. ViT are better than CNNs because they use a lot less computational resources during inference and are more appropriate in principle, since CNNs are designed to learn patterns independently of their orientation (it is nonsensical to translate the axes of spectra because one measures time and the other frequency). In fact, one team in BirdCLEF2021 [17] claimed that ViT are able to be somewhat competitive with CNNs specifically because the former does not learn pitch-invariant patterns. However, the top ten solutions for BirdCLEF2021 all used CNNs.

It is difficult for either method to consider the temporal nature of our data. Actually, these techniques ignore it in the sense that segments of audio are classified in isolation of each other. Hearing a bird call in one segment will probably increase the chances that the same bird call will be heard in the subsequent segment, but without other stages in the pipeline to account for this, the information is lost. As mentioned in Section 2.2, CRNNs have been used to utilise this temporal information.

In *soundscapes*, different birds can often be heard calling simultaneously. To teach a classifier to recognise multiple classes, the Mix Up technique takes examples from different classes and blends them together to create new examples with labels from their constituents. Synthetic (e.g., *pink* or *brown*) noise can also be introduced. This technique was used widely by competitors, as well as other techniques which were specialised to the specific datasets of the competition. We discuss these in the following section.

2.5 Problem-specific challenges

2.5.1 Class imbalance

The *nocall* class was highly imbalanced compared to all the other classes (bird species), occurring in 63.7% of the five-second training segments. To decrease this imbalance, teams used external data as extra *nocall* examples. (See `lib/external-datasets.txt` in the GitHub repository).

A primary focus of our solution was the systematic development of the various models used in our ensemble of classifiers. We emphasised this process because ensemble classifiers have been shown to be more effective than data sampling techniques to enhance the classification performance of imbalanced data [9].

2.5.2 Labelling issues

Training is difficult because *focals* may not necessarily contain calls when divided into segments to generate spectra. They may even contain calls from other birds. When selecting the size of a spectrogram segment, or how much a *focal* ought to be trimmed, the trade-off is between getting a clip which is long enough to make it more likely that a call occurs in it and that is short enough to not have too much background noise distracting the model from the call. To make the prediction, a straightforward approach is to globally pool all a *focal*'s segments into a single logit to be transformed by a sigmoid. The pooling operation would ideally be a compromise between average (which would cause weak predictions to be made uniformly over all the segments) and max (which would be like saying that a bird only calls in a single five-second segment). The teams dealt with segments potentially not having calls in various ways, such as: manually selecting segments where the target bird was present, using stochastic sub-sampling, or only using the results of training as "pseudo-labels" to be processed during later stages into "cleaned labels". The last is called "self-distillation" or "multistep train-segment-shift training" [21]. It regularises predictive uncertainty, similar to label smoothing, thereby increasing predictive diversity without requiring another model to act as a teacher [36].

2.5.3 Domain mismatch

An example of the domain mismatch is that *focals* are recorded with directional microphones, intending to pick up a particular bird, and therefore pick up less noise than the less directed microphones used to record *soundscapes*. Mix Up techniques described in the previous section can help to address this. Another example is that birds are generally further away in the *soundscape* recordings, and high frequencies have lower volume than low frequencies as distance to the microphone is increased. A team from the 2020 competition proposed to reduce the volume of high frequencies to address this[3]. Overall, the authors' intentions for both types of audio are different: the training data is intended to provide the cleanest examples of particular species' calls, whereas the validation/testing data is not specific to particular birds.

2.5.4 Advanced classifier co-operation

Many teams found it beneficial to develop different classifiers for different stages of their workflow. To reduce false positives and false negatives, some used a component to distinguish between segments with calls and those without. To ensure that long- and short-term information is properly utilised, one team [16] determined the list of species present in an entire *focal*, and then tried to detect which species from that list were present in each of the spectrogram-length segments which constituted that *focal*. A common technique for ensembling was to average the logits from the networks and then threshold them. One team commented explicitly that this performed better than the other way around. We follow this method in our performant solution.

2.5.5 Location metadata

There were multiple ways that teams utilised the location metadata provided with the *focals*, such as:

- Disregarding it.
- Weighting predictions which were associated with sites that were geographically closer to the testing site proportionally higher than those that were far away [16].
- Including it as feature(s) in ML systems [25].

2.5.6 Data augmentation

To help their models generalise, teams applied various augmentations to the spectra, such as: constructing the mel filterbank on the fly where the minimum and maximum frequencies were scaled randomly (to adjust the pitch of the sound) [16], and, as described above, dampening higher frequencies and mixing up the training examples with each other and with noise. Surveying some teams' experiments, it seems that the augmentations generally did not improve performance individually, but did when used in different amounts across ensembled networks, perhaps because the networks themselves became more diverse and therefore more robust in aggregate.

2.5.7 Post-processing

There were some interesting post-processing steps that the teams performed, such as:

- Increasing prediction probability according to a particular species' prevalence in the training data
- Distilling labels using smoothing, where the labelling errors which the humans made can be seen as a kind of beneficial regularisation
- Weighting predictions according to location/time-of-year metadata

2.5.8 Summary

The main challenges faced during the competition were:

[3] https://www.kaggle.com/c/birdsong-recognition/discussion/183269+

- Incorporating temporal context. For example: if a specific bird is heard in an audio recording, then it is more likely that the same bird will be heard later in that recording too.
- High-performing SED techniques are not suited to this problem. They are only suited to datasets which have fewer classes and where the classes are much more distinctive, compared to ours.
- Label distribution among audio segments. It is more likely that a bird is not calling in the training data: when the recordings are split into five-second segments, most of those segments do not contain a bird call. This is especially detrimental when labels are distributed to all the segments.
- The labelling is weak. For example: other bird calls which are not labelled may be audible in the recording, or the bird call which the author knew was there may be very difficult to hear in the recording.

