Bioacoustic Classification Framework: Spectral and Cepstral Based Approaches.

Pedro Esteves Martins Bonito Baptista

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor: Prof. Cláudia Martins Antunes

Examination Committee

Chairperson: Prof. Paolo Romano
Supervisor: Prof. Cláudia Martins Antunes
Member of the Committee: Prof. Rui Miguel Carrasqueiro Henriques

November 2021

Acknowledgments

This work was supported by national funds by Fundação para a Ciência e Tecnologia (FCT) through project VizBig (PTDC/CCI-CIF/28939/2017).

I would like to thank my parents for their effort, sacrifice and encouragement over all these years, for always being there for me through thick and thin and without whom this project would not be possible. I would also like to thank my family members and remember those who are no longer with us. Their example and influence played a major part in all the hard work invested in this document.

I would also like to acknowledge my dissertation supervisor Prof. Cláudia Antunes for her insight, support and availability that has made this thesis possible.

Moreover, I would also like to thank Hitachi Vantara, and also Marco Vala, for the summer internship opportunity which was crucial to this work's research.

Last but not least, I would also like to thank all my friends and colleagues.

To each and every one of you – Thank you.

Abstract

The field of bioacoustics plays an important role on preventing and reducing human impact on environment, by enabling the development of tools capable of performing automated analysis of environmental data. Deep learning methods were successful on automating the process of species identification in environmental recordings, requiring nonetheless a large number of training samples per species. Hence, efforts were made to develop high-accuracy methods capable of automating species detection in noisy environments with limited training data. In this document, we address the problem of automating species detection in noisy environments with limited training data, proposing an end-to-end spectral based approach for training a convolutional neural network (CNN) on Mel spectrograms to predict a set of species present in the Rainforest Connection's acoustic recordings. Additionally, we propose a cepstral based framework for training a Long Short-Term Memory (LSTM) network on the Mel-frequency cepstral coefficients (MFCCs), complementing this approach with the motifs extracted by the matrix profile algorithm. Finally, we evaluate the performance of the approaches so that the bioacoustic classification framework can be established.

Keywords

Bioacoustic classification; Deep learning; Convolutional Neural Networks (CNN); Data augmentation; Transfer learning; Long short-term memory (LSTM); Matrix Profile;

Resumo

O campo da bioacústica desempenha um papel crucial na prevenção e redução do impacto humano no ambiente, ao permitir o desenvolvimento de ferramentas capazes de automatizar a análise de dados ambientais. Os métodos de *Deep Learning* foram bem sucedidos na automatização do processo de identificação de espécies em áudios ambientais, necessitando, no entanto, de um número elevado de instâncias de treino por espécie. Por conseguinte, o foco virou-se para o desenvolvimento de métodos capazes de automatizar a detecção de espécies em ambientes ruidosos, dispondo de um conjunto limitado de dados para treino. Neste documento, abordamos o problema de automatizar o processo de detecção de espécies em áudios ambientais, com um conjunto de treino limitado, propondo uma abordagem para o treino de uma *Convolutional Neural Network (CNN)* a partir das propriedades espectrais do som, nomeadamente dos Mel espectrogramas, para identificar as diferentes espécies presentes nos áudios da Rainforest Connection. Apresentamos ainda, uma abordagem alternativa, baseada nas características cepstrais do som, em particular dos *Mel-frequency cepstral coefficients (MFCCs)*, para o treino de uma *Long Short-Term Memory (LSTM) network*, sendo esta complementada pela inclusão de *motifs* extraídos pelo algoritmo *matrix profile*. Finalmente, avaliamos os resultados de ambas as abordagens de forma a definir uma metodologia de classificação de sinais bioacústicos.

Palavras Chave

Classificação Bioacústica; Aprendizagem Profunda; Redes Convolucionais; Data augmentation; Transfer learning; Long short-term memory (LSTM); Matrix Profile;

Contents

List of Figures

List of Tables

1

Introduction

Bioacoustics focuses on the analysis of the sounds produced by or affecting living organisms, especially the ones related to communication. Prior bioacoustic research was heavily dependent on manual labor to segment, detect and label animal activity, present in hours of field recordings. Consequently, recent research overlaps the work developed by Rainforest Connection (NGO) [1] which focuses on developing bioacoustic monitoring systems to ensure the rainforest's conservation, being also a prominent source of environmental audio data.

Deep learning methods have been successful on automatic acoustic identification, through image analysis dedicated architectures, such as convolutional networks. However, they require a large number of training samples per species. This limits applicability to rarer species, which are central to conservation efforts. Thus, the Kaggle competition "Rainforest Connection Species Audio Detection" [2] encouraged contenders to develop solutions capable of automate high-accuracy species detection in noisy soundscapes with limited training data.

In this document, we address the problem of automating species detection in noisy environments with limited training data, thus, we explore two main approaches to build a bioacoustic classification framework. The first, the spectral based one, proposes a framework for training a convolutional neural network (CNN) on Mel spectrograms to predict a set of species present in the Rainforest Connection's acoustic recordings. We leverage transfer learning by using a pretrained model as a way to reduce training requirements, both the amounts of data and time. Finally, we explore several window sizes, data augmentation techniques and predictive thresholds to improve the model's performance. The second, the cepstral based one, proposed an end-to-end pipeline for training a Long Short-Term Memory (LSTM) network on the Mel-frequency cepstral coefficients (MFCCs). Furthermore, we complement this approach with the motifs extracted by the matrix profile algorithm, as a way of improving the performance of the concerned network. Lastly, we explore the standard and the multidimensional implementation of the matrix profile algorithm, experimenting also different window sizes and predictive thresholds.

The best performing approach is the spectral based classification model, both on the chainsaw and on the Kaggle dataset. It includes 5-second-long Mel spectrograms and relies on the SpecAugment method to increase the training set size. Regarding the chainsaw dataset, it achieves an accuracy of 0.97, a mean precision of 0.99 and a mean recall of 0.97. In relation to the Kaggle dataset, it registers an accuracy of 0.97, a mean precision of 0.91 and a mean recall of 0.93.

This paper is organized into six sections. Section 2 encompasses the concepts and procedures associated with audio processing and analysis. Section 3 provides an overview of the current work in motif discovery, namely the Eammon Keogh's matrix profile. It also regards the work related with sound event detection and classification, focusing on methods based on deep learning architectures, such as Convolutional Neural Networks. Section 4 outlines the proposed methodology and section 5 details the

[1] https://rfcx.org/

[2] https://www.kaggle.com/c/rfcx-species-audio-detection/data

used datasets and validates the proposed methodology on them. Finally, section 6 summarizes the main ideas and addresses the future work.

2

Context

Contents

The growth of computational power led to a demand for solutions capable of handling the continuous ever-growing flux of data, highlighting the capability of extracting valuable insights from these data streams, such as detecting repeated patterns or anomalous events. In this sense, the main concerns of this knowledge discovery process are the constrained time and space, as well as the high rate and volume in which data arrives.

Hence, we propose a bioacoustic classification framework that aims to contribute to the process of handling, extracting and classifying meaningful events in audio data streams. This work also focus on exploring the recent progress in time series motif discovery, namely the matrix profile algorithm, firstly introduced in Yeh et al. (2016) by Eamonn Keogh, as a way of complementing the introduced methodology. It is also important to recognize audio data streams as time series, as they are unbounded and ordered sequences of instances (sounds) which arrive over time. Hao et al. (2013), Branco (2020).

