Methods for Training Convolutional Neural Networks to Identify Bird Species in Complex Soundscape Recordings

Konstantin Dmitriev[1,*]

[1]Lomonosov Moscow State University, 1 Leninskie Gory, Moscow, 119992, Russian Federation

Abstract

The task of bird species identification is very important in ecosystem monitoring. Modern methods based on the use of deep learning will allow such research to be carried out cheaply and on a regular basis. However, creating such algorithms is not an easy task due to the wide variety of birds, their calls, recording conditions, and equipment used. In this paper, some methods are presented for training Convolutional Neural Networks (CNNs) that improve the effectiveness of these models. This includes recording length standardization, data augmentation, mixing, sample selection, and weighting.

Keywords

Audio classification, sound event detection, signal processing, convolutional neuron network, augmentations, spectrogram

1. Introduction

Bird species diversity and its change in time serve as good indicators of ecosystem state. Traditional methods of monitoring require the presence of a qualified observer who can manually identify the bird. It's quite hard and expensive to conduct such surveys regularly, especially in the case of large areas. Many birds are small and hard to notice, but they have a loud voice. So, it seems promising to use small and cheap audio recording devices with omnidirectional microphones instead of human observers and to process the recordings using modern methods based on deep learning.

Creating and training such algorithms is a difficult task, however. The first problem is the diversity of bird species and their recording conditions. There are many birds that can imitate other birds or even repeat a sound they once heard and liked. Many animals and insects sound like birds. The second problem is the difference between the available training data and the real recordings to be processed. Usually, the training data is a set of short bird call recordings made by different people at different locations. They try to make the recordings clean and loud, without noise or interference. So, good equipment is used, including directional microphones, and bad recordings are dropped. The third problem is that only weak labels are given that identify the bird's existence in each recording but not the exact call position.

BirdCLEF 2024 is a competition that is supposed to address the mentioned problems [1, 2]. It is a part of the LifeCLEF 2024 conference [3]. The task is to identify bird calls in a set of recordings made in the Western Ghats, India. The training dataset consists of recordings from the xeno-canto project [4]. Each of them has primary and secondary labels. The primary label corresponds to the main bird that can be heard, and the secondary labels are used to mark additional birds that can accompany the main bird. 182 bird species, whose existence needs to be predicted, were selected by the organizers. The predictions must be made for each of the 5-second-long intervals of about 1100 recordings that form the hidden dataset. An additional constraint is that the task must be completed in 2 hours using CPU only. The macro-averaged AUC ROC that discards classes with no true labels was used as a metric in the competition. To prevent overfitting, the full hidden dataset is split into the public and private parts

CLEF 2024: Conference and Labs of the Evaluation Forum, September 09–12, 2024, Grenoble, France

*Corresponding author.

presentatio@mail.ru (K. Dmitriev)

0000-0001-5842-383X (K. Dmitriev)

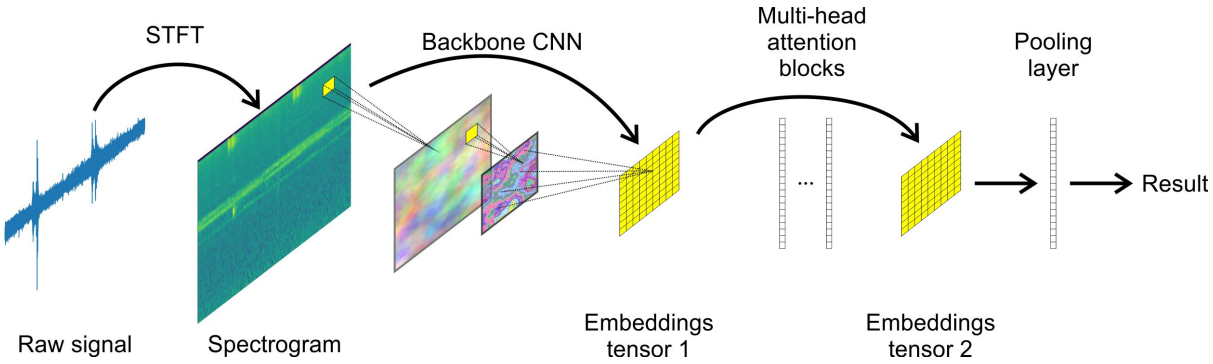

Figure 1: Model architecture.