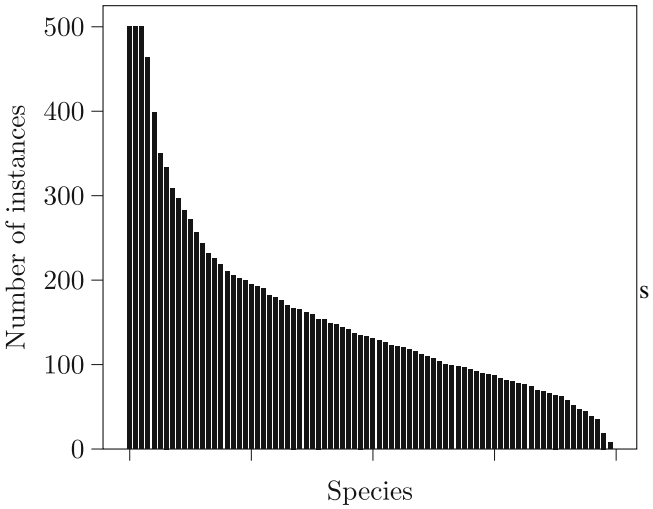


Fig. 4 Number of *focals* per species. Due to space constraints, x-axis labels are omitted and only every fifth bar is plotted

3 Data analysis

In this section, we comment on some statistical properties of the datasets. There are 62,874 *focals* with metadata, such as location and time of recording, available for training. The labels are provided through "Citizen Science" on Xeno Canto[4], where amateurs and professionals contribute through a system of tagging, discussing, flagging and reviewing. They are only at the audio file level, without timestamps of when exactly the call can be heard. There are 397 unique classes to classify, and 264 of them have at least 100 instances. Datasets like this are known as "long-tailed" because there is a class imbalance which can be visualised as shown in Fig. 4. It is generally harder to achieve high performance with such datasets [32]. The competition organisers capped the number of *focals* per species to 500 because they thought that having more would not reveal much more discriminatory information.

There are twenty *soundscapes* available for training/validation, each with a list of bird species that can be heard (or *nocall* if there are none) in every five-second segment. The labels are quite strong since they are provided by The Cornell Lab of Ornithology. The hidden testing data consists of approximately eighty *soundscapes*. There are 397 species but this is a 398-class problem due to the special *nocall* class which is only predicted when all other classes are not predicted. Since there could be many birds calling simultaneously, this is a multi-label problem. The *soundscape* data was recorded only at four separate sites: Colombia (COL), Costa Rica (COR), Sierra Nevada (SNE) and Sapsucker Woods (SSW). The hidden test set contains instances from all four, but the twenty *soundscapes* that were provided for training/validation only contained examples from COR and SSW.

It is not straightforward to know how to use the secondary labels provided in the *focals* during training. The calls of the primary species are dominant because the authors' intentions were to capture them. By definition, the secondary species are difficult to hear in the recordings, so trying to train a model to recognise such a weak signal (so weak that it might be almost indistinguishable from noise) may lower its performance. However, because the *soundscape* data is not recorded to emphasise particular bird calls, models might be helped in transferring to this domain through training on weak signals. We counted the number of times each label appears in the *focal* secondary label lists and found that since there are 62,874 *focals*, there are twelve secondary labels that appear in at least 1% of them. See $diagrams/tangential-analysis.ipynb$ in the source code for more analysis like this.

This analysis provides the following key insights:

- The labels are sometimes provided by amateurs and always at the audio file level, i.e., without granular labelling at different timestamps within the audio file.
- The training data is long-tailed, so we do not have a lot of instances for every class.
- There is a domain mismatch between the training and validation/testing data and it is difficult to train a model which works well in both domains.
- The miscellaneous secondary labels supplied with the training data could be more harmful than useful for training certain predictors.

[4] https://xeno-canto.org/+

4 Solution architecture

This section explains how a baseline solution might be designed and how we constructed our modular, high-performing solution, which can be split into six logical phases.

The baseline solution converts *focals* into spectra to train a CNN, which is used directly to make predictions on *soundscapes*. The network is:

- *convolutional* to reduce the feature vector size of the spectra.
- *residual* to help it converge faster and not suffer from vanishing/exploding gradients.
- *deep* because many layers are required to extract the fine detail of the spectrogram pattern.

4.1 Phases of the performant solution

The performant solution incorporates more datasets and techniques to achieve high performance on the Kaggle Public Leaderboard. The proposed process can be described in six phases and is drawn simply in Fig. 5 and in more detail in Fig. 6:

1. **Convert all the audio we will use into spectra images**: Since our approach benefits from the pattern-recognition machinery which has been advanced in the field of Computer Vision, we convert all the audio data that we use in our solution into the spectrogram image representation. We can then treat our problem as an image classification task and utilise effective CNN workflows. The three main audio repositories to convert are:
   (a) The *freefield1010* data: used in Phase 2 to build the *nocall* discriminator, which will make predictions on the *focals* in Phase 3 to re-weight their probabilities in Phase 4
   (b) The *focals* data: used in Phase 4 to help the multi-label classifier learn the calls of the different species
   (c) The validation data from BirdCLEF2020 which is only tagged with *nocall*: used to enhance the training of the multi-label classifier in Phase 4 by providing more examples of what environments without bird calls sound like

   This process involves splitting each audio file into its constituent segments of equal length and applying the Fourier transformation to those, yielding a spectrogram image from each. This overall process is known as the STFT, as described in Section 1.
2. **Discriminate the presence of calls**: Once all the data is in image format, we can train our first model. It is a binary classifier which just determines whether a bird call can be "seen" in a given spectrogram. Having this expertise helps to augment the subsequent model, which is a multi-label classifier, by increasing our confidence in its species labels. The *freefield1010* dataset is separated into those instances that are tagged with "birds" (where birds can be heard in the audio) from those instances that are not, to create a binary classification dataset. The *nocall* detector that we train with this new dataset is a CNN that learns when a visual pattern in the spectrogram is likely to represent a bird call's frequency intensities or not.
3. **Predict whether *focals* have calls**: the *nocall* discriminator created in Phase 2 is used to produce likelihoods for whether each *focal* segment has a bird call in it. These probabilities will influence the predictions made in Phase 4 to improve their accuracy. Thus we have used some of our data (*freefield1010* divided into instances that had birds calling and those that did not) to train the *nocall* detector, and another part of our data (*focals*), with which to perform inference using that model. In other words, we train an expert binary classifier to label our training data. We will use that list of labels as a new feature in the subsequent two phases. Therefore, this training is a kind of "pre-work" before we train the models which will be used on testing data.
4. **Train the multi-label classifier using a combination of datasources**: In our case, this classifier is actually an ensemble of classifiers, but the modularity of our solution supports any artifact which is capable of outputting a list of species labels. We use the following datasources to train it: the spectra of the *focals* (Phase 1), the spectra of the *nocall* examples from the

