Project report - Machine learning lecture

Bird Species Classification using Audio Signal Processing and Convolutional Neural Networks

Lukas Bertsch
lukas.bertsch@tu-dortmund.de

Due on 31 July 2024

TU Dortmund – Faculty of Physics

Contents

1. Introduction

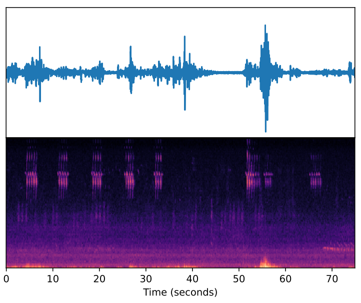
Even if it may not seem so at first glance, birds play a crucial role in our ecosystem. They connect different habitats by spreading seeds and transporting fish spawn. Just like bees and butterflies, some bird species, such as *hummingbirds* and *honeycreepers*, also significantly contribute to the pollination of fruits and flowers. Moreover, birds are an important part of our ecosystem's food chain: for example, most birds feed on invertebrates and thus act as nature's pest control units.
By monitoring bird populations, we can gain valuable insights into the health of our ecosystem. A challenge that arises in this task is the difficulty of spotting birds, especially in dense vegetation. However, their calls can often be heard over long distances without a direct line of sight. Identifying birds by their calls is also challenging due to the vast number of species and subspecies, even in less biodiverse regions, especially among songbirds, which are hard to differentiate. Moreover, classifying different bird species by sound requires expert knowledge and is a time-consuming process.
A solution to this problem could be provided by the automated classification of bird songs in soundscape recordings. Raw audio wave formats are difficult to handle due to their high data density and lack of interpretability. Hence, the common approach in analyzing audio files is to transition to the time-frequency domain and use spectrograms instead. A spectrogram is a visual representation of the time evolution of the frequency spectrum of a signal. It is created by performing a series of short-time Fourier transforms (STFTs) [1] on a signal. The x-axis represents time, the y-axis represents frequency, and the intensity or color of each point represents the amplitude or power of the frequency at that time. To better reflect human perception of sound, the frequency spectrum can be transformed to the mel scale [2], and amplitudes can be converted to decibels (logarithmic scaling). This is also beneficial in machine learning, where log mel spectrograms are used as input features for models such as convolutional neural networks (CNNs). Their ability to capture the characteristics of two-dimensional data makes them especially useful for tasks involving audio classification and recognition using spectrograms [3].
This leads to the research question of this report:

How accurately can different European bird species be classified by analyzing audio recordings using convolutional neural networks and audio processing?

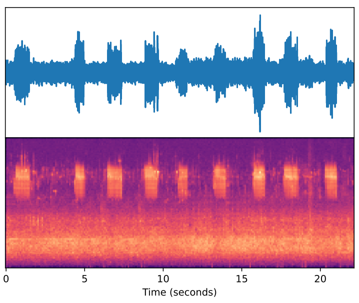
To investigate this question, this report presents the selection of a dataset in Section 2 and a documentation of the solution approach in Section 3. Five individual CNNs are trained using randomly picked spectrogram slices equivalent to 15 s audio clips and used in an ensemble to predict audio recordings of different durations. The results of this study are evaluated and presented in Section 4 and compared to an alternative method, making use of a simpler machine learning algorithm in Section 5. Finally, the discussion and conclusion are provided in Section 6.

2. Dataset

For the task of bird species classification, many datasets are available online. However, these datasets encounter similar limitations for the research topic of this report. Most datasets focus on bird species that are not native to Europe. Further, these datasets often provide a limited amount of data and are already heavily preselected. Therefore, a new dataset is created based on the comprehensive database of bird recordings available at xeno-canto.org [4]. Xeno-canto is a website for sharing recordings of wildlife sounds, particularly bird songs. People from around the world can upload audio recordings to the website. The recordings are licensed under Creative Commons licenses and are free to use for non-commercial purposes. To generate the dataset, the website's API is used to filter recordings and automatically download the files using the tool *scikit-maad* [5]. For this project, recordings of common songbird species in Central Europe are searched for via the API. The recordings are required to have a length between 5 and 300 s, as shorter recordings are not suitable for the approach used in this analysis, and longer recordings require more storage space. For this project, the recording type must include the tag "song" to filter for bird songs and exclude other bird sounds, such as woodpecker hammering. Additionally, the sampling rate of the audio must be greater than 20 000 Hz, and the audio must have sufficient overall quality (quality categories "A" and "B"). Often, other bird species can be heard in the background, which is problematic because this could lead to misclassifications between classes while training the classifier. Therefore, recordings that contain more than two other bird species in the background are removed. Between 100 and 500 files are downloaded for each class. After applying all selection criteria, the dataset includes approximately 17 000 files across 46 bird species classes, referenced by numerical "labels", i.e., numbers from 0 to 45. A list of the scientific and common names of the species corresponding to each label can be found in Appendix A.1. The majority of the selected species are songbirds, but owl, swallow, and pigeon species are also included. In general, the audios can include background noise and calls of other species, as explained previously. The loudness, quality, and sampling rate vary significantly between recordings. Additionally, some recordings might include long segments with no bird sounds. All this increases the difficulty of the classification task but also enhances the robustness of a classifier trained on this data. In Figure 1, the waveforms and spectrograms for two recordings of different bird species are shown. The spectrograms highlight distinct features the CNN can learn to differentiate between these species.
To independently evaluate the classifier after hyperparameter optimization, a train-test split is performed, and 20 % of the data are set aside for exclusive use in the final evaluation. The dataset's class distributions are strongly imbalanced. Some classes have 500 audio files, while others have only around 100. This issue becomes more severe when considering the total audio length of the classes because classes with a lower number of available files often also have shorter audios. Plots showing the number of files and total audio length are presented in Appendix A.2. To address this imbalance, classes with fewer files are upsampled to match the class with the highest number of files during training, and a balanced accuracy score is used in the evaluation.



