

Birdcall Identification Using CNN and Gradient Boosting Decision Trees with Weak and Noisy Supervision

Naoki Murakami¹, Hajime Tanaka² and Masataka Nishimori³

¹Graduate School of Information Science and Technology, University of Tokyo, 7-3-1, Hongo, Bunkyo-ku, Tokyo, Japan

²Program of Medicine, School of Medicine, Hiroshima University, 1-2-3 Kasumi, Minami-ku, Hiroshima 734-8551, Japan

³MNES, 1-2-27, Honmachi, Shinonome, Minami-ku, Hiroshima City, Hiroshima, Japan

Abstract

We present a winning solution to the BirdCLEF2021 challenge, in which we detected bird calls from a sound source and classified the type of birds successfully. Our method consists of a multi-stage pipeline using CNN and Gradient Boosting Decision Trees. Using a unique labeling scheme, our models can learn efficiently from data with noisy and weak labels. By transforming the sound recognition task into a time-series tabular data classification task, the model of our method can learn various information such as the preceding and following sounds, regional characteristics, and seasonality. In addition, it can learn with fewer computational resources. The final submission of our method achieved a row-wise micro averaged F1 score of 0.7620 for the public test set and 0.6932 for the private test set. Our method is versatile enough to be applied to tasks on other time-series data that are labeled only for the whole. This reduces the cost of annotations required for training.

Keywords

Bird Species Recognition, BirdCLEF2021, Machine Learning, CNN, Gradient Boosting Decision Trees, Weak Label, Noisy Label

1. Introduction

The ecology of birds, including their habitats, has a huge relationship with the quality of their environment. By identifying which areas birds appear in, it is possible to track current biodiversity trends. However, to keep tracking them on a large scale is not easy to do because of the burden on observers [1].

BirdCLEF2021 [2], one of the LifeCLEF challenges[3], was a contest to identify bird calls from recordings of soundscapes by training a model mainly using bird sound sources. The better the model it was, the more accurately it would be able to detect bird calls and classify the type of birds from large amounts of audio.

One of the difficulties of this contest was that the domains of the training data and the test data were different. Most of the training data, we call *train_short_audio*, consisted of recordings

CLEF 2021 – Conference and Labs of the Evaluation Forum, September 21–24, 2021, Bucharest, Romania

✉ murakami-naoki167@g.ecc.u-tokyo.ac.jp (N. Murakami); b164181@hiroshima-u.ac.jp (H. Tanaka); nishimori-m@mnes.org (M. Nishimori)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

and metadata of 397 different bird calls uploaded by users to <https://www.xeno-canto.org/>. The peculiarities of the dataset posed the following difficulties:

- Due to incomplete labeling, it was unclear which of the labeled birds was singing within each short segment.
- The label given itself might have been wrong.
- The measure of whether a bird was singing or not was unclear.

Other data provided by BirdCLEF2021 were *train_soundscape*s and *test_soundscape*s. These data contained soundscape recordings from four different recording locations. The major difference between *train_short_audio* and *train/test_soundscape*s were their recording environments and labeling method. Speaking of the labeling method, *train_short_audio* was labeled for the entire sound source, but *train/test_soundscape*s were annotated every 5 seconds. Participants in this contest needed to accurately predict which of 397 different birds were singing and which were not in a 5-second segment. Participants also had to predict *nocall* if none of the birds were singing. However, a simple sound recognition model could not fully handle this challenge because not only one segment but also the segments before and after it might have been meaningful.

There are many different ways to perform sound recognition with deep learning. Some use raw, continuous data of audio sources as is [4], while other approaches transform audio data into images and then use Convolutional Neural Networks (CNN). For example, the audio can be transformed into a spectrogram using the Fourier transforms or other techniques and then trained with CNN [5].

However, in this contest, using only audio data was not the best learning method. It was difficult to make a decision based on the sound alone since it included many factors such as dialect, degree of mastery of the sound, and similar birds. It was possible to classify with high accuracy by taking into account regional and seasonal characteristics, environmental sounds, and segments before and after the audio comprehensively.

