Bird Sound Recognition Using a Convolutional Neural Network

Ágnes Incze*, Henrietta-Bernadett Jancsó*, Zoltán Szilágyi†, Attila Farkas† and Csaba Sulyok*
*Faculty of Mathematics and Computer Science, Babeș-Bolyai University
RO-400084 Cluj-Napoca, Romania
†Codespring Ltd.
RO-400664 Cluj-Napoca, Romania
Email: incze.agnes@yahoo.com

***Abstract*— Convolutional neural networks (CNNs) are powerful toolkits of machine learning which have proven efficient in the field of image processing and sound recognition.**

In this paper, a CNN system classifying bird sounds is presented and tested through different configurations and hyperparameters. The MobileNet pre-trained CNN model is fine-tuned using a dataset acquired from the Xeno-canto bird song sharing portal, which provides a large collection of labeled and categorized recordings. Spectrograms generated from the downloaded data represent the input of the neural network.

The attached experiments compare various configurations including the number of classes (bird species) and the color scheme of the spectrograms. Results suggest that choosing a color map in line with the images the network has been pre-trained with provides a measurable advantage. The presented system is viable only for a low number of classes.

***Keywords*-convolutional neural network; bird sound recognition; transfer learning; audio classification**

I. Introduction

In recent years, algorithmic sound recognition has enjoyed a steady increase in interest [1], [2]. The popularity of deep learning and the many types of neural networks have provided a new unexplored mechanism of approaching such classification problems.

Out of the many niche sound categories suitable for this type of classification, the current research focuses on bird sounds. We set the target audience as nature enthusiasts and ornithologists who would benefit from a hands-on way to tell bird species apart merely by audible traits, as often in the wild it is difficult to spot an elusive songbird.

The current research uses transfer learning [3] to fine-tune an existing neural network to recognize bird sounds. Many such pre-trained networks are taught to recognize general features in images. However, audio is represented in one dimension, while pictures are two-dimensional signals, therefore a representative transformation is needed for compatibility. To this end, we use spectrograms, which are a visual representation of the magnitude returned by the Short Time Fourier Transform (STFT)[4]. STFT is a version of the Discrete Fourier Transform (DFT), which instead of only performing one DFT on a longer signal, splits the signal into partially overlapping chunks and performs the DFT on each using a sliding window. This yields a two-dimensional spectral representation of an audio slice, where time and frequency denote the axes. The spectrogram uses a color map to view the STFT output as an image, which can then be input to an image-based pre-trained network.

We use the compact and performance oriented MobileNet [5] as our starting checkpoint. Our experiments provide a comparative study of relevant configurations of the system, such as the number of classes and the color map used in spectrograms.

Extensive research has been conducted in the recent past dissecting potential approaches to the presented issue. A rise in interest may be attributed to the annual BirdCLEF [6] recognition challenge: a biodiversity data evaluation campaign. The training dataset of BirdCLEF 2017 comprises over 36,000 audio files from 1500 different species, collected from Xeno-canto, with classes not necessarily having an equal number of sound samples. The challenge focuses on recognizing single audible species as well as separating multiple overlayed sounds in field recordings.

In 2017, Kahl et al. [7] undertake the challenge, with their experiments measuring 60% accuracy for overlayed sounds and 68% for recognizing the dominant species. In 2016, Piczak [8] approaches this problem similarly with 41.2% accuracy for multi-labeled and 52.9% for single labeled data. His experiments involve networks trained from scratch, mel-scaled power spectrograms, an upper frequency cap and noise filtering.

The winner of the 2016 BirdCLEF challenge, Sprengel et al. [9] (68.6% for single labeling and 55.5% for multiple labeling) discuss classification with CNN containing five convolutional and one dense layer. As input, spectrograms are generated from the audio files after splitting the noise from the actual bird sound.

The literature also proposes alternative or augmentative methodologies to CNNs, such as unsupervised learning [10], decision tree based feature selection [11], recurrent CNNs [12] or Hidden Markov models [13].

Different approaches to visual representation of audio have also been researched. While spectrograms seem to be an intuitive choice [11], using the raw audio data as input [10], mel-frequency cepstral coefficients (MFCCs) [14] and Constant-Q transformed spectrograms [15] have also been tested. In

2015, Piczak et al. [16] propose that the spectrogram surpasses MFCCs as a sound visualization mechanism in case of CNNs. In his research involving template matching and bagging, Lasseck [11] proposes spectrogram downsampling as a performance improvement without significant detriment to the quality.