(a) “Great Tit” (32) [6].



(b) “European Greenfinch” (29) [7].

Figure 1: Example spectrograms of audio recordings of different bird species.

3. Design and Implementation

As explained in the introduction (Section 1), the aim of this project is to train a convolutional neural network capable of classifying different bird species using spectrogram audio data. In this section, methodological details of the solution approach are explained. First, the general concept of the solution to the research question and the data pipeline are described. After that, the choice of model architecture and hyperparameter optimizations are discussed.

3.1. Solution Concept

The dataset described in the previous section consists of audio recordings of varying durations, which are converted to decibel mel-spectrograms (referred to as “spectrograms” hereafter). To process this data in a CNN, the input shape must be fixed. Therefore, the classifier is trained with spectrograms representing fixed-length audio clips. The duration of these audio clips is a hyperparameter optimized in Section 3.4. To make a prediction for a full-length audio recording, the audio’s spectrogram is split into slices of fixed length. The last slice is either discarded if it corresponds to less than 5 s of audio or randomly padded with zeros to match the expected shape otherwise. The prediction for the entire audio is then calculated as the mean prediction of all spectrogram slices. This procedure has the advantage that empty sequences in the audio and parts where different species than the target species can be heard are averaged out in the prediction, increasing overall accuracy.
Another important aspect of this project is the validation of the training process and hyperparameter optimization. To achieve this, 20 % of the training data are used for validation when training a model. Since this effectively reduces the available training data for the final model, a *stratified k-fold* with $k = 5$ is used. This results in five classifiers, each trained with different validation data, allowing to assess the generalization of the method. Finally, the labels of the test data are predicted using all five classifiers in an ensemble, and the average prediction is taken, which should benefit the accuracy and

robustness of the method. For the Implementation of the neural network, the machine learning libraries *tensorflow* [8], *Keras* [9] and *Scikit-Learn* [10] are used.

3.2. Data Pipeline

A difficulty encountered in this project is processing large amounts of data: in total, over 50 GB of audio data must be analyzed. Ideally, a randomly picked audio clip of fixed duration would be drawn from the training data, the spectrogram computed, and then further passed to the neural network. This approach would allow audio augmentation, such as pitch shifting, to be applied directly to the audio itself but requires substantially more computational resources for data loading. Since training times with this approach are impractical, a different approach is chosen instead. Before starting the training and optimization process, the spectrograms are computed for all audio files and saved to storage in `hdf5` files. This excludes the possibility of audio augmentation but drastically reduces training times and the bottleneck in data loading. As a consequence, the parameters used in the computation of the spectrogram cannot be optimized as hyperparameters and must be set a priori. The parameters for the spectrogram computation are the number of mel frequency bins `n_mels`, which defines the y-axis resolution of the spectrogram; the audio sampling rate `sr`; the window size of the STFT `n_fft`; and the hop length of the STFT `hop_length`. The window size describes the number of sampling points used in one STFT, and the hop length describes the step size of the STFT. The x-axis resolution of the spectrogram is then given by $\texttt{duration} \cdot \texttt{sr}/\texttt{hop\_length} + 1$. The maximum frequency of the spectrogram is determined by $\texttt{sr}/2$, and the minimum frequency is set to 0. The amplitude is restricted between 0 and 80 dB. For the other parameters, the values are chosen to sufficiently resolve distinct audio features while maintaining a low dimensionality. The parameter values are listed in Table 1. All audio files are

Table 1: Parameters of the spectrograms.

Parameter	**Value**
Sampling Rate (`sampling_rate`)	22 050 Hz
Hop Length (`hop_length`)	2048
Number of FFT Points (`nfft`)	4096
Number of Mel Bands (`n_mels`)	128

resampled to the desired sampling rate, and the spectrograms are computed and saved. While training a model, a random slice of the spectrograms is loaded for each recording. If a spectrogram is shorter than the required input size, it is randomly padded with zeros. This procedure results in a constantly changing training dataset, which reduces the possibility of overfitting and increases the diversity of the training data.

3.3. Model Architecture

The architecture of the convolutional neural network is central to the project's success in accurately classifying different bird species. The model needs to have high capacity to effectively extract meaningful audio features from the spectrograms, be robust against noise and variability, and generalize well to the validation data. Furthermore, efficiency and scalability are required to process large amounts of data.
The models that are tested include a simple model consisting of blocks of convolution and dense layers, as well as the `EfficientNetB0` [11], `DenseNet121` [12], and `ResNet50` [13] architectures. EfficientNetB0, DenseNet121, and ResNet50 are architectures designed for image classification and can be directly imported into Keras. Although these models are known for their good accuracy on image classification tasks, they do not produce satisfactory results for this project and are outperformed by the simpler model. The model architecture of the best-performing model tested can be seen in Figure 2. The raw

