Bird Calls Identification in Soundscapes

Yanqing Lu Jinbo Liu Zixuan Wang Jingyi Huang

Abstract

In recent years, bird conservation has received increasing attention, and in order to more easily observe bird species, passive acoustic monitoring (PAM) combined with new analytical methods based on machine learning came into being. In this project, we focus on the task of bird sound identification. To address this task, we employed three different machine learning models: K-Nearest Neighbors (k-NN), Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN). Our findings revealed that CNNs outperform both k-NN and RNN in classifying bird sounds, despite RNNs typically being preferred for sequential data. Surprisingly, RNNs, which excel in tasks requiring temporal dependencies such as speech classification, performed the poorest in our study. This indicates that RNNs may not always be the best choice for all types of sequential data, particularly when long-term dependencies are absent. For bird sound classification, CNNs proved more effective by capturing significant time-frequency features in spectrograms of shorter audio clips, which lack long-term dependencies but possess distinct features. This research highlights the importance of selecting appropriate models based on the specific characteristics of the data in audio classification tasks.

1. Introduction

Biodiversity has always been a hot topic, and as birds are an important part of regional biology, changes in their species can be used as an indicator of ecological changes in the region. The traditional method of examining bird species is based on field surveys conducted by observers over a long period of time, which is not only time-consuming but also requires a lot of funding resources. Therefore, we need other methods to collect information about bird species. One of the good solutions is passive acoustic monitoring (PAM). It is a monitoring method that installs automatic recorders in natural environments to collect acoustic signals from wildlife, which can give us multiple spatial scales and at fine temporal resolution.

Our goal is to analyze the audio data collected by PAM to identify Eastern African bird species. More precisely, we will process continuous audio data and build models to identify bird species based on their sounds.

For the dataset, we have 264 bird species and 16941 audio files, each sound file corresponds to only one bird species. However, the files are not evenly divided in each species. Some species have up to 500 audio samples, while some only have one. Also, every audio file has their quality rating ranging from 0 to 5. High scores mean that single bird species call in the audio is clearly distinguishable. Low scores imply that there is some noise in the audio that may interfere with our analysis.

In the field of sound processing, Mel-Frequency Cepstrum (MFC) is a linear transformation of the logarithmic energy spectrum based on the nonlinear Mel scale of sound frequency. Mel-Frequency Cepstral Coefficients (MFCCs) are the coefficients that make up the MFC. On one hand, we will use MFCCs directly. On the other hand, we convert it into spectrograms. They are used to train different machine learning models.

As for machine learning models, CNN and RNN can capture low-level acoustic features like spectral patterns, allowing them to perceive subtle difference in sound and classify sound signals with high accuracy. Zhejian Chi et al. proposed a CNN using a concatenated spectrogram as input features to increase the richness of features. This method was tested in the datasets ESC-50 and UrbanSound8K with more than 80% classification accuracy(Chi et al., 2019). Jonghee Sang et al. combines CNN with RNN to design convolutional recurrent neural network (CRNN), which is a modified CNN by changing the last layers into an RNN layer. CRNN enables the networks to extract high level features that are invariant local variations and take temporal aggregation of extracted features. The result shows that CRNN improves the accuracy by 7.38% than original CNN(Sang et al., 2018). In contrast, we train CNN model and RNN model respectively to examine which one is better on the bird calls. Lezhenin et al. examine a LSTM model for urban sounds classification and find that it has 84.25% of average accuracy and perform better than the majority of existing solutions(Lezhenin et al., 2019).In 2024, Vashishtha et al. use k-NN and CNN to classify musical instrument sounds.

Table 1. Subsets obtained by applying different filters.

#class/#sample	Rating ≥ 0	Rating ≥ 3	Rating ≥ 4
Count ≥ 0	264/16941	264/14620	261/9952
Count ≥ 100	44/10310	33/7892	20/4453
Count ≥ 200	17/6693	17/5782	11/3169

k-NN model achieves 96% accuracy, while CNN model gets 88% accuracy, which implies that k-NN can be used as a base model for sounds classification(Vashishtha et al., 2024). Based on k-NN, we will use embedding to train the model and try to achieve increases of accuracy on the test.