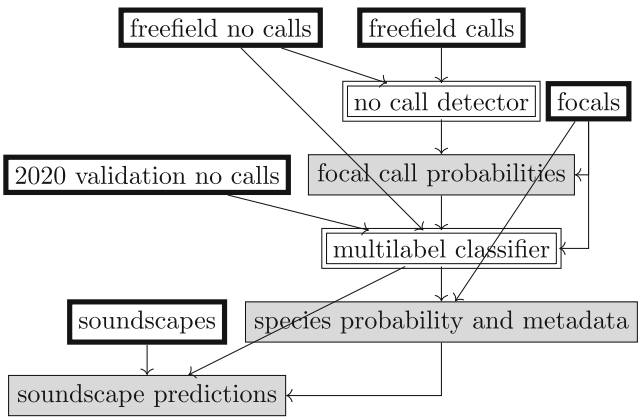


Fig. 5 The proposed performant solution architecture at a high level. Datasets are inputs in this procedure and are shown with bold borders, while the processing machinery (models) have double borders and the output artefacts (tables) are filled with grey. This formatting shows how some inputs are used at multiple stages and how the outputs from one procedure may be used as inputs in another

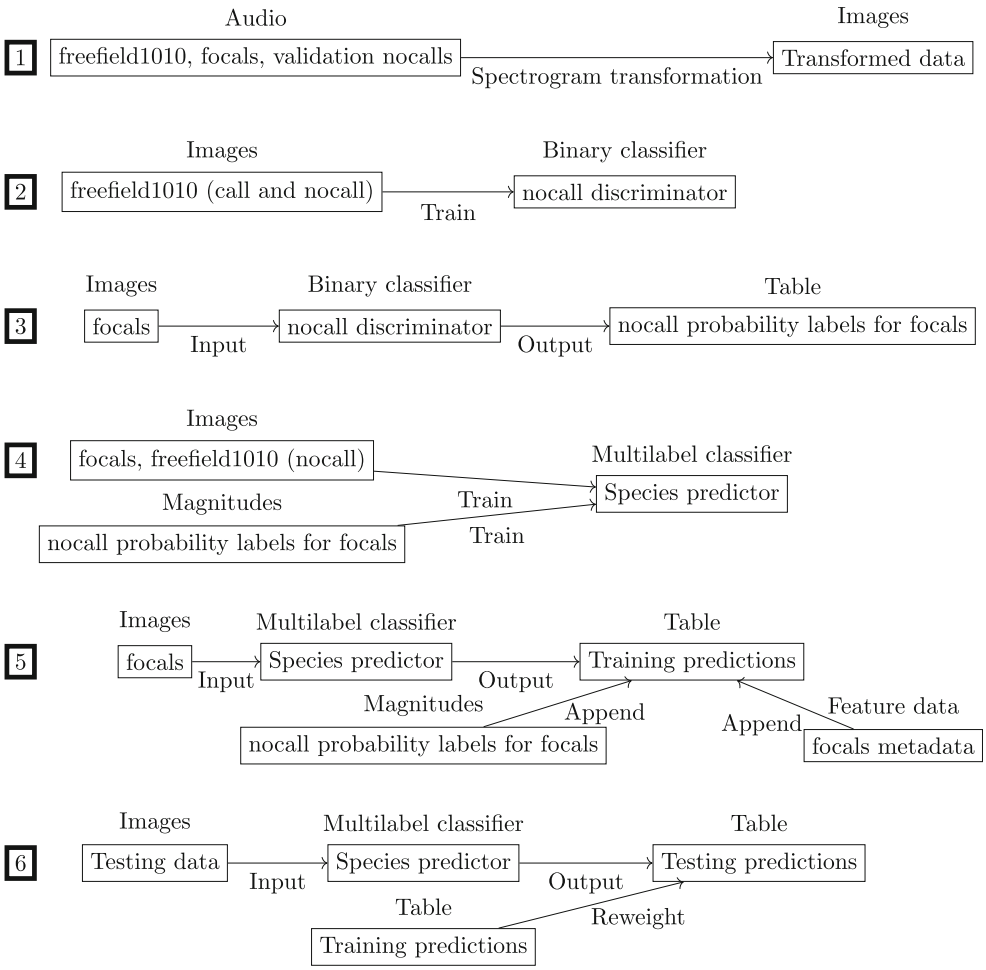


Fig. 6 The proposed performant solution architecture in detail. The numbered phases on the left of the diagram correspond with the phases described in Section 4.1

freefield1010 and BirdCLEF2020 validation datasets, and the list of probabilities for whether each *focal* segment contains *nocall* (Phase 3). This combination mitigates class imbalance by giving the "super class" of all 397 bird classes more *nocall* counter-examples. The labels of the training data which are fed into the multi-label classifier are assigned a weight equal to the *nocall* probabilities, so that, the multi-label classifier treats the labels of segments which have a lower weight "less seriously". This paradigm allows us to fit the secondary labels into our training process by assigning them a low confidence score by default.

5. **Predict each species' probability**: the multi-label classifier model from Phase 4 predicts which species are present in segments of *focals*. We output these results in tables for each model configuration, where each row in those tables represents a single segment from a *focal*. Each will have 397 columns to represent the probability of that species being audible in the segment. The row will be appended with the metadata of the *focal* from which the segment was extracted and the call probability that was assigned by the *nocall* discriminator which was created in Phase 2. This can be thought of as a version of the workflow described in Phase 4 which has been modified to perform testing and post-processing on the results. It is another sub-process which involves inference that is part of the overall training process.
6. **Make predictions on testing data**: the probabilities for each species and the metadata from Phase 5, together with the classifier models from Phase 4, are used to make predictions on the given testing data. The probabilities incorporate information about the popularity of different birds for re-weighting predictions. For example, two candidates may sound similar but one might be a lot more rare to encounter in general, so it should be a less likely prediction. The metadata contains latitudinal and longitudinal information from the *focals* to help make decisions when encountering the *site* information from the *soundscapes*. As a post-processing step, the

technique in which a *nocall* label is added to the lists of predictions with a particularly low confidence, is performed. More details on this technique are given below.

To evaluate the classification performance, the micro-averaged F1-score was used in the competition. This metric evenly blends precision and recall measurements and aggregates the binary classification comparisons into a single score for a multi-label task. It accounts for proportional class contributions, which is useful for our imbalanced dataset, and emphasises common class performance, as opposed to the macro-averaged F1-score, which emphasises uncommon class performance [33].