2.1 Basic Concepts and Notation

The definitions adopted in this document are similar to the ones presented by Eamonn Keogh in Yeh et al. (2016), Yeh et al. (2017b), as this work explores the techniques introduced in his work. Similarly, it is important to begin by defining the data type of interest, time series:

Definition 1. ***Time series:*** *A time series $T \in \mathbb{R}^n$ is a sequence of real-valued numbers $t_i \in \mathbb{R} : T = [t_1, t_2 ..., t_n]$ where n is the length of T.*

For motif discovery, one is not interested in the global properties of a time series, but in the local subsequences:

Definition 2. ***Subsequence:*** *A subsequence $T_{i,m} \in \mathbb{R}^m$ of a time series T is a continuous subset of the values from T of length m starting from position i. Formally, $T_{i,m} = [t_i, t_{i+1} ..., t_{i+m-1}]$.*

The particular local properties that this work focus on is the time series motifs:

Definition 3. ***Time series motif:*** *A time series motif is the most similar subsequence pair of a time series. Formally, $T_{a,m}$ and $T_{b,m}$ is the motif pair if $dist(T_{a,m}, T_{b,m}) \leq dis(T_{i,m}, T_{j,m})$ $\forall$ $i, j \in$ [1,2,. . . ,n-m+1] where $a \neq b$ and $i \neq j$ and dist is a function that computes the z-normalized Euclidean distance between the input subsequences.*

The distance between a subsequence of a time series, with all the other subsequences from the same time series, is stored in an ordered array called distance profile.

Definition 4. ***Distance Profile:*** *A distance profile $D \in \mathbb{R}^{n-m+1}$ of a time series T and a subsequence $T_{i,m}$ is a vector that stores $dist(T_{i,m}, T_{j,m})$ $\forall$ $j \in$ [1,2,. . . ,n-m+1].*

We are interested in the similarity join of all subsequences of a given time series. We define an all-subsequences set of a given time series as a set that contains all possible subsequences from the time series.

Definition 5. ***All-subsequences set:*** *An all-subsequences set A of a time series* T *is an ordered set of all possible subsequences of T obtained by sliding a window of length m across* $T : A = \{T_{1,m}, T_{2,m}, \ldots, T_{n-m+1,m}\}$, *where m is a user-defined subsequence length. We use* $A[i]$ *to denote* $T_{i,m}$.

Similarly, we are interested in the nearest neighbour relation between subsequences.

Definition 6. ***1NN-join:*** *Given two all-subsequences sets A and B and two subsequences A[i] and B[j], a 1NN-join function* $\theta_{1nn}\ (A[i], B[j])$ *is a Boolean function which returns "true" only if B[j] is the nearest neighbor of A[i] in the set B.*

A similarity join set is then the result of the application of the similarity join operator on two input all-subsequences sets.

Definition 7. ***Similarity join set:*** *Similarity join set: given all-subsequences sets A and B, a similarity join set* J_{AB} *of A and B is a set containing pairs of each subsequence in A with its nearest neighbor in* $B : J_{AB} = \{\langle\ A[i], B[j]\ \rangle \mid \theta_{1nn}\ (A[i], B[j])\}$. *We denote this formally as* $J_{AB} = A \bowtie \theta_{1nn} B$.

As in Yeh et al. (2016), the previous four definitions are represented in Fig. 2.1:

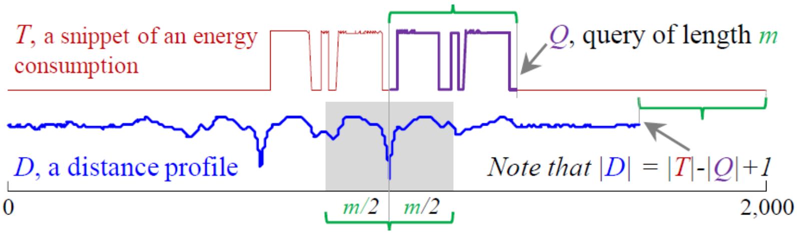


Figure 2.1: A subsequence Q extracted from a time series T is used as a query to every subsequence in T. The vector of all distances is a distance profile.

Finally, the Euclidean distance metric chosen by the author in Yeh et al. (2016), among others, is measured between each pair within a similarity join set and the resultants are stored into an ordered vector, resulting in the matrix profile.

Definition 8. ***Matrix Profile:*** *A matrix profile* $P \in \mathbb{R}^{n-m+1}$ *of a time series T is a meta time series that stores the z-normalized Euclidean distance between each subsequence and its nearest neighbor where n is the length of T and m is the given subsequence length. The time series motif can be found by simply locating the two lowest values in P (they will have tying values).*

Additionally, one can extend the matrix profile Yeh et al. (2016) to find motifs in multidimensional time series, as in Yeh et al. (2017b).

Definition 9. ***Multidimensional time series:*** *A multidimensional time series $T \in \mathbb{R}^{d \times n}$ is a set of co-evolving time series*
$T^{(i)} \in \mathbb{R}^n : T = [T^{(1)}, T^{(2)}, \ldots, T^{(d)}]^T$ *where d is the dimensionality of T and n is the length of T.*

Similarly, the definition of a subsequence in a multidimensional setting becomes the following:

Definition 10. ***Multidimensional subsequence:*** *A multidimensional subsequence $T_{i,m} \in \mathbb{R}^{d \times m}$ of a multidimensional time series T is a set of univariant subsequences from T of length m starting from position i. Formally, $T_{i,m} = [T_{i,m}^{(1)}, T_{i,m}^{(2)}, ..., T_{i,m}^{(d)}]^T$.*

Motif discovery considering all dimensions is generally guaranteed to fail as demonstrated in Yeh et al. (2017b). Generally, only a subset of all dimensions should be considered for multidimensional motif discovery, often referred as subdimensional subsequences:

Definition 11. ***Subdimensional subsequence:*** *A subdimensional subsequence $T_{i,m}(X) \in \mathbb{R}^{k \times m}$ is a multidimensional subsequence for which only a subset of dimensions is selected, where X is an indicator vector that shows which dimension is included, and k is the number of dimension included (i.e., $\|X\|_0 = k$).*

We are only interested in computing the distance between two multidimensional subsequences, using only their corresponding subdimensional subsequences. To measure this relation, one can use the distance function:

Definition 12. ***K-dimensional distance function:*** *The k-dimensional distance function $dist^{(k)}$ computes the distance between two multidimensional subsequences by using only the "best" k out of d dimensions.*
Formally, $dist^{(k)}(T_{i,m}, T_{j,m}) := \min_X dist\Big(T_{i,m}(X), T_{j,m}(X)\Big)$, where $\|X\|_0 = k$.