2. Methods

We formulated the bird calls identification task as a classification problem, and then applied 3 different machine learning algorithms (k-NN, CNN, and RNN) to solve it.

2.1. Data Gathering

Based on attributes like rating, and the number of samples belonging to one species, we could filter all the raw data to obtain several subsets with different sizes and number of classes. We selected the three subsets marked as red in Table 1 to formulate three classification problems with different levels of complexity. These three problems, from simple to complex, are denoted as **Problem 1, 2**, and **3** respectively. It is worth noting that in the training pipelines of our algorithms, we further segment one audio sample to implement data augmentation.

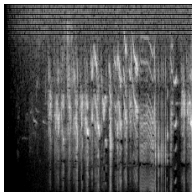

Figure 1. Sample spectrogram generated from bird calls.

2.2. k-NN

The k-nearest neighbors (k-NN) algorithm is a non-parametric, classic supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. For image classification, such as dogs and cats, the same species could have different appearance on images. For example, images labeled 'cat' may have differences in illumination, deformation, occlusion, or background clutter. However, in terms of sound classification, the same species of birds have little variation on bird calls. Therefore, we choose k-NN to be our classic supervised learning classifier because it has advantages in predicting similar data.

2.2.1. Data Pre-processing

We divide a piece of audio into segments every five seconds and use one segment as one datapoint in our dataset. To a five seconds audio segment, first we transfer it to an embedding with size of 313*13 where 313 and 13 represent 313 frames and 13 Mel-frequency cepstral coefficients (MFCCs) respectively. Then, we average the columns and get an embedding with size of 1*13.

2.2.2. Cross validation

After splitting training and testing sets with a ratio of 9:1, we apply a 5-fold cross validation on the training set with the number of neighbors from k=1 to k=25 on Problem 1. As shown in figure 2, our k-NN model gets the highest accuracy when k=1(cross validation on Problem 2 and 3 have the same trend). Therefore, we choose k=1 as the number of neighbors of our k-NN classifier.

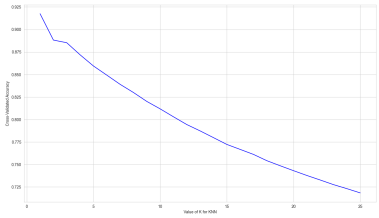

Figure 2. Cross validation of k-NN.

2.3. CNN

Convolutional Neural Network (CNN) is a neural network architecture widely used in computer vision. Given its ability to effectively capture spatial hierarchies of features, we designed a CNN-based algorithm for our problem. Specifically, our model was built on ResNet-18 (He et al., 2016).

2.3.1. Data Pre-processing

Since the format of our raw data is audio, the first step is to convert them into *spectrogram*, which is an image representation of audio signals. We load every bird call sample with a consistent sampling rate, and then split them into multiple fixed-length 5-second chunks. Signals within only a specified frequency range are captured.

In order to match the input size of ResNet-18 without cropping the spectrogram to lose some information, we generated 224x224 pixel spectrograms from all audio chunks. We firstly generated gray-scale spectrograms to avoid the

effect of different audio channels, and then converted them to standard 3-channel tensors with normalization. These image tensors constituted the dataset for our CNN model. We randomly split the dataset into training, validation and testing sets with a ratio of 7:2:1.

2.3.2. CNN Architecture

The architecture of a CNN plays a critical role in its performance and effectiveness for image classification tasks. We chose ResNet-18 as our primary CNN architecture due to its balance between model complexity and performance.

ResNet-18 consists of 18 layers in total, including convolutional layers, batch normalization layers, ReLU activation functions, max-pooling layers, residual blocks, and a fully-connected layer. In order to adapt this model structure to our problem, we modified the dimension of the last fully-connected layer as the number of bird species in each problem. As a result, this model can reasonably output logits over classes in each of our classification problem.

We used the ResNet-18 model pre-trained on the ImageNet-1k dataset (Russakovsky et al., 2015) released by PyTorch (Paszke et al., 2019) as a starting point for training. We found that training our model from the pre-trained model achieved better performance than training from scratch, which benefits from the transfer learning. Since the pre-trained model has already learned rich feature representations from a diverse range of images, this initialization helps the model converge faster and more effectively, as it can focus on learning task-specific features rather than starting from random initialization.

