University of Nottingham
UK | CHINA | MALAYSIA

INTERIM REPORT

Identifying Bird Species by their Calls in Soundscape Recordings

Submitted on 25 April 2022, as part of my work towards the degree:
Master of Science, Computer Science with Artificial Intelligence.

Kyle Maclean
Supervisor: Isaac Triguero

School of Computer Science
The University of Nottingham
United Kingdom

Abstract

Bird species can be recognised by looking for patterns in audio recordings of their calls. A recent machine learning competition challenged participants to create software which is capable of labelling audio segments with the names of bird species which can be heard calling in them. Using the data and metrics of this competition, we present a high-performing solution to accomplish this task. Additionally, a thorough investigation of the relevant Data Science literature is provided as a useful resource for future developments. At the Interim Report stage, majority of the relevant techniques have been discussed, the provided data has been analysed and architectures for the high-performing solution as well as a simple baseline have been outlined.

Contents

1 Introduction and Motivation

Machine learning is very effective at automating ecological management processes. For example, convolutional neural networks (CNNs) have been applied to "camera trap" video data for monitoring wildlife in particular locations [38]. We will be studying audio to identify birds, which is useful in tourism, ornithology, and climate change mitigation [32]. Conservationists can gain insight into how well "rewilding" projects are working by monitoring the diversity and magnitude of bird populations. The birds also contribute to healthy ecosystems by scattering seeds and curtailing pests. Historically, the monitoring process has been manual, with people learning to recognise birds by their vocalisations (calls) since visibility is often low in dense forests.

Since there are many species of birds, people require a lot of training before they can proficiently identify them. Automating the identification process can therefore save on human resources, reduce bias in measurement and potentially reach massive scale [26]. It may also create opportunities to monitor areas which are inaccessible to humans, such as reed ecosystems. From a commercial perspective, there may be a market for smartphone apps that can identify bird calls in real time[1]. If the algorithms are computationally efficient enough to execute on the device without having to stream to a server, then such apps can be used offline. This is beneficial for use cases in non-urban environments which do not have good communications infrastructure, i.e., environments where lots of birds can be found.

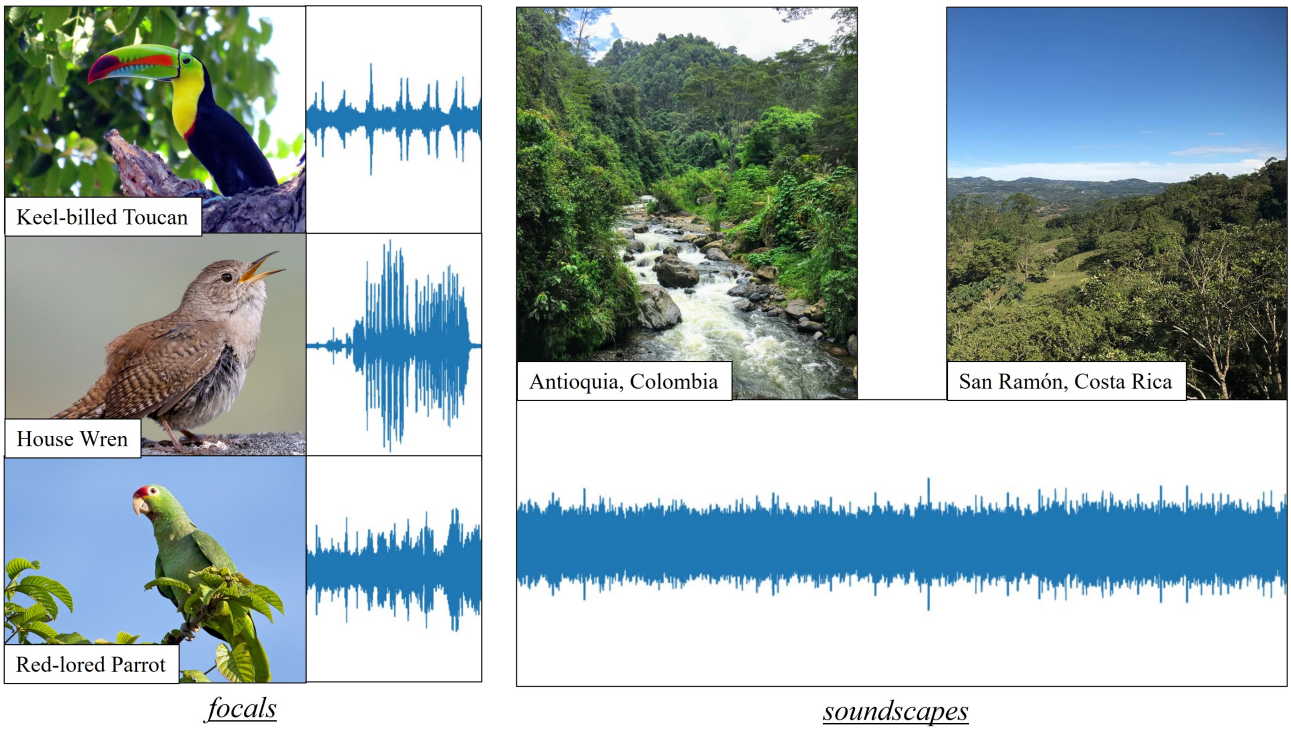


Figure 1: Illustrating the mismatch between the *focals* training data and the *soundscapes* validation/testing data. The waveforms of each bird call on the left exhibit distinct patterns. The waveform on the right is a longer segment of *soundscape* audio recorded in San Ramón, Costa Rica. Due to the background noise (especially the sound of running water in this example), bird calls are less easy to distinguish. Image credit: `ebird.org (Blair Dudeck, Rolando Jordan, Chris Wood), seecolombia.travel, retireforlessincostarica.com`

[1]The *Picture Bird* app (`https://play.google.com/store/apps/details?id=com.glority.picturebird`) has over 500,000 downloads and integrated adverts, which may generate a lot of revenue

Our aim is to explore techniques to create automated software systems which can correctly list all bird species that can be heard calling in a 5-second segment of audio, thus relieving people of this task. This project is based on the Kaggle competition, BirdCLEF2021 [25], in which bird calls had to be classified in *soundscapes* (10-minute long audio files in which different bird calls and background noise can be heard) after having learned the calls of different species from *focals* (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). Therefore, there is a *domain mismatch* between training and validation/testing data, as illustrated in Figure 1. The data forms a *time series* because each recording is composed of audio samples made sequentially through time [8], at the rate of thousands per second in our case. We require supervised learning because the data is annotated with *ground truth* labels [10]. We seek to learn the relationship between the given training data and its labels so that we can classify new data with appropriate labels. There may be multiple labels per instance because there may be multiple birds calling simultaneously. Thus, we model the task of recognising bird calls in audio data as a **multi-label time series classification problem**. In addition to the audio files themselves, we also have some metadata for each recording; some examples are shown in Table 1. The See Section 4 for more information on the data and Section 6 for a discussion on how to score classifiers of such datasets.

primary_label	**secondary_labels**	**latitude**	**longitude**	**date**	**rating**
acafly	['tuftit', 'reevir1', 'ovenbi1']	38.802	-94.6896	15.5.17	3.5
bawwar	['buggna', 'norcar']	40.4327	-79.904	29.4.20	0
carchi	[]	40.0975	-75.4497	16.1.17	3

Table 1: Five selected examples of the metadata available for the *focal* (training) recordings. Other less relevant columns include: **type, scientific_name, author, filename, license, time, url**. The labels are a short code for the common names of the birds. In the primary labels of this example, *acafly* stands for the *Acadian Flycatcher*, *bawwar* for *Black-and-white Warbler* and *carchi* for *Carolina Chickadee*.