Table 1
The CV accuracy score calculated using xeno-canto data for the models with different parameters: the backbone, the embedding dimension D, the number H of attention heads, and the number B of multi-head attention blocks.

resnet18, $H = 8, B = 2$		resnet18, $D = 768, B = 2$		resnet18, $D = 768, H = 8$		$D = 768, H = 8, B = 2$	
D	Accuracy	H	Accuracy	B	Accuracy	Backbone	Accuracy
256	0.755	1	0.651	0	0.604	resnet18	0.741
512	0.748	2	0.693	1	0.697	**seresnext26t_32x4d**	**0.852**
768	0.741	3	0.703	2	0.741	seresnext50_32x4d	0.849
1024	0.661	8	0.741	3	0.721	eca_nfnet_l0	0.679
1536	0.650	16	0.734	4	0.611	efficientnet_b0	0.695
2048	0.520	32	0.714			resnext50_32x4d	0.785
		64	0.779				

(approximately 35% and 65% of the data, respectively). The corresponding public and private scores are calculated independently, and only the public score is known at the competition time.

This article presents the methods that can be used to overcome the aforementioned difficulties and improve the results of bird call recognition.

2. Methods

2.1. Model architecture

The used model is based on the model proposed in [5]. Its scheme is presented in Fig 1. After the signal is loaded and normalized, the spectrogram transform is performed. Then, it is fed to a backbone CNN, followed by B multi-head attention blocks. Finally, a log-sum-exp pooling layer is used to extract the label. The multi-head attention mechanism is described in the paper [6], and it is implemented in PyTorch by the `torch.nn.MultiheadAttention` class. Its main parameters are the number H of parallel attention heads in each of the blocks and the embedding dimension D.

To find the best combination of the model parameters, a number of tests were conducted. Instead of using the macro-averaged AUC ROC score, which became very close to one after a few epochs, the accuracy score was used in a 5-fold cross-validation (CV) scheme. In the tests conducted, the backbone as well as the values B, H, and D were varied. The results are presented in Table 1.

The simple resnet18 [7] backbone was used to check different parameters. Using it, the best results were achieved with $D = 256, H = 64$ and $B = 2$. This slightly differs from the parameters presented in [5], where they were set as $D = 768, H = 8$ and $B = 2$. The results significantly improve with heavier backbones, among which seresnext26t_32x4d [8] seems the best.

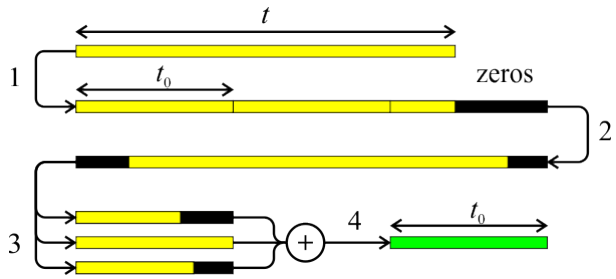

Figure 2: The "Sum" approach to making the lengths of recordings equal.

Table 2
The comparison of different approaches

t_0	"First" approach		"First and last" approach		"Sum" approach	
	Public score	Private score	Public score	Private score	Public score	Private score
5	0.606	0.584	–	–	0.595	0.560
10	0.616	0.563	0.611	0.569	0.607	0.566
20	0.586	0.572	0.599	0.561	0.607	0.571
30	0.614	0.570	–	–	0.620	0.582

2.2. Dealing with different lengths of the recordings

An important problem with the recordings is that they have different lengths. The shortest of them lasts only 0.5 seconds, while the longest is more than 1.5 hours. Different lengths don't allow using these recordings in batches while training the model. This causes the training to be slow.

There are several possible ways to overcome this difficulty. Let t_0 be the fixed final length of each recording processed. If the initial recording is shorter, then it is simply padded with zeros. If it is longer, the "First" approach is to use only the first t_0-long interval of the recording. The "First and last" approach is to use the first $t_0/2$ and last $t_0/2$ intervals stacked together. This seems reasonable because, as a rule, the recordings were processed by their authors before being uploaded to the website. So, one can suspect that irrelevant sounds were cut off from the beginning and the end of each recording. These two approaches are quite popular among the competition solutions. However, a lot of information is lost. Instead, the third approach is proposed, which is called the "Sum" approach and illustrated in Fig 2. It consists of the following steps.