One of the most promising methods for learning such meta-information and time-series information as tabular data is Gradient Boosting Decision Trees (GBDT) [6]. It is characterized by its ability to learn quickly and predict something with high accuracy, and there are efficient implementations such as XGBoost [7], Lightgbm [8], and CatBoost [9].

In this paper, we present a three-step method that achieved highly accurate prediction by learning from sound and metadata. The models of the first stage were used for training of the second and third stages, and the models of the second and third stages were used for inference (Figure 1). Our method has a low learning cost and is versatile enough to be applied to other tasks on time-series data that are labeled only for the whole.

2. Methods

We describe the details of our pipeline consisting of three stages. First, Section 2.1 gives details about the dataset to be used and presents what data will be used at which stage. In Section 2.2, we explain the details of the *nocall* detector in the first stage. The predictions of this model will be used during the training of the second and third stages. Section 2.3 describes the second stage

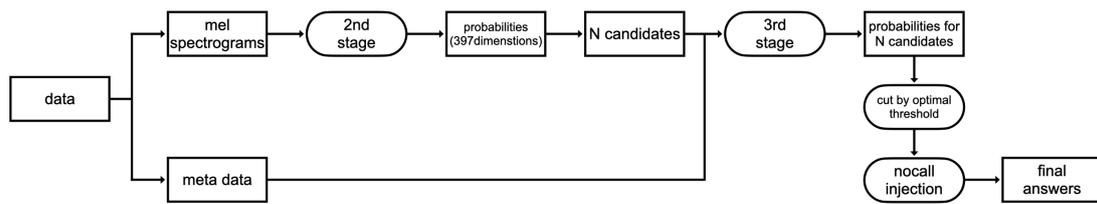


Figure 1: Inference Pipeline of the proposed method. The model of the first stage was used for training of the second and third stages, so it was not used for inference. In the second stage, CNN was used to predict the probability of occurrence for 397 bird species. In the third stage, GBDT was used to predict whether the candidate extracted in the second stage was the correct answer.

model for multi-label classification with CNN using only audio converted to images. Section 2.4 describes the third stage model, which is trained with Gradient Boosting Decision Trees using metadata and predictions from the second stage. In Section 2.5, we describe a threshold optimization using ternary search and a method that mixes *nocall* and bird labels in the final prediction to improve the score while expressing ambiguity, which we call *nocall injection*.

2.1. Data Sets

This section describes the data given in BirdCLEF2021 and the external data sets we used.

Training Data. BirdCLEF2021 provided *train_short_audio* and *train_soundscape*s as training and validation data. *train_short_audio* contained the sounds of 397 bird species that had been uploaded by xeno-canto users. These files were re-sampled to 32 kHz to match the audio in the test set. The label was given only in audio units. While the primary label of the bird that appeared primarily in the audio was given for all audio, the secondary label of other birds that also appeared in the audio might have been missing. It was also possible that the labels themselves were wrong because the data was uploaded by xeno-canto users. It also contained meta-information about the source, such as the latitude and longitude of the recording, and the date. *train_soundscape*s was a set of 20 10-minute files annotated every 5 seconds. This data set was in the same format as the test data. Each file was recorded in either COR (Costa Rica) or SSW (Sapsucker Woods, Ithaca, New York, USA). The location and date of the recording were given.

Test Data. *test_soundscape*s had a total of 80 files consisting of 10 minutes of audio sampled at 32 kHz. Each file was recorded in either COR (Costa Rica), COL (Colombia), SNE (Sierra Nevada, California, USA), or SSW (Sapsucker Woods, Ithaca, New York, USA). The location and date of recording of these files were known. Participants were asked to predict what birds were singing in segments separated by 5 seconds for this audio data. This data was used in the contest to evaluate the results.

Table 1

Data sets used in each stage. BirdCLEF2021 provided *train_short_audio* and *train_soundscapes* as training and validation data. We used *freefield1010* and validation data of BirdCLEF2020 as external data.