Long-standing techniques for *supervised classification* learning include support vector machines (SVMs), neural networks, logistic regression, naïve bayes and decision trees [6]. More recently, random forests, extreme learning machines and deep neural networks have become popular [44]. Some successful techniques for time series data naturally consider its temporal structure through a kind of memory, like recurrent neural networks (RNNs) [22]. However, an alternative paradigm in which the above techniques are used in a non-time series manner has shown the highest performance for bird call recognition. This paradigm involves splitting audio signals into discrete segments (capturing some tune-able range of information over time) and classifying the segments individually. It fits with the constraints of the competition, where, although *soundscapes* are 10-minutes long, participants were required to make a list of bird call predictions for every 5 seconds of the file. Figure 2 shows how one of the *soundscapes* provided for validation can be broken into 5-second segments and labelled with the species which can be heard calling in each segment (this is our goal). Since most machine learning algorithms work best with fixed-size input vectors [16], it makes sense to learn and infer using a feature vector of standardised length, as input. An extensively-used technique to manipulate the vector as a series of audio samples is through converting it into a two-dimensional *spectrogram*, which visualises the intensity of different frequencies in an audio signal over a specified time window. It is produced by applying short-time Fourier transformations (STFT) on (usually overlapping) time windows [37]. A simplified illustration of this particular method is shown in Figure 3.

Spectra have been used in many sound event detection (SED) problems, although this domain is disproportionately unpopular compared to conceptually similar domains [12]. We

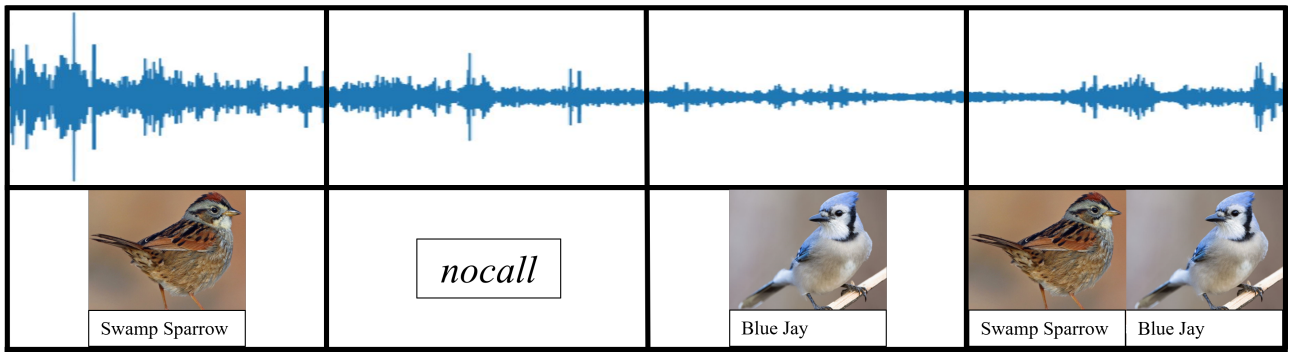


Figure 2: Visualising the *soundscape* (validation/testing) data and labels. The waveform of the audio, with the y-axis showing Decibels (dB), is shown at the top. Note: a lot of wind could be heard in these segments, so most of the spikes represent wind noise; the bird sounds were relatively very quiet. The bottom of the picture shows pictures of each species which was labelled as calling in that 5-second segment. These particular segments are taken between 525 and 545 seconds from `train_soundscapes/28933_SSW_20170408.ogg`. Image credit: `ebird.org (Keenan Yakola, Ryan Sanderson)`

are motivated to contribute to SED as it may present an opportunity to contribute novel ideas and a handy technical resource for future development. To demonstrate its unpopularity:

- the UCR Time Series Archive's three most popular dataset categories are "Image", "Motion" and "Sensor"[2]. This indicates that time series classification algorithms do not have as many opportunities for testing on sound-related datasets
- the field of AutoML, in which the goal is to have a deep learning system be able to "build itself" without human intervention has mainly seen its significant progress in the domains of image recognition, object detection and language modelling [20], and not much attention has been given to SED [5].

The domains of images, languages and sounds are similar in that the goal is to find patterns in time-series data that can be processed naturally by humans, so we might intuitively think that many techniques that are successful in one domain can be applied to the other. It is motivational to explore the extent to which techniques from Computer Vision can be adapted to conceptually-similar audio problems.

Our aim is to investigate the state-of-the-art in this problem domain and present a thorough breakdown of all the necessary concepts and techniques to achieve high performance. The first set of objectives to achieve this is to present software which will:

- define a computationally-efficient machine learning workflow which can learn to effectively distinguish bird calls in audio, with justifications of its architecture through reference to theoretical and empirical studies
- utilise additional external data sources to reduce class imbalance (and therefore false positives at inference) and incorporating the given metadata into the learning process (instead of just using the audio itself)
- register high scores on the Private and Public Leaderboards for the BirdCLEF2021 Kaggle competition

[2] `http://timeseriesclassification.com/dataset.php`

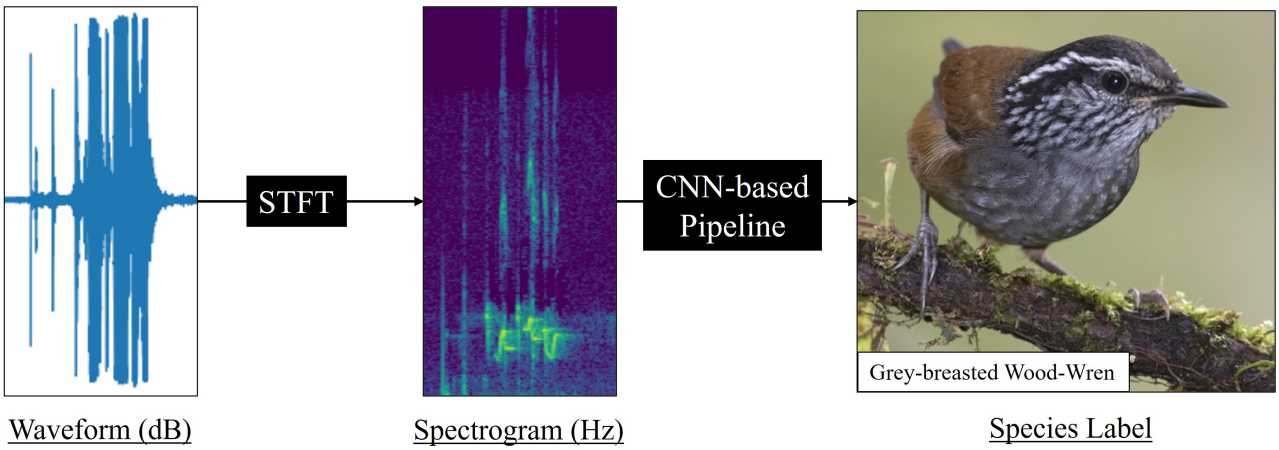


Figure 3: A common method to recognise bird calls. From the left, an audio recording of a bird call represented as decibels/loudness over time is converted into a spectrogram represented as frequency/tonality over time using a short-time Fourier transformation. The spectrogram can be treated as an image and fed into a pipeline involving CNNs which have been pre-trained to perform well on object recognition tasks in images and have had their learning transferred to distinguishing patterns in spectra. The class probabilities in the final network layer can be used to infer which bird species is most likely to have made the call. Image credit: `birdphotos.com`

More information on the proposed software is given in Section 5. Submissions to the competition[3] achieve decent performance but do not present systematic reviews of the literature and instead usually focus on engineering a small set of techniques to perform well on the Public Leaderboard[4]. Accordingly, the second set of objectives is to:

- present a survey of Data Science techniques to reveal and explain methods and optimisations compared to those used in the top competition submissions, which can serve as a resource to guide future work in this problem domain, generally
- distil and describe a simple-yet-effective baseline solution to the problem so that the complex, high-performing solutions from the competition can be better-contextualised

The following section will make headway into achieving the second set of objectives by providing the necessary background to understand some of the concepts discussed later.

