Bird Species Recognition via Neural Architecture Search

Markus Mühling, Jakob Franz, Nikolaus Korfhage, and Bernd Freisleben

Department of Mathematics and Computer Science, University of Marburg,
Hans-Meerwein-Straße 6, D-35032 Marburg, Germany
{muehling,franz,korfhage,freisleb}@informatik.uni-marburg.de

Abstract. This paper presents the winning approach of the BirdCLEF 2020 challenge. The challenge is to automatically recognize bird sounds in continuous soundscapes. In our approach, a deep convolutional neural network model is used that directly operates on the audio data. This neural network architecture is based on a neural architecture search and contains multiple auxiliary heads and recurrent layers. During the training process, scheduled drop path is used as a regularization method and extensive data augmentation is applied to the audio input. Furthermore, species location lists are used in the post-processing step to reject unlikely classes. Our best run on the test set obtains a classification mean average precision score (cmap) of 13.1% and a retrieval mean average precision score (rmap) of 19.2%.

Keywords: Bird Species Recognition · BirdClef 2020 · Neural Architecture Search · Gabor Wavelet Layer

1 Introduction

Automatically identifying bird species in continuous sound recordings is an important task for monitoring the populations of different bird species in forest ecosystems. In the BirdCLEF 2020 challenge [4], which is part of the LifeClef challenge [3], participants have to identify 960 bird species in 5 seconds snippets of continuous audio recordings. The test data contains 153 soundscapes of a length of 10 minutes recorded at four different locations in Peru (Concesion de Ecoturismo Inka Terra), the USA (Sierra Nevada/High Sierra in California and Sapsucker Woods/Ithaca in New York), and Germany (Laubach). Each soundscape contains high quantities of (overlapping) bird vocalizations. In contrast to the BirdCLEF 2019 challenge [5], no other than the provided training data is allowed to build the recognition system. This prohibits fine-tuning convolutional neural networks (CNNs) pretrained on other datasets, as applied in the best approaches of previous BirdClef challenges. This is reasonable, since fine-tuning a CNN pretrained for the task of image classification on the ILSVRC dataset [11] has proven to yield excellent results for many tasks, even for the task of bird recognition using spectogram images.

This restriction makes it even more important to find an optimal neural network architecture for the BirdClef 2020 challenge. For this purpose, we applied a network architecture search (NAS) approach. NAS is a current field of research. It allows to automatically learn an optimal neural network architecture for a specific problem and offers an alternative to the time-consuming task of manual architecture optimization.

The designed architecture is directly applied to the audio input using a Gabor wavelet transformation, similar to Zeghidour et al. [13] in speech recognition. This transformation is integrated into the neural network architecture as a complex 1-D convolutional layer.

The contributions of the paper are as follows:

- A novel neural architecture search approach based on a memetic algorithm is used to find an optimal neural network architecture for the task of bird sound recognition.
- The proposed neural network architecture directly operates on the audio input.

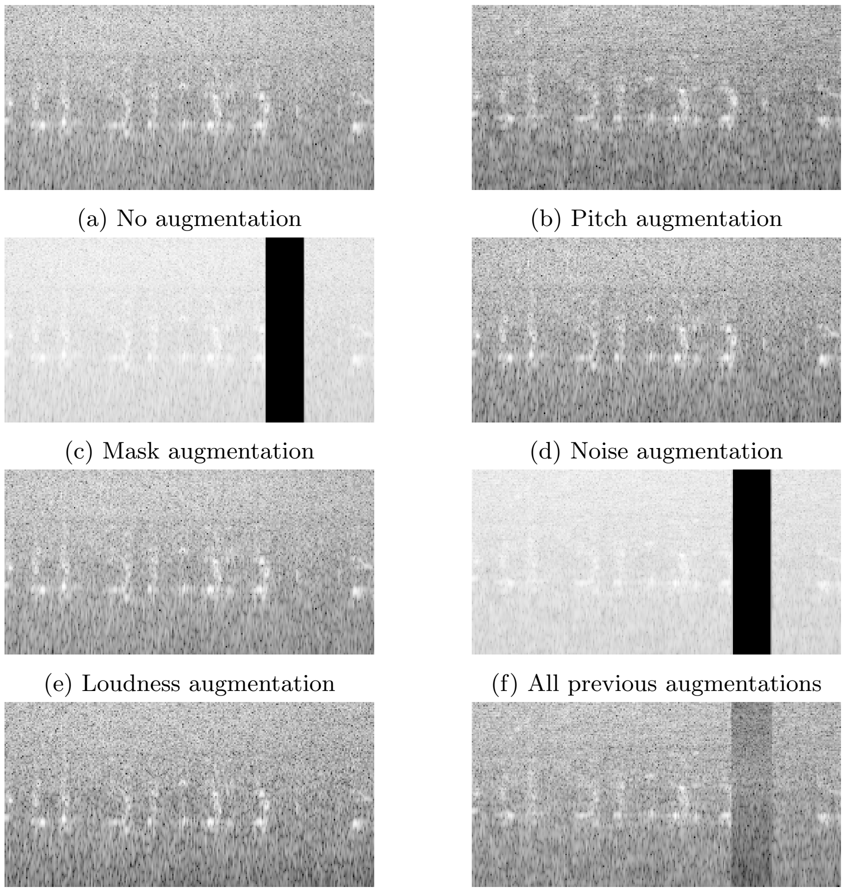
The paper is organized as follows. Section 2 describes the bird recognition approach including data pre-processing, data augmentation, the construction of the neural network architecture, and details of the training process. Experimental results are presented in Section 3. Section 4 concludes the paper and outlines areas for future work.

