Winging It: A Comparative Machine Learning Approach to Bird Sound Classification

Exam Paper for Data Mining, Machine Learning, and Deep Learning

Students:

Ghisini, Tommaso 158389
Mealor, Alexander 158801
Ries, Alexander 158292
Torp, Aleksander 158277

Programme: MSc Business Administration and Data Science
Course Code: CDSCO1004E
Submission Date: 19.05.23
Nr. of Characters: 34 098 characters
Nr. of Pages: 15 pages

Table of Contents

Link to the Jupyter Notebook and Data:

https://drive.google.com/drive/folders/1OKWG4V5kW3-e44b3XfttJ7LVBqO4eUF_?usp=sharing

Backup link:

https://1drv.ms/f/s!AiqIYDThiBNP4WLLT5VrUad6RpTG?e=1l5ab9

Abstract

The monitoring of bird species has the potential for meaningful application far beyond avian conservation efforts, including - but not limited to - environmental change and land usage. Historically onerous tracking efforts have become far less burdensome with the growth of crowd-sourced bird audio recordings and the ability to use this data for species classification. In this paper we address the problem of how such classification efforts can be best performed by applying machine learning techniques. Utilizing the BirdCLEF 2023 dataset, we implement three different models - a Random Forest Classifier (RF), a Convolutional Neural Network (CNN) and a Recurrent Convolutional Neural Network (RCNN) - that build upon audio classification tasks in an evolutionary way. Two different preprocessing pipelines are employed in order to explore different spatial and temporal aspects of audio data. We find that neural networks outperform traditional models with the CNN achieving an accuracy of 0.62. The RCNN performs even more strongly across pipelines on average, suggesting that the temporal features of audio should be utilized in machine learning techniques and particularly when audio is lengthier. Imbalanced data, however, remains a key challenge and requires addressing carefully in bird sound classification preprocessing pipelines. We recommend practitioners experiment with different augmentation techniques and handle noise and silence appropriately for optimal results.

Keywords: Bird audio classification, Convolutional Neural Network, Random Forest, Recurrent Convolutional Neural Network, Bioacoustics

Introduction

The monitoring of individual bird species and broader populations, beyond directly aiding avian conservation efforts, has the potential for meaningful application across many important related spheres. Observably inhabiting many environments, birds are a suitable target for monitoring purposes as both primary and proxy for other species. Environmental change, in particular, can be observed in ways that standard physical or chemical measurement methods cannot capture, with ‘biological, often cumulative and non-linear consequences of many environmental changes acting simultaneously’ (Koskimies, 1989, p. 153). Land planners, governments and business, who increasingly must consider the impact of their resource and land usage, may quantify these effects via ‘large scale, accurate bird recognition’ (Sprengel et al., 2016, p. 547).

The growth of publicly available, crowd-sourced sound recordings of bird species - a shift away from manual documentation by avian experts alone - combined with their distinct vocal activities has rendered audio monitoring a meaningful method of tracking populations via classification. Such crowd-sourcing often provides labeled data en-masse, with enthusiasts categorizing recordings by species. Scaling such monitoring, however, requires the ability to automate the classification process and handle in-field recordings with hurdles such as variable length, environmental noise, and often mixed quality.

As training data has grown in volume, machine and deep learning models - with their capacity to learn audio features - have been key in this automation process. This paper explores the evolution between three methods - RF (serving as our benchmark), CNN and RCNN - in classifying bird species by their vocalizations. This multiclass classification has multiple use cases. First, it can be used to identify ranges of different species in environmental recordings, allowing ecologists to track changes in biodiversity over time. Second, it can be used to proxy individual population sizes in areas by summing classification instances, measuring these trends against land and resource usage. We imagine land planners using such tools to evaluate their decision-making process.

We thus intend to answer the following research question -

How can machine learning algorithms be best utilized to differentiate between bird species based on audio data?

Related Work

The interest for developing mechanisms for classifying bird species has increased, and in recent years more focus has been paid to bird classification in literature relating to problems such as the one in the BirdCLEF competitions. The datasets in these papers range from audio clips on tens up to hundreds of bird species and thousands of hours of sound data. We identify three overarching commonalities in literature; preprocessing data, extracting features, and constructing models.

Preprocessing is a crucial step, usually consisting of several components in varying order. This includes some combination of segmenting, padding, resampling data, normalizing values, dealing with noise, augmentation and class imbalances, as data often comes in varying quality, lengths and sample rates, and since a uniform length is required by deep learning models. The majority split the samples into either three seconds (Incze et al., 2018; Kahl et al., 2021; Madhavi & Pamnani, 2018; Carvalho and Gomes, 2022) or five seconds (Mühling et al., 2020; Piczak, 2016; Kahl et al., 2017) and pad shorter segments with zeroes. Others experiment with repeating signals rather than padding, reporting promising results, although on smaller datasets (Xie et al., 2019; Grill & Schluter, 2017). Resampling is done, in combination with normalization, for reducing data size, down to 22 050 Hz (Carvalho & Gomes, 2022; Mühling et al., 2020; Madhavi and Pamnani, 2018) or to unify and preserve details at 44100 Hz (Incze et al., 2018; Piczak, 2016; Grill & Schluter, 2017). Normalization is also performed before extracting features (Mühling et al., 2020; Carvalho & Gomes, 2022; Madhavi & Pamnani, 2018), afterward (Grill & Schluter, 2017; Piczak, 2016) or both before and after (Incze et al., 2018). Denoising is emphasized by some as an important step, as the environment in which the recordings are made often contain unrelated sounds. Methods involve either applying filters to all the segments removing parts below a threshold (Carvalho & Gomes, 2022; Piczak, 2016), separating into chunks that fall below given thresholds and are either removed or used as basis for augmentation data (Incze et al., 2018;

Mühling et al., 2020; Madhavi & Pamnani, 2018; Kahl et al., 2017). Finally, various methods are utilized for oversampling and augmentation, both for dealing with imbalanced classes and for generalization purposes outside training environments. Augmentation techniques for audio include shifting pitches up or down, masking random parts of the extracted features with zeros, adding white noise or previously filtered noisy segments, time shifting randomly changing loudness, or randomly combine segments by overlapping them (Mühling et al., 2020; Xie et al., 2019; Kahl et al., 2017; Kahl et al., 2021; Grill and Schluter, 2017). The methods vary being applied randomly or sequentially, and helps to some extent combat a common issue of imbalance between classes.

Concerning extracting features - essentially the time frequency representations of audio - there are primarily two representations that are prevalent in this context, mel frequency cepstral coefficients (MFCC) and Mel-Spectrograms (MS). Both involve transforming a signal by either applying discrete cosine or Fourier transformations, respectively. Previously, MFCCs were more commonly used, partially for their ability to reduce dimensionality of the featurespace (Kahl et al., 2021), however MS have become more common due to their overall stronger performance, capability of characterizing acoustic components, and widespread usage (Xie et al., 2019; Carvalho & Gomes, 2022; Kahl et al., 2017; Kahl et al., 2021; Piczak, 2016; Grill and Schluter, 2017).

