**CHAPTER 1**

**INTRODUCTION**

**1.1 Overview**

In recent years, algorithmic sound recognition has enjoyed a steady increase in interest. 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. The target audience are 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 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, spectrograms are used, which are a visual representation of the magnitude returned by the Short Time Fourier Transform (STFT). 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.

In this case, the compact and performance oriented MobileNet is used as a starting checkpoint. The 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 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.

Large scale, accurate bird recognition is essential for avian biodiversity conservation.

It helps us quantify the impact of land use and land management on bird species and is fundamental for bird watchers, conservation organizations, park rangers, ecology consultants, and ornithologists all over the world. Many books have been published to help humans determine the correct species and dedicated online forums exist where recordings can be shared and discussed.

Nevertheless, because recordings, spanning hundreds of hours, need to be carefully analysed and categorised, large scale bird identification remains almost an impossible task to be done manually. It, therefore, seems natural to look at ways to automate the process. Unfortunately a number of challenges have made this task extremely difficult to tackle.

**1.2 Approach**

We use a convolutional neural network with five convolutional and one dense layer. Every convolutional layer uses a rectify activation function and is followed by a max-pooling layer. For preprocessing, we split the sound file into a signal part where bird songs/calls are audible and a noise part where no bird is singing/calling (background noise is still present in these parts). We compute the spectrograms (Short Time Fourier Transform) of both parts and split each spectrogram into equally sized chunks. Each chunk can be seen as the spectrogram of a short time interval (typically around 3 seconds). As such, we can use each chunk from the signal part as a unique training/testing sample for our neural network.

**CHAPTER 2**

**LITERATURE SURVEY**

**Title:** Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification

**Authors:** J. Salamon and J. P. Bello

**Published in:** IEEE Signal Processing Letters, vol. 24,

**Year: 2017**

* The ability of deep convolutional neural networks (CNNs) to learn discriminative spectro-temporal patterns makes them well suited to environmental sound classification. However, the relative scarcity of labeled data has impeded the exploitation of this family of high-capacity models.
* This study has two primary contributions: first, we propose a deep CNN architecture for environmental sound classification. Second, we propose the use of audio data augmentation for overcoming the problem of data scarcity and explore the influence of different augmentations on the performance of the proposed CNN architecture. Combined with data augmentation, the proposed model produces state-of-the-art results for environmental sound classification.
* We show that the improved performance stems from the combination of a deep, high-capacity model and an augmented training set: this combination outperforms both the proposed CNN without augmentation and a “shallow” dictionary learning model with augmentation.
* Finally, we examine the influence of each augmentation on the model's classification accuracy for each class, and observe that the accuracy for each class is influenced differently by each augmentation, suggesting that the performance of the model could be improved further by applying class-conditional data augmentation.

**Title:** Leveraging deep neural networks with nonnegative representations for improved environmental sound classification

**Authors:** V. Bisot, R. Serizel, S. Essid, and G. Richard

**Published in:** IEEE International Workshop on Machine Learning for Signal Processing MLS

**Year: 2017**

* 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\nobreakdash-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\nobreakdash-trained with provides a measurable advantage. The presented system is viable only for a low number of classes.

**Title:** Transfer learning and the art of using pre-trained models in deep learning

**Authors:** D. Gupta.

**Published in:** https://www.analyticsvidhya.com/blog/2017/06

**Year: 2017**

* Neural networks are a different breed of models compared to the supervised machine learning algorithms. Why do I say so? There are multiple reasons for that, but the most prominent is the cost of running algorithms on the hardware.
* In today’s world, RAM on a machine is cheap and is available in plenty. You need hundreds of GBs of RAM to run a super complex supervised machine learning problem – it can be yours for a little investment / rent. On the other hand, access to GPUs is not that cheap. You need access to hundred GB VRAM on GPUs – it won’t be straight forward and would involve significant costs.
* Now, that may change in future. But for now, it means that we have to be smarter about the way we use our resources in solving Deep Learning problems. Especially so, when we try to solve complex real life problems on areas like image and voice recognition. Once you have a few hidden layers in your model, adding another layer of hidden layer would need immense resources.

**Title:** MobileNets: Efficient convolutional neural networks for mobile vision applications,

**Authors:** A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand,

M. Andreetto, and H. Adam,

**Published in:** CoRR, vol. abs/1704.04861

**Year: 2017**

* We present a class of efficient models called MobileNets for mobile and embedded vision applications. MobileNets are based on a streamlined architecture that uses depth-wise separable convolutions to build light weight deep neural networks.
* We introduce two simple global hyper-parameters that efficiently trade-off between latency and accuracy. These hyper-parameters allow the model builder to choose the right sized model for their application based on the constraints of the problem.
* We present extensive experiments on resource and accuracy trade-offs and show strong performance compared to other popular models on ImageNet classification.
* We then demonstrate the effectiveness of MobileNets across a wide range of applications and use cases including object detection, fine grain classification, face attributes and large scale geo-localization.