1. Each recording, having length t, is padded with $t_z = t - t_0 n_{\text{int}}$ zeros, where $n_{\text{int}} = \lceil t/t_0 \rceil$, i.e., the resulting recording contains a whole number of intervals with the length of t_0.
2. As an augmentation, a random circular shift is performed.
3. The recording is split into n_{int} intervals, and they are summed together. The length of the result is equal to t_0.

There is no information loss in the third approach. The overlapping of bird calls that may occur doesn't seem to be a problem since it corresponds to a situation when many birds vocalize at the same time.

The same model (seresnext26t_32x4d; $D = 768, H = 8, B = 2$) was trained using all the described approaches with different t_0 values. Every training was repeated three times with different random seeds, and the "best" of them with the highest score on the public dataset was selected. The resulting public and private scores are presented in Table 2.

The results produced with different approaches are close to each other. However, the scores of the "First" approach are slightly better with low t_0 values. The growth of t_0 doesn't improve the scores of the "First" and "First and last" approaches, but the scores of the "Sum" approach increase, and it becomes preferable with large t_0. At the same time, increasing t_0 makes the model training longer.

Table 3
The score of the model in different groups of bird species.

Group	N_{rec}	Number of bird species	Public score	Private score
1	$0 < N_{\text{rec}} \leq 20$	34	0.543	0.536
2	$20 < N_{\text{rec}} \leq 50$	47	0.667	0.610
3	$50 < N_{\text{rec}} \leq 100$	27	0.652	0.585
4	$100 < N_{\text{rec}} \leq 200$	33	0.672	0.603
5	$200 < N_{\text{rec}} \leq 500$	41	0.569	0.569

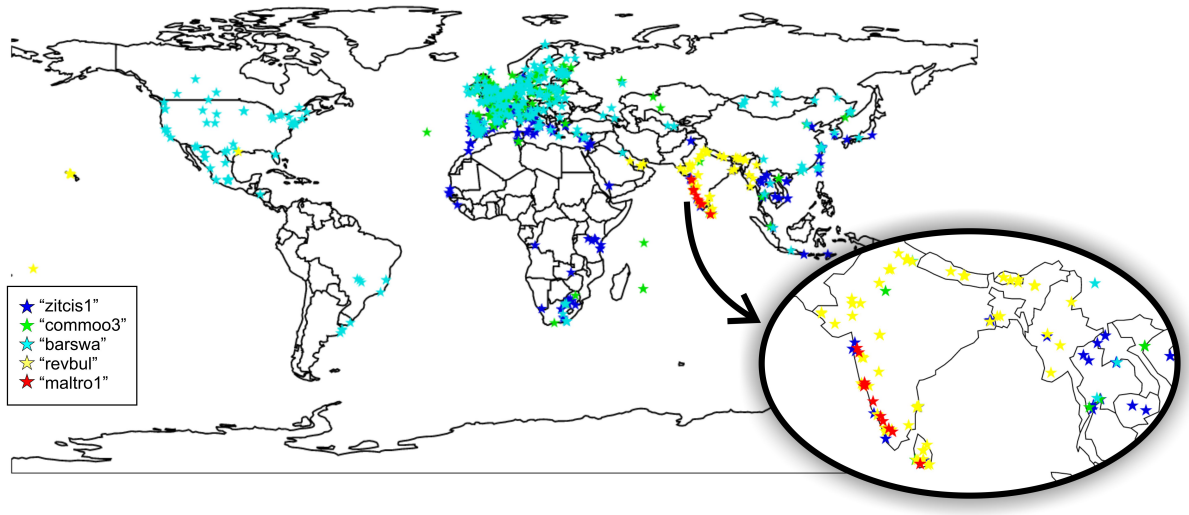

Figure 3: The locations where the recordings were made.

2.3. Train data selection

The train dataset suggested at BirdCLEF 2024 competition contains approximately half of all the recordings from xeno-canto project [4], with primary labels corresponding to 182 target birds. So, the obvious step is to download the absent data and create an additional dataset. Merged together, these two datasets contain about 40 000 recordings.

Using the whole dataset, however, doesn't improve the model score. It seems strange because the higher diversity of data usually causes an increase in the generalization ability of the model. So, one may expect that the additional data corrupts the dataset somehow, making it not correspond to the hidden dataset.