2 Methods

In this section, the proposed system for bird sound recognition is presented. Section 2.1 describes the preprocessing steps. The data augmentation methods used during the training process are specified in Section 2.2. The design of the neural network architecture is explained in Section 2.3, including the Gabor wavelet layer, the NAS approach, and the composition of the neural network using multiple auxiliary heads and recurrent layers. Section 2.4 provides information about the training process.

2.1 Data Pre-processing

First, the audio recordings are split into 5 seconds segments with an overlap of 0.25 seconds. Then, these snippets are classified as either bird sound or noise (i.e., no audible bird sounds) using the heuristic of the BirdCLEF 2018 baseline system [6]. The sets are called T_{signal} (bird sounds) and T_{noise} (noise segments). Finally, the recordings are normalized to -3 db and resampled to 22,050 Hz.



(a) No augmentation
(b) Pitch augmentation
(c) Mask augmentation
(d) Noise augmentation
(e) Loudness augmentation
(f) All previous augmentations
(g) Noise snippets augmentation
(h) All augmentations

Fig. 1: Visualization of the impact of different audio augmentation methods on the resulting spectrogram.

2.2 Data Augmentation

To avoid overfitting and to improve the generalization capabilities of the neural network model to various recording conditions, the following data augmentation methods are applied to the audio segments (see Figure 1):

- Pitch augmentation: Shifting pitch by one to three semitones.
- Mask augmentation: A randomly chosen segment of 0.5 seconds duration is masked with zeros.
- Noise augmentation: White noise is added to the training snippet.
- Loudness augmentation: The volume of a training snippet is increased by a randomly chosen factor within range $[0.25, 4]$ (0.25 leads to a decrease by factor 4, 4 leads to an increase by factor 4).

– Noise snippets augmentation: Each training snippet is a randomly weighted sum of one augmented bird snippet from T_{signal} and four noise snippets from T_{noise}.

First, pitch, mask, noise, and loudness augmentation are applied to the training segments of T_{signal}. For this purpose, between one and four augmentation methods are randomly selected and applied in random order. Second, noise snippet augmentation is used. Figure 1 visualizes the impact of the different data augmentation methods on the resulting audio spectograms.

2.3 Neural Network Architecture

A NAS approach based on a memetic algorithm is used to find an optimal convolutional neural network architecture for the task of bird sound recognition. The overall network architecture is based on a Gabor wavelet layer directly operating on the audio input, similar to Zeghidour et al. [13] for speech recognition. The optimal cell structure found by NAS and multiple architecture heads including recurrent layers are used to take further advantage of temporal information.

Gabor Wavelet Layer. The extraction of the audio spectograms is integrated into the neural network architecture using a Gabor wavelet layer. For this purpose, a complex 1-D convolution with n filters is applied to the audio input where n is the number of frequencies. The weights of the complex kernels are created using Gabor wavelets. The complex 1-D convolution is followed by applying the logarithm and a zero centering normalization.