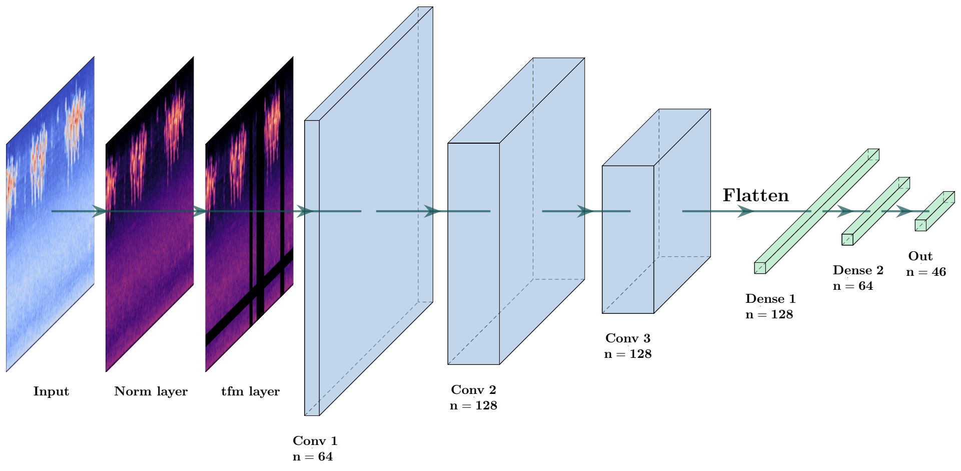


Figure 2: Sketch of the CNN architecture used for this project. Spectrogram from [14].

input spectrograms, with a shape of 162×128, are first normalized using a `ZScoreMinMax` layer from the extension *tensorflow_extra* [15], which rescales the inputs to a mean of 0 and unit variance and then applies min-max scaling between 0 and 1. After normalization, time-frequency masking, implemented in a layer from the same library, is applied to enhance the generalization and robustness of the model. The augmented spectrograms are then processed through a series of convolutional blocks, each consisting of a `Conv2D` layer with a 3×3 kernel, followed by an activation layer and a `MaxPooling2D` layer with a 2×2 pool size. No zero padding is applied in either the convolutional or pooling layers to reduce the dimensionality of the data. The number of filters per convolution layer increases by a factor of 2 and is determined alongside the number of convolution blocks during hyperparameter optimization in Section 3.4. After the convolutional blocks, the data are flattened and passed to "dense blocks", which consist of a dense layer and an activation function. Here, the number of nodes decreases by a factor of 2 in each block.

The output of the neural network is produced in the final dense layer, which uses the *softmax* activation function and has 46 output nodes. For the loss function, *categorical crossentropy* is used. More details on the convolutional and dense blocks are discussed in the next section.

3.4. Hyperparameter Optimization

Due to the long training times of the networks, hyperparameter optimization posed a major challenge. Consequently, a grid search over the entire hyperparameter space, including cross-validation, was not feasible. Instead, the effects of structural hyperparameters are tested on individual models and compared using the tool *Weights & Biases* (WandB) [16]. For other hyperparameters, such as the duration of the audio clips, dropout percentages, or the number of convolutional filters and nodes in a dense layer, a random search using WandB's *sweep* function is performed. The *sweep* function allows to distribute individual runs of the random search to multiple machines, greatly enhancing the possibilities for hyperparameter optimization in this project.
After choosing the basic model architecture as explained in Section 3.3, a first sweep is carried out to determine the activation function used in the convolutional and dense layers, the number of convolutional blocks and filters, and the optimizer of the model. Figure 3 depicts the validation accuracy of 52 individual runs in dependence on the used hyperparameters. Good hyperparameter values can be identified by searching for

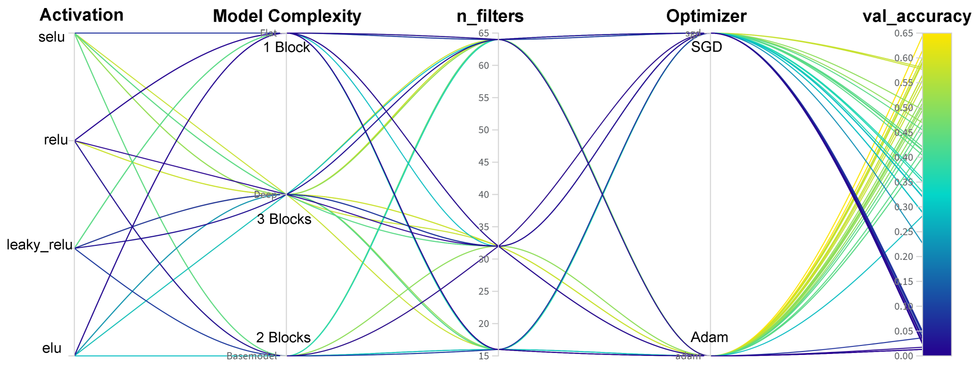


Figure 3: Hyperparamters of the first sweep and their impact on validation accuracy. Since all hyperparameters are discrete in this study, the curves overlap at the nodes, which makes "selu" appear as the best activation function. However, this is not the case.

clusters of brighter colors at the nodes in the graphic. From this search, the number of convolutional blocks (model complexity) is determined to be 3, the optimizer is chosen as "Adam", and the number of filters is further tested with values of 32 and 64, as both yielded good results. Although it does not seem so in Figure 3, the impact of the activation function is not as significant, and therefore, "leaky_relu" is chosen because it was used in the run with the overall best validation accuracy.