The definition of a distance profile is therefore updated to the multidimensional setting and renamed to:

Definition 13. ***k-dimensional distance profile:*** *A k-dimensional distance profile $D \in \mathbb{R}^{n-m+1}$ of a time series T and a subsequence $T_{i,m}$ is a vector that stores $dist^{(k)}(T_{i,m}, T_{j,m}) \; \forall \; j \in [1, 2, \ldots, n-m+1]$.*

Similarly, the motif definition must be readjusted:

Definition 14. ***K-dimensional motif:*** *A k-dimensional motif is the most similar subdimensional subsequence pair of a multidimensional time series when the distance is computed by using the k-dimensional distance function.*

Formally, $T_{a,m}$ and $T_{b,m}$ *is the k-dimensional motif pair if* $dist^{(k)}(T_{a,m}, T_{b,m}) \leq dist^{(k)}(T_{i,m}, T_{j,m})$ $\forall\ i, j \in [1, 2, \ldots, n-m+1]$ *where* $a \neq b$ and $i \neq j.$

As well as the definition of the matrix profile:

Definition 15. ***K-dimensional matrix profile:*** *A k-dimensional matrix profile* $P \in \mathbb{R}^{n-m+1}$ *of a multidimensional time series T is a meta time series that stores the z-normalized Euclidean distance between each subsequence and its nearest neighbor (the distance is computed using k-dimensional distance function), where n is the length of T, d is the dimensionality of T, k is the given number of dimension, and m is the given subsequence length. Formally, the i th position in P stores* $dist^{(k)}(T_{i,m}, T_{j,m})\ \forall\ j \in [1, 2, \ldots, n-m+1]$ *where* and $i \neq j.$

A k-dimensional matrix profile only reveals the location of motifs in time, but it fails to reveal which k out of the d dimension contains the motif pair. To store this information, we define another meta time series called the k-dimensional matrix profile subspace:

Definition 16. ***K-dimensional matrix profile subspace:*** *A k-dimensional matrix profile subspace* $S \in \mathbb{R}^{k \times n-m+1}$ *is a multidimensional meta time series that stores the selected k dimension for each subsequence when computing the distance with others.*

2.2 Rainforest Connection

Rainforest Connection [1] is a non-profit tech startup that builds acoustic monitoring systems to protect rainforests from illegal deforestation, to halt animal poaching and to enable bioacoustic monitoring. It transforms upcycled cell-phones into autonomous, solar-powered listening devices which record all sounds in the forest, enabling the development of solutions that can remotely monitor and detect anomalous activity in a given area.

Once the audio is in the cloud, Google's machine learning framework, TensorFlow, is used to analyse all the auditory data in real-time and listen for chainsaws, logging trucks and other sounds of illegal activity that can help to pinpoint problems in the forest.

More recently, Hitachi Vantara and Rainforest Connection have been working together to develop advanced acoustic algorithms that will help mitigating this environmental problem. So, through this connection, Hitachi made the RFCx audio files available for this work, data that represents the main focus of the carried out exploration and analysis.

[1] https://www.rfcx.org/

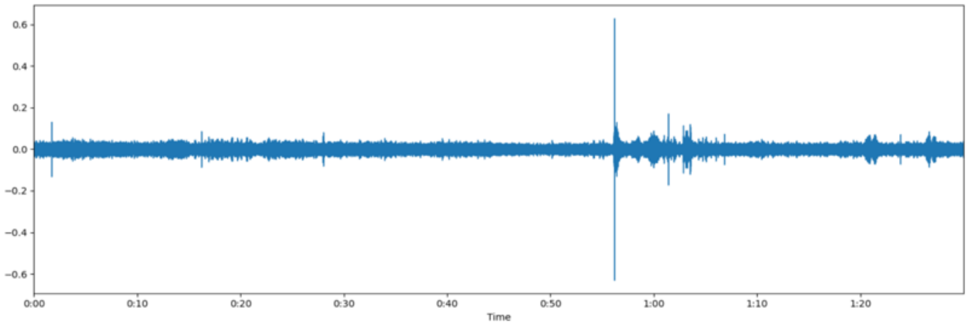


Figure 2.2: Amplitude envelope of a waveform - corresponds to the RFCx's digital sound of the day 18/01/2020 at 02:29:30 with a sampling rate of 22050 Hz.

2.3 Sound Signal Representations

As introduced in Serizel et al. (2018), a *sound signal* is the result of a vibration that propagates as waves through a medium such as air or water. Sounds can be recorded under the form of an electric signal x(t) by means of an electroacoustic transducer such as a microphone. This analog signal x(t) can then be converted to a digital signal x[n] and stored in a computer before further analysis. Therefore, and according to the definition 1, sound signals are time series.

The typical process used to convert an analog signal to a digital one is also detailed in Serizel et al. (2018), a procedure that consists of three major steps, described here, having as reference the previous cited work.

Firstly, there is a *filtering stage* where the analog signal x(t) is low-pass filtered, a process which aims to limit the frequency bandwidth to be contained in the interval [0, B], where B is the cutoff frequency of the low-pass filter.

Secondly, in the *sampling stage*, the signal is digitally sampled at a sampling rate $f_s = 2B$ to avoid the frequency aliasing phenomenon.

Finally, the obtained digital signal is *quantized*, a process in which, for instances, the amplitude of the signal can only take a limited number of predefined values, so that the storage capacity can be preserved. Typically, one uses a sampling frequency of 44100 Hz and a quantization on 16 bits per sample.

Digital sound can then be stored under different formats, such as the uncompressed ones (.wav) which are based on PCM (Pulse Code Modulation), stored using lossless compression (.flac) or using lossy compression (.mp3). Lossy compression may compromise knowledge discovery, and so it should be avoided for such tasks Nordby (2019).

In time domain representations, the identification of events in sound signals usually is a burdensome task, unless there are indistinct events that make the interpretation clearer. Nonetheless, such conditions are rare and as described in Serizel et al. (2018), sound signals are usually converted to the frequency-domain, to facilitate the knowledge discovery process.

In detail, the frequency-domain representation of a signal $x[n]$ on a linear frequency scale can be obtained with the *discrete-time Fourier transform (DFT)* Serizel et al. (2018), Rocchesso (1995).

$$X(f) = \sum_{n=0}^{N-1} x[n]\, e^{\frac{-i\, 2\pi\, f\, n}{N}} \tag{2.1}$$

The spectrum $X(f)$ is f_s-periodic in f, with f_s as the sampling frequency. The frequency $f = \frac{fs}{2}$ represents the Nyquist-frequency Nyquist (1928), Weik (2001).

Moreover, one can fasten the DFT's computation with the *Fast Fourier Transform (FFT)*, reducing the computation complexity from $O(N^2)$ to $O(NlogN)$, being, consequently, a common method of most of the libraries used to process sound signals. It is also important to remark that the FFT provides the frequency distribution of the signal $x[n]$ but disregards the time component.

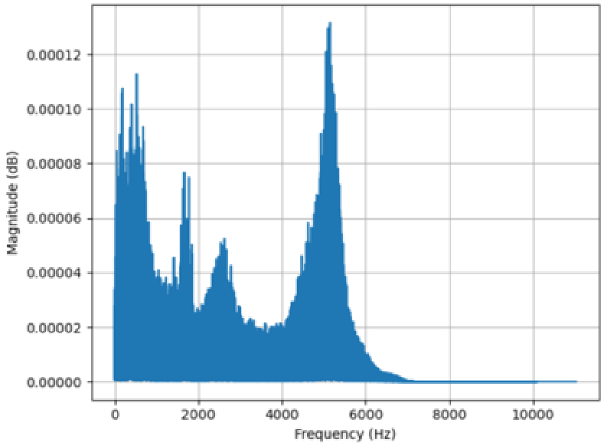


Figure 2.3: Frequency-domain representation (FFT) - corresponds to the RFCx's digital sound of the day 18/01/2020 at 02:29:30 with a sampling rate of 22050 Hz.

Conversely, one can apply to the signal $x[n]$, the DFT on a windowed frame of length N, procedure that is referred to as the *short-time Fourier transform (STFT)* Serizel et al. (2018), Rocchesso (1995). The f^{th} component of the DFT, of the t^{th} frame of $x[n]$ is computed as follows:

$$X(t, f) = \sum_{k=0}^{N-1} w[k]\, x[t\, N + k]\, e^{\frac{-i\, 2\pi\, k\, f}{N}} \tag{2.2}$$

As explained in Serizel et al. (2018), this technique introduces a window function $w[k]$ (e.g., rectangular, Hamming, Blackman,...) which is used to enforce continuity and periodicity at the edge of the frames. The equation 2.2 presents a hop, between frames, equal to the length of the frames (N), which means that there is no overlap between consecutive frames. Nevertheless, one can choose a smaller hop size, in comparison to the frame length, resulting in overlapping frames, that allow a smoother STFT representation and introduce statistical dependencies between frames.