The last step is constructing models and measuring performance. Historically, the approach for tackling classification of birds has relied on using various classifiers, such as support vector machines, random forests, and clustering techniques. The relative drawback of these methods included being forced to reduce the training sets and computational costs, which has been omitted by advancements in deep learning networks that now provide superior performance (Kahl et al., 2017; Kahl et al., 2021). The breakthrough for CNNs in this domain came in a submission to the 2016 BirdCLEF competition, and since then neural networks have become de facto standard (Incze et al., 2018). Since then, various research has explored different architectures and compositions of layers, such as LSTM, recurrent CNN, RCNN (combining convolutional and recurrent networks), and residual networks, yielding promising results (Madhavi & Pamnani, 2018; Kahl et al., 2021; Carvalho & Gomes, 2022) along with a tendency of moving towards ensemble approaches (Kahl et al., 2017; Xie et al., 2019). However, directly comparing the results across research in this domain is difficult, due to a lack of a standard benchmark, data, and metrics, as pointed out by Kahl et al. (2021). Nevertheless, there are some similarities worth noting, such as a using categorical cross entropy as loss function (Xie et al., 2019; Piczak, 2016; Madhavi and Pamnani, 2018), optimizing with ADAM (Xie et al., 2019; Kahl et al., 2017; Grill and Schluter, 2017; Kahl et al., 2021; Mühling et al., 2020) and dynamically decreasing the learning rate (Kahl et al., 2017; Kahl et al., 2021; Grill & Schluter, 2017; Incze et al., 2018).

Methodology

Dataset Description

The data is retrieved from BirdCLEF 2023 competition on Kaggle, (Kaggle, 2023), initiated by Cornell Lab of Ornithology. The competition involves developing a solution for identifying which birds are present in long recordings, as to monitor bird populations. The dataset consists of 5.3 GB sound recordings on 264 species, with a total of 16941 samples, a 10 minutes long test file, a bird taxonomy overview, and metadata on the training files. The majority of the audio samples are taken from xeno-canto.org, a platform and collaborative project for sharing, annotating, and rating wildlife sounds by 1082 unique authors. The aim of this paper however, is to develop a model to classify the sound of one bird given a sample, and we therefore only use the training audio files along with the corresponding metadata. The metadata consists of twelve columns with various data on each record, 16941 samples and 264 species, of which the primary labels, secondary labels, ratings of sound clip quality, and filenames are of most interest.

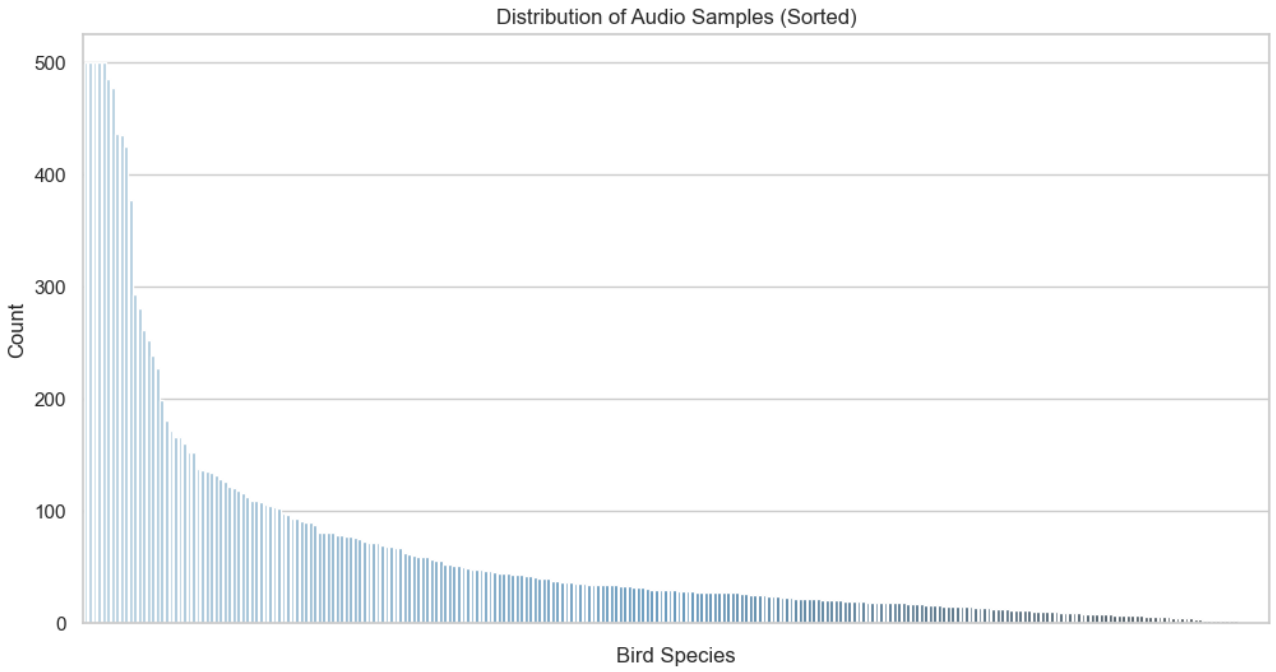


Figure 1: Distribution of Number of Audio Samples Per Species

Primary label pertains to the main bird type associated with a recording. The distribution of recordings per species is highly skewed, ranging from 1 to 500, figure 1, with a mean of 64.17 and standard deviation of 98.75, and 75% having 14 or more samples. Secondary labels, additional species associated with a sample as annotated by the recordist, pertains to 2305 samples (13.61%) having 2 to 10 identified species in them that vary between 195 different bird labels. However, no value registered in this column does not necessarily equal that there are no additional

species in the recordings, and without information on the accuracy of the recorder, they cannot be excluded. In many cases the annotator also registered the type of bird sounds, a set of types which is very heterogeneous and consists of 399 recognized types. The quality of the audio clips are rated on a scale from 0.0 to 5.0, with 5 being the best quality and 0 indicating that the sample has not received any rating yet. With the majority of the samples rated 3 or above, and without knowledge about the unrated clips, no samples are excluded.

Data Analysis Process

The general steps in the process for analyzing the data consists of six stages and two phases, as outlined in figure 2. Audio data first has to be converted and preprocessed into the right format, the different classes have to be augmented where necessary, and then features representing the acoustic information as images are generated. Thereafter, we construct a baseline model along with models of interest, train and validate them, and finally compare how they perform.