**Title:** Life CLEF 2017 lab overview: multimedia species identification challenges,

**Authors:** A. Joly, H. Goeau, H. Glotin, C. Spampinato, P. Bonnet, W.-P. Vellinga,

J.-C. Lombardo, R. Planque, S. Palazzo, and H. Muller,

**Published in:** International Conference of the Cross-Language Evaluation Forum for

European Languages.

**Year: 2017**

* Identifying and naming living plants or animals are usually impossible for the general public and often a difficult task for professionals and naturalists. Bridging this gap is a key challenge towards enabling effective biodiversity information retrieval systems.
* This taxonomic gap was actually already identified as one of the main ecological challenges to be solved during the Rio de Janeiro United Nations ”Earth Summit” in 1992. Since 2011, the LifeCLEF challenges conducted in the context of the CLEF evaluation forum have been boosting and evaluating the advances in this domain.
* Data collections with an unprecedented volume and diversity have been shared with the scientific community to allow repeatable and long term experiments

**Title:** Large-scale bird sound classification using convolutional neural networks

**Authors:** S. Kahl, T. Wilhelm-Stein, H. Hussein, H. Klinck, D. Kowerko, M. Ritter, and M. Eibl,

**Published in:** Working notes of CLEF

**Year: 2017**

* Identifying bird species in audio recordings is a challenging field of research. In this paper, we summarize a method for large-scale bird sound classification in the context of the LifeCLEF 2017 bird identification task.
* We used a variety of convolutional neural networks to generate features extracted from visual representations of field recordings.
* The BirdCLEF 2017 training dataset consist of 36.496 audio recordings containing 1500 different bird species. Our approach achieved a mean average precision of 0,605 (official score) and 0,687 considering only foreground species.

**Title:** Recognizing bird species in audio recordings using deep convolutional neural networks

**Authors:** K. J. Piczak,

**Published in:** Working notes of CLEF,

**Year: 2016**

* This paper summarizes a method for purely audio-based bird species recognition through the application of convolutional neural networks. The approach is evaluated in the context of the LifeCLEF 2016 bird identification task - an open challenge conducted on a dataset containing 34 128 audio recordings representing 999 bird species from South America.
* Three different network architectures and a simple ensemble model are considered for this task, with the ensemble submission achieving a mean average precision of 41.2% (official score) and 52.9% for foreground species.

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 EXISTING SYSTEM**

* [Bird Sound Recognition Using a Convolutional Neural Network](https://ieeexplore.ieee.org/document/8524677)
* [BirdNET – The easiest way to identify birds by sound.](https://birdnet.cornell.edu/)
* A real-time bird sound recognition system using a low-cost microcontroller
* Bird Call Identification using Dynamic Kernel based Support Vector Machines and Deep Neural Networks

# Recognition of bird species based on spike model using bird dataset

* Deep Learning Based Audio Classifier for Bird Species
* [Birds Voice Classification using Deep Residual Network](https://www.ijedr.org/papers/IJEDR1804032.pdf)

**3.2 PROPOSED SYSTEM**

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.

**CHAPTER 4**

**SYSTEM REQUIREMENT SPECIFICATIONS**

Requirements analysis is critical for project development. Requirements must be documented, actionable, measurable, testable and defined to a level of detail sufficient for system design. Requirements can be architectural, structural, behavioural, functional, and functional.

A software requirements specification (SRS) is a comprehensive description of the intended purpose and the environment for software under development.

**4.1 Software Requirements**

Scripting language : Python Programming

Scripting Tool : Anaconda Navigator (Jupyter Notebook) & Google Colab

Operating System : Microsoft Windows 7, 8 or 10

Dataset : UCI Machine Learning Repository

Machine Learning Packages : Numpy, Pandas, Matplotlib, Seaborn Packages etc...

**4.2 Hardware Requirements**

Processor : 3.0 GHz and Above

Output Devices : Monitor (LCD)

Input Devices : Keyboard

Hard Disk : 1 TB

RAM : 8GB or Above

**CHAPTER 5**

**SYSTEM DESIGN**

**5.1 SYSTEM ARCHITECTURE**

The generation of good input features is vital to the success of the neural network. There are three main stages. First, we decide which parts of the sound file correspond to a bird singing/calling (signal parts) and which parts contain noise or silence (noise parts). Second, we compute the spectrogram for both signal and noise part. Third, we divide the spectrogram of each part into equally sized chunks. We can then use each chunk from the signal spectrogram as a unique sample for training/testing and augment it with a chunk from the noise spectrogram.