One team [25] used an interesting technique to improve their score. To the list of predictions, they added the *nocall* label if the other predictions were of a sufficiently low confidence, even though this yields in a prediction list that can never be totally correct because there cannot simultaneously be calls and *nocall* in a training example. The justification is that data points with only one correct label will be given no score if that label is not predicted but a data point with multiple correct labels will get a non-zero score if at least one of them was predicted. The *nocall* label is popular, so predicting it more often is strategically advantageous. We copy this score-optimising strategy in the performant solution. A straightforward way to address this loophole in the metric is to add some custom logic to penalise predicting *nocall* in conjunction with other classes.

4.2 How the key challenges of this competition are mitigated

Section 3 showed how the dataset presents challenges in that it is labelled by amateurs at a non-ideal granularity, has a long-tailed class distribution, creates a domain mismatch, and that some features may confuse the learning process. These are some strategies to deal with those challenges:

- The undesirable effects of labels being at the audio file level can be mitigated by globally pooling all the segments into which a focal is divided, the "self-distillation" technique, and others as described in Section 2.5.2.
- The technique which most significantly mitigated the negative effects on learning potential of the long-tailed class distribution is the introduction of more *nocall* instances. Additionally, the organisers of the competition have already applied undersampling to deal with the class imbalance. This is evident in Fig. 4 where the number of instances for each species is shown to be capped at 500.
- The ability of CNNs to transfer their learning between mismatched domains [2] is one reason why our solution and almost every solution submitted to the Kaggle competition were based on CNNs.
- A popular design choice was to not use the information provided by the secondary labels because of the risk that they may add more confusion than desirable discriminatory powers to the process. We took the approach of [25] to model our lack of confidence with the secondary labels by assigning a lower weight to them than the primary labels in our system of weighting the labels that were used to train the multi-label classifier.
- Oversampling is a popular technique to deal with imbalanced data in general [3]. However, teams in this competition did not benefit much from this technique, and the majority of the top solutions did not use it. Data augmentation techniques were popular among the competitors though; we normalised and decreased the standard JPG compression amounts in our solution by following the successes of [25].

5 Results and discussion

This section's results pertain to the training and evaluation of the performant solution. We did not achieve significant increases in performance compared to the top Kaggle solutions. Our solution performed approximately 4% worse than the best score on the Public Leaderboard but was able to do better than the best score on the Private Leaderboard by approximately 1%. Table 1 shows the differences. Note that our goal is not to outperform the Kaggle solutions, but to investigate the impact of different solution components and provide a convenient framework for similar future study. The competition did not have any comparison of the techniques which the competitors used; this paper addresses that gap.

5.1 Developing multi-label classifiers

The multi-label classifier can utilise up to four different datasets in the training process. They are (as shown in Fig. 5): *focals*, *focal call probabilities*, *freefield1010 no call* and *2020 validation no calls*. The first two are crucial because they are derived from the training data provided in the competition. The last two contain recordings of ambient noise in which no birds can be heard calling. They can help to give the model more examples of the dominant *nocall* class. We evaluate all four combinations of including or not including these two datasets as architectural components in the training process. Since we repeat all experiments in this section for each of the four configurations, Figs. 7 to 9 have the same legend, where ¬ denotes "not using", "f" denotes the *freefield1010* dataset and "v" denotes the dataset

Table 1 Comparing some of our preliminary solutions with the best on Kaggle

Solution	Private Score	Public Score
Ours: ResNeSt-26 #1	0.7045	0.7554
Ours: ResNeSt-50 #1	0.6968	0.7482
Ours: ResNeSt-50 #2	0.6942	0.7448
Team: Dr. 北村の愉快な仲間たち	0.6932	0.7736
Ours: ResNeSt-26 #2	0.6926	0.7678
Team: new baseline	0.6893	0.7998
Team: Shiro	0.6891	0.7919
Team: Third time's the charm.	0.6864	0.7897
Team: Kramarenko Vladislav	0.682	0.7897
Ours: simple baseline	0.5511	0.6131

We sort by Private Leaderboard score because that is how the winners of the competition were decided

constituting the *nocall* instances from the 2020 competition. The micro-averaged F1 score is calculated using five-fold cross validation using the *soundscape* data.

5.1.1 Which is the best CNN?

There are three popular pre-trained CNNs which were used most frequently in the competition for transfer learning [34]. They are: *ResNeSt-26*, *ResNeSt-50* and *efficientnet*. We train them for 100 epochs and analyse the statistics of the scores for the best 50 epochs with these fixed hyper-parameters: $\{\alpha = 0.5, \neg g\}$ (see Section 5.1.2 for more details).

Figure 7 shows that *ResNeSt-50* is better than the other two CNNs because its lower quartiles are greater than all the other upper quartiles. It is less obvious which is better between the other two, but we may say that *ResNeSt-26* is more "consistent" because: its minima are greater than *efficientnet*'s except for $\{\neg f, v\}$; its maxima are mostly less than *efficientnet*'s, and the differences between its lower and upper quartiles are less than *efficientnet*'s. It makes sense that the network with the most layers performs the best. The values in this comparison are shown in Table 2.

5.1.2 How do different hyper-parameters perform?

We explore the performance of different hyper-parameters for *ResNeSt-50*, which had the highest performance among those tested in Section 5.1.1. We are not particularly interested in the "best" hyper-parameters because it is beneficial to use an ensemble of models with a mixture

Table 2 Tabular representation of the statistics graphed in Fig. 7

Models	Datasets	q0	q1	q2	q3	q4	mean
ResNeSt-26	**¬f, ¬ v**	0.5046	0.5097	0.5135	0.5198	0.5264	0.5145
ResNeSt-26	**¬f, v**	0.5055	0.5119	0.5170	0.5205	0.5328	0.5169
ResNeSt-26	**f, ¬ v**	0.5039	0.5079	0.5126	0.5177	0.5357	0.5140
ResNeSt-26	**f, v**	0.5034	0.5074	0.5091	0.5128	0.5237	0.5106
ResNeSt-50	**¬f, ¬ v**	0.5206	0.5251	0.5287	0.5348	0.5472	0.5301
ResNeSt-50	**¬f, v**	0.5269	0.5333	0.5388	0.5424	0.5491	0.5380
ResNeSt-50	**f, ¬ v**	0.5224	0.5257	0.5293	0.5339	0.5447	0.5302
ResNeSt-50	**f, v**	0.5195	0.5283	0.5335	0.5363	0.5458	0.5324
efficientnet	**¬f, ¬v**	0.4904	0.5003	0.5061	0.5121	0.5284	0.5067
efficientnet	**¬ f, v**	0.4930	0.5053	0.5082	0.5180	0.5301	0.5103
efficientnet	**f, ¬ v**	0.4931	0.5012	0.5089	0.5178	0.5386	0.5095
efficientnet	**f, v**	0.4994	0.5060	0.5128	0.5212	0.5444	0.5144

q0 is the zeroth quartile (minimum), q1 is the first quartile (lower quartile), q2 is the second quartile (median), q3 is the third quartile (upper quartile), q4 is the fourth quartile (maximum). The values are F1 scores