Furthermore, we experimented with audio spectograms generated by applying the FFT, as provided by the baseline system of the BirdClef challenge 2018 [7]. While our experiments on this dataset showed that using the Gabor wavelet layer did not lead to quality improvements compared to using audio spectograms generated by the FFT, the implemented Gabor wavelet layer has some advantages. First, data augmentation methods can be also flexibly applied to the audio input. Second, arbitrary frequencies can be extracted in contrast to the FFT with equidistant spacing. This allows us to directly use mel-scaled frequencies.

Neural Architecture Search. NAS is a recent field of research. The aim of NAS is to automatically find an optimal neural network architecture on a specific task. The NASNet architecture [14] is the first design that outperformed handcrafted, manually optimized architectures for the task of image classification. The main idea is to break down the search space and search for cells that form the building blocks for the overall network architecture. Other approaches primarily exhibit runtime improvements [14, 9, 1, 10, 2]. While the approaches of Chen et al. [1] and Real et al. [10] use an evolutionary search method, Dong et al. [2] start with a huge, over-parameterized network which is then shrunk and locally optimized step by step.

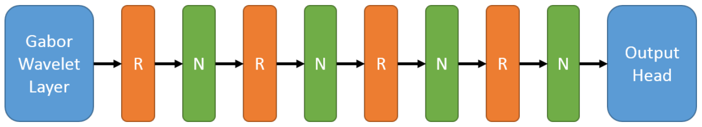


Fig. 2: Stacked network with reduction cells (R) and normal cells (N).

In the following, we describe an NAS approach to find an optimized architecture for bird sound recognition. The approach uses a memetic search algorithm that combines local optimization with an evolutionary algorithm. Like Zoph et al. [14], we do not search for full network architectures but instead for relatively small structures called cells that are later stacked in a predefined manner to build the full network. NAS approaches are typically categorized according to the used search space, the search strategy, and the performance estimation/evaluation strategy, which are described in the following paragraphs.

Search Space. As previously described, we search for cell structures that are later upscaled to the full network architecture. Like the NASNet search space [14], a cell consists of blocks and operations. While a cell in the NASNet search space consists of a fixed number of blocks and operations, our cell structure is more flexible. A cell can be composed of a variable number of blocks, and each block can contain a variable number of operations. The set of allowed operations to perform is as follows:

- identity
- depth-wise separable convolution
- normal convolution
- max pooling
- average pooling

Kernel sizes for the convolution and pooling operations are restricted to the following sizes: $\{3 \times 3, 5 \times 5, 7 \times 7\}$.

The outputs of the operations within a block are merged with an add layer that forms the output of the corresponding block. Therefore, each operation contains some extra layers to satisfy shape constraints. The output of blocks that have not been used as input to any other operation within the cell are merged using either an add or a concatenate layer to form the output of the cell. The input of an operation can be either the output of one of the previous two cells or the output of a preceding block of the current or previous cell.

Performance Estimation. For performance estimation, a network architecture is constructed from the cell. First, a normal and a reduction cell are derived from the cell structure. While the normal cell retains the spatial size of its predecessor cell (i.e., every convolution or pooling operation has stride 1×1 and appropriate padding), the reduction cell reduces the spatial size by a factor of 2

Algorithm 1: Neural architecture search

```
Input: initial population $P$, number of cells to visit $r$
for $cell$ $in$ $P$ do
    // transform cell to full network
    $cell.network$ := cell-to-cnn($cell$, $F$=8);
    $cell.accuracy$ := train-and-eval($cell.network$);
    $cell.flops$ := compute-flops($cell.network$);
end
// compute fitness values of population members
$compute\text{-}fitness(P)$;
for $round = 1$ $to$ $r$ do
    $parents$ := $select\text{-}parents(P)$;
    // perform the crossover operation
    $child\text{-}cell$ := create-child($parents$);
    // perform mutation operations
    $child\text{-}cell$ := mutate-child($child\text{-}cell$);
    // add child cell to population
    $P$.append($child\text{-}cell$);
    $child - cell.network$ := cell-to-cnn($child - cell$, $F$=8);
    $child - cell.accuracy$ := train-and-eval($child - cell.network$);
    $child - cell.flops$ := compute-flops($child - cell.network$);
    $compute\text{-}fitness(P)$;
end
return max($P$, key=lambda $cell$ : $cell.fitness$);
```

(i.e., convolutions and pooling operations have stride 2×2). Second, the overall network is stacked, as shown in Figure 2.