A *sound spectrogram* Hao et al. (2013) is an image of the time-varying spectral representation, produced by applying the STFT to successive overlapping frames of an audio sequence. The horizontal dimension corresponds to time and the vertical dimension corresponds to frequency. The relative spectral intensity of a sound at any specific time and frequency is indicated by the color/grayscale intensity of the image, as illustrated in Fig. 2.4. One can reduce the dimensionality of a raw STFT representation, as it can contain more than 1024 bins, often strong correlated, by processing the spectrogram with a filter-bank of 40-128 frequency bands.

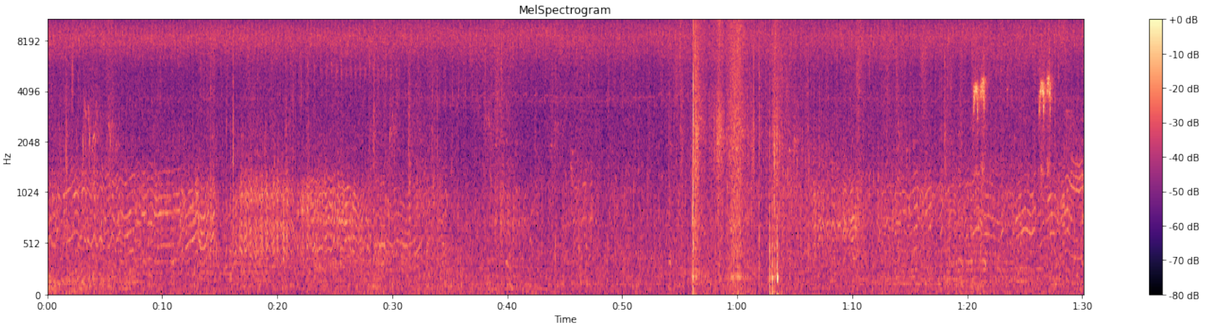


Figure 2.4: Mel spectrogram representation (STFT) - corresponds to the RFCx's digital sound of the day 18/01/2020 at 02:29:30 with a sampling rate of 22050 Hz.

2.4 Feature Extraction

The typical pipeline in sound processing is to transform the raw data into features that characterize audio signals, via feature extraction. However, for sound event analysis purposes, feature extraction often depends on feature engineering to carefully craft features from low level representations, by using domain expert knowledge. Generally, audio features can be separated into two categories: time-domain and frequency domain. The most common features are presented in Serizel et al. (2018), being some of them explained in this section, narrowing however the level of detail.

Firstly, to better process and analyze audio data, one often represents sound signals as *frames*, that consist of smaller groups of samples across time. The frame length must be set long enough to contain enough relevant data, but not so long that temporal variations disappear. In speech recognition, for example, the typical frame length choice is 25 ms.

The *temporal features* are computed on the temporal waveform, being its computation rather straightforward. The time domain envelope can be seen as the boundary within which signal is contained. An example is the *root mean square (RMS)* feature that computes the energy from the sound signal, being a reliable indicator for silence detection.

Also, in the temporal domain, the *zero crossing rate (ZCR)* is given by the number of times the signal amplitude crosses the zero value, and it is useful for discriminating periodic signals, which have small ZCR values, from signals that may be corrupted by noises, which present high ZCR values. Furthermore, to represent the different characteristics of the time domain waveform's shape, one can use the temporal waveform moments, further detailed in Serizel et al. (2018).

The *spectral shape features* are the result of deriving features from the frequency representation of the signal, for example, from the spectrogram, being commonly used, as the perception of sound often relies on its frequency content. In Serizel et al. (2018), the author points the most common spectral features, such as the spectral envelope, similar to the time domain envelope, mapped however to the frequency domain, as it can be seen as the boundary within which spectrum of a signal is contained.

Nevertheless, the two aforementioned type of features are rarely used separately, as they are mostly designed to model specific aspects of the signal. Consequently, the temporal and spectral shape features are often considered and evaluated together, as one set of features, commonly referred to as *low-level features*.

The *cepstral features*, as explained in Serizel et al. (2018), allow the decomposition of the signal according to the source-filter model, being widely used to model speech production. The signal is decomposed into a carrier and a modulation, in which the first represents the source, and in speech includes the glottal excitation. The latter represents the filter, and in speech includes the vocal tract and the position of the tongue. The *Mel-frequency cepstral coefficients (MFCCs)* are the most common cepstral coefficients and audio signals have been traditionally characterized by this particular feature. As detailed in Serizel et al. (2018), the filter banks used for the MFCCs' computation, typically 12 to 30, approximate some important properties of the human auditory system, being the main reason to these features success with structured sounds, such as speech and music. Nonetheless their performance degrades in the presence of noise and when analyzing noise-like signals that have a flat spectrum.

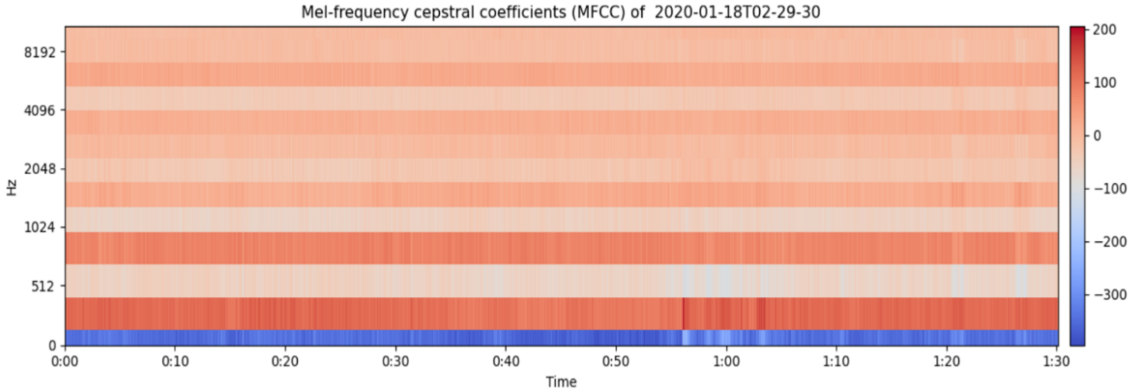


Figure 2.5: MFCCs representation - corresponds to the RFCx's digital sound of the day 18/01/2020 at 02:29:30 with a sampling rate of 22050 Hz.

The *perceptually motivated features* are an additional set of features which can also be used for audio classification. Loudness, sharpness and perceptual spread are three examples of these audio features, as referred in Serizel et al. (2018).

Finally, it is also important to introduce the *Mel spectrogram*, shown in Fig 2.4. In this particular spectrogram, that has 128 coefficients, the frequencies are converted to the *Mel scale*. This scale provides a linear scale for the human auditory system, enabling a more approximate depiction of what humans perceive. It is noteworthy because it allows the use of well-researched image classification techniques, as this feature is an image representation of the sound signal.

3

Literature Review

Contents

A vast number of tasks concerning time series usually fall under motif or anomalous event (discord) detection. Most of the algorithms that concern the latter term are explained in Branco (2020) and, analogously, some of the concepts introduced there are here described, as they can be adapted to motif discovery.

Similarly, the methods that conduct motif discovery, in terms of their outcome, can be divided in two main categories:

1. The outcome is a continuous score which describes the level of trust in the classification of the motif.

2. The outcome is a binary label, that states if a motif was found or not. Usually, is issued when the value of the analysed variable surpasses a previously set threshold, which is a tuning parameter.