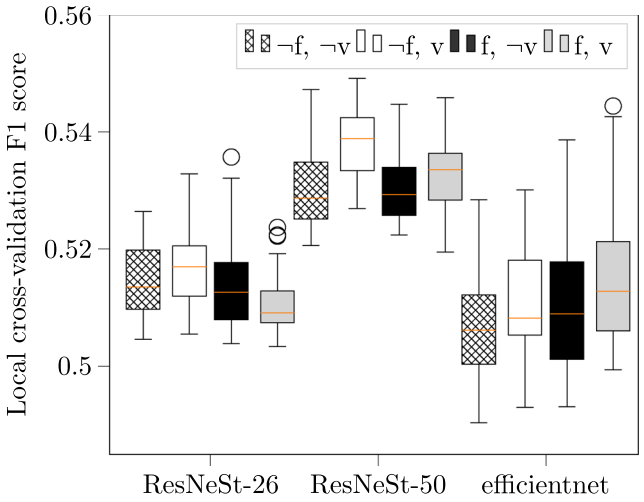
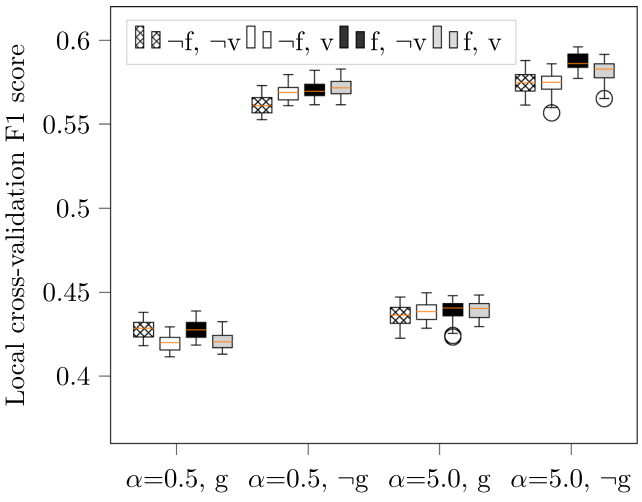
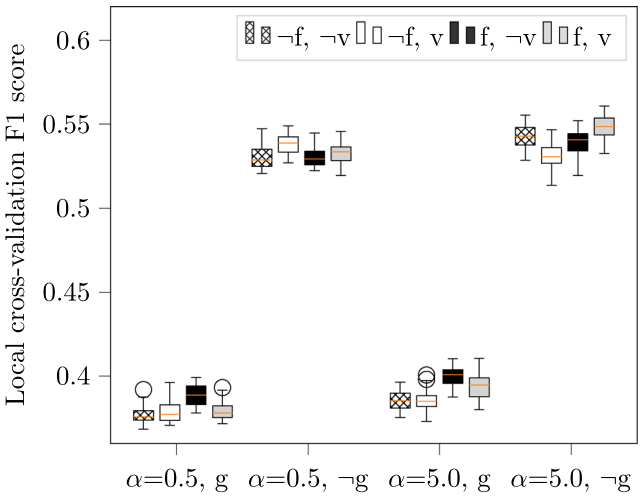


Fig. 7 Determining the best CNN when different supplemental datasets are or are not included during training. The box-and-whisker diagram plots the F1 scores



(a) Not including the *nocall* detector component: $\neg nc$



(b) Including the *nocall* detector component: nc

Fig. 8 Performance of *ResNeSt-50* using different hyper-parameters

of hyper-parameters among them, as demonstrated in Section 5.2.1. However, we would like to compare their performance to help us decide which to use in our ensemble.

Figure 8 splits the results of evaluating the combinations into two graphs. The hyper-parameters are:

- Whether to include the *nocall* detector's probabilities for weighting the predictions of the multi-label classifier to help reduce false positives. The models in Fig. 8a do not use the *nocall* detector ($\neg nc$) while the models in Fig. 8b do (nc).
- The α coefficient of the Mix Up data augmentation process where some spectra and their labels are combined to create new training instances to help the models learn to identify simultaneous bird calls.
- Whether to split the recordings uploaded by the same author on Xeno Canto across different folds ($\neg g$) or not (g). By not allowing this split, i.e. "grouping by the author" (g), it becomes more likely that different microphones and recording idiosyncrasies would have been used to record the audio between the training and validation sets because it is more likely that the same author will use the same microphone and recording habits. This may prevent over-fitting to specific microphones.

The performance with the *nocall* detector is consistently lower than without it. This observation is discussed in contrast to the results from the hidden test sets on Kaggle in Section 5.2.1. When the same author's data is allowed to appear in the training and validation sets, we observe the highest performance increase in these experiments. This is expected because the model may be learning peculiarities of different authors instead of the bird call patterns. The performance of the different α coefficients is similar, but the 5.0 value slightly outperforms the 0.5 value more often.

5.1.3 How do different hyper-parameters affect training time?

It is common to examine "online" inference time of ML systems because this indicates what resources are required when the systems are deployed and will be relevant for the entire life cycle of the systems. It is also useful to consider "offline" training time even though it is a "once off" factor because the resource costs can be significant when there is a lot of data and the models are complex.