The output head is a global average pooling layer, followed by a densely connected layer with softmax activation. The convolutions of the first reduction cell have F filters. The number of filters is then doubled after every reduction cell.

To determine the quality (or fitness) of a cell, a full network is derived from each cell with $F = 8$ initial filters. Then, the model is trained and evaluated on a subset of the BirdCLEF 2020 training data set. For this purpose, 300 training samples are randomly selected for each of 50 randomly chosen classes. The resulting data set with $15,000$ audio samples is split into training and validation set with a validation split of 0.1. Samples of the same audio file are either all in the training set or in the validation set. The model is trained for 20 epochs using the ADAM optimizer and a cosine learning rate scheduler. Finally, the accuracy of the model is calculated on the validation set.

Search Strategy. The idea of the search algorithm is to use an evolutionary algorithm and local optimization. This memetic search algorithm starts with a population of cells randomly drawn from the search space.

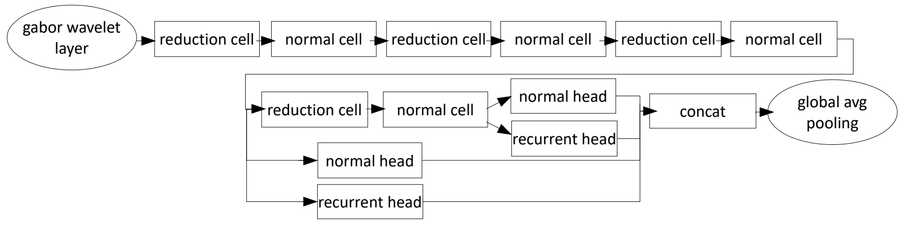


Fig. 3: General structure of the full network

The fitness of a cell c is calculated as follows:

$$c.fitness = 0.7 \cdot \frac{c.accuracy - acc_{mean}}{acc_{std}} + (1 - 0.7) \cdot \frac{flops_{mean} - c.flops}{flops_{std}}$$

where $c.accuracy$ is the performance estimation and $c.flops$ is the number of floating point operations of cell c. acc_{mean}, acc_{std}, $flops_{mean}$ and $flops_{std}$ are the mean and the standard deviation of the validation accuracy and flops, respectively, of all cells of the current population.

The tournament selection method is used to select two cells of the current population based on their fitness values. Cells with high fitness values have a higher chance to be chosen. The two selected cells are used to create a new cell by merging all blocks. Considering the order of the blocks, all operations are joined, so that block i of the new cell contains all operations of the i-th blocks of the parents whereby the input connections of the operations are preserved. In a local optimization step, the most important operations per block are identified similar to the approach of Dong et al. [2]. For this purpose, a network is derived from the child cell, trained for two epochs and shrunk to the most important operations.

Afterwards, mutation operations are applied to the new cell, for example, changing operation types, input connections or adding and removing blocks. Finally, the cell is added to the population, and the fitness values are recomputed. This procedure is repeated for a certain number of rounds, and the cell with the best fitness value is returned. The search algorithm is described in Algorithm 1.

Multi-Head Model. In contrast to natural images, audio spectrograms contain temporal information. It seems reasonable to use this information, since the individual sounds of a bird's song may follow a certain chronological order. For this reason, we added output heads with recurrent layers to the network architecture. Altogether, the final network architecture contains two output heads, each with and without recurrent layers: one pair at the end of the network and one pair at an intermediate stage. Output heads without a recurrent layer contain a global average pooling layer followed by a densely connected layer with softmax activation, recurrent output heads contain two consecutive GRU layers with 128 units each, followed by a global average pooling layer and a densely connected layer with softmax activation. Each output head has its own loss function in the training phase, whereas for inference the output heads are merged using a concat layer followed by global average pooling. Figure 3 shows the structure of the network including all output heads.

2.4 Training Methodology

We trained our models using a weighted loss function consisting of a softmax log loss for each output head. The loss terms of the intermediate output heads are weighted by a factor of 0.6, whereas the loss terms of the output heads at the end of the network are weighted by a factor of 1. The ADAM optimizer [8] is used for the training process with a cosine learning rate scheduler. Furthermore, scheduled drop path [14] where the drop rate d_r is linearly increased to 0.4 throughout the training process is used as a regularization method. Within the scheduled drop path, each operation of each cell in the network gets dropped (i.e., its output is set to zero) by the probability $d_r \cdot \frac{c}{n}$ if the network has n cells and if we consider an operation of the c-th cell of the network.