It is also important to remark that the previous scoring mechanisms can also be adapted to work together. In addition, Branco (2020) gives an overview of the class of algorithms commonly used in this particular domain, being, consequently, a strong reference for our definitions.

Machine learning algorithms can be characterized as *offline* or *online learning algorithms*. The first set of algorithms has access to data beforehand, whereas the models created by the second set of algorithms are continuously updated with the produced data. Thus, online learning algorithms are of extreme importance, mainly due to the need of handling streaming data.

Furthermore, these algorithms can also be grouped based on their need to have prior knowledge, that is, have labelled data, or not. *Unsupervised, Semi-Supervised*, and *Supervised* tasks have no labels, a few labeled objects or full labeling of data, respectively.

The annotation process, in the last two referred tasks, can be costly and require a thorough manual work, being the two main reasons for labelled data not being available in some cases. Hence, Unsupervised learning approaches can overcome this problem, and may represent a more fitting option to perform motif detection, mostly, owing to the lack of labelled data associated with environmental audio files. In Branco (2020), more detailed information it is provided on each of these methods, such as their pros and cons.

Supervised techniques require labeled data to predict a given outcome and, generally, two main concerns come up with this particular approach. The first arises when the model is skewed towards the majority class, problem that does not occur when the dataset in question is balanced. The second is related to the difficulty of obtaining labelled data, as mentioned previously.

Moreover, in Semi-Supervised techniques, only a part of the observations is labelled. As described in Branco (2020), in the context of anomaly detection it is frequent to have just one of the possible outcomes labelled, either the normal or the abnormal one, being the first labels easier to acquire.

Similarly, this can also be the case with the RFCx's data, as the raw audio files often do not go through any annotation process, in order to obtain the mentioned labels. Additionally, the diversity of sounds present in the recorded files, sometimes hardly indistinguishable from each other, make this task even more difficult, having also a strong probability of having incomplete profiles, given the amount of available data.

Conversely, apart from not requiring labelled data, Unsupervised techniques also make no distinction between train and test data. As explained in Branco (2020), algorithms in this category make the assumption that normal observations are more frequent than abnormal ones, making, therefore, a clear separation between what is normal and what is not. The main reason to this is that if the previous assumptions did not stand, techniques in this category would have high false alarm rates. As a consequence, numerous works use the raw audio waveforms as input features, process often referred as ”end to end” learning. In Dai et al. (2017), a Convolutional Neural Network was trained on raw audio data to match Log-Mel spectrogram features in an audio classification task. Also, Hitachi Vantara has successfully developed an approach in which the FFT is used to describe the overall shape of the spectral envelope, computing then the *Kernel Density Estimation* for each of the FFT results. Subsequently, the frequency is divided into buckets and the probability density function is derived, for each one of them, being the *PCA* performed afterwards to present only the most representative dimensions. Finally, the components are clustered, being each one of them represented by a *Gaussian Distribution (Gaussian Mixture Model)*, and the probability of the data belonging to each one of these distributions computed. The high probability clusters are chosen and a manual annotation process takes place to map the distributions to actual labels, which results in the identification of four major disturbance events, such as the presence of human sounds or chainsaw sounds.

3.1 Sound Event Detection and Classification

To better understand the two main research areas that fall within *computational auditory scene analysis* it is important to get acquainted with the terms introduced in Imoto (2018), Stowell et al. (2015). A *frame* (or sound clip), as mentioned previously, indicates the unit of analysis and may contain several events that may overlap in time. Moreover, in an *acoustic scene* the label describes the place where the sound was recorded (park, office, kitchen) whereas in an *acoustic event* it refers to the sound type (rainwater, music, noise, scream, etc.).

The first research direction is the *acoustic scene classification*, that aims to provide a textual label that characterizes the acoustic environment. Naturally, it will not be the focus of this work, as the setting in which the sounds are recorded is well known (rainforests).

Oppositely, the other research direction is *sound event detection and classification*. The goal of *sound (acoustic) event classification* is to determine which acoustic event appears in an audio sample, not taking into account its corresponding time and its number of occurrences. Nevertheless, as acoustic events can overlap temporally, acoustic event classification sometimes is not a practical problem. On the other hand, *sound event detection* labels temporal regions within an audio recording, with their start and end time, as well as with the event's type.

The previous task can be divided into *monophonic event detection* where only the most prominent event is identified and *polyphonic event detection* where multiple events are allowed at the same time, that is, can be overlapped. A single classifier can be used for the first case whereas one can consider separate classifiers per event type, or a multi-label classifier as a joint model for the second one.

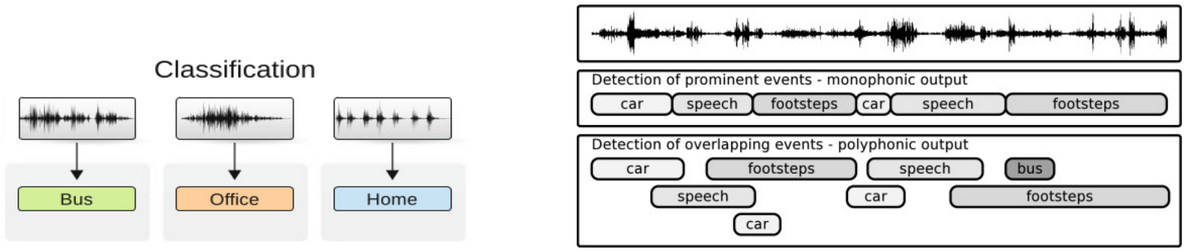


Figure 3.1: Left: Sound event classification - Right: Sound event detection (monophonic and polyphonic)

The referred classifiers, in sound event detection, ideally, have each one of the acoustic events instances in the training data, labeled with their start and end time. This type of labels is referred to as *strong labels*, nevertheless, acquiring them is a costly process that also requires careful attention to detail by the annotator.

On the other hand, the labels that do not contain any data about the temporal location of each event or the number of occurrences in the recording are called *weak labels*. In comparison, collecting weakly labelled data takes much less time, since the annotator only has to mark the active sound event classes and not their exact boundaries. Moreover, when performing sound event detection with small frames as inputs, one may end up with frames that do not have a direct label. This is known as a *Multiple Instance Learning (MIL)* problem where instances are grouped into a 'bag' and labels are associated to bags instead of being linked to each individual instance, a setting further detailed in section 3.8.

All in all, the previous tasks assume the availability of labels, which might not be the case when considering the particularities of environmental data, as the recordings may lack annotation.

3.2 Matrix Profile

Following the definitions introduced in section 2.1, the *matrix profile* algorithm proposed by Eamonn Keogh will be further explained in this section, having as reference Yeh et al. (2016), Yeh et al. (2017b), Dau and Keogh (2017), Yeh et al. (2017a), Yeh et al. (2016), Zhu et al. (2016). This method represents a significant progress in motif discovery and its properties, also listed in Yeh et al. (2016), are the following:

- It is exact, providing no false positives or negatives
- It is simple and parameter-free
- Its computational space is $O(n)$, with a small constant factor.
- It is still extremely scalable considering that it is exact, enabling thus the computation of its results in an anytime fashion, allowing ultra-fast approximate solutions
- Once the similarity join is computed for a dataset, one can incrementally update it in an efficient manner.
- The proposed method provides full joins, therefore not requiring the definition of a similarity threshold.
- It is parallelizable, both on multicore processors and in distributed systems

The general intuition behind this approach is that all distance profiles, with the trivial match region not included, are upper bound approximations to the matrix profile. Thus, to obtain the matrix profile, one can compute all the distances profiles and extract only the minimum value at each location. Furthermore, some algorithms are available to enable the fast computation of the distance profiles, hence of the matrix profile.