2 Background

To increase accessibility, this section presents some Computer Science and Data Science knowledge which Section 3 assumes. It sometimes make sense that a single datapoint be given multiple labels. An interesting application is to assign a list of emotions to pieces of music which exhibit them. There are many ways to define the accuracy of systems which accomplish such **multi-label classification**, and one of the considerations is how many labels of the entire list the system was able to discern [40]. Different classification models may have different strengths and weaknesses. Instead of only choosing one and therefore losing out on some strengths of other models which are left unused, researchers have found it beneficial to consider

[3]See the bird-related papers listed under the heading "LifeCLEF: Multimedia Life Species Identification" here: `http://ceur-ws.org/Vol-2936/`

[4]Follow this link and click on the Public Leaderboard button to arrange the solutions by the metric which the competitors used to evaluate their performance during the competition: `https://www.kaggle.com/c/birdclef-2021/leaderboard`

the outputs of multiple classifiers simultaneously. This is called **ensembling**, and its most simple method takes an average vote of the outputs from all the ensembled classifiers. Some other methods include *output coding*, *bagging* and *boosting* [13]. Different images of the same object may portray it differently. The object's pixel representation could be scaled, translated, rotated, illuminated differently, or projected (affine). So when extracting features of the object to compare for similarity, it is useful to process the features such that they become *invariant* to these changes. The first extremely popular and effective method to do this is **scale-invariant feature transform (SIFT)**. It uses statistics about how image gradient intensity changes in local areas of the image to determine points of interest, which are invariant to the above transformations [29]. Improvements on this technique include SURF and ORB [39].

Neural networks are a series of layers of nodes. Each of the first layer's nodes processes a component of the input data. Subsequent layers' nodes perform transformations on the data according to learn-able weight and threshold parameters, which are adjusted according to their influence on the output using the backpropagation algorithm. **Convolutional neural networks (CNNs)** are neural networks which incorporate:

- convolutional layers, which require the most computation because they apply a kernel / filter / feature extractor on most cells in the matrix of input data to compress the information into a smaller matrix
- pooling layers, which also also reduce the size of the data by combining its components but their functions are typically standard aggregation operations
- the final fully-connected layer, which considers the extracted features from the data in previous layers to output a series of probabilities that the data belongs to each of the possible classes [18]

See Figure 4 for an example the convolution process. In theory, deeper neural networks can learn more complicated relationships in data. However, it is often more difficult to train deeper networks because of the vanishing/exploding gradient problem, which prevents convergence. One popular method to overcome this is to program the nodes to learn residual functions of their inputs. These are called **residual networks (ResNets)** and are implemented by fitting an identity mapping (outputting what was input) between nodes in adjacent layers and then adjusting this relationship instead of starting from an arbitrary relationship. A layer performing an identity transformation on its input can also be thought of as a "skipped" layer, since it would make no difference if that transformation was not performed at all. Layers that start the training as skipped in this sense do not add time (computational) or space (parameter storage) complexity and can therefore help deeper networks to not become so bogged down with computational complexity [19]. Neural networks in their standard form are capable of learning and making predictions on datapoints independently. Some datapoints may be sequential in nature (and may form a time series). To utilise the information that a previous datapoint's prediction might influence a subsequent datapoint's prediction, **recurrent neural networks (RNNs)** have been developed. They maintain a "memory", in which previous inputs are represented as hidden states when processing later inputs. A common way to visualise this is to draw an arrow pointing from nodes in the network to themselves. A variant of the backpropagation algorithm called backpropagation through time (BPTT) is required to add the errors at each timestep, instead of just considering the errors for each input individually. This extra complexity makes RNNs more susceptible to exploding/vanishing gradient issues. A more sophisticated RNN variant introduces "bidirectionality", in which future data is used to improve current-state accuracy [22].

When the order in which observations are made is fundamental to the meaning of some data, then that data is said to form a **time series**. For example, the series of phonemes in human

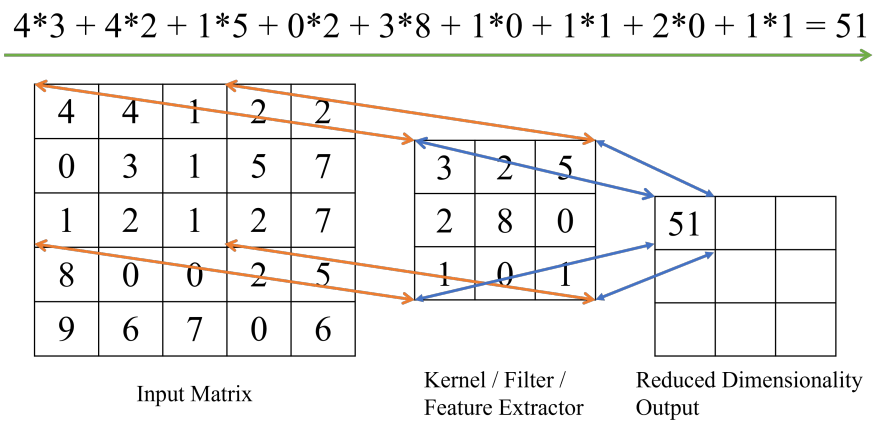


Figure 4: Illustrating the process of convolving a single cell in an input matrix. Note that the the values of the kernel / filter / feature extractor are trained and that the single cell in the output represents a compression of the information contained in the 3*3 cells of the input.

speech is a time series because it is the order in which the phonemes are said that gives meaning to what a person is saying [10]. When comparing different time-series patterns, it may be the case that one pattern should be recognised as another but it is somehow deformed, for example, it may have been emitted at a different speed. To match patterns which should be matched but are otherwise deformed relative to each other, **Dynamic Time Warping** applies nonlinear transformations so that the patterns can be more easily aligned [33]. To overcome issues around interpret-ability, efficiency and accuracy of standard time-series classification algorithms, the **shapelet** primitive was developed. The idea is to only use a part of the time-series signal which is best at discriminating classes. Considering just these maximally distinctive subsequences of the time series is more efficient than also comparing features which do not contribute significant discriminatory powers [43].

3 Related Work

Here we contextualise our problem domain's situation within the larger taxonomy of Data Science. The subsections traverse down the hierarchy shown in Figure 5, leading from broader to narrower levels of relevancy. Section 3.1 starts generally with Time Series Data and Signal Processing. Section 3.2 drills into audio signals specifically. Section 3.3 deals with a particular category of mixed type audio called Environmental Sound Recognition. Section 3.4 covers the most immediately relevant category of bird call recognition in *soundscapes*. Finally, Section 3.5 explains some of the techniques used to address the peculiarities and conditions which the competitors of BirdCLEF2021 faced.