Data Set	First Stage	Second Stage	Third Stage
<i>train_short_audio</i>	no	train	train
<i>train_soundscapes</i>	no	validation	validation
<i>freefield1010</i>	train and validation	train	no
validation data of BirdCLEF2020	no	train	no

External Data. Since further improvement in accuracy could be achieved by using data other than that provided by BirdCLEF2021, we used two additional external data sources. The first was *freefield1010*, which was annotated again in the Bird Audio Detection challenge [10], instead of the originally published annotation of 7,690 10-second audio samples [11]. The data without birdsong was divided into 5,755 segments, while the data with birdsong was divided into 1,935 segments, and every 10 seconds of audio was given a label. The second was the validation data in BirdCLEF2020 [12]. There was a total of 2 hours of audio data, but we split it into 5-second segments and used only the segments where no bird calls appeared.

Pre-processing. Table 1 shows the stages in which the data was used. All datasets were divided into segments of 5 or 7 seconds. Using as long a segment as possible increases the likelihood that the birdcall will be included in the segment and will match the label added to the entire audio. Therefore, we decided to use 7-second segments when training the first and second stage models. For the BirdCLEF2020 validation data, we used segments divided into 5-second segments, since the data was labeled every 5 seconds. The training data was converted to mel spectrograms in advance. Although this had the disadvantage that data augmentation was not possible for the original audio data, it was superior in that it reduced the cost of converting the audio to a mel spectrogram during training time, and allowed faster training. We used 128 mel-scaled frequencies in the range of 0Hz to 16000Hz. When converting the audio to a mel spectrogram, we set the length of the FFT window to 3200 and the hop length to 800. For example, a 7-second segment would be converted to an image with a resolution of 128x281 (frequencies × time).

2.2. First Stage

In this stage, we created *nocall* detectors using ResNeXt50 [13], which is a kind of CNN-based model. In our pipeline, the *nocall* detector was used in two ways:

- To weight the labels of segments that were predicted to have no birds singing during the second stage of training.
- To create labels for training during the third stage of training.

As mentioned in 2.1, the data given by BirdCLEF2021 was not sufficient, so we used *freefield1010* [11] to create the *nocall* detector. The sound source was given every 10 seconds, but only the sound source for 7 seconds from the start of the audio was used. This was because the

train_short_audio used for learning in the second and third stages were separated every 7 seconds. Since the data set was converted to images beforehand, it became difficult to use the data augmentation used in normal sound recognition, such as adding noise. However, we used the usual augmentations for images, such as flip, normalization, decreasing JPEG compression of an image, to improve the generalization performance.

The model was trained using the ADAM optimizer and a Cosine Annealing Warm Restarts scheduler.

2.3. Second Stage

We detected and classified the birds that appeared in the segment. Multi-label classification was performed on the transformed mel spectrogram using CNN-based ResNeSt50 [14].

Data. For the training data, we used *train_short_audio* separated by 7 seconds. In the case of long audio, only the first 10 segments were used. For some models, we also used *freefield1010* and the validation data from BirdCLEF2020. For these data, only the *nocall* data was used, and all 397 dimensions of the target were set to 0. For the validation data from BirdCLEF2020, since it was separated every 5 seconds, we padded the mel spectrogram with zeros before and after (in random ratios) during training so that it could be converted to a 128x281 image. Data augmentation was difficult to apply to raw sound because the dataset had been converted to images beforehand. Instead, the mixup [15] was used as data augmentation. This would improve the generalization performance and also allowed us to better recognize that multiple birds might be singing at the same time.

Labels. We used the *nocall* detector created in the first stage to modify the labels for the data used for training in this stage. Specifically, each segment was multiplied by the output value of the first stage (the closer to 0, the fewer birds were singing, and the closer to 1, the more likely multiple birds were singing), and the weight of the segment with no birds singing was set to be smaller. In addition, for datasets with secondary labels, we used a value of 0.6 times the predicted value of the *nocall* detector as the target. This was because birds with secondary labels were considered to occur less frequently than birds with primary labels, and therefore might not have appeared throughout the audio.