Our methodology deviates from the presented approaches by incorporating transfer learning using a pre-trained CNN; using a different geographical region for filtering recordings from Xeno-canto to that of BirdCLEF; linearly represented magnitude spectrograms as well as balancing the training data by using an equal number of recordings per class.

Transfer learning for bird song recognition has recently been proposed by Fritzler et al. [17] who state that fine-tuning a pre-trained network may even perform better than one with no prior training.

The structure of the current document is as follows: Section II describes the methodology and the workflow of setting up and fine-tuning a MobileNet architecture based frozen inference graph. Section **??** discusses the architecture of the application which provides a way to access the evaluation, while Section III contains the experiments conducted and the results thereof. Section IV draws the appropriate conclusions and discusses possible future improvements.

II. Methodology

In this section we discuss the general concepts and methodologies involved in the forming of the current bird sound recognition system. The two steps involved in the system setup (see Figure 1) are the offline training of the CNN using the appropriate data and the online evaluation for a single sound.

The first step evoked during the model creation is the scripted gathering of the dataset recordings from Xeno-canto. Upon download, these recordings arrive in MP3 format; they are converted to WAV and separated into training and test datasets (further details in Sections II-B).

This is followed by the preprocessing of the sound files: the WAV files are split into chunks of equal length and normalized. In order to work only with segments containing relevant information, a threshold filtering is applied, discarding chunks which are not loud enough. A Short-time Fourier transform (STFT) is then performed on the sounds, followed by a normalization. Finally, the spectrograms are created by converting the STFT output into an image using a color map (see Section II-C).

In order to train the model for classification, the labeled spectrogram images are grouped into TFRecord format files. The output of the training phase is an inference graph, which allows evaluating new incoming spectrograms.

The main application of the trained model is to classify a single recording of a bird sound. The user records the sound and uploads it to the server, which converts it into the WAV format and performs the preprocessing. The output will be 0 or more spectrograms depending on the threshold value; they represent the input of the inference graph. The results provided by the evaluation after an averaging are returned to the user.

A. Corpus

Xeno-canto is an open website dedicated to sharing bird sounds; users upload their own recordings and label them by genus, species, subspecies, location, type, quality (from A to E, where A is the best quality of the sound) etc. Sounds may also be classified as "calls" or "songs". The calls are short, alarming bursts males use to signal their territories, while songs are longer and mainly used by males to attract female birds. In case the species of the bird is not known, the uploaded sound may be flagged as a "mystery" and further descriptions may be provided, so that other users can assist in identification. Currently the site hosts almost 400,000 recordings spanning over 6,000 hours, representing close to 10,000 species, courtesy of over 4,000 uploaders.

Since many different labels (genus, species and subspecies) are supported by Xeno-canto, for our experiments we use a combination of two to define our classes: genus and species. We include both type of sounds (calls and songs) in the corpus, but filter based on quality to only include rank A.

The data gathering process for our experiments employ automated scripts making use of Xeno-canto's RESTful API [1]. This API returns JSON responses for REST requests, containing information about the sounds: among others the number of recordings, species, pages, the current page as well as the recordings themselves. Some labels allow you to filter between the recordings such as the country and quality.

B. Automated data download

In order to train the model, a homogeneous dataset is needed, which contains a considerable amount of labeled recordings. We automate the selection and download process with the help of a script employing one of the following strategies:

- We can provide a file containing the name(s) of the species; this infers the number of classes.
- The file can be generated, by giving the number of European bird species we want to download recordings for.
- We can choose not to specify the names or the number of bird species, in this case the script will download all recordings of all European species known by the Xeno-canto site.

Configurable options include the location where the recordings, the generated spectrograms and the TFRecord files are all stored.

The current experiments apply the second strategy. Given a number of classes, the script sends requests to the Xeno-canto API in order to collect all the recordings of European bird species which have the most recordings labeled with good quality. It caches the classes with the most recordings for further use. It proceeds to count the number of recordings for each species, saving the minimum count. Only the minimum for each species is downloaded; for classes with more recordings,

[1]API described here: https://www.xeno-canto.org/article/153, last visited: 21.04.2018