3.1 Time series data and signal processing

Historically, Time Series Forecasting (the task of predicting future values based on previous values) has had a lot of attention, but recently, Time Series Classification (TSC) has increased in popularity, with hundreds of algorithms being published over the last few years [21]. Traditional approaches to TSC used mainly nearest neighbour (NN) classifiers [3]. Over time, more and more complicated ensembling architectures involving multiple NN classifiers (and other techniques such as Dynamic Time Warping) were developed as it became clear that ensembles often performed better than all their constituent classifiers [2]. After that, higher-performing techniques using *shapelet transformations* were developed. A *shapelet* is a subsection of time

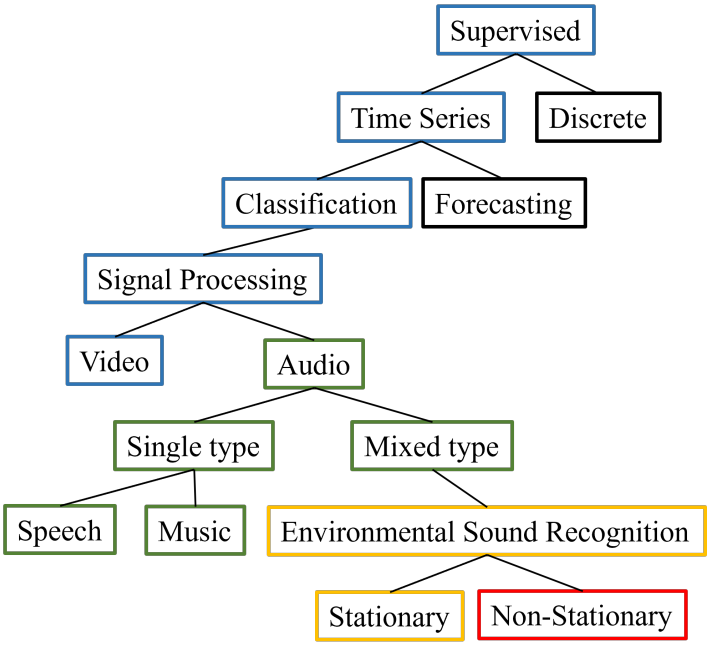


Figure 5: Taxonomy of the problem context. Each coloured rectangle is associated with a specific section: blue with Section 3.1, green with Section 3.2, yellow with Section 3.3, and red with Section 3.4.

series data which can be compared to other shapelets of the same length extracted from the series using a distance measure [43]. This concept is very valuable for bird call recognition because an audio recording can naturally be broken up into *shapelets* encompassing the start and end of each call. Ultimately, the highest accuracy was generally achieved with an ensemble of 37 classifiers called HIVE-COTE [28]. Alas, it is very computationally expensive to train that many classifiers.

Researchers sought a high-performing *and* efficient classification technique and so tried to move away from large ensembles. After the success of deep CNNs in computer vision [27], it seemed promising to utilise them in other time series signal processing domains. Before the advent of deep CNNs, image features were extracted using methods such as SIFT, SURF, ORB, etc., and then classified with models such as SVMs [39]. Deep CNNs brought great performance to natural language processing [4] and speech recognition [35]. These two types of tasks are similar in that they both involve sound where previous samples greatly influence the meaning of subsequent samples. This property is shared by bird call recognition tasks and is therefore logical to investigate the technique for this purpose as well. The performance is consistently high for signal-processing tasks because the convolutional layers provide a general machinery that can be specialised through training to compress information down to what is relevant for discriminating a particular signal. 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 [36]. The convolutions are also very capable of picking out features independently of where they are in the space of the image [1]. In our case, it is important that we are able to accurately recognise a bird call wherever it may occur in time. The earlier layers in a CNN learn more "primitive" features, such as edges, and the later layers learn more sophisticated shapes. We do not expect networks with more than 100 layers to be necessary for bird call recognition because the patterns are not at a high enough level for those deeper layers to be effective.

3.2 Understanding audio

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. We can use these measures to understand audio at three different levels [11]:

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 [9]. ZCR is usually only applied to speech recognition and music understanding because it measures the rate that the signal crosses negative to positive magnitudes [17]. Examples of single type audio are isolated speech, music, etc. Some of the "cleanest" *focal* recordings in the BirdCLEF2021 dataset can be considered examples of this type, because different bird species can exhibit different patterns in the ZCR statistic. However, to show its ineffectiveness for this dataset, we randomly sampled 10 *focals* for the *Keel-billed Toucan* and calculated ZCR statistics for each of them, arriving at the following values:

```
177011, 361624, 41041, 143295, 950122, 2949, 497421, 485385, 84918, 102374
```

The magnitudes of these values are very different, so the ZCR does not work for call recordings with as much noise as ours have; therefore, they can be considered *mixed type audio*. A survey from 2006 found that for this type, SVMs tended to perform well because of their ability to map data into higher spaces [9]. This survey found that most neural network methods until then had performed relatively poorly. This is probably due to the lack of expertly pre-trained CNNs, which have the ability to reduce the number of parameters required and to converge much faster than neural networks without convolutions, as discussed in Section 3.1. These techniques to not deal with the temporal nature of the audio through the implementation of a memory, so it is also worth exploring techniques that have incorporated residual neural networks (RNNs).

One method to consider the temporal aspect more explicitly is to use a set of CNNs to extract features containing time and frequency information in a way that is invariant to local disturbances in the signal. These features are then been fed into a set of RNNs to understand the temporal context. These are known as convolutional recurrent neural networks (CRNN), and have shown high performance for sound event detection (SED) [5]. A version of this strategy which achieves higher accuracy uses depth-wise separable convolutions instead of CNNs and dilated convolutions instead of RNNs, allowing the capture of longer-term temporal context without increasing the number of parameters [15]. We mention the last in particular because it draws attention to the difference between the usual SED tasks and ours. The SED which these papers address is to annotate an audio file with overlapping segments of sound events. For example, given a 20 second audio clip, determine that *footsteps* were heard between seconds 4 and 15, *thunder* was heard between seconds 12 and 17, and *rain* was heard throughout. It is therefore 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 10. Also, many of these bird calls sound similar, whereas the common SED classes are often quite different, as shown in the example. Most common mixed type audio fits into the category of environmental sound recognition (ESR).

3.3 Environmental sound recognition

The term, *environmental sounds* is used to describe the sorts of sounds that are often in the background of other recordings. They can be artificial or made by humans and include quotidian things like rivers flowing, house fans, rustling leaves, rain, vehicle noises, etc. A survey from 2014 [7] described solutions for these problems 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, *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, or cepstral (which uses an inverse Fourier transformation of the logarithm of a spectrum), features to work best to classify these sounds. A particularly popular feature is the mel-frequency cepstral coefficient (MFCC). Mel-frequency is the preferred range among researchers in our problem domain because of its conceptual similarities with how humans are thought to perceive audio. Also, the result of a Fourier transformation is the preferred way to analyse the audio in many different domains. Environmental sounds can also be described as *non-stationary*, where the *stationarity* assumption mentioned above is violated. The techniques to classify these sounds are more sophisticated:

1. wavelet-based techniques are now decades-old and use a representation of audio in time and frequency space using short waves, instead of the now-popular Fourier transform-based methods which decompose the audio

2. matching pursuit (MP)-based techniques use sparse signal representations with overcomplete dictionaries. MP features reveal patterns at larger time-frequencies and so can be used in conjunction with MFCC for non-stationary sound

3. spectrum-based methods utilise the Fourier transformation, which separates the frequency components of (in our case) audio

Spectrum-based methods have been shown to consistently outperform MFCC on its own and be comparable to a particular variant of MP. A potential justification that the survey gave for systems which used neural networks to learn spectral patterns not being the highest-performing was due to the spectrogram being cursed by high dimensionality. However, since this survey was conducted in 2014, it did not account for later advancements in deep network training, such as the residual learning framework [19], which led to the proliferation of deep pretrained networks which can deal high-dimensionality data by compressing it through many convolutional layers.