3 Experiments

In this section, we present the results of the two official and two post challenge submissions.

3.1 Evaluation Metrics

The task of the challenge is to identify all audible bird species in 5 second snippets of continuous soundscape recordings from four different locations. With the BirdClef challenge 2020 [4], the following two metrics are used to measure the performance of a submitted run:

– Classification Mean Average Precision (cmap):

$$cmap = \frac{\sum_{c=1}^{C} AveP(c)}{C}$$

where C is the total number of classes and $AveP(c)$ is the average precision of class c.

– Retrieval Mean Average Precision (rmap):

$$rmap = \frac{\sum_{x=1}^{X} AveP(x)}{X}$$

where X is the total number of audio snippets of all test files and $AveP(x)$ is the average precision of snippet x.

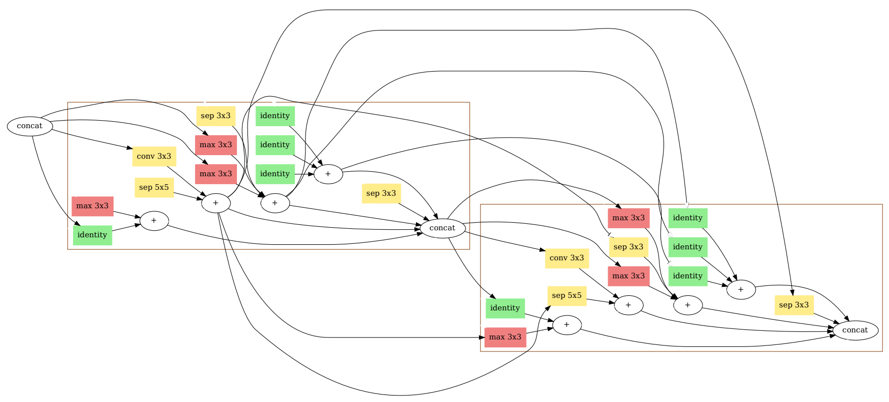


Fig. 4: Structure of the cell used in Runs 1 and 2

3.2 Data Sets

The training set consists of more than 70,000 recordings across 960 bird species classes contributed by the Xeno-canto community. Exactly one foreground bird species is assigned to each audio recording file. Additionally, metadata such as recording location, recording date, elevation, and recording quality is provided.

The validation set contains 12 soundscape files recorded at two different locations (one in Peru, one in the USA). Each soundscape file has a duration of 10 minutes and is divided into 5 second snippets. For our evaluation, a list of audible bird species is assigned to each 5 seconds snippet of the validation data.

The test set consists of 153 soundscape files of 10 minutes duration recorded at four different locations (one in Peru, two in the USA, and one in Germany).

3.3 Results

Two network architectures found by the NAS method described in Section 2.3 are evaluated. Altogether, we submitted five runs: two official runs with the first architecture and three post challenge runs with the second architecture.

The first network architecture was found by running the search algorithm described in Section 2.3 for 100 rounds. Figure 4 shows the cell architecture of this network. The overall network was constructed as described in Section 2.3 with the number of initial filters set to $F = 16$ and trained for 30 epochs, as described in Section 2.4. We used 128 mel-scaled frequencies in the range of 170 Hz to 10,000 Hz. The width of the kernel of the Gabor wavelet layer was set to $w = 1024$, resulting in spectograms with a resolution of 256x128. The training took 65 hours on four Nvidia TITAN XP GPUs.

The trained network takes about 43 seconds to analyze a 10 minutes soundscape file. However, most of the time (≈ 40 s) is consumed by the non-optimized

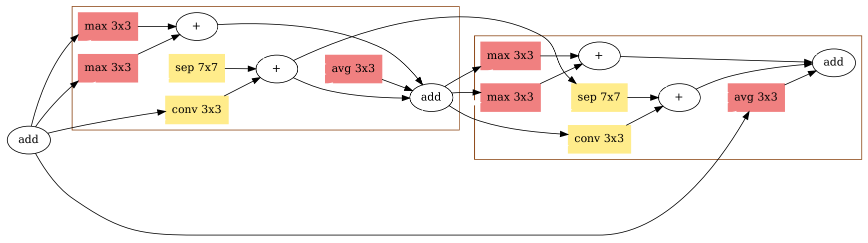


Fig. 5: Structure of the cell used in Post-Submission Runs 1, 2, and 3

single-threaded pre-processing step. The inference time of the network is only $\approx$ 3 s, which corresponds to 40 snippets per second.