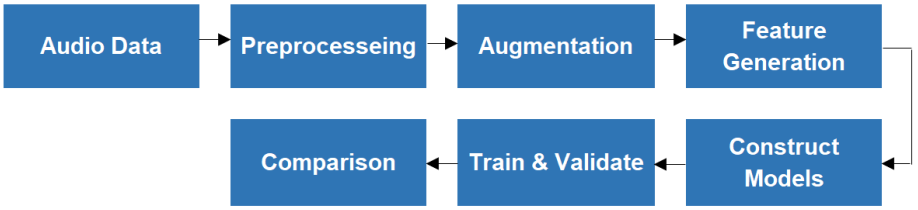


Figure 2: Overview of the Steps in the Data Analysis Process

Data Pre-Processing

Preprocessing of the BirdCLEF data is a crucial step, as we are required to extract the necessary features from the diverse and large raw audio provided. The preprocessing is inspired by common practices in literature. Additionally, multiple steps undergo an iterative testing process in which accuracy between a simple preprocessing pipeline (downsampling and trimming) and the additional method(s) are compared via a *testing model*. In line with typical approaches to audio classification, we utilize a simple CNN as a testing model with which to optimize our preprocessing pipeline. Two convolutional layers of 32 and 64 filters followed by MaxPooling layers are used for feature extraction (later discussed) from input data, flattened and then passed into a 128 neuron dense layer with ReLU activation function. A dropout layer regularizes before a final dense layer outputs class probabilities via softmax activation function. Steps which contributed to better performance of the testing model were retained in the final pipelines, of which we define two for latter comparison.

Downsample: All audio by default has a recorded sample rate of 32,000hz, yet the literature (Venier et al., 2017) suggests that for the purpose of bird song classification, much of this information is unnecessary - bird song typically does not reach frequencies beyond 16,000hz. Thus

we downsample to 16,000 hz which additionally helps to reduce the data load required to further process. We also note previous competition winners apply similar frequency caps (Piczak, 2016).

Trim: To further reduce audio clip size we apply a trimming function which cuts silence - determined by a threshold decibel level at 20 decibels - at the beginning and end of the sequence.

Normalize: Normalization of audio clips is a prevalent practice, and there are various experimentations of normalizing before or after extracting features or compensating for not doing so by using batch normalization layers in CNN models (Piczak, 2016; Kahl et al., 2017; Grill & Schlüter, 2017; Madhavi & Pamnani, 2018). We normalize the audio using Librosa's normalize method (which applies peak normalization to the samples) before extracting the features. This is key for our audio as volume varies considerably between samples, thus the signals generated by later feature extraction techniques - which serve as input into the models - require normalization.

Reduce noise: A noise reduction algorithm - based upon a 'spectral gating' method and contained in the noisereduce Python library - is applied in order to decrease the impact of environmental noise which features throughout many of the audio clips. The first two seconds of each clip are taken as a noise signal with which to calculate noise thresholds for different frequency bands. A mask, generated by these thresholds, is then applied to the entire audio clip in order to denoise elements unrelated to the primary bird song. Later experimentation between feature extraction processes implementing this noise reduction method saw it lead to general classification outperformance, thus the method was retained. We note similar approaches to bird audio classification tasks by others (Incze et al., 2018; Mehyadin et al., 2021), including previous BirdCLEF competition winners (see Incze et al. for examples).

Standardize length: The length of the audio samples is highly variable, with an average length of 40.88 seconds and standard deviation of 69.61 seconds. The majority of the clips, 16087 (94.96%), have a length equal to or shorter than 120 seconds, 704 (4.16%) is between two and five minutes, and 150 (0.89%) are between five and forty minutes, as portrayed in figure 3.

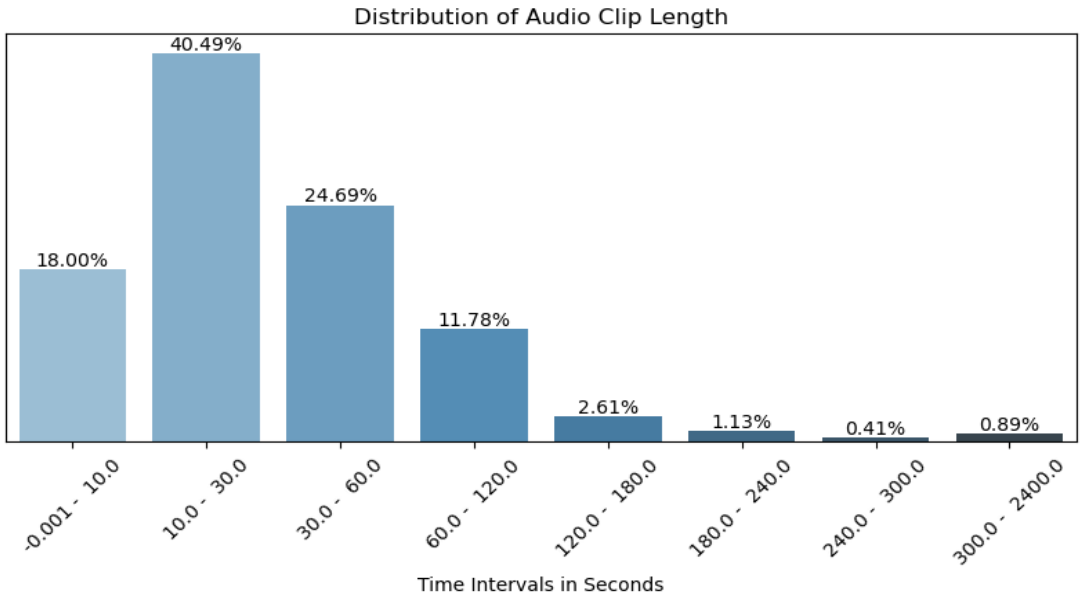


Figure 3: The Distribution of Time Length of Audio Samples in Intervals

It is common practice to divide audio clips into segments of equal length, although the specific length varies (Kahl et al., 2017; Madhavi & Pamnani, 2018; Mühling et al., 2020) We calculate the average audio clip length across all species and apply a padding and truncation method which shapes all clips into this size. Uniformity in clip length is necessary for future feature extraction and in terms of uniform dimensionality of input data to CNN models.

Segment: Another approach to processing audio data, particularly in bird sound classification, is slicing the audio into shorter segments or 'snippets'. This method has been widely used in literature (Xie & Zhu, 2023; Kahl et al., 2021) and for applications such as the BirdNet tool (Cornell University & Chemnitz University of Technology, 2023). The concept is to divide the original, longer audio files into smaller segments, typically ranging from one to five seconds in length. The primary advantage of slicing lies in data augmentation. By creating multiple snippets from a single, longer audio recording, the volume of training data is increased. This is especially useful when the dataset is limited or imbalanced. Furthermore, slicing might help in avoiding overfitting, as the model has a broader range of examples to learn from (Xie & Zhu, 2023).

Figure 4 illustrates the effects of different preprocessing steps.