3.4 Bird call recognition in soundscapes

The majority of recent work to recognise bird calls in *soundscapes* is based on spectra. A general workflow is to separate audio into segments (BirdCLEF2021 required 5-second segments to be classified), use short term Fourier transform (STFT) to transform that segment into a spectrogram using appropriate parameters, apply image transformations/filtering/augmentations to the spectrogram, use some kind of event detection system to find groups of pixels that meet some structural requirements, and then classify the group of pixels into a bird species class. CNNs are usually incorporated due to their ability to learn the last two stages with high accuracy. A standard use of them would take the processed spectrogram an X*Y*C image and, after convolution and pooling, output a a 1*1*N channel image, where X,Y is the image size, C is the number of channels, and N is the number of classes to predict. Modern approaches usually use ensembles of high-performing CNNs to make such predictions and aggregate the results.

As an alternative to CNNs, vision transformers (ViT), which have been largely used for natural language processing have recently been applied to image recognition [14]. In ViT, a *vanilla/pure* transformer yields classifications after direct application to a spectrogram. ViT are advantageous because they use a lot less computational resources than CNNs during inference and are more appropriate in principle than CNNs, because the latter 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 [24] claimed that ViT are able to be somewhat competitive with CNNs specifically because the former do not learn pitch-invariant patterns. However, the top 10 solutions for BirdCLEF2021 all used CNNs.

It is difficult with the above methods to consider the temporal nature of *soundscape* 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 3.2, RNNs have been used to utilise this temporal information.

In *soundscapes*, different birds can often be heard calling simultaneously. A good classifier must therefore be able to list all these birds. One way to teach this to neural networks is with Mix Up, where examples from different classes are blended together to create new examples with labels from their constituents. Synthetic (e.g., *pink* or *brown*) noise can also be introduced.

3.5 Problem-specific challenges

The *nocall* class was highly imbalanced compared to all the other classes (bird species), occurring in 63.7% of the 5-second training segments. To decrease this imbalance, teams have used audio from the *Rain Forest* [5], *freefield1010* [6], *Chernobyl BiVA* [7] and *BirdVox-full-night* [8] datasets as extra *nocall* examples. The *Rain Forest* dataset was particularly beneficial because of how many sounds that were not birds but sounded like birds (e.g., frogs) it introduced.

The training stage is difficult because when dividing a given *focal* into segments for the generation of spectra, not every segment will contain calls from the primarily-/secondarily-labelled birds, or indeed may contain calls from other birds. When selecting the size of a spectrogram segment, or how much a *focal* ought to be trimmed, the general 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 would be to globally pool all the segments for an entire *focal* recording 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 in each *focal*) and max (which would be like saying that a bird only calls in a single 5-second segment in each *focal*). The teams dealt with the potential lack of a call in a segment in various ways; some manually selected segments where the target bird was present, some used stochastic sub-sampling methods, and some [31] only used the results of training on these troublesome examples as "pseudo-labels" to be processed during later stages into "cleaned labels". The last part can be called a "self-distillation approach" or "multistep train-segment-shift training procedure". It regularises predictive uncertainty, similar to label smoothing, and thereby increases predictive diversity without requiring another model to act as a teacher [45].

[5] `https://kaggle.com/c/rfcx-species-audio-detection`

[6] `https://archive.org/details/freefield1010`

[7] `https://catalogue.ceh.ac.uk/documents/be5639e9-75e9-4aa3-afdd-65ba80352591`

[8] `https://wp.nyu.edu/birdvox/birdvox-full-night/`

One example of the domain mismatch between training and testing data 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*. The Mix Up techniques described in the previous section can help to address this. Another 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 in forests. A team from the 2020 competition [9] proposed to to reduce the volume of high frequencies to address this. Overall, the recordists' intentions for both types of audio are different: the Xeno Canto contributors are trying to provide the cleanest examples of particular species' calls, whereas the *soundscape* data is intended to be non-specific to particular birds.

Many teams found it beneficial to develop different classifiers for different stages of their workflow. To reduce false positives and false negatives, one team [30] had a stage trained specifically to distinguish between segments with calls and those without. To ensure that long- and short-term information is properly utilised, another team [23] 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*. The latter team also commented explicitly on their ensembling technique which many other teams used: they found that averaging the logits from the networks and then thresholding performed better than the other way around.

Regarding how to utilise the location metadata which is provided with the *focals*, some teams totally disregarded it, others developed handcrafted systems to, e.g., weight predictions which were associated with sites closer (according to haversine distance for spheres) to the testing site proportionally higher than than those that were far away [23], while others included the information as feature(s) in machine learning systems [34].

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), 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 the did when used in different amounts across ensembled networks, perhaps because the networks themselves became more diverse and therefore more robust in aggregate.

Some exemplar post-processing steps were: increasing prediction probability according to a particular species' prevalence in the training data, distilling labels using smoothing (the labelling errors which the humans made can be seen from a beneficial perspective as a kind of regularisation), and weighting predictions according to location/time-of-year metadata.

4 Preliminary analysis of the data

The competitors in BirdCLEF2021 were given two datasets for training/validation, one containing *focals* and another containing *soundscapes*. The hidden test dataset consisted only in *soundscapes*. We define a *focal* (not the terminology used by the competition organisers) to be a seconds-to-minutes long audio file 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. There are 62,874 *focals* with metadata, such as latitude, longitude and time of recording, available for training. The labels are provided through "crowd sourcing" on a platform called Xeno Canto[10] in which amateurs and professionals contribute through a system

[9] `https://www.kaggle.com/c/birdsong-recognition/discussion/183269`

[10] `https://xeno-canto.org/`

of tagging, discussing, flagging and reviewing. They are weak because the label is only at the audio file level, without timestamps of when exactly the call can be heard. In Figure 6, a world map is plotted with the latitudinal and longitudinal information to reveal the locations where most of the Xeno Canto submissions were made. There is a bias towards the Americas and Europe (compared to Africa and Asia, for example). However, all the *soundscape* recordings came from the Americas, so the data may be beneficial for this problem but not for an unbiased, global bird species predictor. There are 397 unique classes to classify, and 264 of them have at least 100 examples. With preliminary intuition, we may think that having at least 100 examples of each class for this data would be sufficient enough to learn to distinguish them. However, approximately one third of the classes have this number of examples. Datasets like this are often referred to as "long-tailed" because there is a class imbalance (relatively fewer examples of some of the classes) which can be visualised as a long tail, as shown in Figure 7. Research has shown that it is harder to achieve high performance on such problems [42].