To find out the reason for the described behavior, the AUC ROC probing technique is proposed. This technique is based on splitting the target data into several parts and masking the model predictions so that only one part of the data is scored each time. In the BirdCLEF competitions, the bird species can be grouped together. For example, let $N = 182$ be the total number of bird species. One can select a group of $0 < n < N$ bird species. During the submission, the predictions corresponding to the rest $N - n$ species are set to zero. The constant prediction produces an AUC ROC score of 0.5. So, the resulting AUC ROC score S is equal to $S = (S_n n + 0.5(N - n))/N$, and the AUC ROC score S_n of the selected group is equal to $S_n = (SN - 0.5(N - n))/n$. Using this simple formula, it is possible to estimate the model performance for different groups of species. If these scores differ significantly and the size of each group is large enough, one may assume that the feature used for group selection is important.

One of the possible features that can affect the model's performance is the number N_{rec} of available recordings of each bird species, i.e., the frequency of its occurrence in the dataset. Following the proposed AUR ROC probing technique, the species were split into five groups (Table 3). It can be

assumed that the model will work well for those birds for which the training set contains many records and poorly in the other case. However, the situation is different. Group 5 with at least 200 recordings of each bird species, has almost as low scores as Group 1 with a maximum of 20 recordings.

One can conclude that, for some reason, the model has significant difficulties when dealing with common birds. One of the reasons for this is that common birds may be present in many recordings in the background while they are not marked, even with secondary labels. Another possible reason is the geographical distribution of the places at which the recordings were made. Indeed, many common birds were recorded in Europe, America, or Africa, far away from the region of interest. Birds can have local dialects. Also, a bird, which is common in Europe, can be rare in India.

The geographic information can be easily taken into account since GPS coordinates are provided. The locations of places at which five bird species were recorded are presented in Fig 3. Here, "zitcis1", "commoo3" and "barswa" are the primary labels of common birds, with the number of recordings equal to 500 in the competition dataset. The fourth bird, "revbul" is medium-rare and has 101 recordings, while the fifth, "maltro1", is a rare bird with 17 recordings. However, it is noticeable that almost all recordings of common birds were made outside India, while "revbul" and "maltro1" are endemic birds. As a result, the number of recordings of all common and rare species that are made in the Indian region is quite low and significantly less than that of medium-rare birds. This observation explains the differences in model scores across different groups of species.

To handle this observation, the algorithm for train data selection is proposed, which consists of the following steps.

1. Prepare the whole dataset with all the recordings from the xeno-canto project [4] that contain the target bird species calls.
2. For each recording, calculate its distance L_{WG} to the Western Ghats region. This can be done, for example, by placing a large number of points in the Western Ghats region and calculating the minimal distance between these points and the point where the recording was made.
3. Specify the maximum distance L_{max} and drop all the recordings with larger distances.
4. Calculate the number of recordings $\widetilde{N}_{\text{rec}}$ for each bird species in the resulting dataset.
5. Calculate the distance weight w_L for each of the recordings. This weight is a decreasing function of L_{WG}. For example, $w_L = 1 + \cos(\pi L_{\text{WG}}/L_{\text{max}})$ may be used.
6. Calculate the class imbalance weight for each of the recordings. This weight is a decreasing function of $\widetilde{N}_{\text{rec}}$. For example, $w_{\text{imb}} = 1/\widetilde{N}_{\text{rec}}$ may be used.
7. The weight of each of the recordings in the final dataset is the product of distant and class imbalance weights: $w = w_L \cdot w_{\text{imb}}$.

The value of L_{max} is important. In the current competition, setting $L_{\text{max}} = 4000$ km was a good choice. From a geographical point of view, it allows to discard European, American, and most African data while covering Southern Asia. Using the lower L_{max} decreases the diversity of species, and with its higher value, the training set includes irrelevant data. As a result, the public score of the model worsens in both cases.

2.4. Additional noise sources

As it was mentioned earlier, there is a huge domain shift between the competition data used for training and for model scoring. The training dataset contains the recordings from the xeno-canto project [4]. As a rule, these are recordings of high quality. However, the model is supposed to work well with the data recorded with an omnidirectional microphone in a noisy environment. The unlabeled soundscapes dataset is provided with recordings similar to those used for model scoring.