The models were trained using the ADAM optimizer and a Cosine Annealing LR scheduler. The batch size was 100, and label smoothing [16] was performed to adjust the target value to a maximum of 0.995 and a minimum of 0.0025. To avoid overfitting on noises specific to the equipment used to record the sound, some models were trained and cross-validated using the data grouped by the authors so that it did not appear commonly in the training and validation data. In the next stage, we used the model from the epoch with the highest row-wise F1 score on *train_soundscape*s. Details of the evaluation method are found in Section 3.1.

As the data set had weak labels, it was not enough to fix the labels using the first stage models. In the second stage, we still had the following issues:

- It was unknown which bird was singing in that segment (primary or secondary labels).

- It was only known with probability whether a bird was singing in that segment.

Since we needed a mechanism to determine which bird was singing with high accuracy while reducing noise to some extent, we used the third stage to sort the prediction results of the second stage.

2.4. Third Stage

In this stage, based on the results of the second stage, we extracted the top N (we used 5) bird species that were likely to be singing in each segment, and combined information other than singing (date, latitude, longitude, and whether the bird was singing in the previous or following segment). LightGBM was trained to perform binary classification. The closer the output was to 0, the less likely it was that the target bird was singing in the target section, and the closer it was to 1, the more likely it was. Several new features were generated by transforming the metadata.

To solve the problem of noisy labels in the second stage, we created a problem in which the correct answer was the intersection of the following "Correct answer set" and the "Candidate set." In this way, we converted the task to one in which the correct labels could be automatically assigned with high accuracy while taking advantage of the characteristics of the second stage model using image features.

- Correct answer set: Primary labels and secondary labels for segments judged to be singing, *nocall* for segments judged not to be singing.
- Candidate set: Top N values of the probability of each bird singing as predicted by the second stage model from the mel spectrogram ($N = 5$ was adopted).

Taking advantage of the fact that both candidates and correct answers rarely had anything in common, this method allowed us to automatically annotate with high accuracy.

By converting the data into tabular data for each segment divided into several-second intervals, we could make predictions that took into account various information such as time-series, seasonality, and regionality in the segment. Details of the features are found in Appendix A.

2.5. Post-processing

After the completion of binary classification in the third stage, we determined the optimal threshold value by ternary search, and performed *nocall injection* to improve the score while leaving ambiguity by mixing *nocall* and bird labels in the final prediction.

Threshold Optimization. A threshold was set for the predicted probability of binary classification in the third stage. Candidates with probability values higher than the threshold were included in the final output, and if none of the candidates were included in the segment, we called it *nocall*. Assuming that the score was a convex function of the threshold, the threshold was set to the highest score by ternary search, and since the optimal threshold range was multiplied by $2/3$ each time the score was calculated, it was possible to determine the optimal threshold after several dozen score calculations. The optimal threshold was calculated using

*train_soundscape*s, and the same value was used when inferring *test_soundscape*s.

Nocall Injection. After determining the threshold, we decided whether to add *nocalls* to the segments that did not become *nocalls*. Specifically, for the birds that were decided to be included in the final output by setting the threshold, *nocall* was added to the rows with low total predicted probability. This had the following two advantages:

- It was able to unambiguously predict what was not highly predictable in the third stage.
- Due to the nature of the row-wise micro averaged F1 score, mixing *nocall* and bird labels might increase the score.

As an implementation, we thought it would be convex here as well and used ternary search.

3. Experiment

3.1. Evaluation Metrics

The BirdCLEF2021 contest dealt with the task of identifying all singing birds in segments of 5 seconds each. Submissions were evaluated based on their row-wise micro averaged F1 score. The F1 score is calculated as follows:

$$F1 = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} = \frac{tp}{tp + 1/2(fp + fn)}$$

Until the end of the contest, only the score calculated using 35% of the data (public score) was shown, while the remaining 65% of the data (private score) was hidden. The final ranking was determined using the private score.

3.2. Results

First Stage. The training of one model took about 90 seconds per epoch, using a Tesla P100 for the GPU. Models were trained for 10 epochs, and the one with the best score was selected. Cross-validation was performed by dividing the data into five parts with equally distributed labels, and the average percentage of correct answers was 0.89857.