Figure 9 shows the average time taken to train one epoch of each model on our computing cluster, where up to two

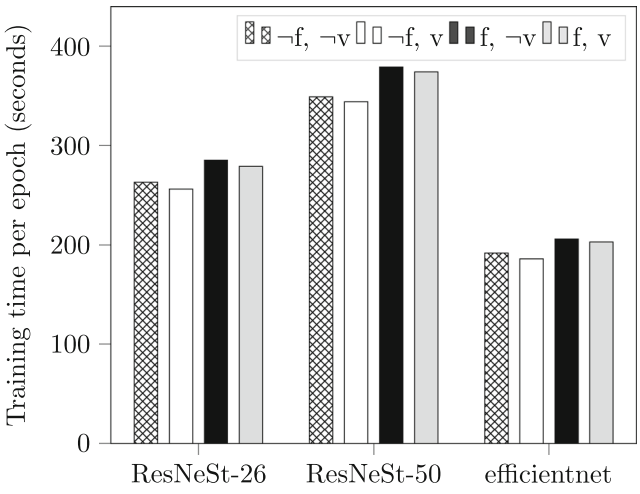


Fig. 9 The average time taken to train one epoch for a variety of CNNs and supplemental datasets as hyper-parameters

NVIDIA RTX 2080 Ti GPUs, 24GB RAM and four CPU cores were available. Grouping by CNN, we find that the better a model performs (see Fig. 7), the slower it is to train. Since *efficientnet* is significantly faster to train than *ResNeSt-26*, we would recommend using *efficientnet* when training resources are at a premium because the two perform similarly. The two models not using f are 8% faster than the other two that do, probably because it takes a significant amount of time to consider the 7,690 instances in the f dataset. The p-value in a Student's T-Test comparing the training times for these two groups is 0.607, so we fail to reject the null hypothesis that there is no difference in training time between them.

5.2 Testing the entire system on Kaggle

In the following experiments, we test the entire system where only the configuration of the multi-label classifier ensemble is varied. The results come from the Kaggle Public Leaderboard. Since the competition is over, we have access to the Private Leaderboard scores, but it is not good practice to tune solution settings using this dataset. We could not learn much from doing so anyway, because the Public and Private testing datasets should be drawn from the same distribution. Our best Public scores were 4% worse and our best Private scores were 1% better than the best on the Leaderboards.

5.2.1 How does ensemble diversity affect performance?

The models in an ensemble can have the same or different hyper-parameters. We hypothesised that having a high "diversity" in the ensemble (i.e., where each ensemble consists of different amounts of models which each have different hyper-parameters) would lead to better overall performance because models trained with different hyper-parameters may be better, in aggregate, at classifying a wider variety of test set inputs. The reason for this may be because more hyper-parameter diversity may be able to accurately model the outputs of more diverse inputs. To test this hypothesis, we chose the best-performing epoch for each of three different levels of model amounts and diversity in hyper-parameter combinations to constitute an ensemble. In other words, we trained ensembles which were composed of three different amounts and configurations of models and compared them. Figure 10 shows the performance of these. In its legend, the first letter refers to each of the three diversity levels, which are defined as:

$$
\begin{aligned}
&'l' : low : \{\alpha = 5.0, \neg g, \neg v, \neg f\} \\
&'m' : medium : \{\alpha = 0.5, \neg g, \neg v, \neg f\}, \{\alpha = 5.0, \neg g, \neg v, \neg f\}, \\
&\{\alpha = 5.0, g, \neg v, \neg f\}, \{\alpha = 5.0, g, v, f\} \\
&'h' : high : \{\alpha = 0.5, \neg g, \neg v, \neg f\}, \{\alpha = 0.5, \neg g, \neg v, f\}, \\
&\{\alpha = 0.5, \neg g, v, f\}, \{\alpha = 0.5, g, \neg v, \neg f\}, \{\alpha = 0.5, g, v, f\}, \\
&\{\alpha = 5.0, \neg g, \neg v, \neg f\}, \{\alpha = 5.0, \neg g, v, f\}, \{\alpha = 5.0, g, v, \neg f\}, \\
&\{\alpha = 5.0, g, \neg v, \neg f\}, \{\alpha = 5.0, g, v, f\}
\end{aligned}
$$

Along with diversity, we also compare the performance of models using or not using a *nocall* detector. The difference between the $\{nc\}$ and $\{\neg nc\}$ pairs (distinguished by black and grey colouring in Fig. 10) was the opposite of what we found during local validation testing (discussed in Section 5.1.2). *nc* leads to a performance increase of between 1.218 and 1.811 times, depending on ensemble

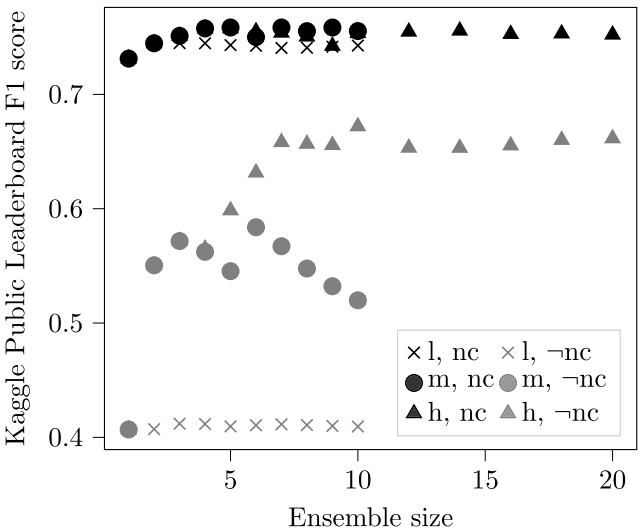


Fig. 10 Performance of the entire system on the hidden test with different multi-label classifier ensemble diversities. The black points represent solutions which included the *nocall* detector while the solutions in grey did not. The three letters (*l, m, h*) represent different solution diversities, about which more details are given in the text body of Section 5.2

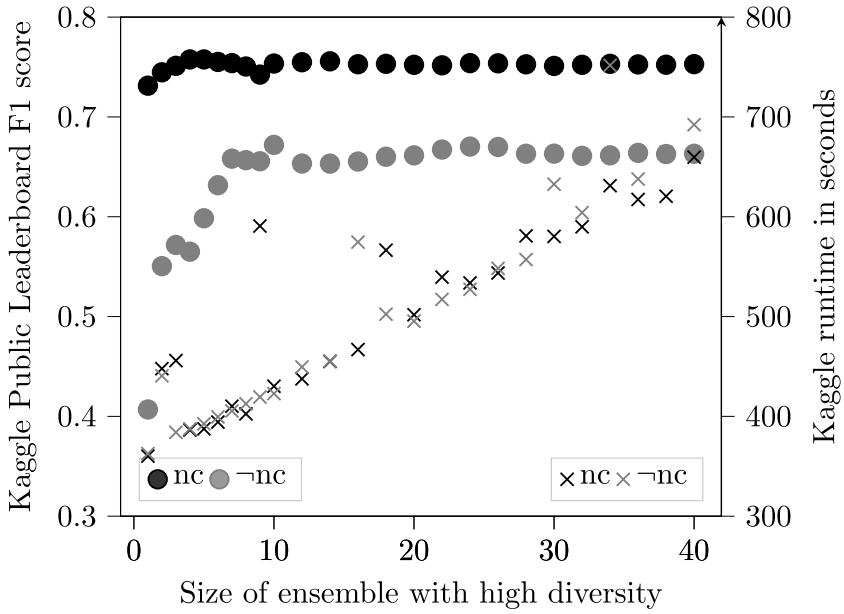


Fig. 11 Kaggle runtime (left, scatteered dots) compared to accuracy (right, scattered crosses) with different high-diversity *ResNeSt-50* ensemble sizes