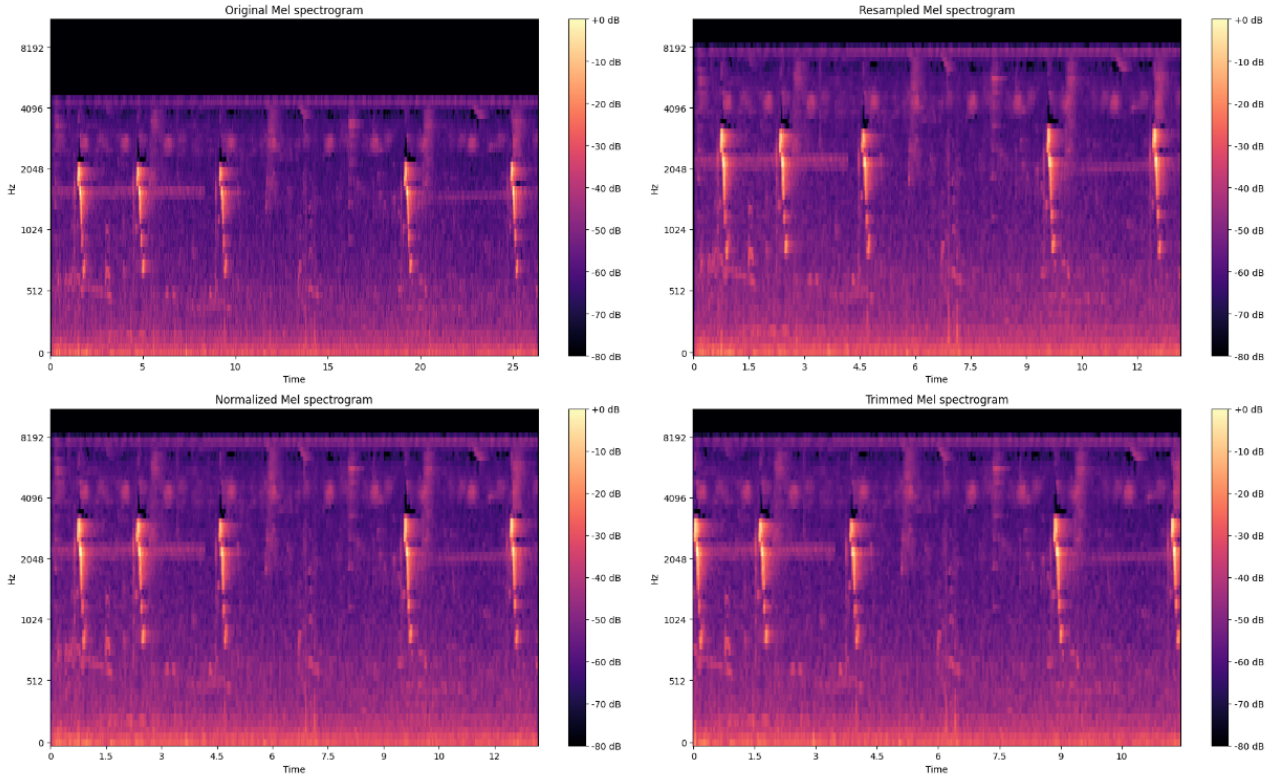


Figure 4: Visualization of the Effect of the Preprocessing Techniques

Data Feature Extraction

A standard approach to audio classification problems is the conversion of raw audio inputs to image data, whose features can be readily extracted and serve as input into deep learning models. Doing so allows for the use of conventional and well-studied deep learning techniques developed to handle images, most particularly CNN. Between the two most commonly extracted audio-to-image features - MFCC and MS - Carvalho & Gomes (2022) find that MS outperform on bird audio classification tasks. Experimentation with both feature types on the *test model* produced similar results for the BirdCLEF 2023 dataset, thus we also opted to use MSs as our signal image with which to extract features.

MS were generated via the Librosa library primarily using default parameters - n_fft (the number of Fast Fourier Transform points (FFT)) was left at 2048 and hop_length (the step size between FFTs) remained at 512. We experimented with the n_mels parameter (the number of Mel frequency bands in which the frequency spectrum is divided) between the values of 128 and 64. This decision was made in light of evidence provided by Xie et al., (2019), whereby a slight decrease in accuracy on baseline models on lower n_mels is a reasonable trade-off for the considerable decrease in computational load it provides. Using fewer Mel bins allows us to later train more complex models without training times becoming unruly.

Data Augmentation and Imbalance:
Imbalance in the dataset is material, with some bird species containing very few audio samples with which to train and test. Beyond the initial filtering of the dataset to remove tail species (see Pipelines) imbalance and lack of data for various species persists and so a process of oversampling via augmentation of existing audio samples is used. In audio classification, data augmentation is a particularly powerful tool for the reduction of overfitting.

Specifically, we apply audio augmentation on the raw input signals in the training set, experimenting with time stretches, time shifts, pitch shifts, and addition of noise. Like Kahl et al. (2021), with experimentation upon the *testing mode*l, we find that noise addition is most effective in improving generalizability and thus opt to implement this into the final pipelines. We implement Gaussian noise of randomized intensity to assist with model robustness against actual noise in the testing data. The test set only undergoes a process of feature extraction in order to prevent data leakage. Figure 5 visualizes a sample before and after injecting noise.

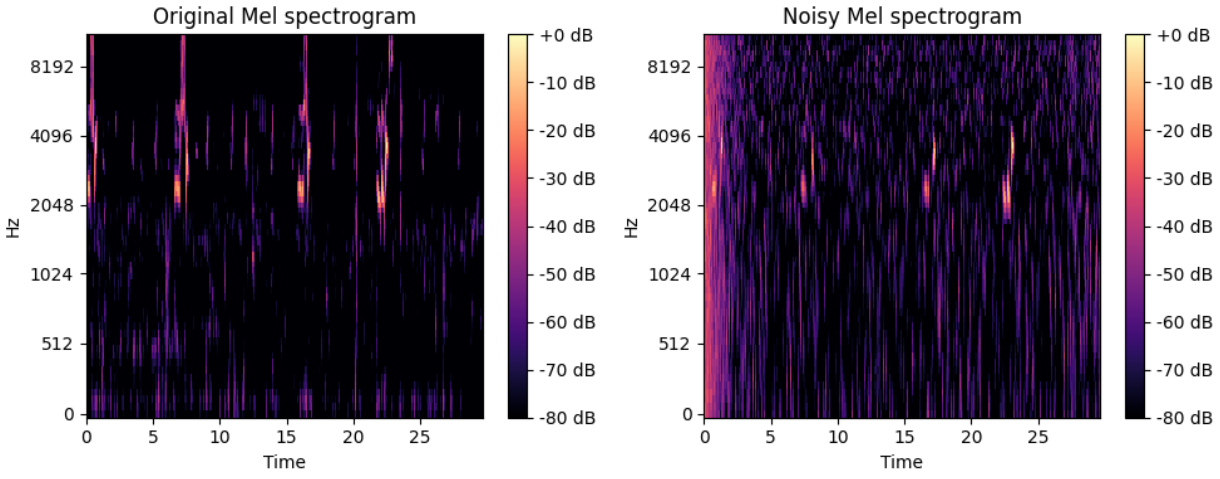


Figure 5: Visualization of the Effect of Data Augmentation by Noise Injection

We also opt to set a threshold level to which augmentations are generated - essentially creating a ‘floor’ for the number of available samples within species. Like other preprocessing parameters, the threshold sample level was determined by experimentation, determining the optimal level by performance comparison on the *testing model*.