The two official runs differ only in the post-processing step.

Official Run 1. In the first run, the species location lists provided by the challenge organizers are used to reject impossible species from the model's predictions. This run obtains a cmap of 12.8% and a rmap of 19.3%, which is 8.6% better than the cmap of the second best submission (4.2%) and thus wins the challenge.

Official Run 2. For the second run, we used species lists per location and season, derived from the training data to remove unlikely species from the model's predictions. This led to a slight decrease in cmap and a slight increase in rmap, resulting in a cmap value of 12.7% and a rmap value of 19.8%.

After the challenge, we ran the NAS algorithm for another 50 rounds and found a new promising cell architecture. The network corresponding to the newly found cell with initial filters set to $F = 16$ was used to submit three further runs. Figure 5 shows how two consecutive cells of the network are connected. While in the Post-Submission Runs 1 and 2 the species lists per location and season are used, Post-Submission Run 3 only applies the species lists per location in the post-processing step.

Post-Submission Run 1. The network used in this run was trained in the same way as the network of Run 1 but only for 10 instead of 30 epochs, since this seemed to be sufficient. This run obtains a cmap of 12.6% and a rmap of 20.3%.

Post-Submission Run 2. For this run, the width of the kernel of the Gabor wavelet layer was reduced from $w = 1024$ to $w = 512$, resulting in higher resolution spectrograms with $428x128$ pixels. The network was initialized with the weights obtained by Post-Submission Run 1 and fine-tuned for another 5 epochs. This resulted in a small improvement of the cmap and rmap values (cmap = 12.7%, rmap = 20.6%).

	validation		test	
	cmap	rmap	cmap	rmap
Run 1	14.8%	21.8%	12.8%	19.3%
Run 2	14.8%	22.2%	12.7%	19.8%
Post-Submission Run 1	15.1%	24.4%	12.6%	20.3%
Post-Submission Run 2	16.2%	24.0%	12.7%	20.6%
Post-Submission Run 3	17.8%	23.4%	13.1%	19.2%

Table 1: Comparison of the submitted runs in terms of cmap and rmap on validation and test set.

Post-Submission Run 3. In contrast to Post-Submission Run 1, this run uses a different sampling strategy. Instead of sampling from the preclassified bird sound snippets (T_{signal}), the batches are sampled from the audio files extracting a random five second snippet per file. In this case, the number of training samples per epoch equals the number of mono-species audio files. Due to the smaller number of samples per epoch, the network was trained for 100 epochs. This run obtains a cmap of 13.1% and a rmap of 19.2%, which is the best result of the challenge in terms of cmap.

3.4 Discussion

Table 1 shows a comparison between all the runs on the validation and test set. It is evident that even though the network used for the post-submission runs scored significantly better on the validation set, this is only partially transferred to the test set. Post-Submission Run 2, for example, obtains a validation cmap of 16.2% and a validation rmap of 24.0%, but yields a test cmap of only 12.7%, which is 0.1% worse than the test cmap of Run 1, although Run 1 obtains a significantly lower validation cmap of only 14.8%. The 2.2% better validation rmap score is, on the other hand, reflected on the test set with a test rmap of 20.6% in comparison to only 19.3% of Run 1. Interestingly, Post-Submission Run 3 yielded the best cmap score both on the validation and test set without distinguishing between bird sound and noise snippets during the training phase. The extraction of snippets at random positions and the different weighting of classes in the training phase seems to be beneficial for the cmap score with 13.1% on the test data, while the rmap score decreases to 19.2%. Both network architectures used for the submissions have a similar inference time of approximately 40 snippets per second.

4 Conclusion

The presented approach won the BirdCLEF 2020 challenge with a cmap of 12.8% and a rmap of 19.3%. In the post-challenge phase, the scores have been further improved to 13.1% cmap and 20.6% rmap. However, the task of recognizing all bird species in soundscapes is far from being solved.