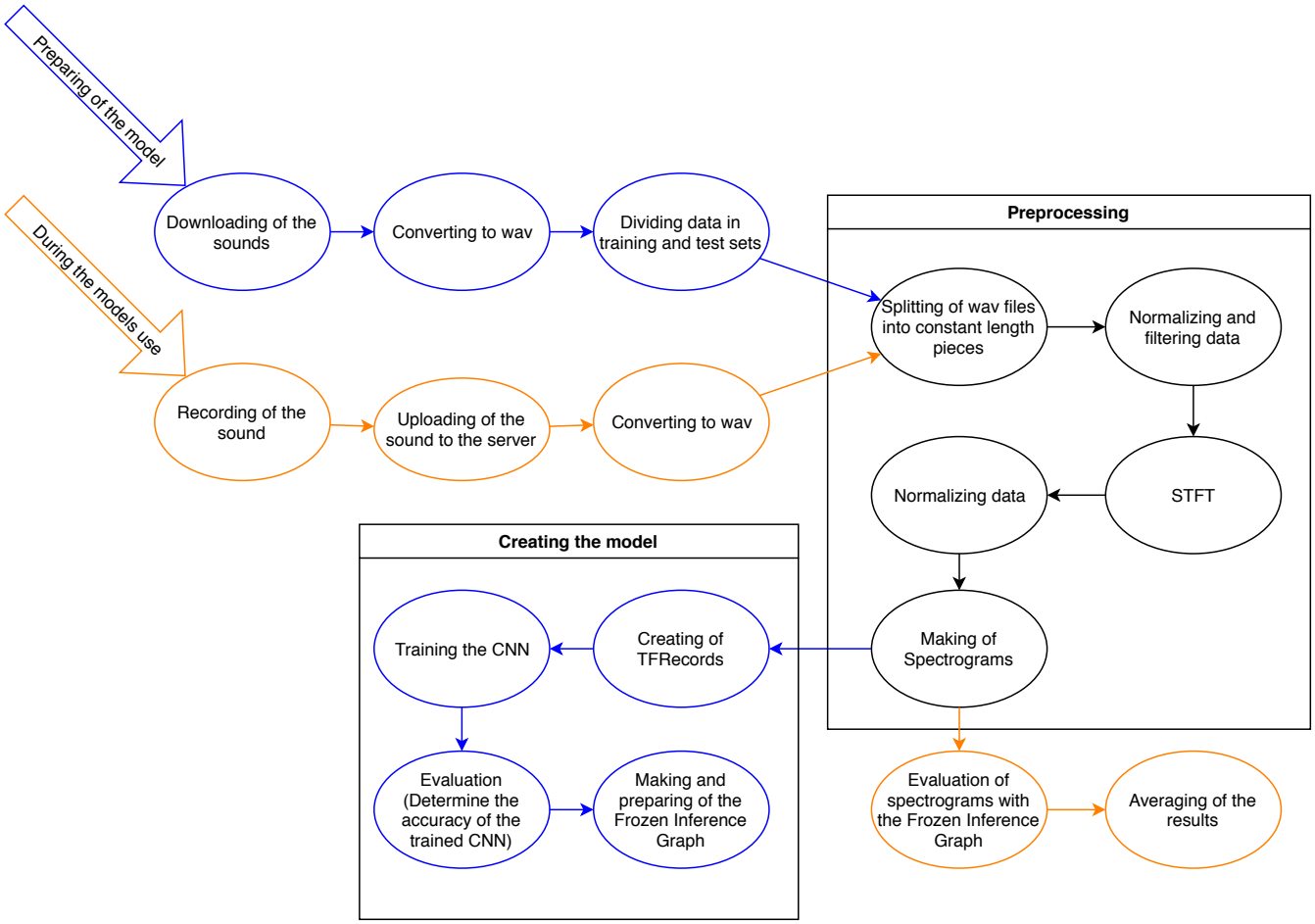


Fig. 1: Model workflow in the stages of preparation (blue) and evaluation (orange). Certain steps, such as the preprocessing, overlap(black) during both scenarios.

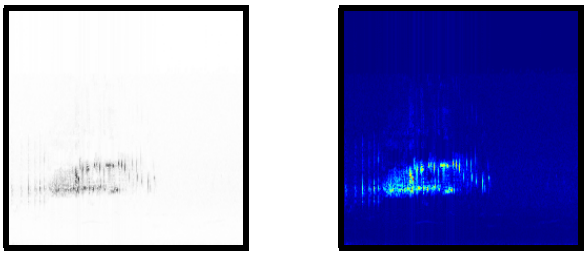
the used ones are randomly selected. Thus every class will have the same number of samples to be used for training and testing.