Pipelines

Two preprocessing and augmentation pipelines are created to capture differing temporal and spatial information of the audio. Data imbalance is also handled slightly differently given the nature of class imbalance each produces, as shown in figure 6.

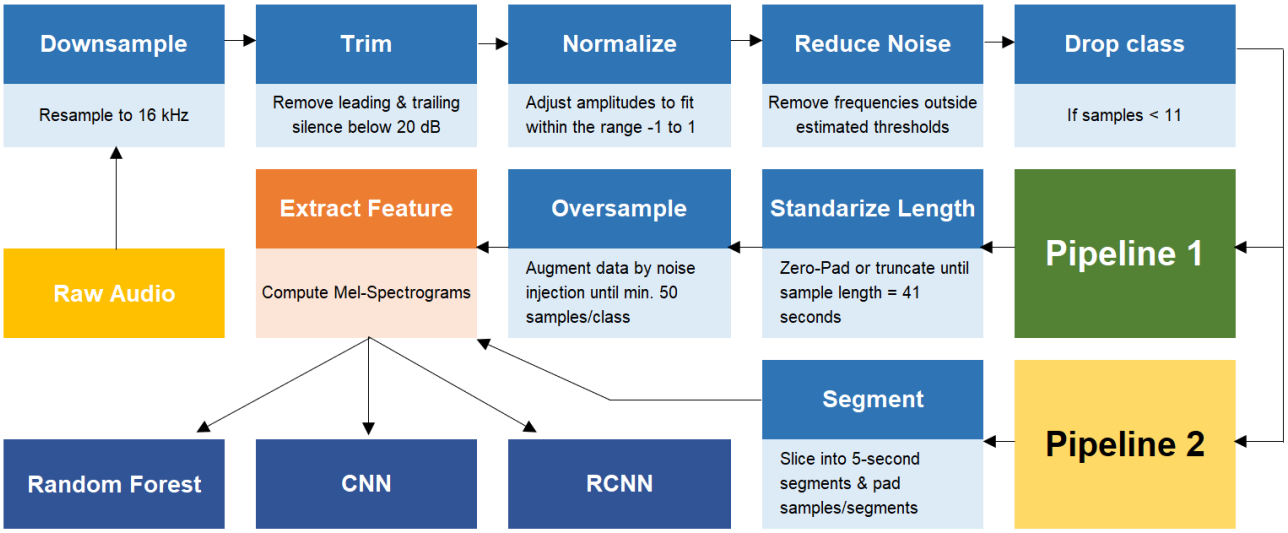


Figure 6: Flowchart depicting the system and the Pipelines

Both pipelines implement many of the same preprocessing steps - downsampling, trimming, normalizing and noise reduction. All bird species containing 10 or fewer samples are automatically removed (leaving 217 bird species to classify), which helps to limit issues around quantity of data in training and testing sets for low sample classes, and aids efforts to handle imbalance between species. Both extract Mel spectrograms. The key differences can be summarized as:

Pipeline 1 (P1) standardizes the length of all audio files to the dataset average (around 41 seconds), truncating or padding files to this point. Since the distribution of species remains unchanged, we oversample, augmenting via noise addition to a threshold level of 50 samples to reduce imbalance.

Pipeline 2 (P2) slices all audio files into 5 second snippets with padding applied to leftovers. There are no further augmentation steps as overall sample sizes increase materially.

Train, test and validation splits (80/10/10) are done prior to the preprocessing steps in order to prevent data leakage from the augmentation and oversampling processes. Following the feature extraction process, data are later labeled via one-hot-encoding of its respective primary species. This categorical labeling allows for multiclass classification.

Modelling Framework

We implement three classification methods, beginning with an RF Classifier - which serves as our baseline - followed by a CNN and a RCNN (see Appendix 1).

RF is chosen as our starting point for a number of reasons. Given the relative complexity of the audio features, its relatively short training times and ability to handle non-linear data and interaction terms serve as an ideal standard with which to compare. As a baseline, we opt for the default

hyperparameters provided by scikit-learn - n_estimators (or number of trees) set to 100 and max_depth (or decision tree splits) determined by the model itself - notably these are very deep (582) for P1.

The second model, a CNN loosely based upon the BirdNet architecture proposed by Kahl et al (2021), is a relatively deep structure with a single pre-processing block of 32 filters of size 3 x 3 with padding, followed by 4 convolutional layers with filters that double in size. All use a ReLU activation function, with batch normalization and max pooling applied to regularize outputs and parameters. This growing complexity of layers allows features to be extracted at different layers of abstraction. The model moves to a classification block in which filter sizes are further reduced and dropout instances help to regularize before finally reaching the fully connected layer - 3D outputs are flattened and passed into a dense layer corresponding to the number of bird species for classification. A softmax activation function outputs the probability distribution over all species.

An RCNN is adopted as a third model. This hybrid approach, combining a CNN and features of a Recurrent Neural Network allows the model to learn what Gupta et al. (2021) refer to as the 'temporal' component of sound across time, alongside the spatial component that a standard CNN may capture. The network is broadly the same structure as the CNN, except convolutional outputs are finally reshaped to fit the LSTM layer including timesteps of the MS in order to capture temporal patterns in the signal sequences. A dense layer with softmax activation function also outputs the probability distribution over all species.

When training the CNN and RCNN, we use the ReduceLROnPlateau callback which dynamically adapts the learning rate of the Adam optimizer. Once the validation loss stops improving, the learning rate is halved up to a lower bound of 0.0001. All of our classifiers work upon the assumption that only one bird species features in a recording.

Results

	RF		CNN		RCNN	
	P1	P2	P1	P2	P1	P2
Accuracy	.22	.27	.48	.62	.59	.63
Macro Avg. Precision	.08	.15	.25	.47	.43	.41
Weighted Avg. Precision	.16	.27	.49	.63	.62	.63
Macro Avg. Recall	.06	.06	.24	.38	.41	.38
Weighted Avg. Recall	.22	.27	.48	.62	.49	.63
Macro Avg. F-1 Score	.05	.06	.23	.39	.40	.37
Weighted Avg. F-1 Score	.15	.21	.47	.61	.59	.62
Epochs	-	-	30	10	40	20
Running Time	9 min	25 min	11 min	7 min	15 min	17 min
Specifications	12 Cores Intel® Xeon CPU		NVIDIA A100 Tensor Core GPU		NVIDIA A100 Tensor Core GPU	

Figure 7: Performance Metrics for all Classifiers and Pipelines

We consider model and pipeline performance using multiple evaluation metrics as seen in figure 7. Starting with accuracy, RCNN performs best, with 0.63 for P2 and 0.59 for P1, followed closely by the CNN with an accuracy of 0.62 for P2 and 0.48 for P1. The RF classifier, the baseline model, performs relatively poorly with an accuracy of just 0.27 for P2 and 0.22 for P1.