2.3.3. Experiment Pipeline

We used *grid search* to tune several hyperparameters (e.g., learning rate, batch size) of CNN, mainly based on the model performance (validation accuracy). We chose Adam optimizer (Kingma & Ba, 2014), which features adaptive learning rates for each parameter, fast convergence, and robustness to noisy gradients in deep learning models.

In the training phase, we evaluated the CNN model by training and validation accuracies at the end of each epoch. We saved the model with the highest validation accuracy. Then in the test phase, we tested our best model on the test set with several evaluation metrics.

2.4. RNN

Recurrent Neural Networks (RNNs) are good at capturing temporal dependencies, which is crucial for tasks where context over time is significant. Their efficacy in handling sequence-based tasks, such as speech recognition and music generation, is widely proven. Given that bird sounds are also a kind of sequential data, it is intuitive to employ an RNN-based model for their classification. This approach leverages the inherent strengths of RNNs in processing and learning from time-series data, making it a suitable choice for this task.

2.4.1. Data Pre-processing

Initially, the audio data is saved in ogg format. In order to prepare the dataset for the RNN based architecture, we need to do a series of data processing. Firstly, each audio file was loaded with a consistent sampling rate to ensure uniformity in signal processing across the dataset and the audio signals were converted to monophonic format to simplify the data and avoid discrepancies that might arise from multiple audio channels. Then, each file was truncated to a fixed duration. This kind of standardization improves the comparability of features extracted from different files. Following the loading process, the audio signals were segmented into smaller, fixed-length chunks. The length of each segment was determined based on the typical temporal audio content being analyzed, aiming to capture enough audio information while maintaining relatively low computational requirements.

Mel-Frequency Cepstral Coefficients (MFCCs) are a feature representation widely utilized in the field of audio signal processing, particularly for speech and sound recognition tasks. The effectiveness of MFCCs arises from their ability to capture the essential characteristics of the audio spectrum in a way that mimics the human auditory system. The process involves transforming the audio signal into the frequency domain, applying the Mel scale to emphasize perceptually important aspects of sound, and then using the discrete cosine transform (DCT) to compress the spectrum into a set of cepstral coefficients. These coefficients form a compact representation of the sound that is used for various tasks in audio analysis. Our feature extraction focused on deriving Mel-frequency cepstral coefficients (MFCCs) from each audio segment. The extraction process was carefully tuned with parameters such as the number of coefficients, the window size for the Fourier transform, and the hop length between successive frames. The resultant feature sets represent the spectral properties of the audio segments in a form that is suitable for training RNN based models.

The structured and processed features, along with their corresponding labels derived from the annotations of the audio files, were compiled into a dataset ready for later model training. The whole dataset is divided into training-set: validation-set: test-set =7:2:1.

2.4.2. RNN Architecture

Our proposed model is based on Long Short-Term Memory networks (LSTMs), which are an advanced type of RNN, designed to address the limitations commonly encountered in traditional RNNs. The model structure is as follows:

Table 2. Evaluation metrics across different models and problems.

	Accuracy	Macro F1	Weighted F1
k-NN (Problem 1)	0.94	0.92	0.94
CNN (Problem 1)	**0.98**	0.97	0.98
RNN (Problem 1)	0.71	0.70	0.71
k-NN (Problem 2)	0.89	0.88	0.89
CNN (Problem 2)	**0.97**	0.97	0.97
RNN (Problem 2)	0.58	0.55	0.57
k-NN (Problem 3)	**0.86**	0.80	0.86
CNN (Problem 3)	0.85	0.73	0.85
RNN (Problem 3)	0.38	0.20	0.34

1.LSTM Layers: The first layer and second layers are both LSTM with 64 neurons. For the first layer, it processes the input sequence while retaining the ability to pass information across longer time spans, and returns sequences to allow the next LSTM layer to further do the feature extraction process, which is crucial for learning from the temporal structure of the audio data. Following the first layer, another LSTM layer with 64 neurons is used, but it does not return sequences at this time. This design choice enhances the model's ability to capture information from the entire sequence before moving to classification, providing a synthetical representation of the input features.