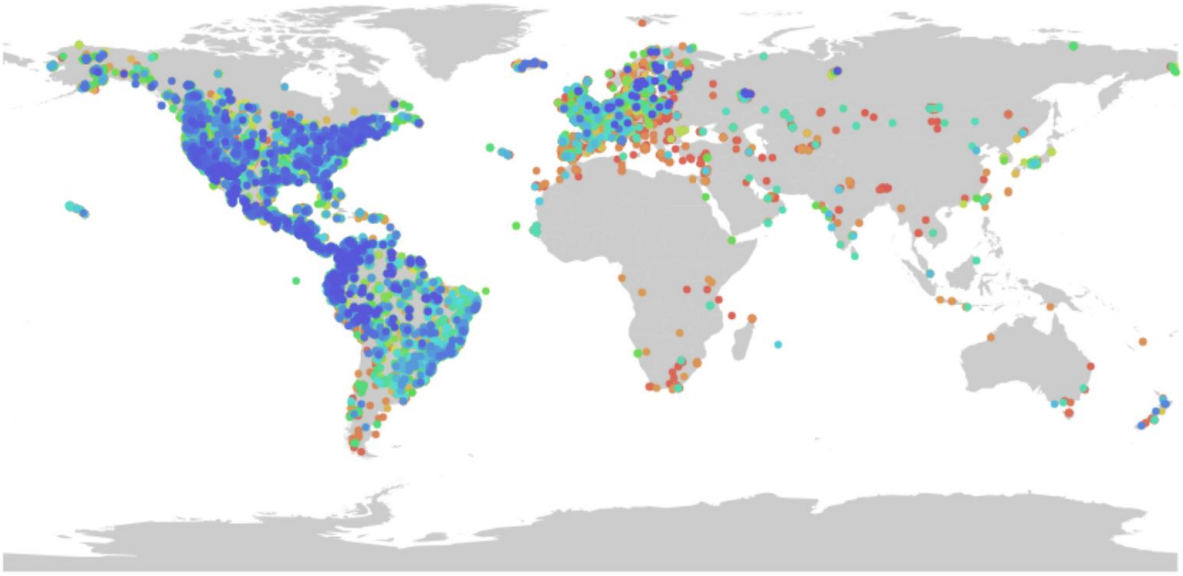


Figure 6: Each unique colour represents a unique species' recording location, taken from the latitudinal and longitudinal information with which each *focal* is tagged. Note that there are therefore 397 unique colours, but some are just slightly different shades of others.

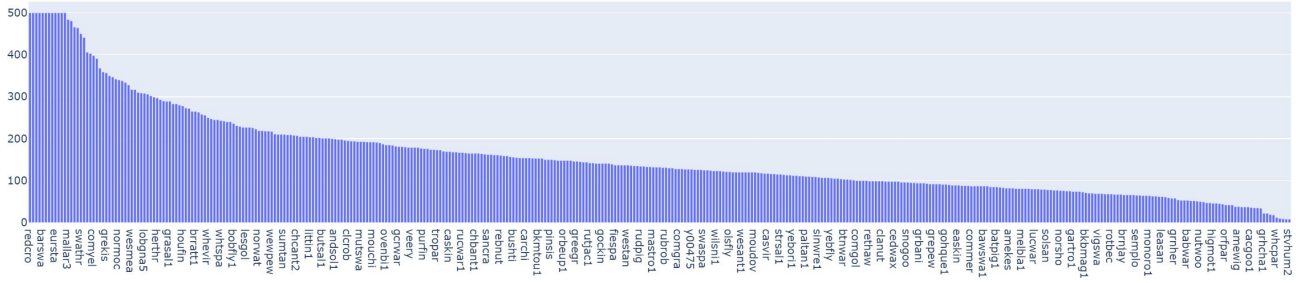


Figure 7: The number of *focals* per species, with some labels on the x-axis ommitted due to space constraints. Note that the competition organisers decided to cap these numbers at 500, because it is thought that having more than 500 class examples will not improve the score in this problem.

We define a *soundscape* to be a 10-minute long audio file in which different bird calls can be heard, sometimes with significant natural (e.g., rain) and/or anthropogenic (e.g., car)

background noise. The labels are quite strong since they are provided by The Cornell Lab of Ornithology. There are 20 *soundscapes* available for training, each with a list of bird species that can be heard (or *nocall* if there are none) in every 5-second segment. The intention may be for teams to use this as validation data, but a minority of the high-performing teams used it during training as well. The hidden testing data consists only in approximately 80 *soundscapes*. Therefore, there is a "domain mismatch" between the training and testing data since the audio data is drawn from different contexts. There are 397 species, but this is a 398-class, multi-label classification problem due to the special case that the *nocall* class is only predicted when all other classes are not predicted. The final system should receive as input a 5-second audio segment and return a list of the birds whose calls can be heard in that segment, possibly based on an internal 397-dimensional probability vector with threshold(s). Since there could be many birds calling simultaneously, we would like to be able to return a list of zero or more labels. Note, however, that we will actually always return one or more labels, because of the special label called *nocall*, representing the absence of bird calls. The *soundscape* data was 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 small amount of *soundscapes* that were provided for training/validation only contained examples from COR and SSW. However, as is shown in Figure 6, the *focal* data came from Xeno Canto users recording all over the world.

It is not straightforward to know how to use the secondary labels provided in the metadata of the *focals* during training. The recordists' intentions were to capture the primary species, so its calls will be very dominant. 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 12 secondary labels that appear in at least 1% of them.

Extra analysis which is not entirely relevant to our proposed solutions is also available[11].

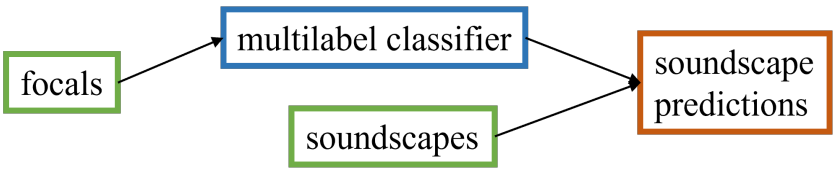
5 Solution Architecture

The presented software shall be a series of Jupyter notebooks and regular Python scripts which are compatible with recent versions of the Kaggle computing environment as defined in their Dockerfile[12]. They constitute machine learning pipelines which can be submitted to BirdCLEF2021 on Kaggle. We present two such pipelines, one to function as a simple baseline and one which is intended to be highly performant. The diagrams in Figure 8 show their proposed architectures. Many of the ideas and code which led to these architectural designs incorporate insight from the literature review and other publicly-available competition submissions. In particular, the Jupyter notebooks of the following Kaggle users were particularly helpful: `https://www.kaggle.com/{startjapan, kneroma, yasufuminakama, kami634, hidehisaarai1213, cpmpml}`.

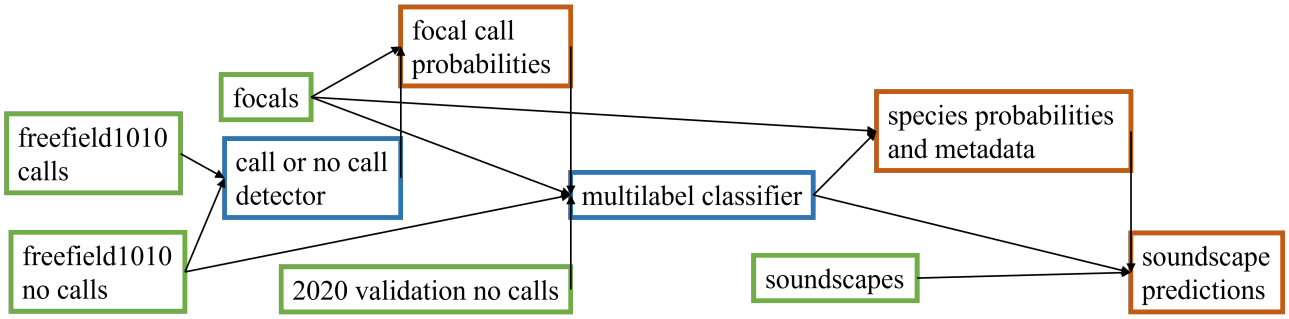
5.1 Baseline solution

The intention with the baseline solution is to simplify all the techniques we have reviewed into a minimalist set of operations which result in decent performance. The provided *focals* audio is

[11] `https://projects.cs.nott.ac.uk/psykjm/dissertation-public/-/blob/main/tangential-analysis.ipynb`
[12] `https://github.com/Kaggle/docker-python`



(a) A proposed baseline solution architecture.