For precision, recall, and F-1 score, the CNN and RCNN outperforms RF in all cases. The RCNN achieves the highest scores for macro average precision, weighted average precision, weighted average recall, and weighted average F-1 score for P1 while it achieves very similar results to the CNN for P2. The CNN also performed reasonably well, consistently outperforming the RF in these metrics. However, the improvement between RCNN and CNN is mainly present in P1 with an average increase of 0.13 in the metrics presented in Figure 7. See Appendix 2 for training and validation loss over epochs for both CNN and RCNNs. In sum, both the CNN and RCNN vastly outperform the baseline model on every metric for both pipelines.

Insights

The first key insight from our results is that neural network structures (CNN and RCNN) considerably outperform traditional machine learning models - our baseline RF - for bird sound classification tasks. This illustrates the shift to neural networks as the de facto standard in this domain, as highlighted by Incze et al. (2018), and suggests that CNNs should be utilized by default.

Secondly, different approaches of data preprocessing and augmentation between the two data pipelines most clearly impact the performance of the CNN model. By segmenting the audio samples into several shorter MS, P2 increased the number of images for the CNN to train on. The higher data volume gives the CNN more opportunities to identify patterns and extract features from the data. P1 presented fewer data inputs compared to P2, using large MS for the audio samples. This provides a comprehensive visual representation of a whole audio sample but challenges the CNN's ability to effectively extract relevant features due to the densely packed information. In light of these differences, it was observed that P2 outperformed P1. This indicates that for optimal training of a CNN model with bird audio data, it's advisable to segment the audio into smaller parts, as it produces a greater volume of data for the model to learn from.

Thirdly, the performance gains that come from the introduction of a recurrent element to CNNs are more material for longer audio clips. This is illustrated clearly when comparing the CNN and RCNN performances of P1 versus P2. P1 improves on every metric when run through an RCNN model, whilst P2 remains relatively unchanged. This suggests that the spatial and temporal features that the RCNN hybrid model captures are learned more effectively when audio input and thus sequences are lengthier. We interpret the P2 slicing as potentially cutting bird sounds at key points, and introducing segments of noise that contain little temporal information from which to learn. Thus, RCNNs should be utilized when clip length is longer.

The fourth key insight is that class imbalance still greatly impacts performance in both pipelines, irrespective of our augmentation methods. In order to treat each class of our imbalanced dataset equally, we assign greater weight to the macro average metrics as this ensures that minority class performance is not overshadowed by majority classes. Even after handling class imbalance in our preprocessing and augmentation steps, there are clear differences between the macro and weighted averages within all models. Oversampling in P1 via noise addition tended to improve accuracy in our experimentation, however there remain clear signs of class imbalance in other metrics. Slicing audio files in P2 increases the overall volume of training data considerably, and whilst this improves overall performance, the gap between macro and weighted averages remains large. This suggests that larger sample sizes contribute positively, but further work beyond oversampling and slicing needs to be done to handle imbalance.

Discussion

Referring to the initial research question - '*How can machine learning algorithms be best utilized to differentiate between bird species based on audio data?'* - we find that treating the task as an image classification problem in the general case - utilizing CNNs - is far superior to baseline traditional models. However, adding a recurrent element in the context of lengthier audio data can lead to further outperformance still. This suggests that machine learning techniques that utilize both the spatial and temporal features of bird audio may better differentiate between bird species.

The consideration of false positives and false negatives is critical in bird sound classification dependent on the end goal. False positives denote instances where a bird's sound is erroneously assigned to a particular species despite not belonging to that species. Addressing this concern entails prioritizing precision, as a high precision value indicates a low occurrence of false positives, thereby signifying a more dependable system for accurately identifying bird species based on their vocalizations. Conversely, false negatives arise when the sound of a bird, which pertains to a specific species, is not correctly identified as such. In scenarios where the objective is to ensure the detection of all species, particularly in conservation efforts with potentially severe implications, the focus should be on achieving high recall. The relative importance of precision and recall can fluctuate based on the specific goals and contextual factors of the application at hand. In conservation initiatives where the primary aim is to detect rare species and avoid any oversight, attaining a high recall becomes increasingly crucial, as false positives can be easily spotted when reviewing the results.

RF was the easiest model to implement, requiring minimal configuration. It runs on CPU and the training time increases with larger amounts of data. The training took 9 minutes with the long MS, while the training with sliced audio data took 25 minutes. The CNN model is a sequential architecture with multiple convolutional and pooling layers and has a total of 5,078,744 parameters. It includes 8 convolutional layers that increase in depth from 32 to 512 and end by dropout and dense layers. The training was most efficient with the sliced audio data from P2, achieving an accuracy of 62% in 10 epochs. However, training the model with non-sliced audio data required 30 epochs, making it more time-intensive. The RCNN model is a combination of the layers from the

CNN and RNN layers, capturing spatial and temporal dependencies. With 5,365,081 parameters, it has a similar number of parameters as the CNN but has a slower training process and requires more epochs. Number of epochs, running time and utilized hardware for model training are also displayed in figure 7.

Preprocessing plays a vital role in our analysis; however, we encountered some limitations that need to be addressed. Initially, we employed a single oversampling technique, injecting white noise, for augmentation across all samples. While we did explore various techniques, we noticed that incorporating noise had a significant impact on the prediction accuracy of our models. It is important to acknowledge that confining ourselves to a sole augmentation technique may not fully exploit the potential benefits it offers. To optimize classification accuracy and overall generalization of our models, it is worth considering the exploration of different augmentation techniques selected at random. Furthermore, a notable aspect overlooked in our preprocessing approach was the absence of handling segments consisting of pure noise or silence, which is a commonly discussed theme in the literature. This oversight might offer an explanation for the inability to achieve higher classification performance and could have introduced bias into our results. Therefore, future research should address these preprocessing limitations to gain a more comprehensive understanding of the data and to improve the performance of machine learning models.

Conclusion & Future Work

In sum, we trained three different models on bird audio data in order to classify different species based upon their sound. We began by structuring two different preprocessing pipelines that aimed to isolate different aspects of the audio data. The extracted features were input into three models - a baseline ‘traditional’ machine learning model in the form of a RF classifier, and two neural networks - a CNN and an RCNN - in comparison. We found that both neural networks vastly outperformed the baseline in all aspects and particularly when clips were sliced into smaller chunks. However, for lengthier audio clips, the RCNN produced significant improvements in classification ability.

For potential future work, an opportunity to explore is the generation of multiple Mel spectrograms from a single audio sample using varying parameters such as mel bins and hop length. This approach gives varying perspectives on the same audio data, capturing different acoustic features that may contribute to improved classification performance when training a CNN, enhancing the accuracy and robustness of a bird sound classification model.