In a second sweep, another 29 models are trained to study the impact of time-frequency masking (see 3.3) and batch normalization layers, the number of convolutional filters, and the duration of the audio clips used in training. Batch normalization layers are placed between the convolutional or dense layers and their respective activation functions and scale the inputs to zero mean and unit variance. This is expected to accelerate training, improve convergence, and help stabilize the learning process. However, batch normalization is not included in the final model, as it causes strong oscillations in the validation accuracy, as shown in Figure 4. The impact of the other hyperparameters in this sweep can be seen in Figure 5. The number of filters is determined to be 64, and the

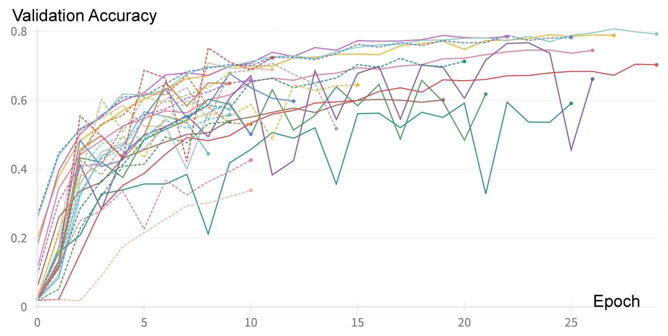


Figure 4: Validation accuracy against epochs for the runs of the second sweep. For some curves, oscillations due to batch normalization can be observed.

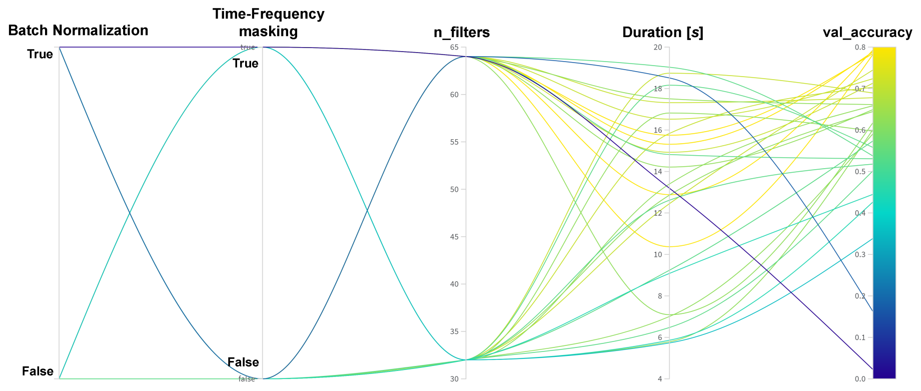


Figure 5: Hyperparamters of the second sweep and their impact on validation accuracy.

duration is chosen as 15 s; shorter durations seem to reduce, and longer durations do not benefit the validation accuracy. In the final sweep, 22 models are trained to determine suitable values for the dropout percentages of dropout layers, which are placed after the last convolutional layer (dropout1) and the first dense layer (dropout2). Including more dropout layers is found to significantly reduce the model's overall performance. The best dropout values are dropout1 = 0.0828 and dropout2 = 0.2718. A plot of the results of this sweep can be found in Appendix A.3.

4. Results

After determining the best hyperparameters of the model, a final set of models is trained and evaluated on the test data. As explained in Section 3.1, a stratified k-fold with $k = 5$ is employed to train five different models exploiting all available training data while maintaining the ability to check for generalization on a validation data subset. The models are trained for a minimum of 30 epochs, and early stopping is used to halt training if the validation accuracy does not increase for six consecutive epochs. The model states with the highest validation accuracy are saved and used for evaluation. Figure 6 shows the mean loss and accuracy of all five models on training and validation data. As overfitting becomes apparent starting from epoch 40, the models are retrained

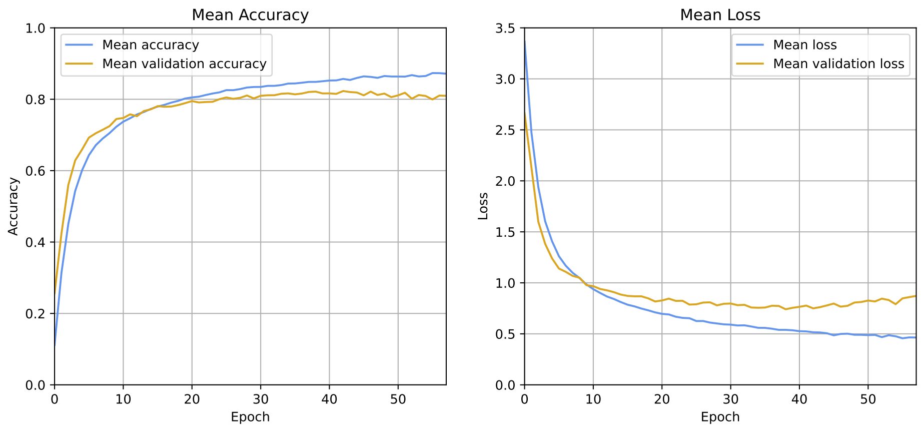


Figure 6: Mean loss and accuracy of the five classifiers on training and validation data.