The *Scalable Time series Anytime Matrix Profile (STAMP)*, further detailed in Yeh et al. (2016), is used to compute the matrix profile, having an overall time complexity of $O(n^2\ logn)$. It takes advantage of the *Mueen's ultra-fast Algorithm for Similarity Search (MASS)* which is an Euclidean distance similarity search algorithm that computes the distance profile in $O(n\ logn)$ time. It provides exact solutions, it is a scalable algorithm, is incrementally maintainable and allows for fast approximate solutions, making it an anytime algorithm. In addition, the *Scalable Time series Ordered-search Matrix Profile (STOMP)* Zhu et al. (2016) is a significant faster version of the STAMP algorithm, requiring only $O(n^2)$ time. It is particularly useful when one is willing to forego the anytime property, which in some cases is not useful. Recently introduced, *SCRIMP++* (Zhu et al. (2018)) is an $O(n^2)$ time algorithm that is also an anytime algorithm, combining the best features of STOMP and STAMP.

Finally, some remarks on the algorithms evaluated by the author. The algorithm's performance it is not compromised by the subsequence length m, that is, all of them are time independent of the data. Moreover, oppositely to motifs, a time series discord is the subsequence that has the biggest distance to its nearest neighbor. Lastly, the mentioned methods, used to compute the matrix profile, can be found online. [1].

[1] https://matrixprofile.org/

3.3 Motif Discovery with Matrix Profile

Once the matrix profile is computed, motif discovery becomes trivial as the locations of the two (tying) minimum values correspond precisely to the locations of the first motif pair, as depicted in Fig. 3.2 and in Dau and Keogh (2017).

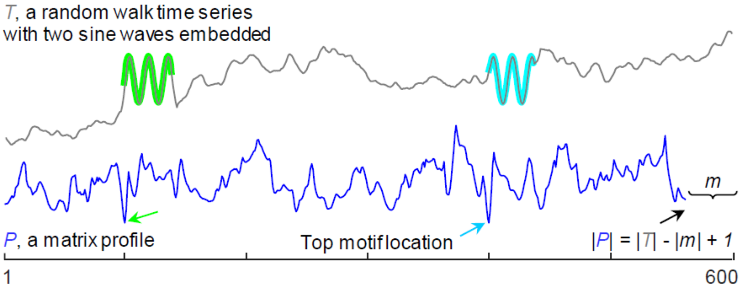


Figure 3.2: A time series T and its self-join matrix profile P. The two minimum values of the matrix profile correspond to the first motif pair.

Additionally, this technique can be applied to audio data, as showcased in Yeh et al. (2016), namely with the MFCCs. More concretely, the authors evaluate the algorithm with the 2^{nd} MFCC at 100Hz, extracted from the raw audios of two popular songs. In this experiment, they are able to find a highly conserved subsequence which corresponds to the baseline of the first song, and that was plagiarized by the second. Thus, the ability of finding conserved structures in apparently disparate time series, namely with audio features, defines the importance of this approach Yeh et al. (2016) to our work.

In Branco (2020), the author performs online anomaly detection in data streams, by combining the matrix profile with *all pair similarity search (APSS)*. Analogously, a similar approach can be followed, adapting, nonetheless, the proposed method to search for motifs instead. Moreover, the main shortcoming of the matrix profile is that it is essentially visual, and as stated in Branco (2020), one must find the parameters that best suit the problem, such as the window size, the number of top motifs, among others. The overview of the algorithm proposed in Branco (2020) is the following:

Adapted Matrix Profile overview:
Similarly, before receiving any data, an initialization of a set of parameters is required, being followed by the process here described, which runs indefinitely:

1. **Receive Data Point:** timestamp and value.
2. **Maintain Dataset:** upon a new data point's arrival, a dataset of a predefined size is maintained.
3. **Evaluate Anomaly:** the matrix profile and the top K discords are extracted from the dataset, with a preestablished window size (W).

4. **Anomaly Score:** A frame with the top k discords is kept and is where the anomalies are stored. The anomaly score issued is 1, when the value returned by the similarity function proposed, surpasses a predefined threshold, resulting in the insertion of the new anomalous pattern in the anomaly database or its replacement whenever the same it is full.

Hence, given that with the matrix profile's computation one can extract motifs as easily as it can extract discords, the Adapted Matrix Profile algorithm proposed in Branco (2020) can be a rather straightforward solution to conduct motif discovery over data streams.

Lastly, the *annotation vector (AV)* may be a resourceful manner to manipulate the motif search in a way that it can be used to discover more meaningful motifs. As shown in Dau and Keogh (2017), this is achieved by combining the matrix profile with the annotation vector to produce a new matrix profile, often referred as the *"Corrected" Matrix Profile*, that correctly incorporates the contextual bias for the problem at hand. In particular, the AV is a time series, with the same length as the matrix profile, consisting of real-valued numbers between [0 - 1]. A low value indicates that the subsequence starting at that index is not a desirable motif, and therefore should be biased against. Conversely, higher values mean that the subsequence at that location should be favored for the potential motif search.

3.4 Multidimensional Matrix Profile

This section introduces *Multidimensional Motif Discovery*, having as reference the algorithm proposed by Eamonn Keogh in Yeh et al. (2017b).

The classic unidimensional matrix profile motif discovery is able to correctly find, on the first two dimensions, the motifs at locations 150 and 350, as depicted in Fig 3.3 and in Yeh et al. (2017b).

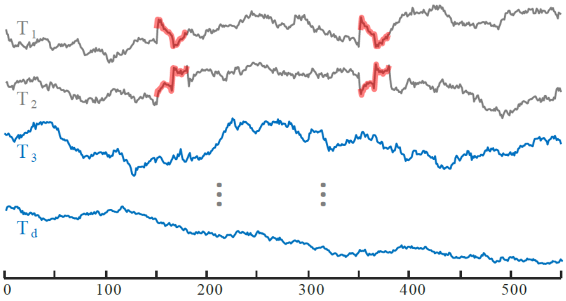


Figure 3.3: A example of a multidimensional time series. Both of the first two dimensions have a motif of length 30 embedded at locations 150 and 350.

Nonetheless, and considering the definitions used in section 2.1, motifs can be readjusted to *Multidimensional Time Series* data (MTS). The author demonstrates that when in the presence of irrelevant dimensions, namely with just eight, the algorithm does not perform well in the motif discovery task.

Furthermore, the rareness of motifs increases with the growth of dimensions, as higher dimensionality masks the motifs that exist in a subspace of the data. In Yeh et al. (2017b), the prevalent problem of several industries is also introduced, which consists of suspecting the presence of motifs in some subset of a time series, but not knowing which dimensions or how many dimensions are involved.

The author begins to compare the proposed framework to similar work in this particular domain. In Minnen et al. (2007), the proposed method was robust to a small number of irrelevant dimensions, but in comparison, the proposed solution is capable of handling higher dimensionality and more irrelevant dimensions.

Moreover, another possible solution could be to perform multidimensional motif discovery by "transforming multidimensional time series data into one-dimensional time series data", as in Tanaka et al. (2005). Despite being a rather straightforward approach it requires all or most of the dimensions to be relevant, also having the algorithm's speed and accuracy dependent on the tuning of numerous parameters.

To sum up, all the approaches prior to the introduced framework had at least one of the listed shortcomings, slow, approximate and brittle to irrelevant dimensions, whereas the proposed solution is fast, exact and robust to hundred of irrelevant dimensions.