The downloaded audio is converted into WAV with a sample rate of 44100Hz and separated into 80% training dataset and 20% test dataset.

C. Sound Preprocessing

During the preprocessing algorithm, the data is normalized and split into three second long chunks, the last shorter one of which is discarded. Every segment is evaluated by thresholding to make sure it is not too quiet: a configurable value is compared to the RMS gain of the chunk; if the gain is lower the chunk is discarded.

The remaining segments are subject to a Short Time Fourier Transform (STFT) to extract spectro-temporal information. Since the used neural network model needs fixed-size 224x224 pixels input, we use a window size of 448 during the STFT. We also use a Hamming window with an overlap of 224 (half of the window size, also configurable). After the STFT is done, we retain only its magnitude, because the phase information is not needed for the spectrogram portrayal. The



(a) Grayscale color map (b) Jet color map

Fig. 2: Spectrogram of a sound of the bird Emebriza Calandra, rendered through different color maps.

normalized STFT output is subsampled/decimated to yield an exact 224x224 matrix. Gamma correction may be applied at this point, which amplifies middling information on an even keel, possibly accentuating soft details.

The spectrogram image is rendered using a color map: a scale of colors mapped to values in the $[0, 1]$ interval. Our experiments compare the use of two different colors maps (see Figure 2):

- *grayscale* - White represents 0, black represents 1, with shades of gray in-between. This linear representation should intuitively translate well to machine understanding.
- *jet* - An RGB color map where blue corresponds to 0, red is 1 and the values around 0.5 are yellow. This representation is often used for illustrating spectrograms, since it is naturally understandable for human vision. Its non-linear scale may hinder its potential for computational understanding.

The spectrogram is saved in PNG format to a configurable location.

D. Pre-trained networks: TF-Slim and MobileNet

The training and evaluation algorithm of the convolutional neural network is implemented using the TensorFlow deep learning framework [18]. It supports computationally demanding parallelizable algorithms while also giving the possibility of execution on CPU, GPU or TPU.

The experiments are also assisted by the TF-Slim TensorFlow library [19], which provides an easily usable environment for creating, training and evaluating neural networks. It contains auxiliary functions for writing and reading TFRecords, creating and evaluating inference graphs and downloading well known datasets such as CIFAR-10, MNIST and ImageNets. TF-Slim also allows working with popular convolutional neural networks, such as Inception, ResNet, VGG, MobileNets. TF-Slim provides several public and compatible pre-trained networks (checkpoints) which greatly reduce training times [3].

Our experiments use the MobileNet [5] pre-trained network. The accuracy of the model is assessed to be 70.9% based on training on the ILSVRC-2012-CLS image classification dataset. The chosen version of MobileNet accepts 224x224 size colored (RGB, 3 channels) images as input. The reasons for choosing this particular architecture were as follows: it was specifically created for mobile devices, it has a relatively small architecture, and it is fast in evaluation.

The base of its architecture is the Depthwise Separable Convolution. This layer separates the conventional convolutional network into two different layers: a depthwise convolution, which in this case uses one single filter for each input; and a 1x1 pointwise convolution, which is applied on the depthwise convolution. This process executes both filtering and convolving in two layers, which usually is computed in one layer. Even though the convolution is performed in two steps, the computations are more simple, thus the used computational resources are reduced.

The MobileNet consists of 28 layers. The first layer is a conventional convolutional network, which is followed by depthwise convolutional layers and pointwise convolutional layers. Each convolutional layer is followed by a Batch Normalization [20] and a ReLU activation function. The last three layers are the average pooling, fully connected (FC) and Softmax layers.

E. Training the network

Training the network is a process that requires learning data as input, and creates a model that can be evaluated by test data. Our learning and test data is the output of the download and preprocessing procedure. The program writes these data into TFRecord files. It is necessary for the TFRecord files in case of both the learning and the test dataset to contain various sample of the classes specified. Thus we shuffle our dataset along with the corresponding tags before writing them into TFRecord files.

The TFRecord is a file used in Tensorflow. Since it is simple to use, it is a popular data archiver among Tendorflow users. It contains informations about the data such as in our case the spectrograms alongside with the labels (tag of the bird species), size of the picture and informations about encoding and decoding. One TFRecord file can contain multiple data structures described beforehand. We can specify how many sample of data we want to store in one TFRecord.