using a higher dropout percentage (dropout1 = 0.15, dropout2 = 0.3) and including additional L2 regularization in the dense layers with a strength of `l2_lambda = 0.0001`. The mean loss and accuracy for the retrained models is shown in Figure 7.
The scores for the accuracy and balanced accuracy on the test dataset are

$$\text{accuracy} = 90.02\,\% \qquad\qquad \text{balanced accuracy} = 88.85\,\%$$

with a statistical baseline of 3.34 % for randomly guessing the most frequent class. The values on training data are 93.88 % (93.82 %) for the (balanced) accuracy score. In Figure 8, the confusion matrix for predictions on the test data is shown. The majority of entries lie on the diagonal, indicating that the overall classification is successful. However, some classes are less accurately classified than others. Class 24, the Eurasian Tree Sparrow ("Passer montanus"), has the lowest accuracy at 45.45 %, with 38.64 % of cases mistakenly classified as the House Sparrow ("Passer domesticus") (33). This appears reasonable, as the two species are closely related and can be easily mistaken. Other examples are the long-eared owl (36) at 71.79 %, which is mistaken with an Eurasian eagle-owl in 17.95 %. Further, for at least 5.14 % of the wrongly classified data, the predicted bird species is

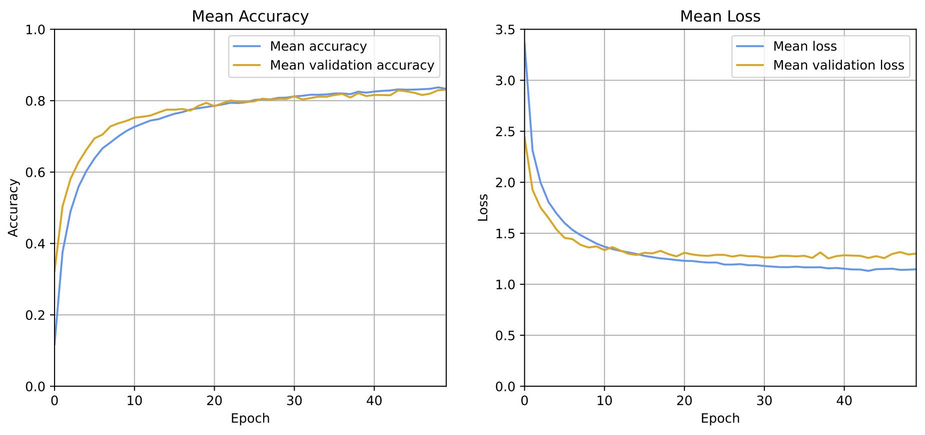


Figure 7: Mean loss and accuracy of the five classifiers after retraining the model with more regularization.

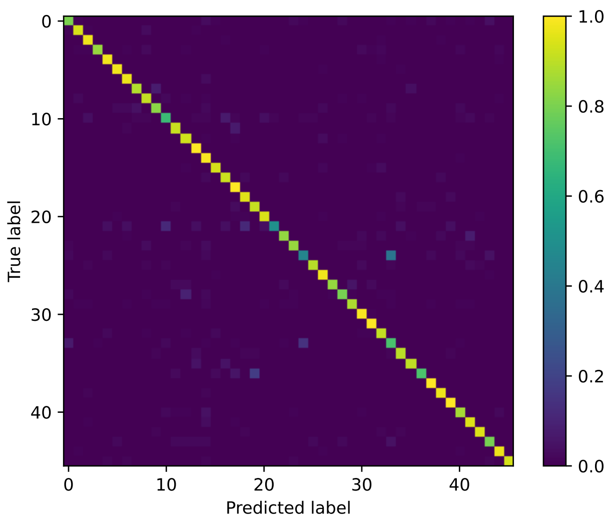
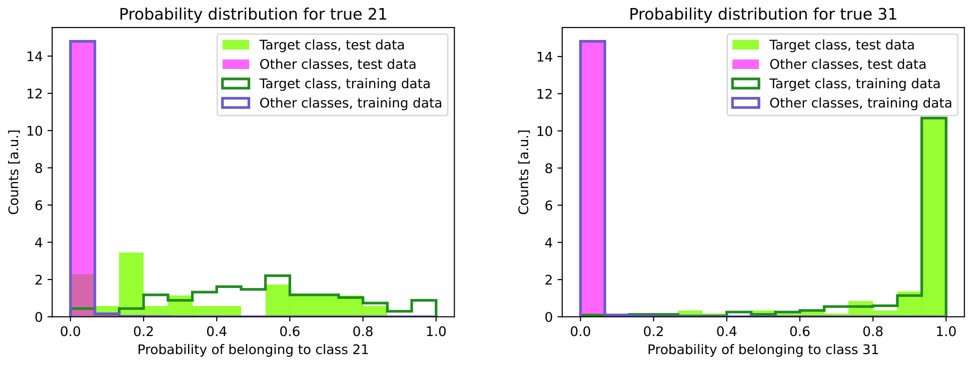


Figure 8: Confusion matrix of the predictions on test data.

listed in the category "also", which lists other bird species present in the recording. For 57.43 % of the wrong classifications, the label with the second-highest probability is the correct label. Nevertheless, the classification output has some errors. In Figure 9, the classifier's output for the true label and all other labels is plotted for training and test data for two examples of good and poor classification. Especially for the example of the



(a) Poor separation. **(b)** Good separation.

Figure 9: Two examples of classification output for training and test data for the target class and all other classes. One example shows a poor- and the other a good separation.

poor classification performance, overfitting is present to some extent, as the classification on training data seems to have a better separation between the true and wrong labels than for the test data. However, this might also result from random fluctuations given the relatively small test dataset.