Listening to the unlabeled recordings, it is possible to make a list of present noise sources. They include car traffic and horns, aircraft noises, sirens, human voices, music, frogs, and cicadas, as well as broadband noises of rain, wind, or even uncertain nature. The example spectrogram of such an unlabeled sounscape is presented in Fig 4. One has to introduce all these kinds of interferences to the

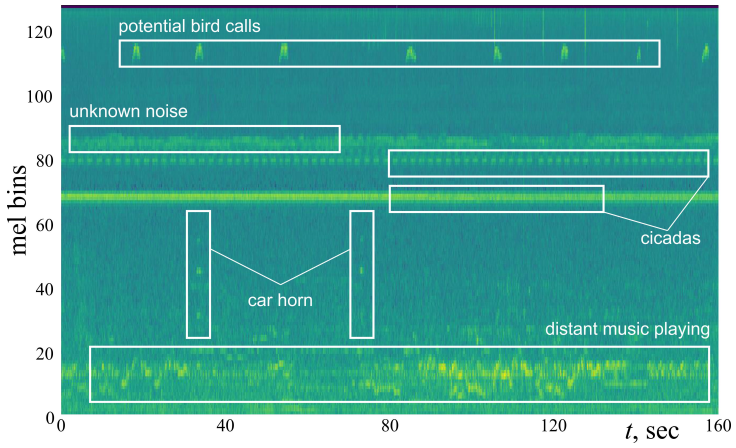

Figure 4: The spectrogram of unlabeled soundscape 101125218.ogg with various sound sources.

model to increase its generalization ability and reduce the domain shift. At the same time, the addition of an extra sound to the recording must not contain bird calls that can disorient the model.

There are several datasets that can help introduce these noises; for example, the Vehicle Type Sound Dataset [9], the Noise Audio Data Dataset with short sounds of different natures [10], the Rain Forest Dataset [11] with recordings of several frog species, and the Hindi Speech Classification Dataset with recordings of short phrases [12]. As an addition, manual selection of recordings not containing bird calls can be used [13]. Although the use of a dataset with the regional dialects spoken in the Western Ghats alongside Hindi may seem more appropriate, it's quite hard to find a sufficient number of such recordings distributed freely. At the same time, the influence of including these dialects on model performance seems minor.

The broadband noises are quite hard to add. On the one hand, many existing recordings of rain and wind noises contain bird calls, which have to be manually filtered. On the other hand, these noises are nonstationary, so they can't be precisely modeled with any kind of simple stationary noise. A similar situation takes place with the sounds produced by cicadas.

To deal with this situation, it is proposed to use unlabeled soundscapes. These recordings, however, contain bird calls that must be excluded. The idea of doing it is based on the fact that background noise as well as cicadas form patterns on the spectrogram that slowly change over time. The patterns of bird calls and other noises are irregular. The first step of the algorithm is taking the 1D Fourier transform of the spectrogram along the time axis. On the second step, only the components with the largest absolute values remain, while the others are put to zero. In the third step, the inverse 1D Fourier transform is performed, and the result is multiplied by random noise. The described filtering procedure significantly reduces the amount of information a spectrogram contains, and its "thin structure" disappears, including bird calls. The example is presented in Fig 5.

2.5. Standard augmentations and post-processing

The "standard" augmentations can be used in the BirdCLEF competition. They are performed on spectrograms and include XY masking, random grid shuffle, and recording mixing. XY masking selects a few rectangular areas in the spectrogram and sets the data inside it to a constant. Random grid shuffle splits the spectrogram into a grid and shuffles all its cells. This transform can be used only along the time axis, and the size of each cell must be greater than the length of a potential bird call, say, 5 seconds. Recording mixing is the technique of adding two or more recordings together before passing them to a model. In this case, the resulting recording contains all the birds from the initial recordings. Its weight is also the sum of the initial weights. These augmentations make the training dataset more diverse.