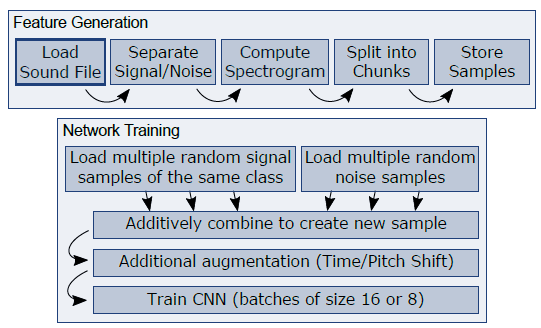


Fig:5.1 Overview of the pipeline for training the neural network. CNN stands for convolutional neural network. During training, we use a batch size of 16 training examples per iteration. However, due to memory limitations of the GPU, we sometimes have to fall back to batches of size 8.

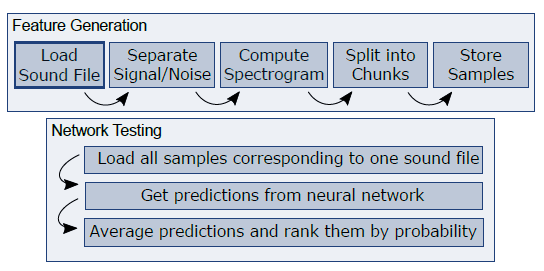


Fig 5.1.2.: Overview of the testing pipeline. Note that we get multiple predictions per sound file (one prediction per chunk/sample) which we can average to obtain a single prediction per file.

**1 Signal/Noise Separation**

To divide the sound file into a signal and a noise part, we first compute the spectrogram of the whole file. Note that all spectrograms in this project are computed in the same way. First the signal is passed through a short-time Fourier transform (STFT); this is done using a Hanning window function (size 512, 75% overlap). Then the logarithm of the amplitude of the STFT is taken. However, the signal/noise separation is the exception to this rule because here, we do not take the logarithm of the amplitude but instead divide every element by the maximum value, such that all values end up in the interval [0; 1]. With the spectrogram at hand, we are now able to look for the signal/noise intervals.

**2 Dividing the Spectrograms into Chunks**

Compute a spectrogram for both the signal and noise part of the sound file. Afterwards we split both spectrograms into chunks of equal size (we use a length of 512). The splitting is done for three reasons. For one, we need a fixed sized input for our neural network architecture.

We could pad the input but the large variance in the length of the recordings would mean that some samples would contain over 99% padding.

**3 Data Augmentation**

Because the number of sound \_les is quite small, compared to the number of classes (the training set (of 24'607 \_les) contains an average of only 25 sound \_les per class), we need additional methods to avoid over fitting. Apart from drop-out, data augmentation was one of the most important ingredients to improve the generalization performance of the system. We apply four different data augmentation methods.

**4 Network architecture**

The network contains 5 convolutional layers, each followed by a max-pooling layer. We insert one dense layer before the final soft-max layer. The dense layer contains 1024 and the soft-max layer 1000 units, generating a probability for each class. We use batch normalization before every convolutional and before the dense layer. The convolutional layers use a rectify activation function. Drop-out is used on the input layer (probability 0.2), on the dense layer (probability 0.4) and on the soft-max layer (probability 0.4). As a cost function we use the single label categorical cross entropy function (in the log domain).