References

Carvalho, S., & Gomes, E. F. (2022). *Automatic Classification of Bird Sounds: Using MFCC and Mel Spectrogram Features with Deep Learning*. In Vietnam Journal of Computer Science (Vol. 10, Issue 01, pp. 39–54). World Scientific Pub Co Pte Ltd. https://doi.org/10.1142/s2196888822500300

Cornell University & Chemnitz University of Technology. (2023). *BirdNET Sound ID.* https://birdnet.cornell.edu/

Grill, T. & Schlüter, J. (2017). *Two convolutional neural networks for bird detection in audio signals*. 25th European Signal Processing Conference (EUSIPCO), pp. 1764-1768, doi: 10.23919/EUSIPCO.2017.8081512.

Gupta, G., Kshirsagar, M., Zhong, M., Gholami, S., & Ferres, J. L. (2021). *Comparing recurrent convolutional neural networks for large scale bird species classification*. In Scientific Reports (Vol. 11, Issue 1). Springer Science and Business Media LLC. https://doi.org/10.1038/s41598-021-96446-w

Incze, A., Jancsó, H. B., Szilágyi, Z., Farkas, A., & Sulyok, C. (2018, September). Bird sound recognition using a convolutional neural network. In *2018 IEEE 16th international symposium on intelligent systems and informatics (SISY)* (pp. 000295-000300). IEEE.

Kaggle. (2023). *BirdCLEF 2023 Identify bird calls in soundscapes*. https://www.kaggle.com/competitions/birdclef-2023

Kahl, S., Wilhelm-Stein, T., Hussein, H., Klinck, H., Kowerko, D., Ritter, M., & Eibl, M. (2017). *Large-Scale Bird Sound Classification using Convolutional Neural Networks*. In *CLEF (Working Notes)* (Vol. 1866).

Kahl, S., Wood, C. M., Eibl, M., & Klinck, H. (2021). *BirdNET: A deep learning solution for avian diversity monitoring*. In Ecological Informatics (Vol. 61, p. 101236). Elsevier BV. https://doi.org/10.1016/j.ecoinf.2021.101236

Koskimies, P. (1989). *Birds as a tool in environmental monitoring*. Annales Zoologici Fennici, *26*(3), 153–166. http://www.jstor.org/stable/23734578

ksanjeevan. (2020, 17th May). *CRNN Audio Classification Models*. https://github.com/ksanjeevan/crnn-audio-classification#models

Madhavi, A., & Pamnani, R. (2018). *Deep learning based audio classifier for bird species*. J. Sci. Res., 3, 228-233.

Mehyadin, A. E., Abdulazeez, A. M., Hasan, D. A., & Saeed, J. N. (2021). *Birds Sound Classification Based on Machine Learning Algorithms*. In Asian Journal of Research in Computer Science (pp. 1–11). Sciencedomain International. https://doi.org/10.9734/ajrcos/2021/v9i430227

Mühling, M., Franz, J., Korfhage, N., & Freisleben, B. (2020). *Bird Species Recognition via Neural Architecture Search.* In *CLEF (Working Notes)* (pp. 1-13).

Piczak, K. J. (2016, September). *Recognizing Bird Species in Audio Recordings using Deep Convolutional Neural Networks*. In *CLEF (working notes)* (pp. 534-543).

Sprengel, E., Jaggi, M., Kilcher, Y., & Hofmann, T. (2016). *Audio based bird species identification using deep learning techniques* (No. CONF, pp. 547-559).

Venier, L. A., Mazerolle, M. J., Rodgers, A., McIlwrick, K. A., Holmes, S., & Thompson, D. (2017). *Comparison of semiautomated bird song recognition with manual detection of recorded bird song samples*. In Avian Conservation and Ecology (Vol. 12, Issue 2). Resilience Alliance, Inc. https://doi.org/10.5751/ace-01029-120202

Xie, J., & Zhu, M. (2023). *Acoustic Classification of Bird Species Using an Early Fusion of Deep Features.* In Birds (Vol. 4, Issue 1, pp. 138–147). MDPI AG. https://doi.org/10.3390/birds4010011

Xie, J., Hu, K., Zhu, M., Yu, J., & Zhu, Q. (2019). *Investigation of different CNN-based models for improved bird sound classification*. *IEEE Access*, *7*, 175353-175361.

Appendix

Appendix 1

CNN - Model Summary

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 64, 1278, 32)	320
batch_normalization (BatchNormalization)	(None, 64, 1278, 32)	128
re_lu (ReLU)	(None, 64, 1278, 32)	0
max_pooling2d (MaxPooling2D)	(None, 32, 639, 32)	0
conv2d_1 (Conv2D)	(None, 32, 639, 64)	18496
batch_normalization_1 (BatchNormalization)	(None, 32, 639, 64)	256
re_lu_1 (ReLU)	(None, 32, 639, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 16, 319, 64)	0
conv2d_2 (Conv2D)	(None, 16, 319, 128)	73856
batch_normalization_2 (BatchNormalization)	(None, 16, 319, 128)	512
re_lu_2 (ReLU)	(None, 16, 319, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 8, 159, 128)	0
conv2d_3 (Conv2D)	(None, 8, 159, 256)	295168
batch_normalization_3 (BatchNormalization)	(None, 8, 159, 256)	1024
re_lu_3 (ReLU)	(None, 8, 159, 256)	0
max_pooling2d_3 (MaxPooling2D)	(None, 4, 79, 256)	0
conv2d_4 (Conv2D)	(None, 4, 79, 512)	1180160
batch_normalization_4 (BatchNormalization)	(None, 4, 79, 512)	2048
re_lu_4 (ReLU)	(None, 4, 79, 512)	0
max_pooling2d_4 (MaxPooling2D)	(None, 2, 39, 512)	0
conv2d_5 (Conv2D)	(None, 1, 38, 512)	1049088
batch_normalization_5 (BatchNormalization)	(None, 1, 38, 512)	2048
re_lu_5 (ReLU)	(None, 1, 38, 512)	0
dropout (Dropout)	(None, 1, 38, 512)	0
conv2d_6 (Conv2D)	(None, 1, 38, 1024)	525312
batch_normalization_6 (BatchNormalization)	(None, 1, 38, 1024)	4096
re_lu_6 (ReLU)	(None, 1, 38, 1024)	0
dropout_1 (Dropout)	(None, 1, 38, 1024)	0
conv2d_7 (Conv2D)	(None, 1, 38, 212)	217300
batch_normalization_7 (BatchNormalization)	(None, 1, 38, 212)	848
dropout_2 (Dropout)	(None, 1, 38, 212)	0
flatten (Flatten)	(None, 8056)	0
dense (Dense)	(None, 212)	1708084