2.Dense and Dropout Layers: After sequential processing through LSTM layers, the network transitions to a dense layer and ReLU activation to introduce non-linearity, enhancing the model's ability to learn complex patterns in the data. A dropout layer with a rate of 0.3 follows to prevent overfitting by randomly omitting a subset of features during training.

3.Output Layer: The final layer is a dense layer with a softmax activation function, which outputs the probability distribution over the class labels for the classification task.

3. Results

3.1. k-NN

We apply k-NN classifiers on Problem 1, 2, and 3 and get test accuracies of 0.94, 0.89, and 0.86 respectively. Figure 3 shows the confusion matrix of the test results of Problem 1. We also calculate macro F1-score and weighted F1-score, as shown in table 2. From the results we can see that our k-NN classifier has good performance on all problems. Although the species to be classified in Problem 3 increase from 11 to 264, roughly a factor of 20 compared to the results of Problem 1, the accuracy and F1-score only decrease by about 0.1. The reason why our model has high accuracy is not only that the embedding of audios well represents the important features of the bird calls data but also that embeddings can be well utilized by k-NN classifiers to perform classification tasks using spatial similarity.

Table 3. Validation accuracies with different hyperparameters.

Batch Size\Learning Rate	1e-3	1e-4	1e-5
16	0.828	0.870	0.839
32	0.842	0.875	0.845
64	0.874	0.884	0.834

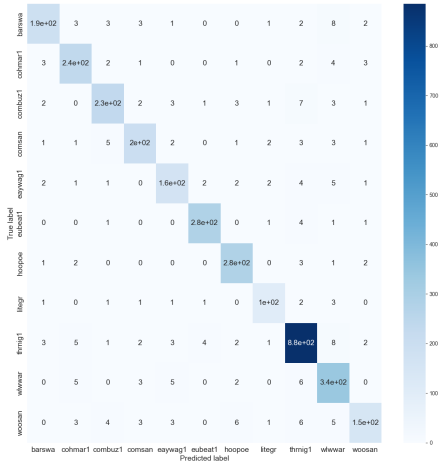

Figure 3. Confusion matrix of k-NN for Problem 1.

3.2. CNN

We conducted experiments for all the three problems following the CNN experiment pipeline in Section 2.3.3. Compared with the other two algorithms, CNN generally performed

3.2.1. Hyperparameters Tuning

As mentioned in Section 2.3.3, we used grid search to find the optimal combination of hyperparameters for our CNN model. We explored the learning rate grid (1e-3, 1e-4, 1e-5) and the batch size grid (16, 32, 64), training and validating CNN for 10 epochs. The validation accuracies for Problem 1 are shown in Table 3. For this problem, the relationship between accuracy and hyperparameter value is not linear. Based on the observation, we chose learning rate = 1e-4, batch size = 64 for our CNN model.

3.2.2. Result Analysis

After 10 training epochs, our CNN model achieved test accuracy of 0.976 on Problem 1 (11-class), 0.971 on Problem 2 (33-class), and 0.850 on Problem 3 (264-class). Looking at the macro and weighted F1 score in Table 2, these two

values are close for Problem 1 and 2, which indicates that CNN has a relatively balanced performance across classes. As a comparison, the macro F1 score is significantly lower than the weighted F1 score for Problem 3. This possibly implies that the CNN classifier performs poorly on minority classes while performing well on majority classes.

Compared with k-NN, CNN got slightly higher test accuracies on Problem 1 and 2, while a little lower accuracy on Problem 3. The following factors could contribute to this result: 1) The dataset exhibits non-linear separability, so that k-NN can accurately capture the distance features of data points to perform competitively with CNN. 2) The complexity of ResNet-18 leads to overfitting, especially for a relatively complex problem with noises.

Compared with RNN, the test accuracy of CNN is higher on the three problems. This may result from the data characteristic, such that CNN can more efficiently extract hierarchical features from the audio data. Besides, the network architecture enables CNN to leverage parallelization and GPU acceleration for faster convergence.