diversity. This is most likely due to the inclusion of the *focal* metadata which occurs in Phase 5 (see Section 4), which helps to make better predictions based on time (season) and location (geography). Other potential reasons could be:

- There is a difference between the *soundscape* data provided for training/validation and the hidden test set. For example:
  - There could have been more noise in the test set instances, meaning that the *nocall* detector could have been a lot more valuable as an expert system dedicated to avoiding being fooled by these noises.
  - Since the training/validation data is only sampled from two of the four recording locations, we may have missed the opportunity to learn features which were present in the other two locations.
- Other hyper-parameters which we did not test that could have reduced/neutralised/reversed this performance difference could already have been discovered by the competitors and fixed. For example:
  - We also used the ADAM optimiser and a Cosine Annealing Warm Restarts scheduler for training without experimenting with alternatives.
  - Competitors may have tried other popular CNNs but found that the three which were used most commonly in the competition had the best test set performance despite having lower validation performance.

Unfortunately, we cannot investigate these other potential reasons because we do not have access to the hidden test set.

Note that whenever the ensemble size is larger than the number of hyper-parameter combinations, we choose the next-best epochs as the other models.

The best score recorded during testing is 1.272 times the value of the best score recorded during validation. $\{m, \neg nc\}$ decreases in performance after more than six models are present. In the nc configurations, the m ensembles are 1.012 times as good as the s ensembles but the h ensembles are only 0.996 times as good as the m ensembles. The top four scores overall were achieved with m ensembles. Therefore, for nc, it seems that increasing up to, approximately, the top four unique model configurations, increases performance, but adding more models with worse hyper-parameters hinders performance after that. However, for $\neg nc$, we see that h continues to climb in performance up to about ten models. This may be because there is still a lot of performance to gain when the system does not benefit from the *nocall* detector's expertise. With that expertise, we seem to reach the peak in performance with this paradigm much faster.

5.2.2 What is the relationship between performance and inference time?

Figure 11 helps in assessing the trade-off between performance and "online" inference time requirements. The dots show how performance increases with ensemble size and then plateaus at around size ten. The crosses show the somewhat linear relationship between inference time and ensemble size. We thought it necessary to test up to forty

models in the ensemble because a small amount of data is anomalous and we wanted to be sure that the general trend is linear. These anomalies may be due to inconsistent loads on Kaggle's servers.

6 Conclusion

In this work, we have investigated the identification of bird species in audio recordings. To do so, we have focused on the BirdCLEF2021 Kaggle competition, analysing related works and discussing the most popular solutions adopted by the participants in order to understand how to solve this challenge. We have utilised publicly-available competition solutions to devise a code-base which can be altered in a modular fashion to easily experiment with different architectural components. We conducted a thorough set of experiments, which, in addition to standard parameter tuning, investigated the individual contributions of entire architectural components.

We chose *ResNeSt-26* for the *nocall* detector because it performed the best during preliminary experiments. We did not change it in the experiments that were performed for later components in the pipeline. However, *ResNeSt-50* was found to be better as the multi-label classifier. Thus, there is an opportunity to explore other models for the *nocall* detector.

In summary, our study yields the following key insights:

1. It is beneficial to use the *nocall* detector due to its high performance on the testing dataset even though not using it may seem better according to its performance on the validation dataset.
2. Testing dataset performance is correlated to the degree of diversity in the multi-label classifier ensembles when the *nocall* detector is not used, but ensemble diversity is much less important when the *nocall* detector is included. This may indicate a ceiling in what can be reliably learned from our datasets.
3. Whether the *nocall* detector was used or not, performance on the testing dataset plateaus after ten ensembled multi-label classifiers.

Acknowledgements This work is partly supported by projects A-TIC-434-UGR20 and PID2020-119478GB-I00.

Funding Funding for open access publishing: Universidad de Granada/CBUA. I. Triguero's work is supported by projects A-TIC-434-UGR20 and PID2020-119478GB-I00. I. Triguero is currently enjoying a Maria Zambrano fellowship at the University of Granada.

Data Availability The datasets analysed during the current study belong BirdCLEF2021 Kaggle competition and are available at https://www.kaggle.com/c/birdclef-2021/overview.

Declarations

Competing interests The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Bagnall A, Lines J, Bostrom A (2017) The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Min Knowl Disc 31(3):606–660
2. Bird JJ, Kobylarz J, Faria DR et al (2020) Cross-domain MLP and CNN transfer learning for biological signal processing: EEG and EMG. IEEE Access 8:54,789–54,801
3. Buda M, Maki A, Mazurowski MA (2018) A systematic study of the class imbalance problem in convolutional neural networks. Neural Netw 106:249–259
4. Cakir E, Parascandolo G, Heittola T (2017) Convolutional recurrent neural networks for polyphonic sound event detection. IEEE/ACM Transactions on Audio, Speech and Language Processing 25(6):1291–1303
5. Chachada S, Kuo CCJ (2014) Environmental sound recognition: a survey. APSIPA Transactions on Signal and Information Processing 3:e14
6. Chen L, Gunduz S, Ozsu MT (2006) Mixed type audio classification with support vector machine. In: 2006 IEEE international conference on multimedia and expo, pp 781–784
7. Dandashi A, AlJaam J (2017) A survey on audio content-based classification. In: 2017 International conference on computational science and computational intelligence (CSCI), pp 408–413
8. Dosovitskiy A, Beyer L, Kolesnikov A et al (2021) An image is worth 16x16 words: transformers for image recognition at scale. In: International conference on learning representations (ICLR 21)
9. Feng W, Huang W, Ren J (2018) Class imbalance ensemble learning based on the margin theory. Appl Sci 8(5):815
10. Fernández A, García S, Galar M et al (2018) Learning from imbalanced data streams. In: Fernández A, García S, Galar M (eds) Learning from imbalanced data sets. Springer International Publishing, Cham, pp 279–303
11. Gouyon F, Pachet F, Delerue O (2000) On the use of zero-crossing rate for an application of classification of percussive sounds. Proceedings of the COST G-6 Conference on Digital Audio Effects
12. Gu J, Wang Z, Kuen J et al (2018) Recent advances in convolutional neural networks. Pattern Recogn 77:354–377
13. He K, Zhang X, Ren S (2016) Deep residual learning for image recognition. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR)