5. Alternative Method

To compare the solution presented in this report to an alternative method, one of the simplest machine learning algorithms is chosen: the k-Nearest Neighbors (kNN) classifier. The kNN algorithm computes the Euclidean distance in the feature space between a datapoint to classify and the saved training data. Therefore, the spectrograms cannot be used as input for this algorithm. Instead, 45 audio features are computed on the raw audio wave forms. These features include the zero crossing rate, measures for the loudness of the signal, spectral features, statistical features like the mean and standard deviation, but also features used in music analysis, like spectral contrast and chroma features. For all features, the mean with respect to their time evolution is taken. The complete list of features can be found in Appendix A.4. Despite the high dimensionality of the feature space, these features can directly be used to train the kNN classifier. Feature reduction by e.g. using principal component analysis is not required and is found to reduce the accuracy on validation data. Since the computed features have different magnitudes, scaling is needed. Different scaling methods are tested and a *Quantile Scaler* from the library *Scikit-Learn* [10] with an output normal distribution is found to yield the highest accuracy on validation data. Again, a stratified k-fold with $k = 5$ is performed and five individual kNN classifiers are fitted with their respective training data. For each classifier, different values for k neighbors are tested on the validation data and the one with the highest validation accuracy is chosen. The five kNN classifiers are evaluated on the same test data as the CNN by taking the averaged prediction of the different classifiers. The confusion matrix of the kNN ensemble is shown in Figure 10. As can

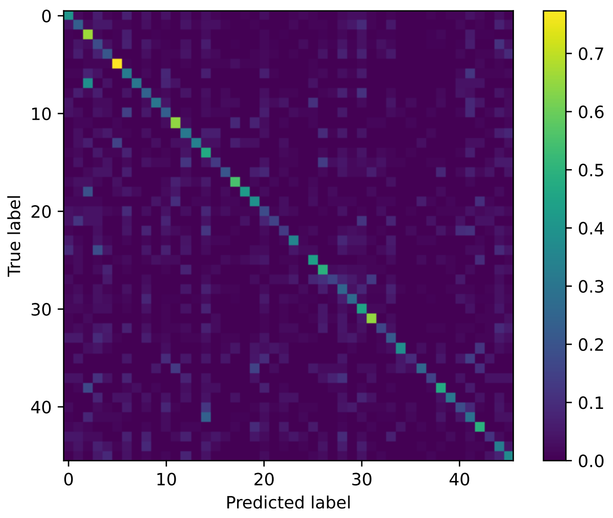


Figure 10: Confusion matrix of the predictions on test data for the kNN ensemble.

be seen, the prediction is significantly less accurate than the neural networks prediction. The (balanced) accuracy reads 33.49 % (31.95 %) for the kNN ensemble.

6. Conclusion

This project investigated the research question of how accurately a CNN can classify 46 different European songbird species based on spectrograms computed from audio recordings. After building a robust and efficient data pipeline, different model architectures were tested, and hyperparameter optimization for the most promising model architecture was performed. For the evaluation, a total of five networks were trained using a stratified k-fold. The predictions on a previously separated test dataset showed an (balanced) accuracy score of 90.02 % (88.85 %). To compare this approach to a simpler method, an ensemble of k-Nearest Neighbors classifiers was trained using 45 audio features extracted from the raw waveforms of the audio. Here, the evaluation returned values of 33.49 % (31.95 %) for the (balanced) accuracy score. The convolutional neural network clearly outperforms the kNN classifier at this task. This is not surprising, as the kNN relies on pre-computed audio features that do not contain as much information as the 162×128 sized spectrograms used in the CNN, which is designed to learn the most meaningful

features on its own. Still, the *k*NN's performance is surprisingly good, compared to the statistical baseline of 3.34 %. An advantage of the *k*NN classifier are the lower fitting and prediction times compared to the CNN. After the audio features are computed, the entire training process for the *k*NN takes just a few seconds, whereas the convolutional neural network requires several hours of training.
Considering the number of classes, the CNN demonstrated excellent performance in distinguishing between the different bird species. An increase in accuracy could potentially be achieved by using a more complex model architecture and further optimizing the hyperparameter space. However, this would require more computational resources and time. Additionally, a larger dataset and higher data quality could improve the results. Nevertheless, the research question can be answered positively: Modern machine learning algorithms, such as neural networks, are well-suited for complex classification tasks and can aid humans in research, such as monitoring bird species populations. In conclusion, the project can be viewed as a successful demonstration of the strengths and practical benefits of machine learning methods in real-world applications.

References

[1] Leon Cohen. "On the Short-Time Fourier Transform of Stationary Stochastic Processes." In: *IEEE Transactions on Information Theory* 10.3 (1964), pp. 119–124. DOI: 10.1109/TIT.1964.1053656.

[2] S. S. Stevens, J. Volkmann, and E. B. Newman. "A scale for the measurement of the psychological magnitude pitch." In: *Journal of the Acoustical Society of America* 8.3 (1937), pp. 185–190. DOI: 10.1121/1.1915893.

[3] François Chollet. *Deep Learning with Python.* Manning Publications, 2017.

[4] Bob Planqué and Willem-Pier Vellinga. *xeno-canto - Sharing wildlife sounds from around the world.* Available at https://xeno-canto.org/ (2024/25/07).

[5] Juan Sebastián Ulloa et al. "scikit-maad: An open-source and modular toolbox for quantitative soundscape analysis in Python." In: *Methods in Ecology and Evolution* (2021). DOI: https://doi.org/10.1111/2041-210X.13711%7D.

[6] Paul Kelly. *ID: XC901667.* Available at https://xeno-canto.org/901667 (2024/26/07).