One possible reason could be the discrepancy between training and test data. The training data consists of sound files of various lengths. Only the most audible bird is labeled in each sound file. This means that there could be samples in the training data where multiple birds or even a bird other than the labeled one is audible. This can be misleading in the training process. However, the task for validation and testing is to identify all audible birds in a snippet. This is a much harder task than just recognizing the most audible bird in a recording. A more fine-grained multi-label annotation of the training data could significantly improve the quality of the bird sound recognition models.

There are several possibilities to improve our approach with the existing training data. There is room for improvement in the data augmentation step that has proven to be very important in previous challenges. For example, we could apply further data augmentation methods on the audio data or even apply some augmentation on the spectrograms produced by the Gabor wavelet layer. Another option is to learn the weights of the complex 1-D convolution kernel of the Gabor wavelet layer to produce better spectrograms. or to use a self-supervised pre-training approach to learn a more suitable audio representation [12] and adapt it for birdcall identification. Furthermore, it may be beneficial to run the neural network search algorithm for a longer period of time to find a network that works even better on this task. Finally, the performance could be improved at the cost of longer training and inference runtimes by scaling up the network's capacity and using stronger regularization.

5 Acknowledgement

This work is funded by the Hessian State Ministry for Higher Education, Research and the Arts (HMWK) (LOEWE Natur 4.0).

References

1. Chen, Y., Meng, G., Zhang, Q., Xiang, S., Huang, C., Mu, L., Wang, X.: Reinforced evolutionary neural architecture search (2018)
2. Dong, X., Yang, Y.: Searching for a robust neural architecture in four gpu hours. In: Proceedings of the IEEE Conference on computer vision and pattern recognition. pp. 1761–1770 (2019)
3. Joly, A., Goëau, H., Kahl, S., Deneu, B., Servajean, M., Cole, E., Picek, L., Ruiz De Castañeda, R., é, Lorieul, T., Botella, C., Glotin, H., Champ, J., Vellinga, W.P., Stöter, F.R., Dorso, A., Bonnet, P., Eggel, I., Müller, H.: Overview of lifeclef 2020: a system-oriented evaluation of automated species identification and species distribution prediction. In: Proceedings of CLEF 2020, CLEF: Conference and Labs of the Evaluation Forum, Sep. 2020, Thessaloniki, Greece. (2020)
4. Kahl, S., Clapp, M., Hopping, A., Goëau, H., Glotin, H., Planqué, R., Vellinga, W.P., Joly, A.: Overview of birdclef 2020: Bird sound recognition in complex acoustic environments. In: CLEF task overview 2020, CLEF: Conference and Labs of the Evaluation Forum, Sep. 2020, Thessaloniki, Greece. (2020)

5. Kahl, S., Stöter, F.R., Goëau, H., Glotin, H., Planque, R., Vellinga, W.P., Joly, A.: Overview of BirdCLEF 2019: Large-Scale Bird Recognition in Soundscapes. In: Working Notes of CLEF 2019 - Conference and Labs of the Evaluation Forum. vol. CEUR Workshop Proceedings, pp. 1–9. CEUR (Sep 2019)
6. Kahl, S., Wilhelm-Stein, T., Klinck, H., Kowerko, D., Eibl, M.: Recognizing birds from sound - the 2018 birdclef baseline system. arXiv preprint arXiv:1804.07177 (2018)
7. Kahl, S., Wilhelm-Stein, T., Klinck, H., Kowerko, D., Eibl, M.: Recognizing birds from sound - the 2018 birdclef baseline system. arXiv preprint arXiv:1804.07177 (2018)
8. Kingma, D., Ba, J.: Adam: A method for stochastic optimization in: Proceedings of international conference on learning representations (2015)
9. Liu, C., Zoph, B., Neumann, M., Shlens, J., Hua, W., Li, L.J., Fei-Fei, L., Yuille, A., Huang, J., Murphy, K.: Progressive neural architecture search. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 19–34 (2018)
10. Real, E., Aggarwal, A., Huang, Y., Le, Q.V.: Regularized evolution for image classifier architecture search **33**, 4780–4789 (2019)
11. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3), 211–252 (2015)
12. Schneider, S., Baevski, A., Collobert, R., Auli, M.: wav2vec: Unsupervised pretraining for speech recognition. arXiv preprint arXiv:1904.05862 (2019)
13. Zeghidour, N., Usunier, N., Kokkinos, I., Schatz, T., Synnaeve, G., Dupoux, E.: Learning filterbanks from raw speech for phone recognition. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (2018)
14. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8697–8710 (2018)