Creating the TFRecord files is the last step of preparing the corpus to train the network. The implementation of algorithm that performs the training and the pre-trained MobileNet CNN is provided by the TF-Slim library. Thus our training starts by a simple function call. In the course of the different learning processes we performed, the parameters discussed in III section, are constant, indifferent from the number of classes or the type of color map used in creating the spectrograms.

Evaluating the trained model is also executed by a function from TF-Slim. It's main result is the accuracy of the model.

The output of the training are checkpoints. These files contain informations about the state of the trained model. They can be used to continue training from the last training step with different hyper parameters. An another usage of the checkpoints are to create inference graphs from them.

Inference graphs are used to classify one single input. Creating and handling the inference graphs are also resolved by the TF-Slim package.

III. Experiments and results

The conducted experiments cover different configurations in order to observe the efficiency of the network. Parameters altered between runs include the number of classes (2, 10 and 50) and the color map used for creating the spectrograms. The latter takes on the values "grayscale" (see Figure 2a) and the RGB encoded "Jet" (see Figure 2b). This results in 6 different configurations.

We train networks using each configuration 5 times for 10,000 steps, using fixed hyperparameter values. The learning rate is set to 0.0001, the weight decay is 0.00004, and the size of the batches in which we split the dataset is 32. Since sampling the recordings from the Xeno-canto site during the download was randomized, the dataset is diversified.

The loss drop of each individual configuration is visualized on Figure 3. The system shows a steady loss drop in all scenarios, with no significant difference between the two color maps. As expected, a higher number of classes leaves the loss function at higher values.

Figure 4 contains the accuracy and recall values achieved by the trained models. As expected, the model performs well for 2 classes, but the accuracy decreases for a higher number of classes. Already for 10 classes, the accuracy drops below 40% for both color maps.

The trainings based on the Jet color map show higher accuracies by an average of 7.4%. The MobileNet checkpoints used in the experiments used a network which had lower layers specifically pre-trained for color images; this may explain the better response of the network to the colored spectrograms. Furthermore, increasing the number of classes widens the gap in the favor of the Jet color map, making it a more viable competitor when using more classes.

The 5 different training instances per configuration yield negligible differences as presented by the minimal deviations in Figure 4. Thus we can conclude that the influence of the color map and the class count is reproducible in a stable manner.

IV. Conclusion and Future Work

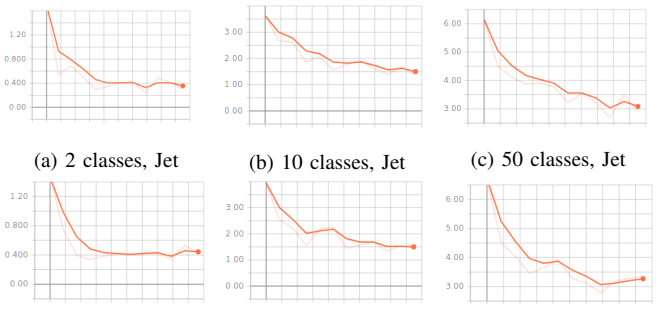
The field of bird sound recognition using machine learning methods has seen a steady increase in recent years, with most works focusing on training various neural networks from the start. As an alternative and possibly more performant solution, the current research proposed transfer learning: fine-tuning a pre-trained network using a visual representation of sound; in our case, spectrograms.

We have successfully demonstrated the viability of the MobileNet network fine-tuned with images containing data much different from the images used in its low level pre-training. Initial experiments show that reasonable accuracies may be achieved when using only 2 classes.

Our experiments also compare the color maps used in spectrogram creation. The results suggest that RGB spectrograms are more effective than their linear black and white counterparts, possibly due to the lower layers of MobileNet being trained on colored images. This difference between the color and grayscale accuracies increase steadily when using more classes.

In order to achieve higher accuracies, the following improvements should be considered. Using a bigger, more robust pre-trained convolutional neural network such as ResNet may help accuracies reach larger numbers. To properly differentiate and compare RGB spectrograms and grayscale spectrograms, in the case of the gray spectrograms the lower layers should be trained on grayscale images.

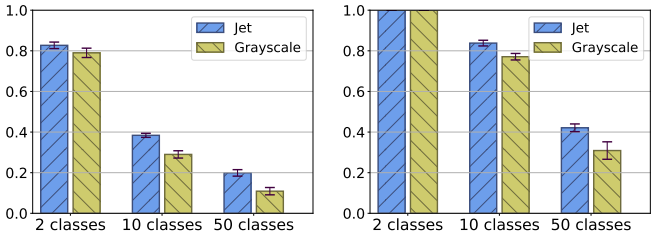
Furthermore, during the preprocessing stage, we propose noise reduction as well as filtering to discard extreme frequencies not present in bird sounds [8]. Testing different gamma values could also increase the amount of helpful information the spectrograms contain.