[7] Albert Noorlander. *ID: XC561205.* Available at https://xeno-canto.org/561205 (2024/26/07).

[8] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* Software available from tensorflow.org. 2015. URL: http://tensorflow.org/.

[9] François Chollet et al. *Keras.* https://keras.io. 2015.

[10] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[11] Mingxing Tan and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks." In: *Proceedings of the 36th International Conference on Machine Learning*. 2019. arXiv: `1905.11946`.

[12] Gao Huang et al. "Densely Connected Convolutional Networks." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 4700–4708. arXiv: `1608.06993`.

[13] Kaiming He et al. "Deep Residual Learning for Image Recognition." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778. arXiv: `1512.03385`.

[14] Jorge Leitão. *ID: XC862478*. Available at `https://xeno-canto.org/862478` (2024/26/07).

[15] Awsaf Hossain. *tensorflow_extra: Extensions for TensorFlow*. `https://github.com/awsaf49/tensorflow_extra`. Version 1.0.2. Accessed: 2024-07-26. 2022.

[16] Lukas Biewald. *Experiment Tracking with Weights and Biases*. Version 0.17.4. Software available from wandb.com. 2020. URL: `https://www.wandb.com/`.

A. Appendix

A.1. List of bird species

Table 2: List of bird species used in this project. The classification label, the scientific name, as well as the english and german names of the species are listed.

Label	Scientific Name	English Name	German Name
0	*Hirundo rustica*	Barn Swallow	Rauchschwalbe
1	*Phoenicurus ochruros*	Black Redstart	Hausrotschwanz
2	*Turdus merula*	Common Blackbird	Amsel
3	*Fringilla coelebs*	Common Chaffinch	Buchfink
4	*Phylloscopus collybita*	Common Chiffchaff	Zilpzalp
5	*Cuculus canorus*	Common Cuckoo	Kuckuck
6	*Luscinia megarhynchos*	Common Nightingale	Nachtigall
7	*Tringa totanus*	Common Redshank	Rotschenkel
8	*Phoenicurus phoenicurus*	Common Redstart	Gartenrotschwanz
9	*Gallinago gallinago*	Common Snipe	Bekassine
10	*Sturnus vulgaris*	Common Starling	Star
11	*Columba palumbus*	Common Wood Pigeon	Ringeltaube
12	*Prunella modularis*	Dunnock	Heckenbraunelle
13	*Botaurus stellaris*	Eurasian Bittern	Rohrdommel
14	*Sylvia atricapilla*	Eurasian Blackcap	Mönchsgrasmücke
15	*Cyanistes caeruleus*	Eurasian Blue Tit	Blaumeise
16	*Pyrrhula pyrrhula*	Eurasian Bullfinch	Gimpel
17	*Streptopelia decaocto*	Eurasian Collared Dove	Türkentaube
18	*Numenius arquata*	Eurasian Curlew	Großer Brachvogel
19	*Bubo bubo*	Eurasian Eagle-Owl	Uhu
20	*Oriolus oriolus*	Eurasian Golden Oriole	Pirol
21	*Garrulus glandarius*	Eurasian Jay	Eichelhäher
22	*Sitta europaea*	Eurasian Nuthatch	Kleiber
23	*Spinus spinus*	Eurasian Siskin	Erlenzeisig
24	*Passer montanus*	Eurasian Tree Sparrow	Feldsperling
25	*Scolopax rusticola*	Eurasian Woodcock	Waldschnepfe
26	*Troglodytes troglodytes*	Eurasian Wren	Zaunkönig
27	*Lophophanes cristatus*	European Crested Tit	Haubenmeise
28	*Carduelis carduelis*	European Goldfinch	Stieglitz
29	*Chloris chloris*	European Greenfinch	Grünfink
30	*Erithacus rubecula*	European Robin	Rotkehlchen
31	*Regulus regulus*	Goldcrest	Wintergoldhähnchen
32	*Parus major*	Great Tit	Kohlmeise

Continued on next page

Label	Scientific Name	English Name	German Name
33	*Passer domesticus*	House Sparrow	Haussperling
34	*Athene noctua*	Little Owl	Steinkauz
35	*Charadrius dubius*	Little Ringed Plover	Flussregenpfeifer
36	*Asio otus*	Long-eared Owl	Waldohreule
37	*Anthus pratensis*	Meadow Pipit	Wiesenpieper
38	*Turdus viscivorus*	Mistle Thrush	Misteldrossel
39	*Vanellus vanellus*	Northern Lapwing	Kiebitz
40	*Turdus iliacus*	Redwing	Rotdrossel
41	*Turdus philomelos*	Song Thrush	Singdrossel
42	*Strix aluco*	Tawny Owl	Waldkauz
43	*Motacilla alba*	White Wagtail	Bachstelze
44	*Phylloscopus trochilus*	Willow Warbler	Fitis
45	*Emberiza citrinella*	Yellowhammer	Goldammer