The developed framework is able to handle the following type of queries given a large k-dimensional time series:

- **Guided Search**: Find the best motif on k dimensions, where the integer k is given by the user, but which k dimensions to use is unspecified.
- **Constrained Search**: Find the best motif on k dimensions, but explicitly include (or exclude) a given subset of dimensions.
- **Unconstrained Search**: Find the best motif on k dimensions, where k is not given by the user but is the "natural" subset of the data that has motifs.

The *mSTAMP* algorithm, detailed in the aforementioned article, can compute the k-dimensional matrix profile, where k represents the k combinations of dimensions from all the d dimensions. This combinatorial search space can be searched efficiently in a greedy way, resulting in the computation of the k-dimensional matrix profile, for every possible setting of k, simultaneously, in $O(d\ log\ dn^2)$ time and $O(dn)$ space.

In particular, the algorithm consists on the computation of the *k*-dimensional distance profile for a given subsequence under every possible setting of k (from 1 to d). Hence, as showcased in Yeh et al. (2017b), it is sufficient to justify the algorithm's overall correctness by demonstrating the correctness of the computed k-dimensional distance profile.

Regarding the algorithm's scalability, as the authors claim, is something which is inherited from the use of the matrix profile, which can be efficiently computed with the previously mentioned computational methods, such as the STAMP (Yeh et al. (2016)) and STOMP (Zhu et al. (2016)) algorithms, or with their GPU versions (Zhu et al. (2016)).

Finally, this particular approach illustrates its potential to successfully perform motif discovery on audio data, since it is capable of dealing with audio features, such as the MFCCs and the Mel spectrogram features. Particularly, the author evaluates the algorithm with sound signals, having rather promising results, which might indicate the possibility of exploring the mSTAMP in our particular problem. In fact, motif discovery was conducted with the multidimensional variant of the matrix profile and applied to the Mel spectrogram of the song ”Never gonna give you up” by Rick Astley. The results are quite positive as the algorithm was able to discover the chorus of the song when applying the matrix profile to all dimensions, with a five-second subsequence length. In addition, the mSTAMP was also applied in subspaces ranging from 1 dimension to 32 and while most of the high dimensional motifs matched part of the chorus, the one-dimensional and two-dimensional motif pairs represented only drum patterns.

In the light of the example, one can now understand one of the advantages of this technique, in which, once the multidimensional matrix profile is computed, one can explore it for different dimensionalities without additional cost, that is, one can easily decide the correct number of dimensions for the specific problem at hand. Also, the source code of the mSTAMP algorithm is available online [2].

3.5 Motif Discovery in Audio Data

In section 3.3 and 3.4, we have reported the capability of the matrix profile and its multidimensional version, performing successful motif discovery in audio features. This section addresses the research done by Eammon Keogh, in relation to the existing audio motif discovery techniques, also describing the proposed algorithm, that poses significant advantages in comparison to the existent ones, having as reference Hao et al. (2013).

The most commonly used approach to find repeated patterns in audio is to “use string-matching techniques on a symbolic representation learned from the data” (Aucouturier and Sandler (2002)). However, symbol extraction algorithms fail to generalize for different sounds.

Additionally, commercial music applications were developed taking advantage of the work done in fast audio searches, often referred to as *audio thumbnailing* or *audio fingerprinting*. As the author states, such technique assumes the existence of a ”platonic ideal” sound snippet, a master recording of a song. Nevertheless, sound instances may differ due to different encodings, or as for the case of Shazam/SoundHound [3], due to background noise present in the recording process. In particular, the two snippets are assumed not to have time warping.

[2] https://sites.google.com/view/mstamp/

[3] Shazam/SoundHound are commercial mobile phone based music identification services. A cell phone’s built-in microphone is used to gather a brief sample of music being played. An acoustic fingerprint is created based on the sample, and is compared against a central database for a match.

The main difference between the proposed work and the prior audio motif discovery approaches is that the first does not make any assumptions about the intrinsic properties of the objects, when finding repeated patterns in audio sequences. In detail, it does not need the common set of possible features considered by researchers Hsu et al. (2001), that include the tempo, loudness, pitch, among others. This is because the mentioned process requires a considerable amount of feature engineering, existing also evidences that they do not generalize well across diverse sound types.

Moreover, researchers Glaze and Troyer (2007) attempt to find repeated patterns in bird songs, by extracting features from bird syllables, process that demands significant human intervention, which the proposed work avoids.

The Eammon Keogh' work Hao et al. (2013) redefines the already presented concept of motif to *audio motif*, introducing also a new similarity measure, the *CK distance* Campana and Keogh (2010). The algorithm is built on the assumption that similar sounds produce similar images when transformed into spectrograms, and through the use of the CK distance measure, patterns can be revealed. The obtained results show that this function measures similarity in an identical way to the human notions of sound similarity.

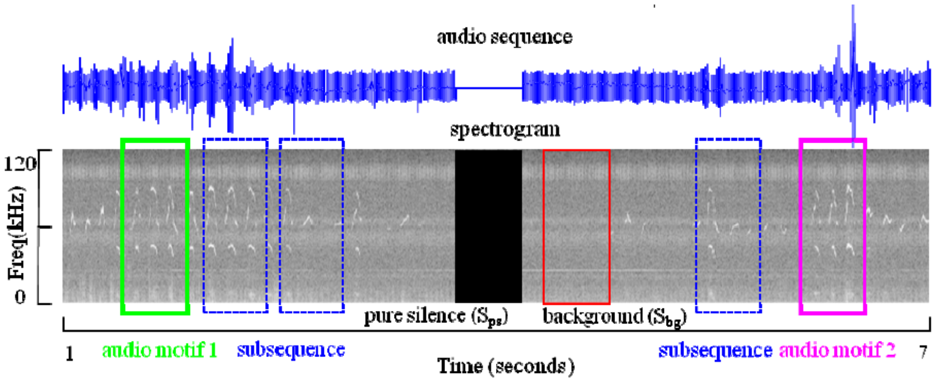


Figure 3.4: Illustration of the definitions introduced in Hao et al. (2013).

As shown in Fig. 3.4, this algorithm takes into consideration cases that are quite often in environmental data, such as sections of pure silence, representend as S_{ps} and sections with constant background sounds, representated as S_{bg}. One can find the formal definitions of the explored algorithms, such as the *Brute-force Algorithm* and the *Probabilistic Early Abandoning Audio Motif Discovery (PEAMD)*, in Hao et al. (2013).

Finally, the algorithm's scalability is also there demonstrated, as well as the capability of discovering motifs in bird songs, achieving promising results in this last field. The proposed methods can be found online [4].

[4] https://sites.google.com/site/audiomotif/

3.6 Neural Network-based Approaches

This section addresses the Supervised and Semi-Supervised learning algorithms, based on neural network approaches, having as baseline Nordby (2019), Maccagno et al. (2017).

In recent years, *Neural networks* assumed a dominant role in machine learning applications, being the *perceptron* Rosenblatt (1958), introduced by Frank Rosenblatt (1958), the basis of this type of networks. The simplest representation of a neural network is the *Multi-Layer Perceptron (MLP)*, being composed of an input layer, one or more hidden layers and an output layer. In detail, each layer consists of a number of neurons, that have as output the weighted sum of the inputs, offset by a bias and followed by an activation function f, as Fig. 3.5 depicts. The numerous adopted activation functions are further explained in Nordby (2019), as well as the typical training process of neural networks.