(b) A proposed performant solution architecture.

Figure 8: Diagrammatic representations of the data processing and machine learning pipelines to make predictions on *soundscapes*. Datasets are boxed in green, models are boxed in blue and output tables are boxed in orange.

converted into melspecs and then used to train a deep, residual, convolutional neural network classifier. To justify this choice of model, it will be:

- deep because the patterns of the calls in the melspecs are fine-grained and detailed (therefore requiring a lot of layers to extract high-level features)
- residual because this will help the deep network to converge faster and not suffer from vanishing/exploding gradients
- convolutional because the melspec pixels constitute a large feature vector which should be reduced to enable feasible learning.

The *soundscapes* given for testing will be segmented and transformed into melspecs and fed into this classifier. The resulting probabilities will be thresholded to produce a binary classification for each of the 398 classes.

5.2 Performant solution

The intention with the performant solution is to incorporate more datasets and techniques to achieve high performance on the Kaggle leaderboards. The proposed process can be described in six phases:

1. Convert all the audio we will use into melspecs (images): the three main audio repositories to convert are:
   (a) the *freefield1010* data for use in Phase 2 to build the *call or nocall* discriminator, which will make predictions on the *focals* in Phase 3 to re-weight their probabilities from Phase 4
   (b) the *train_short_audio* data (referred to in this paper as *focals*) containing call examples of all the bird species relevant to the competition. It is used in Phase 4 to help the multilabel classifier learn the calls of the different species

(c) the validation data from BirdCLEF2020 (the previous year's competition) which is only tagged with *nocall*. It is used to enhance the training of the multilabel classifier in Phase 4 by providing more examples of what natural environments without bird calls sound like

2. Learn to discriminate the presence of calls: The *freefield1010* dataset[13] contains audio clips from the Freesound audio archive[14] where each clip has at least the "field-recording" tag. Out of these, we separate those that are tagged with "birds" from those that are not. Thus is constructed a binary classification dataset with which to train a classifier to distinguish the special *nocall* class from audio in which bird calls (or other bird sounds) can be heard. The reason for this is to augment the next phase with an "expert" on the label that is disproportionately dominant in the BirdCLEF2021 competition, *nocall*. The result of this phase is a model which is trained to classify such data.

3. Predict whether *focals* have calls: the *call or nocall* discriminator created in Phase 2 is used to produce likelihoods for whether each *focal* segment has a bird call in it. The output is a table with one row per *focal* file. The *nocalldetection* column will contain a space-delimited list of probabilities for each segment as predicted on its corresponding melspec, where higher values indicate higher chances of there being a call. These probabilities will weight the predictions made in Phase 2 to improve their accuracy. The intention is to improve erroneous probabilities from the multilabel classifier by incorporating the expertise of the binary label classifier.

4. Train the multilabel classifier using a combination of datasources: the melspecs of the *focals* generated in Phase 1, the *nocall* examples from the *freefield1010* and BirdCLEF2020 validation datasets, and the list of probabilities for whether each *focal* segment contains *nocall* which was generated in Phase 3. The reason for combining all these datasources is to mitigate class imbalance by giving the "super class" of all 397 bird classes more *nocall* counter-examples.

5. Predict each species' probability: the multilabel classifier model trained in Phase 4 is used to predict which species are present in segments of *focals*. We output these results in a table where each row represents a single segment from a *focal*. Tables will be maintained for each model and model configuration that made the predictions. 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 *call or nocall* discriminator from Phase 2.

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 reason for including the probabilities is that we want to incorporate information about the popularity of various birds to weight down the unpopular ones. 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 reason for including the metadata is so that the latitudinal and longitudinal information from the *focals* can help to make decisions when encountering the *site* information from the *soundscapes*.

[13] `https://archive.org/details/freefield1010`
[14] `https://freesound.org/browse/tags/`

6 Experimental Framework

In multilabel classification, we encounter the notion of partial correctness when some labels for a datapoint are correctly predicted but others are not. Additionally, since the training labels are divided into a single *primary* label and multiple *secondary* labels, we could also use metrics suited for labels with rankings, such as *one error*, which counts how many times the label predicted to be the top-ranked is not present in the true label set [41]. The metric by which the competition is scored is the micro-averaged F1-score. F-score is way of blending the precision and recall measurements, giving equal importance to both. If a model has more false positives than true positives, then precision is worse. If a model has more false negatives than true positives, then recall is worse. F1-score is usually applied to binary classification problems, but there are two popular ways to aggregate multiple F1-scores into a single metric for problems with more than two classes:

1. macro-averaged F1-score, which takes the simple average of all F1-scores for each class and is therefore more useful in datasets with more uniform class distributions because the importance of each class is considered equally

2. micro-averaged F1-score, which accounts for the proportional contributions of all classes by summing the true positives, false positives and false negatives for the entire system and is therefore more appropriate for imbalanced datasets, like the one in this competition

One team [34] used an interesting technique exploit the workings of the micro-averaged F1-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. They did this because according to the F1-score formula, a datapoint with only one correct label will be given no score if that label is not predicted but a datapoint with multiple correct labels will give a non-zero score if at least one of them was predicted. Assuming that only one label was missed in the latter case, we see that the penalty for missing a single label is different according to the number of expected (correct) labels. Since *nocall* is a popular label which only appears on its own, it makes strategic sense to include it as a prediction more often. This helps in preventing some zero scores for datapoints that are labelled with *nocall*. Since this project was started after the competition ended, we had access to the "Private Leaderboard" on Kaggle, which introduced bias to our work because we could now optimise our solution to perform well on that Private Leaderboard. The reason for withholding the performance of solutions on the Private Leaderboard until the end of the competition is to prevent teams from overfitting to the testing data. We are therefore not checking our solution performance on the Private Leaderboard "too often"; instead relying mainly on local cross-validation and micro-averaged F1-score calculations using the training and validation data.

7 Progress update

Out of the five tasks which were set out in the Work Plan from the Dissertation Proposal to be done before the December break, four have been achieved, mainly in the desired order:

- ☑ 18 Oct. to 01 Nov.: Run competition solutions on cluster
- ☑ 18 Oct. to 15 Nov.: Craft pre-processing system
- ☑ 18 Oct. to 15 Nov.: Utilise additional datasources

☐ 01 Nov. to 13 Dec.: Beat 1st place solution from competition

☑ 29 Nov. to 13 Dec.: Write Interim Report

Although achieving a better score than the best from the competition was not achieved, an architecture which should perform well in theory has been detailed in Section 5.2. Also, a first version of the baseline solution has been implemented and achieves a Private Leaderboard score of 0.5511.

Part of the reason I chose a Kaggle project was to get experience interacting with the Kaggle platform: I have become proficient in utilising the free GPU computation time to perform experiments and make submissions, with 11 successful submissions made so far. An unexpectedly fun part of this semester's work was setting up University's Computer Vision Laboratory's GPU Cluster to run Jupyter Notebooks interactively. I received some one-on-one support to accomplish this and contributed a list of detailed instructions for how to set it up so that future users can easily accomplish it too. There was a delay in getting access to the necessary resources on the cluster, but I did not lose out on working time because there was plenty of research to do while I waited. I suspect that a lot of parameter tuning will be required to outperform the current best solutions, and access to the cluster will be useful for that. The research, development and writing process has gone largely as expected and I am satisfied with the approaches I have taken to complete the work. A challenge has been understanding some of the top competition solutions. In particular, the team who won did not expect to win and have said that the explanation of their solution was therefore rushed. In addition, their code is documented almost entirely in Japanese, and I think the translations I have made using Google Translate are not perfect. I have found that many teams reused each other's code (which some continually published during the competition), and I suspect some of them did it without fully understanding what they were reusing, so there are some conceptual inconsistencies which also make it hard to understand many solutions. There is a risk that I will not be able to develop a solution which outperforms the best from the competition; if it seems like this is going to be the case next semester, I will look towards other outcomes, like developing a system which can perform efficient inference on mobile devices.