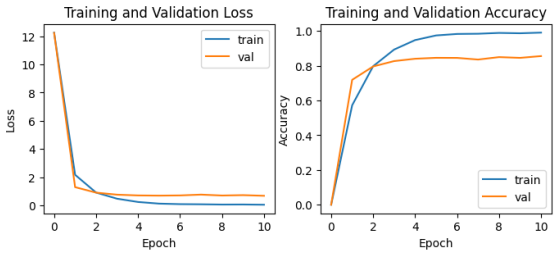

Figure 4. Learning curve of CNN on Problem 3.

3.3. RNN

We did experiments on all 3 problems. In our experiments, batch size is set to 32 and learning rate is set to 1e-3. Basically, the accuracy curves by using our RNN based model are similar for all these three problems, for spatial reasons we only show the accuracy curves for Problem 3.

3.3.1. Result Analysis

We can see that accuracy gradually increases with more epochs, but even after 50 epochs, the accuracy of the RNN based model is still far below the accuracy achieved by the CNN method at less than 10 epochs. In addition, the RNN spent five times as long in Problem 3 as the CNN model. This suggests that RNN based models may not be well suited to classification tasks based on bird sounds, which is an atypical sequence data. We discussed further in 4.3.2. Another notable phenomenon is that at the beginning of training, the accuracy on the validation set is always slightly higher than on the training set, which we discussed in 4.3.1.

Table 4. k-NN test accuracies of spectrogram and flattening embedding methods.

	Spectrogram	Flattening embedding
Problem 1	0.51	0.74
Problem 2	0.40	0.64
Problem 3	0.33	0.55

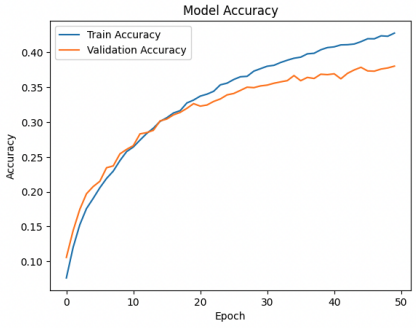

Figure 5. Learning curve of RNN on Problem 3.

4. Process

In this section, we will discuss our works, explorations and iterations that led to the final solution.

4.1. k-NN

In the beginning, we start training our k-NN model with the spectrogram dataset. For a 48*128 spectrogram, we flatten it into a 6,144 dimension feature vector to train our k-NN model. As shown in table 4, we only get accuracies of 0.51, 0.40, and 0.33 on the test set of the three problems. Afterwards, we use embedding to train our k-NN model. However, before using the average version embedding, first we tried to flatten a 313*13 Embedding to a 4069 dimension feature vector for training. The accuracies of flattening version embedding of the three problems are 0.74, 0.64, and 0.55. Graph 6 shows the increases of accuracies on the three problems after each improvements on methods.

4.2. CNN

In addition to ResNet-18, we also tried other CNN models. Firstly, we tried two manually-built simple CNN model, they have 3 and 6 convolutional layers followed by 2 fully-connected layers, respectively. They achieved test accuracies of 0.563 and 0.608. Furthermore, we explored another classic CNN architecture, AlexNet (Krizhevsky et al., 2012), which achieved 0.727 test accuracy.

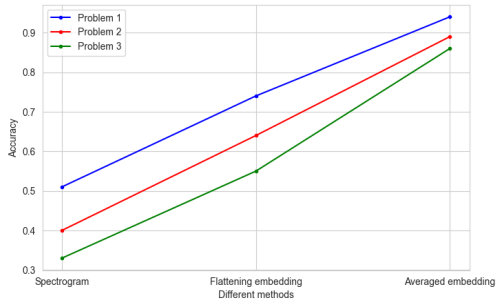

Figure 6. k-NN improvement.

Compared with the first 2 CNN models, ReseNet-18 has more complex network structure to be able to capture more informative patterns and representations in the data. Compared with AlexNet, ResNet-18 is more lightweight in terms of the number of parameters, while performing better.

4.3. RNN

4.3.1. Accuracy of Anomaly

In our training process, especially for the RNN based model, an interesting observation was that the training set accuracy was consistently lower than validation set accuracy in some initial epochs. Although counterintuitive at the first glance, it's reasonable after our careful consideration.