14. Ignatov AD (2018) Real-time human activity recognition from accelerometer data using convolutional neural networks. Appl Soft Comput 62:915–922
15. Ismail Fawaz H, Forestier G, Weber J (2019) Deep learning for time series classification: a review. Data Min Knowl Disc 33(4):917–963
16. Schlüter J (2021) Learning to monitor Birdcalls from weakly-labeled focused recordings. CEUR Workshop Proceedings 29362936(CLEF 2021 Working Notes)
17. Puget J-F (2021) STFT transformers for bird song recognition. CEUR Workshop Proceedings 2936(CLEF 2021 Working Notes)
18. Li J, Pedrycz W, Gacek A (2022) Time series reconstruction and classification: a comprehensive comparative study. Appl Intell 52(9):10,082–10,097
19. Lin L, Xu B, Wu W et al (2019) Medical time series classification with hierarchical attention-based temporal convolutional networks: a case study of myotonic dystrophy diagnosis. In: IEEE conference on computer vision and pattern recognition workshops, CVPR workshops 2019, Long Beach, CA, USA, June 16-20, 2019, pp 83–86
20. Liu W, Wang H, Shen X et al (2022) The emerging trends of multi-label learning. IEEE Trans Pattern Anal Mach Intell 44(11):7955–7974
21. Shugaev MV, Tanahashi N, Dhingra P (2021) BirdCLEF 2021: building a birdcall segmentation model based on weak labels. CEUR Workshop Proceedings 2936(CLEF 2021 Working Notes)
22. Morales G, Vargas V, Espejo D et al (2022) Method for passive acoustic monitoring of bird communities using UMAP and a deep neural network. Eco Inform 72:101, 909
23. Mumuni A, Mumuni F (2021) CNN architectures for geometric transformation-invariant feature representation in computer vision: a review. SN Computer Science 2(5):340
24. Musaev M, Khujayorov I, Ochilov M (2020) Image Approach to Speech Recognition on CNN. In: Proceedings of the 2019 3rd international symposium on computer science and intelligent control. Association for Computing Machinery, New York, NY, USA, ISCSIC 2019, pp 1–6
25. Murakami N, Tanaka H, Nishimori M (2021) Birdcall identification using CNN and gradient boosting decision trees with weak and noisy supervision. CEUR Workshop Proceedings 2936(CLEF 2021 Working Notes)
26. Qin J, Pan W, Xiang X (2020) A biological image classification method based on improved CNN. Eco Inform 58:101,093
27. Singer E, Reynolds DA (2015) Domain mismatch compensation for speaker recognition using a library of whiteners. IEEE Signal Process Lett 22(11):2000–2003
28. Smith JO (2011) Spectral Audio Signal Processing. Stanford University, CCRMA
29. Sun L, Lyu G, Feng S et al (2021) Beyond missing: weakly-supervised multi-label learning with incomplete and noisy labels. Appl Intell 51(3):1552–1564
30. Tarekegn AN, Giacobini M, Michalak K (2021) A review of methods for imbalanced multi-label classification. Pattern Recogn 118:107,965
31. Tuia D, Kellenberger B, Beery S et al (2022) Perspectives in machine learning for wildlife conservation. Nat Commun 13(1):792
32. Wang T, Li Y, Kang B (2020) The devil is in classification: a simple framework for long-tail instance segmentation. In: Computer vision – ECCV 2020. Springer International Publishing, Cham, pp 728–744
33. Yang Y, Liu X (1999) A re-examination of text categorization methods. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '99. ACM Press, Berkeley California, United States, pp 42–49
34. Zhang H, Wu C, Zhang Z et al (2022) Resnest: Split-attention networks. In: 2022 IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW), pp 2735–2745
35. Zhang Y, Kang B, Hooi B et al (2021) Deep long-tailed learning: a survey. https://doi.org/10.48550, arXiv.2110.04596
36. Zhang Z, Sabuncu M (2020) Self-Distillation as instance-specific label smoothing. In: 34th Conference on neural information processing systems (NeurIPS 2020), Vancouver, Canada
37. Zhao Y, Xu S, Huang Z et al (2022) Temporal and spatial characteristics of Soundscape ecology in urban forest areas and its landscape spatial influencing factors. Forests 13(11):1751
38. Zhou ZH (2017) A brief introduction to weakly supervised learning. Natl Sci Rev 5(1):44–53

Kyle Maclean received his M.Sc. in Computer Science with Artificial Intelligence from the University of Nottingham, UK in 2022. He is currently working as a Software Engineer. His research interests include computer vision and natural language generation. He grew up in the KwaZulu-Natal Drakensberg, South Africa.

Isaac Triguero received his M.Sc. and Ph.D. degrees in Computer Science from the University of Granada, Granada, Spain, in 2009 and 2014, respectively. He is a currently enjoying a Distinguished Senior Research Fellowship at the University of Granada, taking a teaching break from his position as an Associate Professor of Data Science at the University of Nottingham. His work is mostly concerned with the research of novel methodologies for big data analytics. Dr Triguero has published more than 90 international publications in the fields of Big Data, Machine Learning and Optimisation (H-index=33 and more than 4500 citations on Google Scholar). He is a Section Editor-in-Chief of the Machine Learning and Knowledge Extraction journal, and an associate editor of the Big Data and Cognitive Computing journal, and the IEEE Access journal. He has acted as Program Co-Chair of the IEEE Conference on Smart Data (2016), the IEEE Conference on Big Data Science and Engineering (2017), and the IEEE International Congress on Big Data (2018). Dr Triguero is currently leading a Knowledge Transfer Partnership project funded by Innovative UK and Unilever that investigates interpretable machine learning models for sustainable business operations.