References

[1] S. Albawi, T. A. Mohammed, and S. Al-Zawi. Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6, 2017.

[2] A. Bagnall, J. Lines, and A. Bostrom. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.

[3] A. Bagnall, J. Lines, and J. Hills. Time-Series Classification with COTE: The Collective of Transformation-Based Ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 2015.

[4] D. Bahdanau, K. Cho, and Y. Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *ICLR 2015*, 2016.

[5] E. Cakir, G. Parascandolo, and T. Heittola. Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 25(6):1291–1303, 2017.

[6] R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 161–168, Pittsburgh, Pennsylvania, USA, 2006. Association for Computing Machinery.

[7] S. Chachada and C.-C. J. Kuo. Environmental sound recognition: a survey. *APSIPA Transactions on Signal and Information Processing*, 3:e14, 2014.

[8] C. Chatfield. Time-series forecasting. *Significance*, 2(3):131–133, 2005.

[9] L. Chen, S. Gunduz, and M. T. Ozsu. Mixed Type Audio Classification with Support Vector Machine. In *2006 IEEE International Conference on Multimedia and Expo*, pages 781–784, 2006. ISSN: 1945-788X.

[10] P. Cunningham, M. Cord, and S. J. Delany. Supervised Learning. In M. Cord and P. Cunningham, editors, *Machine Learning Techniques for Multimedia: Case Studies on Organization and Retrieval*, Cognitive Technologies, pages 21–49. Springer, Berlin, Heidelberg, 2008.

[11] A. Dandashi and J. AlJaam. A Survey on Audio Content-Based Classification. In *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 408–413, 2017.

[12] J. Dennis, H. D. Tran, and E. S. Chng. Overlapping sound event recognition using local spectrogram features and the generalised hough transform. *Pattern Recognition Letters*, 34(9):1085–1093, 2013.

[13] T. G. Dietterich. Ensemble Methods in Machine Learning. In *Multiple Classifier Systems*, Lecture Notes in Computer Science, pages 1–15, Berlin, Heidelberg, 2000. Springer.

[14] A. Dosovitskiy, L. Beyer, and A. Kolesnikov. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *CoRR*, 2021.

[15] K. Drossos, S. I. Mimilakis, and S. Gharib. Sound Event Detection with Depthwise Separable and Dilated Convolutions. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, Glasgow, UK, 2020. IEEE.

[16] P. Duboue. *The art of feature engineering: essentials for machine learning*. Cambridge University Press, Cambridge ; New York, NY, first edition edition, 2020.

[17] F. Gouyon, F. Pachet, and O. Delerue. 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*, 2000.

[18] H. Habibi Aghdam and E. Jahani Heravi. *Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification*. Springer International Publishing : Imprint: Springer, Cham, 1st ed. 2017 edition, 2017.

[19] K. He, X. Zhang, and S. Ren. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[20] X. He, K. Zhao, and X. Chu. AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212:106622, 2021.

[21] H. Ismail Fawaz, G. Forestier, and J. Weber. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019.

[22] L. C. Jain. *Recurrent neural networks: design and applications*. CRC Press, Boca Raton, FL, 2000. OCLC: 60557840.

[23] Jan Schlüter. Learning to Monitor Birdcalls From Weakly-Labeled Focused Recordings. *CEUR Workshop Proceedings*, 2936(CLEF 2021 Working Notes.), 2021.

[24] Jean-Francois Puget. STFT Transformers for Bird Song Recognition. *CEUR Workshop Proceedings*, 2936(CLEF 2021 Working Notes.), 2021.

[25] A. Joly, H. Goëau, and S. Kahl. Overview of LifeCLEF 2021: an evaluation of Machine-Learning based Species Identification and Species Distribution Prediction. *LifeCLEF 2021 Workshop*, 2021.

[26] S. Kahl, C. M. Wood, and M. Eibl. BirdNET: A deep learning solution for avian diversity monitoring. *Ecological Informatics*, 61:101236, Mar. 2021.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *Commun. ACM*, 2012.

[28] J. Lines, S. Taylor, and A. Bagnall. Time Series Classification with HIVE-COTE: The Hierarchical Vote Collective of Transformation-Based Ensembles. *ACM Transactions on Knowledge Discovery from Data*, 12(5):52:1–52:35, 2018.

[29] D. Lowe. Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 1999.

[30] Marcos V. Conde, Kumar Shubham, and Prateek Agnihotri. Weakly-Supervised Classification and Detection of bird Sounds in the Wild. A BirdCLEF 2021 Solution. *CEUR Workshop Proceedings*, 2936(CLEF 2021 Working Notes.), 2021.

[31] Maxim V. Shugaev, Naoya Tanahashi, and Philip Dhingra. BirdCLEF 2021: building a birdcall segmentation model based on weak labels. *CEUR Workshop Proceedings*, 2936(CLEF 2021 Working Notes.), 2021.

[32] B. C. McComb. *Monitoring animal populations and their habitats: a practitioner's guide.* CRC Press, Boca Raton, 2010. OCLC: 839658581.

[33] M. Müller. Dynamic Time Warping. In *Information Retrieval for Music and Motion*, pages 69–84. Springer, Berlin, Heidelberg, 2007.

[34] Naoki Murakami, Hajime Tanaka, and Masataka Nishimori. Birdcall Identification Using CNN and Gradient Boosting Decision Trees with Weak and Noisy Supervision. *CEUR Workshop Proceedings*, 2936(CLEF 2021 Working Notes.), 2021.

[35] R. Prabhavalkar, O. Alsharif, and A. Bruguier. On the compression of recurrent neural networks with an application to LVCSR acoustic modeling for embedded speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5970–5974, Shanghai, China, 2016. IEEE Press.

[36] J. Qin, W. Pan, and X. Xiang. A biological image classification method based on improved CNN. *Ecological Informatics*, 58:101093, 2020.

[37] J. O. Smith. *Spectral audio signal processing.* Stanford University, CCRMA, Stanford, Calif, 2011.

[38] M. A. Tabak, M. S. Norouzzadeh, and D. W. Wolfson. Machine learning to classify animal species in camera trap images: Applications in ecology. *Methods in Ecology and Evolution*, 10(4):585–590, 2019.

[39] S. A. K. Tareen and Z. Saleem. A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. In *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pages 1–10, 2018.

[40] G. Tsoumakas and I. Katakis. Multi-Label Classification: An Overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3):1–13, 2007.

[41] G. Tsoumakas, I. Katakis, and I. Vlahavas. Mining Multi-label Data. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer US, Boston, MA, 2010.

[42] T. Wang, Y. Li, and B. Kang. The Devil Is in Classification: A Simple Framework for Long-Tail Instance Segmentation. In *Computer Vision – ECCV 2020*, pages 728–744, Cham, 2020. Springer International Publishing.

[43] L. Ye and E. Keogh. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 947–956, Paris, France, 2009. Association for Computing Machinery.

[44] C. Zhang, C. Liu, and X. Zhang. An up-to-date comparison of state-of-the-art classification algorithms. *Expert Systems with Applications*, 82:128–150, 2017.

[45] Z. Zhang and M. Sabuncu. Self-Distillation as Instance-Specific Label Smoothing. *NeurIPS*, 2020.