(a) 2 classes, Jet (b) 10 classes, Jet (c) 50 classes, Jet

(d) 2 classes, grayscale (e) 10 classes, grayscale (f) 50 classes, grayscale

Fig. 3: Losses reported during the training process



(a) Accuracy (b) Recall

Fig. 4: Accuracy and recall achieved by our experiments. The results are averaged over 5 runs for each configuration, with the standard deviation overlayed.

References

[1] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, pp. 279–283, 2017.

[2] V. Bisot, R. Serizel, S. Essid, and G. Richard, "Leveraging deep neural networks with nonnegative representations for improved environmental sound classification," *IEEE International Workshop on Machine Learning for Signal Processing MLS*, 2017.

[3] D. Gupta. (2017) Transfer learning and the art of using pre-trained models in deep learning. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/06/transfer-learning-the-art-of-fine-tuning-a-pre-trained-model/

[4] J. Allen, "Short term spectral analysis, synthesis, and modification by discrete fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 3, pp. 235–238, Jun 1977.

[5] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017.

[6] A. Joly, H. Goëau, H. Glotin, C. Spampinato, P. Bonnet, W.-P. Vellinga, J.-C. Lombardo, R. Planque, S. Palazzo, and H. Müller, "LifeCLEF 2017 lab overview: multimedia species identification challenges," in *International Conference of the Cross-Language Evaluation Forum for European Languages*. Springer, 2017, pp. 255–274.

[7] S. Kahl, T. Wilhelm-Stein, H. Hussein, H. Klinck, D. Kowerko, M. Ritter, and M. Eibl, "Large-scale bird sound classification using convolutional neural networks," *Working notes of CLEF*, 2017.

[8] K. J. Piczak, "Recognizing bird species in audio recordings using deep convolutional neural networks," *Working notes of CLEF*, 2016.

[9] E. Sprengel, M. Jaggi, Y. Kilcher, and T. Hofmann, "Audio based bird species identification using deep learning techniques," *Working notes of CLEF*, 2016.

[10] D. Stowell and M. D. Plumbley, "Audio-only bird classification using unsupervised feature learning," *Working Notes of CLEF*, 2014.

[11] M. Lasseck, "Improved automatic bird identification through decision tree based feature selection and bagging." *Working Notes of CLEF*, 2015.

[12] E. Çakir, S. Adavanne, G. Parascandolo, K. Drossos, and T. Virtanen, "Convolutional recurrent neural networks for bird audio detection," in *Proceedings of the 25th European Signal Processing Conference (EUSIPCO)*. IEEE, 2017, pp. 1744–1748.

[13] P. Jančovic, M. Köküer, M. Zakeri, and M. Russell, "Bird species recognition using HMM-based unsupervised modelling of individual syllables with incorporated duration modelling," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 559–563.

[14] A. Joly, V. Leveau, J. Champ, and O. Buisson, "Shared nearest neighbors match kernel for bird songs identification - LifeCLEF 2015 challenge," *Working Notes of CLEF*, 2015.

[15] B. Fazekas, A. Schindler, T. Lidy, and A. Rauber, "A multi-modal deep neural network approach to bird-song identification," *Working Notes of CLEF*, 2017.

[16] K. J. Piczak, "Environmental sound classification with convolutional neural networks," *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, 2015.

[17] A. Fritzler, S. Koitka, and C. M. Friedrich, "Recognizing bird species in audio files using transfer learning," *Working Notes of CLEF*, vol. 2017, 2017.

[18] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning." in *OSDI*, vol. 16, 2016, pp. 265–283.

[19] N. Silberman, "TF-Slim: A lightweight library for defining, training and evaluating complex models in TensorFlow," 2017. [Online]. Available: https://github.com/tensorflow/models/tree/master/research/slim

[20] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift." in *ICML*, ser. JMLR Workshop and Conference Proceedings, F. R. Bach and D. M. Blei, Eds., vol. 37. JMLR.org, 2015, pp. 448–456. [Online]. Available: http://dblp.uni-trier.de/db/conf/icml/icml2015.html#IoffeS15