The model predictions may be post-processed using sliding window averaging. This approach assumes that if there is a bird call in a certain time interval, the probability of the same bird call in the

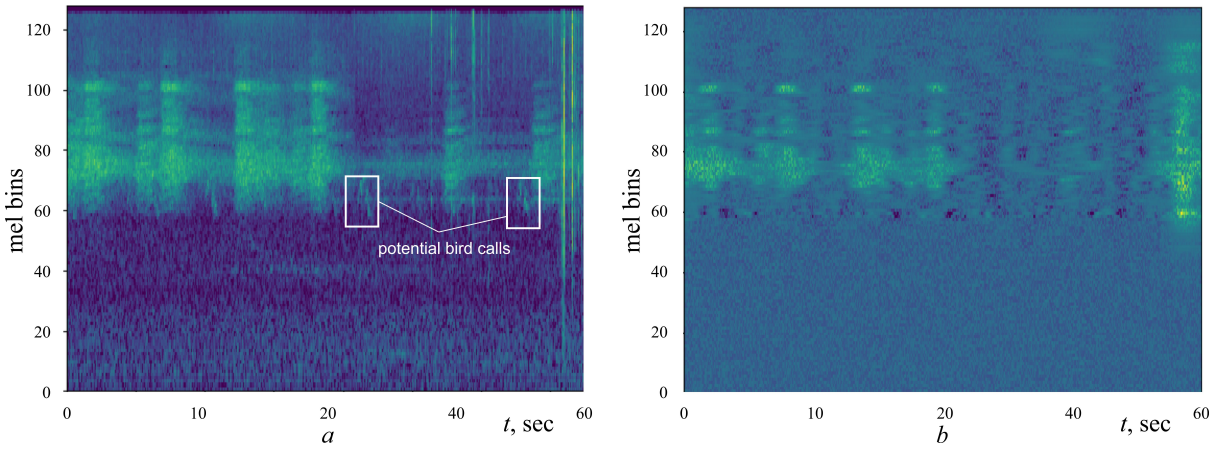

Figure 5: The spectrogram of unlabeled soundscape 1000308629.ogg (a) before and (b) after filtering procedure.

Table 4
The results of applying sliding window averaging.

α	Public score	Private score	α	Public score	Private score
0.000	0.681	0.616	0.250	0.693	0.621
0.100	0.687	0.619	0.275	0.694	0.621
0.150	0.690	0.620	0.300	0.694	0.621
0.200	0.692	0.621	0.350	0.693	0.621
0.225	0.693	0.621	0.400	0.692	0.620

Table 5
The inference time of different frameworks per 240-seconds-long recording.

PyTorch	ONNX	OpenVino	OpenVino with HT	OpenVino with TPE
3.3 sec	2.8 sec	2.5 sec	2.0 sec	2.0 sec

neighboring intervals is also high. So, the final prediction for the current interval is a sum of predictions for the current, previous, and next intervals with the weights of $1 - 2\alpha, \alpha$, and α respectfully. The coefficient α is an averaging parameter, which is often set to 0.25 in BirdCLEF competitions. The results of using different α are presented in Table 4. It can be seen, that the value of $\alpha = 0.25$ is indeed optimal, however, the public score is maximized by $\alpha = 0.3$.

2.6. Inference time optimization

The inference time on the CPU is one of the crucial factors in the competition. However, it was noticed that the models became extremely slow after training. For example, the used model (seresnext26t_32x4d; $D = 768, H = 8, B = 2$) processed the 240-seconds-long recording in 60-70 seconds after training, while the untrained model did the same job in 3.5 seconds. The training procedure doesn't change the model architecture and only adjusts its weights.

After some research, the problem was localized. In the model computational graph, some of the paths are unnecessary, and the corresponding weights must be set to zero during training. However, using L2 regularization makes these weights very low but not exactly zero. As a result, not only do these paths consume computational resources, but the computations with such low values are extremely slow. To prevent this behavior, one has to retrain the model with an additional L1 regularization term, which causes the small weights to be exactly zero. A simple solution for an already-trained model is weight rounding. To do it, one has to convert the model precision from float32 to float16 and then back to float32. Weight rounding can be performed with one line of code in PyTorch: `model.half().float()`.

Table 6
The results of training the same model with different random seeds.

	Public score	Private score
Seed 1	0.662	0.616
Seed 2	0.645	0.615
Seed 3	0.634	0.620
Seed 4	0.669	0.619
Average	0.653 ± 0.017	0.618 ± 0.002

Table 7
The results of applying different methods in the BirdCLEF 2024 competition.