**5.2 DATAFLOW DIAGRAM**

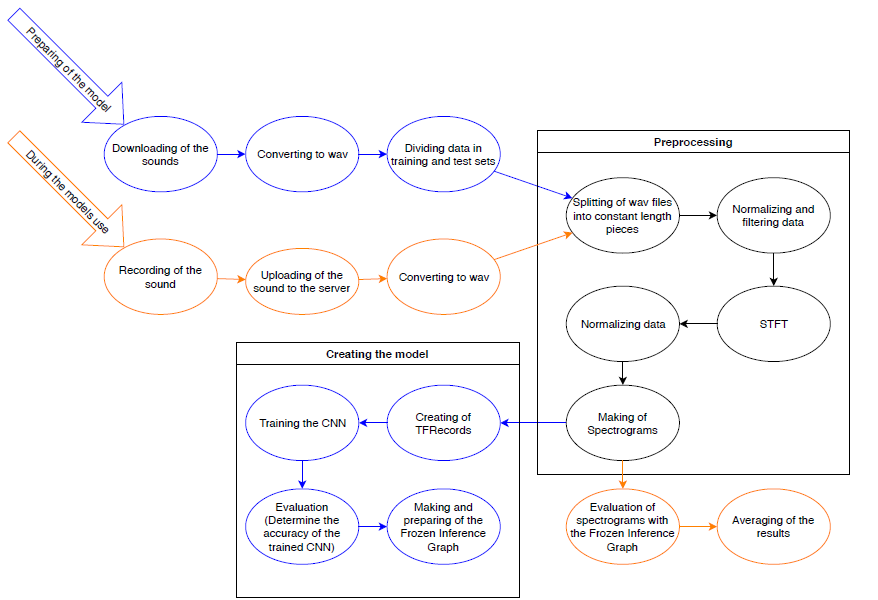
****

Fig.5.3 Model workflow in the stages of preparation (blue) and evaluation (orange). Certain steps, such as the preprocessing, overlap (black) during both scenarios. the general concepts and methodologies involved in the forming of the current bird sound recognition system are presented. The two steps involved in the system setup 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. 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 normalization. Finally, the spectrograms are created by converting the STFT output into an image using a color map.

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 is returned to the user.

**B. Automated data download**

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

* Providing a file containing the name(s) of the species; this infers the number of classes.
* The file can be generated by providing the desired number of European bird species for which to download recordings.
* It is not necessary 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 Xenocanto site.
* Configurable options include the location where the recordings, the generated spectrograms and the TFRecord files are all stored.

**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, a window size of 448 is used during the STFT. Furthermore, a Hamming window with an overlap of 224 (half of the window size, also configurable) is deployed.

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

The experiments are also assisted by the TF-Slim TensorFlow library, 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 allow working with popular convolutional neural networks, such as Inception, ResNet, VGG, and MobileNets. TF-Slim provide several public and compatible pre-trained networks (checkpoints) which greatly reduce training times.

**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. The 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 the dataset is shuffled along with the corresponding tags before writing them into TFRecord files.

**5.3 Sequence Diagram**

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

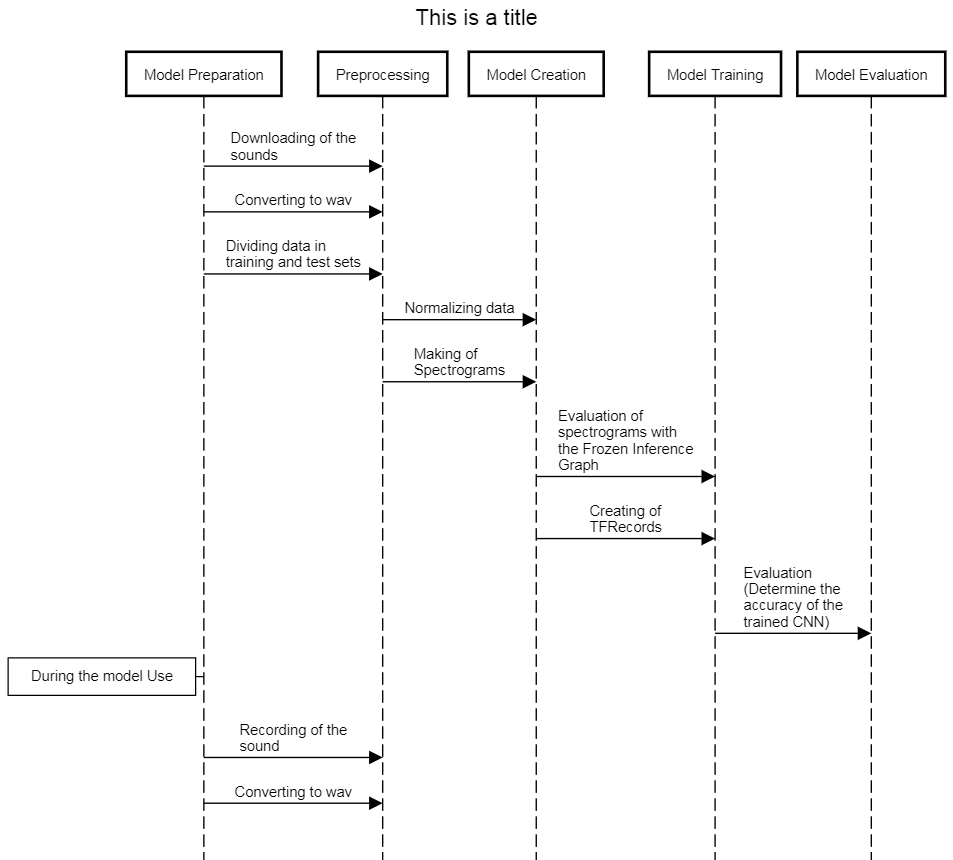
****

Fig 5.4

**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.
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. Go¨eau, H. Glotin, C. Spampinato, P. Bonnet, W.-P. Vellinga, J.-C. Lombardo, R. Planque, S. Palazzo, and H. M¨uller, “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 uses 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. C¸ akir, 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. Janˇcovic, M. K¨ok¨uer, 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. 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.
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 forlarge-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.
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.
21. API Flaticon for social media icons.