The first reason is the dropout layer in our model. In our case, we applied a layer with a dropout rate of 0.3 after the Dense layer. This means that during each training epoch, a portion of the network's connections are temporarily dropped out, effectively reducing the model's capacity to learn from the training data. Conversely, during validation, Dropout is disabled, and the model utilizes its full ability, thus performing better on the validation data than on the training data.

Secondly, as we all know, neural networks require several epochs over the training data to adequately learn the features. In the beginning, the model is actually learning to generalize from a small amount of information relative to the complexity of the task. And also, the model is trained with small batch data, the data from each batch can vary significantly. Some batches tend to contain some outliers or "hard data", which can reduce overall training accuracy. When it comes to evaluation, the model is evaluated using the entire validation set at each epoch, which behaves more smoothly and consistently.

However, after some epochs, the train accuracy always becomes higher than the validation accuracy, which is the normal case.

4.3.2. Atypical Sequence Data

RNN or its variant LSTM is usually a great choice to learn the features of sequence data. However, it can be found from our experiment that compared with RNN based models, CNN achieves higher accuracy in bird sound classification. So why CNN architecture over RNN architecture?

First of all, the bird sound data is naturally represented in time-frequency and this makes it suitable for spectrogram representation. Bird specific call patterns, such as specific frequencies and rhythms of song, can be represented in the spectrogram as visual textures and shapes. CNN can effectively recognize these patterns, even in different background noises.

Secondly, RNN outperforms CNN only when there are long-term dependencies. However, such long-term dependence may not be as significant in bird call recognition tasks as it is in other sequence data tasks. The identification of birds can rely primarily on features within shorter audio clips, including specific frequency patterns, tones, and rhythms, which can be efficiently captured by spectral analysis within a short time window. From Survey of Audio Classification Using Deep Learning (Zaman et al., 2023), it says "CNNs have proven to be highly effective for extracting spatial features from audio signals, making them suitable for tasks such as music genre classification and environmental sound classification. The remarkable success of CNNs in this field is due to their ability to capture high-level features from the audio data. RNNs are particularly well-suited for tasks that require temporal dependencies, such as speech classification and audio-sequence classification."

Last but not least, CNNs are more computationally efficient when dealing with fixed-size inputs, such as spectrograms, because they can process spatial hierarchies in the data in parallel. In contrast, RNNs must process time steps sequentially, which leads to the low efficiency. In our experiments, for the same size data set, our CNN based model is 2-5 times faster than the RNN model.

5. Contributions

In conclusion, our exploration of k-NN, CNN, and RNN for bird sound classification has provided valuable insights into the efficacy of machine learning models in this domain. Through rigorous experimentation and analysis, we have demonstrated the capability of each method in capturing and leveraging temporal and spatial features of bird sound data. Our findings underscore the importance of selecting appropriate algorithms tailored to the characteristics of the dataset. Furthermore, this study contributes to the broader understanding of machine learning-based techniques for audio classification tasks, paving the way for future research and applications in wildlife and conservation studies.

References

Chi, Z., Li, Y., and Chen, C. Deep convolutional neural network combined with concatenated spectrogram for environmental sound classification. pp. 251–254, 2019. doi: 10.1109/ICCSNT47585.2019.8962462.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

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

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Lezhenin, I., Bogach, N., and Pyshkin, E. Urban sound classification using long short-term memory neural network. In *2019 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pp. 57–60, 2019. doi: 10.15439/2019F185.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Sang, J., Park, S., and Lee, J. Convolutional recurrent neural networks for urban sound classification using raw waveforms. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 2444–2448, 2018. doi: 10.23919/EUSIPCO.2018.8553247.

Vashishtha, S., Narula, R., and Chaudhary, P. Classification of musical instruments' sound using knn and cnn. In *2024 11th International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 1196–1200, 2024. doi: 10.23919/INDIACom61295.2024.10498772.

Zaman, K., Sah, M., Direkoglu, C., and Unoki, M. A survey of audio classification using deep learning. *IEEE Access*, 2023.

A. Code

https://drive.google.com/drive/folders/1irJ15Hah906U0nPN1nvR2uVXyPeFprGB?usp=drive_link

B. Dataset

https://www.kaggle.com/competitions/birdclef-2023/data

C. Acknowledgements

We extend our deepest gratitude to the data processing template, the sample code and the librosa library for audio data processing.