Second Stage. Ten ResNeSt50s were trained, varying on the following conditions:

- mixup α
- Whether to cross-validate the *nocall* detector (No: Hold-out, Yes: CV)
- Whether to use BirdCLEF2020 validation data in training or not
- Whether or not to use *freefield1010* during training
- Whether to group authors so that they do not appear in different folds. (No: StratifiedKFold, Yes: StratifiedGroupKFold)
- The fold id of *train_short_audio* used for verification. If the data were split by grouped authors, we split the data into 5 parts and assigned 0-4 ids. When the data was not split by author, the data was also split into 5 parts and given the id of 0'-4'.

Table 2

Second Stage Models. Scores and main properties of models used for submitted runs. ResNeSt50 was used as the backbone for all models. The F1 score was calculated for *train_soundscape*s. The F1 score, calculated as the simple average of the predictions of the models from M1 to M10, was 0.788.

Model ID	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
mixup α	0.5	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0
use BirdCLEF2020	No	Yes	Yes	Yes						
use ff1010	No	Yes	Yes	Yes						
<i>nocall</i> detector cv	No	No	Yes							
grouped by author	No	No	Yes							
validation fold id	0'	0'	0	1	2	3	4	0	0	1
epochs	27	13	33	34	34	20	34	78	84	27
F1	0.741	0.750	0.755	0.737	0.750	0.755	0.737	0.751	0.757	0.754

For each model, the threshold was determined by ternary search, and the epoch with the best F1 score during validation in *train_soundscape*s was used in the final submission. More details about the condition settings when training the model and the number of epochs and scores are shown in Table 2. The F1 score, calculated as the simple average of the predictions of the models from M1 to M10, was 0.788.

We trained the models on Tesla P100 or V100 GPUs; for ResNeSt50, it took 6-7 minutes per epoch on P100. Although we did not include ResNeSt26 in the final submission for the contest, ResNeSt26 performed better than ResNeSt50 in the second stage and required less training time (4-5 minutes per epoch) on P100.

Third Stage. For training, we used the inference results of M4 in the second stage and metadata for the validation data of *train_short_audio*. When extracting candidates, we selected the top 5 candidates with the highest prediction probability from the inference results for 397 bird species in the second stage. For inference, we used the simple average of the 10 models in Table 2 to improve accuracy.

It took about 15 minutes to complete the submission, including learning the tabular data.

When the threshold was optimized, the CV score was 0.833286 and after the *nocall injection* it was 0.837063. The row-wise micro averaged F1 score for the final submission achieved 0.7620 for the public test set and 0.6932 for the private test set.

3.3. Discussion

The uniqueness of our work was that we used the intersection for the labels for entire audio files and the segment-level candidates, as the new segment-level labels to train the tabular data in the third stage. This had the following advantages:

- Dealing with weak and noisy supervision: It could be used to learn accurately from data that was only labeled in audio units or contained errors.

- Taking into account time-series information: Image classification task could be converted to a time-series tabular data classification task, improving accuracy.
- Taking into account metadata: The accuracy was improved by being able to incorporate a variety of information such as before/after information, regional characteristics, and seasonality.
- Reducing computational cost: Our solution was fast, which saved time and computational resources in learning and inference.

Although our method performed well in the contest, there was still room for improvement. For example, it could learn effectively even with noisy and weak labels, but the accuracy could have been further improved by using more data that was accurately annotated.

For the first stage, in addition to the addition of other data, further performance improvement could have been achieved by changing the CNN model and ensembles.

For the second stage, due to time constraints, we were not able to incorporate another highly accurate model that we tried, such as ResNeSt26. In addition, only ResNeSt50 was used, and other models were not used in the ensemble.

For the third stage, the second stage models used for training and inference were different. That is, there was only one model used to infer the probability values for *train_short_audio* and 10 models used to infer the probability values for *train_soundscapes*. The fact that these models did not match might cause differences between the training data and inferred data in the third stage, so unifying them could have further improved the accuracy. When extracting candidates, the top five candidates with the highest prediction probability for each segment were extracted, which did not take into account the fact that the ease of appearance of birds differed for each segment. It might have been possible to extract a large number of candidates for segments where many birds were likely to appear, and a smaller number for segments where not so many birds were likely to appear. There was also room for improvement in feature engineering. For example, since there were interactions (co-occurrences) between birds, it might be possible to add new features that included relevant information such as birds that tended to sing together or birds that coexisted.