A.2. Class distributions for training and test data

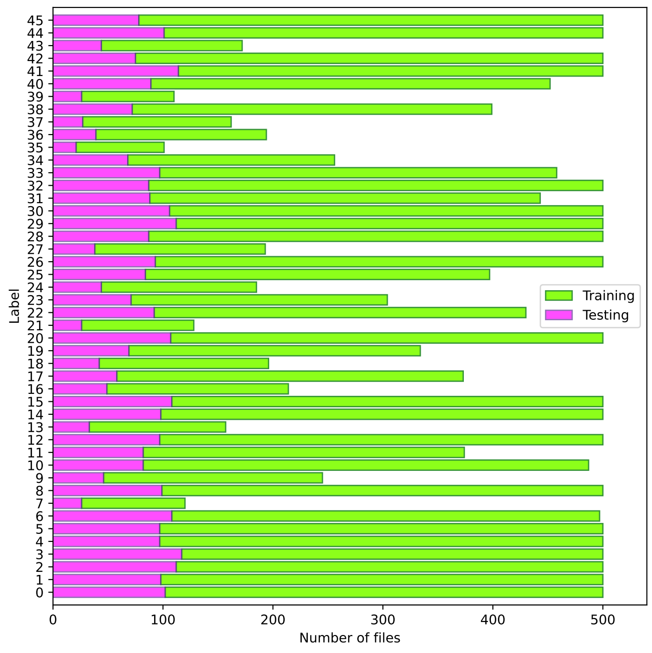


Figure 11: Class distributions of the number of files for training and test dataset (stacked). As can be seen, the classes are imbalanced.

A.3. Additional material on the hyperparameter optimization

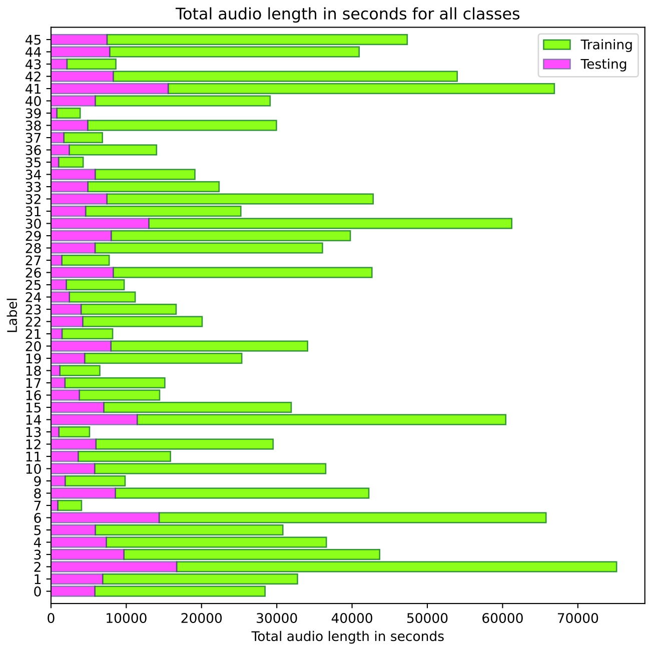


Figure 12: Class distributions of the total audio length for training and test dataset (stacked). The imbalance gets more severe due to the fact, that classes with a lower number of files tend to also have shorter recordings.

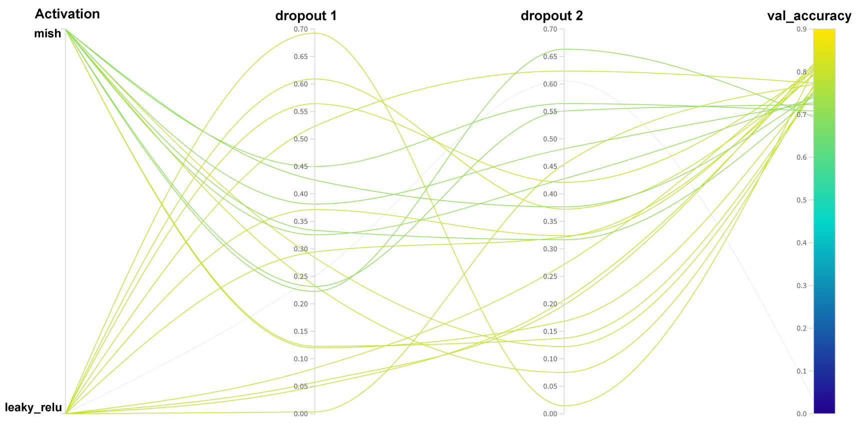


Figure 13: Hyperparamters of the third sweep and their impact on validation accuracy. A new activation function, the "mish" activation function is also tested but not found to yield better results. The study on the dropout percentages is inconclusive.

A.4. Audio features used for the kNN classifier

Table 3: List of Extracted Audio Features

Feature (count)	**Description**
Zero Crossing Rate	The rate at which the signal changes sign.
Root Mean Square Energy	The square root of the average of the squared values of the signal, representing signal power.
Spectral Centroid	The ”center of mass” of the spectrum, indicating where the bulk of the energy is concentrated.
Spectral Bandwidth	The width of the spectrum, representing the spread of frequencies around the centroid.
Spectral Flatness	A measure of how flat or peaky a spectrum is. High flatness indicates noise-like signals.
Spectral Rolloff	The frequency below which a certain percentage of the total spectral energy is contained.
Mean	The average amplitude of the audio signal.
Standard Deviation	The amount of variation or dispersion of the audio signal.
Skewness	The asymmetry of the signal amplitude distribution.
Kurtosis	The ”tailedness” of the signal amplitude distribution.
Mel-frequency Cepstral Coefficients (10)	A set of coefficients that represent the short-term power spectrum of the audio signal.
Spectral Contrast (7)	The difference between peaks and valleys in the spectrum, calculated in sub-bands.
Chroma Features (12)	The distribution of energy across the 12 pitch classes of the chromatic scale.
Tonnetz (6)	Harmonic and tonal features representing the tonal centroid features of the signal.