Method	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7
Competition data	+		+	+	+	+	+
Additional data		+	+	+	+	+	+
Select the recordings with $L_{\text{WG}} < 4000$ km				+	+	+	+
Additional noise sources					+	+	+
Distant and class imbalance weights						+	+
Sliding window averaging							+
Public score	0.620	0.560	0.622	0.663	0.672	0.681	0.684
Private score	0.582	0.509	0.568	0.633	0.622	0.616	0.637

Further acceleration is possible with the help of powerful frameworks, such as ONNX and OpenVino. The model was converted to ONNX format and then exported to OpenVino. The inference time summary is presented in Table 5. OpenVino seems faster than ONNX, but one may notice that it doesn't use all available CPU cores when running. To do so, hyperthreading (HT) must be switched on. The alternative is to run multiple threads with multiprocessing. In Python, this can be done in several ways, for example, by using the ThreadPoolExecutor class (TPE). As expected, the results are the same as with the use of HT. It should be noted that many laptops have multiple cores and support HT now, so using it may accelerate the model outside of the competition environment.

3. Results

The main difficulty of the BirdCLEF 2024 competition is an unstable public score. The minor changes in the model and its training procedure may significantly increase or decrease this score. This makes it quite hard to test different approaches. For example, the results of training with exactly the same model and data and different random seeds are presented in Table 6. On the one hand, the standard deviation of the public score may be several times larger than the improvement of the score with the use of some clever technique. On the other hand, this makes it hard to introduce a reliable CV. It can't be consistent with the public score because it is unstable, and with the private score because there is a huge domain shift between the data. So, the reasonable way is to conduct many experiments, take the average of the public score, and hope that this will not cause overfitting to the public score.

The methods described in Section 2 were applied consequently, and the results are presented in Table 7. The most significant improvement was caused by the use of geographical data.

The inference time of the resulting models was 28 minutes, so an ensemble of four models with the best public scores was made. The public score of this ensemble was 0.713 (13th place in the competition public leaderboard). However, the selected models overfit, and the private score of the ensemble was as low as 0.616. Despite the unlucky model selection, the presented methods seem good and may be successfully used in the future competitions and applications.

References

[1] S. Kahl, T. Denton, H. Klinck, V. Ramesh, V. Joshi, M. Srivathsa, A. Anand, C. Arvind, H. CP, S. Sawant, V. V. Robin, H. Glotin, H. Goëau, W.-P. Vellinga, R. Planqué, A. Joly, Overview of BirdCLEF 2024: Acoustic identification of under-studied bird species in the western ghats, Working Notes of CLEF 2024 - Conference and Labs of the Evaluation Forum (2024).

[2] Birdclef 2024 – birdcall species identification from audio, 2024. URL: https://www.kaggle.com/competitions/birdclef-2024.

[3] A. Joly, L. Picek, S. Kahl, H. Goëau, V. Espitalier, C. Botella, B. Deneu, D. Marcos, J. Estopinan, C. Leblanc, T. Larcher, M. Šulc, M. Hrúz, M. Servajean, et al., Overview of lifeclef 2024: Challenges on species distribution prediction and identification, in: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, 2024.

[4] Xeno-canto sharing wildlife sounds from around the world, 2024. URL: https://xeno-canto.org.

[5] M. V. Shugaev, N. Tanahashi, P. Dhingra, U. Patel, Birdclef 2021: building a birdcall segmentation model based on weak labels, CEUR Workshop Proceedings 2936 (2021). URL: https://ceur-ws.org/Vol-2936/paper-141.pdf.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need (2023). URL: https://arxiv.org/abs/1706.03762. arXiv:1706.03762.

[7] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, CoRR abs/1512.03385 (2015). URL: http://arxiv.org/abs/1512.03385. arXiv:1512.03385.

[8] J. Hu, L. Shen, S. Albanie, G. Sun, E. Wu, Squeeze-and-excitation networks, 2019. URL: http://arxiv.org/abs/1709.01507. arXiv:1709.01507.

[9] Vehicle type sound dataset, 2024. URL: https://www.kaggle.com/datasets/brinkor/vehicle-type-sound-dataset.

[10] Noise audio data dataset, 2024. URL: https://www.kaggle.com/datasets/javohirtoshqorgonov/noise-audio-data.

[11] Rainforest connection species audio detection data, 2021. URL: https://www.kaggle.com/competitions/rfcx-species-audio-detection/data.

[12] Hindi speech classification dataset, 2024. URL: https://www.kaggle.com/datasets/vivmankar/hindi-speech-classification.

[13] Nocall manual classification dataset, 2024. URL: https://www.kaggle.com/datasets/janmpia/nocall-manual-classification.