Since our method could learn to include information from the previous and following segments, as well as meta-information such as region and time, further improvement could be expected depending on how the features were created.

4. Conclusion

We have presented a method for audio recognition consisting of a multi-stage pipeline. The first stage is a binary classification of whether a bird is singing or not for label correction. The second stage is to predict which bird is singing from the sound source. In the third stage, we select the candidates extracted in the second stage.

By building a multi-stage pipeline and devising a semi-automatic labeling scheme, it is possible to learn effectively even from weak and noisy teacher data. Our method is superior in terms of computational cost, yet can learn accurately from data in different domains that are labeled only per audio. Our semi-automatic labeling method in the third stage makes it possible to label segments of audio by simply annotating the entire audio. Therefore, assuming that our

method is used, the cost of annotation can be reduced. It is a great deal of effort for a person to annotate each segment in detail, and the annotation itself is prone to have errors. Therefore, it is a great advantage to reduce the effort required for annotation. In addition, our method may be able to detect preliminary sounds of the bird itself (e.g., the sound of rustling in the bushes), even if the bird itself is not singing. This is difficult for humans to determine, so if we can identify them as candidates, they may be useful for future research.

Our method is also versatile enough to be applied to other tasks on time-series and series data labeled to the whole. For example:

- Identifying a lesion slice from multiple slices in MRI.
- Identifying which scenes were good from video ratings.

Acknowledgments

We would like to express our sincere gratitude to many people for their support in conducting this research. First of all, we would like to thank The Cornell Lab of Ornithology for organizing the BirdCLEF2021 Kaggle competition. Without them, we would not have been able to start this research. Secondly, we would also like to thank the other competitors who provided us with many suggestions during the competition. Finally, we would like to express our gratitude to Ms. Sue Goldberg who works for the University of Cincinnati Foundation, Professor Hiroshi Imai at Graduate School of Information Science and Technology, University of Tokyo, and Dr. Tetsuya Tanimoto at Medical Governance Research Institute in Tokyo, Japan, for proofreading this paper.

References

- [1] W. D. Robinson, A. C. Lees, J. G. Blake, Surveying tropical birds is much harder than you think: a primer of best practices, *Biotropica* 50 (2018) 846–849. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1111/btp.12608>. doi:<https://doi.org/10.1111/btp.12608>.
- [2] S. Kahl, T. Denton, H. Klinck, H. Glotin, H. Goëau, W.-P. Vellinga, R. Planqué, A. Joly, Overview of BirdCLEF 2021: Bird call identification in soundscape recordings, in: *Working Notes of CLEF 2021 - Conference and Labs of the Evaluation Forum, 2021*.
- [3] A. Joly, H. Goëau, S. Kahl, L. Picek, T. Lorieul, E. Cole, B. Deneu, M. Servajean, R. Ruiz De Castañeda, I. Bolon, H. Glotin, R. Planqué, W.-P. Vellinga, A. Dorso, H. Klinck, T. Denton, I. Eggel, P. Bonnet, H. Müller, Overview of LifeCLEF 2021: a system-oriented evaluation of automated species identification and species distribution prediction, in: *Proceedings of the Twelfth International Conference of the CLEF Association (CLEF 2021), 2021*.
- [4] Y. Aytar, C. Vondrick, A. Torralba, Soundnet: Learning sound representations from unlabeled video, in: D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, volume 29, Curran Associates, Inc., 2016. URL: <https://proceedings.neurips.cc/paper/2016/file/7dcd340d84f762eba80aa538b0c527f7-Paper.pdf>.
- [5] Brian McFee, Colin Raffel, Dawen Liang, Daniel P.W. Ellis, Matt McVicar, Eric Battenberg, Oriol Nieto, *librosa: Audio and Music Signal Analysis in Python*, in: Kathryn Huff, James