RCNN - Model Summary

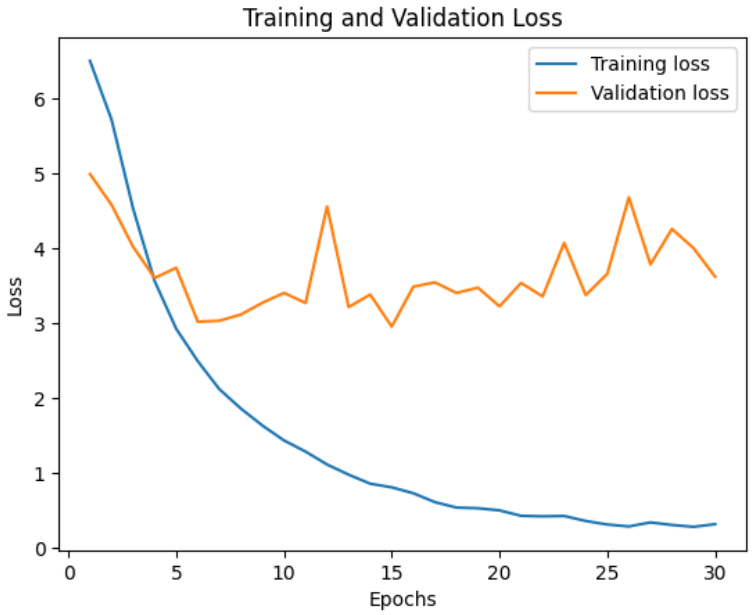
Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d_8 (Conv2D)	(None, 64, 1278, 32)	320
batch_normalization_8 (BatchNormalization)	(None, 64, 1278, 32)	128
re_lu_7 (ReLU)	(None, 64, 1278, 32)	-
max_pooling2d_5 (MaxPooling2D)	(None, 32, 639, 32)	-
conv2d_9 (Conv2D)	(None, 32, 639, 64)	18 496
batch_normalization_9 (BatchNormalization)	(None, 32, 639, 64)	256
re_lu_8 (ReLU)	(None, 32, 639, 64)	-
max_pooling2d_6 (MaxPooling2D)	(None, 16, 319, 64)	-
conv2d_10 (Conv2D)	(None, 16, 319, 128)	73 856
batch_normalization_10 (Bat chNormalization)	(None, 16, 319, 128)	512
re_lu_9 (ReLU)	(None, 16, 319, 128)	-
max_pooling2d_7 (MaxPooling2D)	(None, 8, 159, 128)	-
conv2d_11 (Conv2D)	(None, 8, 159, 256)	295 168
batch_normalization_11 (BatchNormalization)	(None, 8, 159, 256)	1 024
re_lu_10 (ReLU)	(None, 8, 159, 256)	-
max_pooling2d_8 (MaxPooling2D)	(None, 4, 79, 256)	-
conv2d_12 (Conv2D)	(None, 4, 79, 512)	1 180 160
batch_normalization_12 (BatchNormalization)	(None, 4, 79, 512)	2 048
re_lu_11 (ReLU)	(None, 4, 79, 512)	-
max_pooling2d_9 (MaxPooling2D)	(None, 2, 39, 512)	-
conv2d_13 (Conv2D)	(None, 1, 38, 512)	1 049 088
batch_normalization_13 (BatchNormalization)	(None, 1, 38, 512)	2 048
re_lu_12 (ReLU)	(None, 1, 38, 512)	-
dropout_3 (Dropout)	(None, 1, 38, 512)	-
conv2d_14 (Conv2D)	(None, 1, 38, 1024)	525 312
batch_normalization_14 (Bat chNormalization)	(None, 1, 38, 1024)	4 096
re_lu_13 (ReLU)	(None, 1, 38, 1024)	-
dropout_4 (Dropout)	(None, 1, 38, 1024)	-
reshape (Reshape)	(None, 76, 512)	-
lstm (LSTM)	(None, 512)	2 099 200
dropout_5 (Dropout)	(None, 512)	-
batch_normalization_15 (BatchNormalization)	(None, 512)	2 048
dense_1 (Dense)	(None, 212)	108 756

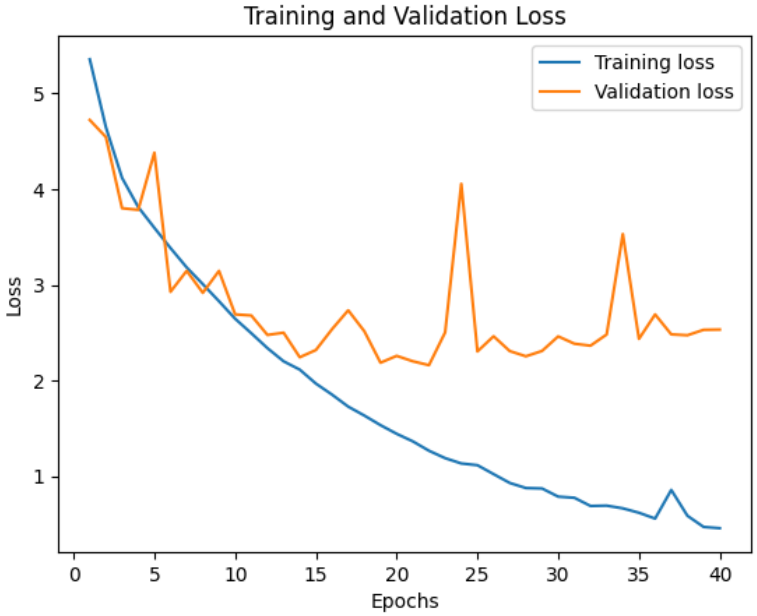
Total params: 5,362,516
Trainable params: 5,356,436
Non-trainable params: 6,080

Appendix 2

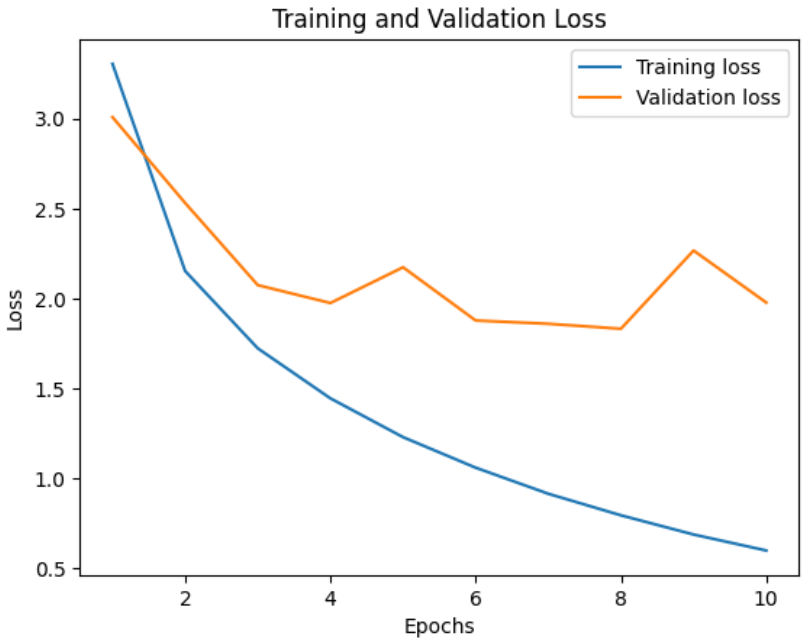
CNN - P1



RCNN - P1



CNN - P2



RCNN - P2