- Bergstra (Eds.), Proceedings of the 14th Python in Science Conference, 2015, pp. 18 – 24. doi:10.25080/Majora-7b98e3ed-003.
- [6] J. H. Friedman, Greedy function approximation: A gradient boosting machine., *The Annals of Statistics* 29 (2001) 1189 – 1232. URL: <https://doi.org/10.1214/aos/1013203451>. doi:10.1214/aos/1013203451.
- [7] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 785–794. URL: <https://doi.org/10.1145/2939672.2939785>. doi:10.1145/2939672.2939785.
- [8] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu, Lightgbm: A highly efficient gradient boosting decision tree, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, volume 30, Curran Associates, Inc., 2017. URL: <https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf>.
- [9] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorigush, A. Gulin, Catboost: unbiased boosting with categorical features, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, volume 31, Curran Associates, Inc., 2018. URL: <https://proceedings.neurips.cc/paper/2018/file/14491b756b3a51daac41c24863285549-Paper.pdf>.
- [10] D. Stowell, M. D. Wood, H. Pamula, Y. Stylianou, H. Glotin, Automatic acoustic detection of birds through deep learning: The first bird audio detection challenge, *Methods in Ecology and Evolution* 10 (2019) 368–380. URL: <https://besjournals.onlinelibrary.wiley.com/doi/abs/10.1111/2041-210X.13103>. doi:<https://doi.org/10.1111/2041-210X.13103>.
- [11] D. Stowell, M. Plumbley, An open dataset for research on audio field recording archives: freefield1010, *Journal of the Audio Engineering Society* (2014).
- [12] S. Kahl, M. Clapp, W. A. Hopping, H. Goëau, H. Glotin, R. Planqué, W.-P. Vellinga, A. Joly, Overview of BirdCLEF 2020: Bird Sound Recognition in Complex Acoustic Environments, in: *CLEF 2020 - 11th International Conference of the Cross-Language Evaluation Forum for European Languages*, Thessaloniki, Greece, 2020. URL: <https://hal.inria.fr/hal-02989101>.
- [13] S. Xie, R. Girshick, P. Dollár, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, in: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5987–5995. doi:10.1109/CVPR.2017.634.
- [14] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, H. Lin, Z. Zhang, Y. Sun, T. He, J. Mueller, R. Manmatha, et al., Resnest: Split-attention networks, arXiv preprint arXiv:2004.08955 (2020).
- [15] H. Zhang, M. Cisse, Y. N. Dauphin, D. Lopez-Paz, mixup: Beyond empirical risk minimization, in: *International Conference on Learning Representations*, 2018. URL: <https://openreview.net/forum?id=r1Ddp1-Rb>.
- [16] R. Müller, S. Kornblith, G. E. Hinton, When does label smoothing help?, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, volume 32, Curran Associates, Inc., 2019. URL: <https://proceedings.neurips.cc/paper/2019/file/f1748d6b0fd9d439f71450117eba2725-Paper.pdf>.

A. Features used in third stage

- "year": The year when the audio was recorded.
- "month": The month in which the audio was recorded.
- "sum_prob": The sum of the probability values of the 397 dimensions for the segment.
- "mean_prob": Mean of the 397 dimensional probability values for the segment.
- "max_prob": Maximum 397-dimensional probability values for the segment.
- "prev_prob" - "prev6_prob": Probability values for the candidate birds in the previous N segments.
- "prob": Probability value for the candidate bird in the target segment.
- "next_prob" - "next6_prob": Probability values of the candidate bird in the next N segments.
- "rank": The probability that the bird of interest has the highest probability value out of 397 in the segment.
- "latitude": The latitude of the location where the audio was recorded.
- "longitude": The longitude of the location where the audio was recorded.
- "bird_id": Which of the 397 birds is the target.
- "seconds": The number of seconds in the segment.
- "num_appear": The number of files where the bird is the primary label in *train_short_audio*
- "site_num_appear": How many times the bird in *train_short_audio* appeared in the region
- "site_appear_ratio": "site_num_appear" divided by "num_appear"
- "prob_diff": Difference between prob and average of prob values for 3 segments

B. Online Resources

The source code is available via GitHub: <https://github.com/namakemono/kaggle-birdclef-2021>.