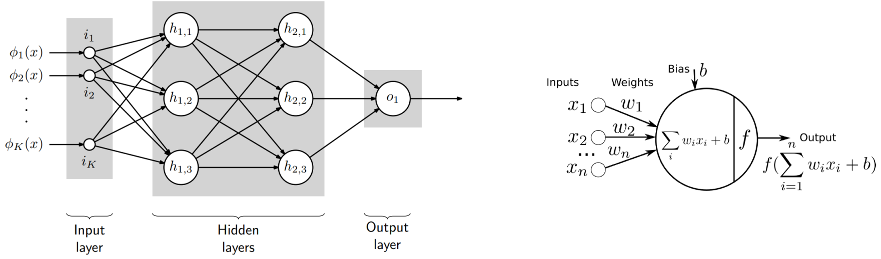


Figure 3.5: Left: Multi-Layer Perceptron with one hidden layer - Right: Computational principle of an artificial neuron on the right

3.6.1 Convolutional Neural Networks

Recently, *Convolutional Neural Networks (CNNs)* have outperformed the former models in visual recognition tasks, namely in large-scale image and video recognition, mostly due to the late availability of large public datasets of images, such as the ImageNet.

Moreover, time series motif detection and image segmentation are closely related, as observing and selecting repeated patterns in a time series is in everything similar to looking at an image and marking the desired image segments where, if present, the related patterns are located. Hence, this intuition is explored in Long et al. (2015), being a *fully convolutional network (FCN)* proposed to perform image segmentation. U-Net Ronneberger et al. (2015) improved upon the FCN architecture and proved to be successful in the segmentation of neuronal structures in electron microscopic images, being later applied to other tasks such as to biomedical image segmentation, automated driving, etc. In Wen and Keyes (2019), transfer learning techniques and time series augmentation strategies were followed to build a CNN model capable of carrying out anomaly detection in streaming data, using time series segmentation.

As previously mentioned, several CNN architectures were applied to the ImageNet Krizhevsky et al.

(2012) dataset to perform image classification, in particular fully connected *Deep Neural Networks (DNNs)* such as AlexNet Krizhevsky et al. (2017), VGG Liu and Deng (2015), resNetHe et al. (2016), among others. As aforementioned, the MFCCs and spectrograms can be thought as image representations of sound and recent works Briggs et al. (2012), Grill and Schlüter (2017), Hershey et al. (2017), Liaqat et al. (2018), have achieved promising results by exploring audio classification settings with these features. CNNs are able to exploit the adjacency properties of audio signals and recognize patterns in the spectrum image, achieving state-of-the-art performance in sound event detection and classification. However, the obtained results may not be directly employed to environmental data, due to its intrinsic characteristics.

Nonetheless, a preliminary work Liu et al. (2019) on the RFCx's data proposes two models designed to conduct sound event detection (SED): Aug-VGGish and FCN-VGGish. The mentioned models are applied to the ESC-50 public dataset and to the RFCx data, classifying chainsaw sounds in the latter, attaining promising results in both. Furthermore, a competition [5], which is a result of a partnership between Kaggle and RFCx, is taking place this year where contenders are encouraged to develop models that automate the detection of several species in the RFCx recordings. There are also multiple competitions on Kaggle whose objective is to perform audio classification on datasets such as ESC-50 and UrbanSound8k, and, in all, one can find multiple audio processing methodologies and models architectures, namely CNNs, that achieved significant results on the aforementioned task.

CNNs specialize in processing structured data, having a sequential (1D) or grid-like (2D or 3D) structure and being composed by two main layers:

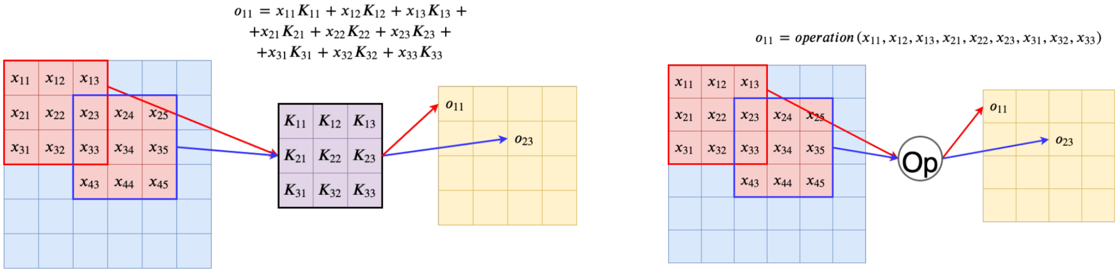
- **Convolutional layer:** The convolutional layer is where the convolution operation is applied on the input. A $k_h \times k_w$ filter (or kernel) matrix $\mathbf{K}_{ij}$ is passed over the input matrix $\mathbf{x} \in \mathbb{R}^{H \times W}$, and the convolution of $\mathbf{x}$ by $\mathbf{K}$ is a matrix $\mathbf{o} = (\mathbf{x} * \mathbf{K})$ where the coordinates are defined as:

$$o_{ij} = (\mathbf{x} * \mathbf{K})_{ij} = \sum_{h=1}^{k_h} \sum_{w=1}^{k_w} x_{(i+h-1)(j+w-1)} \, K_{hw} \tag{3.1}$$

  The output is frequently referred to as the *feature map*, and Fig. 3.6 is an example of this computation. One can think of each step as a dot product between the kernel and the image's window, being the result high if the window is similar to the kernel. This tells us that convolving an image with a kernel corresponds to searching for occurrences of the feature, represented by that kernel, in the image. Finally, the filter is shifted by a predefined stride along its dimensions. This layer is frequently used together with a non-linear activation function, providing some non-linearity to the network. Most often, the ReLU function is used, which converts all negative values to zero, keeping the positive ones.

[5] https://www.kaggle.com/c/rfcx-species-audio-detection/overview

- **Pooling layer:** The pooling layer is meant to reduce the dimensionality of the input, by combining the output of neuron clusters at one layer into one single neuron in the subsequent layer. This is achieved by downsampling the input's spatial dimensions (width and height), while maintaining the depth dimension. There are several pooling functions such as average pooling, weighted average pooling, and max pooling, being the last the most common one, and which output is the maximum value within the pooling window, as Fig. 3.6 illustrates.



$o_{11} = x_{11}K_{11} + x_{12}K_{12} + x_{13}K_{13} + {} + x_{21}K_{21} + x_{22}K_{22} + x_{23}K_{23} + {} + x_{31}K_{31} + x_{32}K_{32} + x_{33}K_{33}$
$o_{11} = operation(x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33})$
x_{11} x_{12} x_{13} x_{21} x_{22} x_{23} x_{24} x_{25} x_{31} x_{32} x_{33} x_{34} x_{35} x_{43} x_{44} x_{45}
K_{11} K_{12} K_{13} K_{21} K_{22} K_{23} K_{31} K_{32} K_{33}
o_{11}
o_{23}
Op

Figure 3.6: Left: Convolution operation of two output positions - Right: Pooling operation of two output positions.

The last layer of the network is a fully connected one, that returns the final classification, being a loss function subsequently applied to the classification output, in order to train the network with the back-propagation algorithm.

CNNs, in comparison to traditional fully connected neural networks that feature very dense interactions, allow sparse interactions. This results on a kernel smaller than the input, enabling the convolutional layer to learn local patterns using only meaningful features. In addition, they allow parameter sharing, as each member of the kernel is usually used at every position of the input. As a consequence, these models have the capability of learning not only translation invariant patterns, that is, are able to recognize a pattern in different positions of the input, but also spatial hierarchies of patterns, that is, subsequent layers learn more complex patterns that those acknowledged in the previous layers.

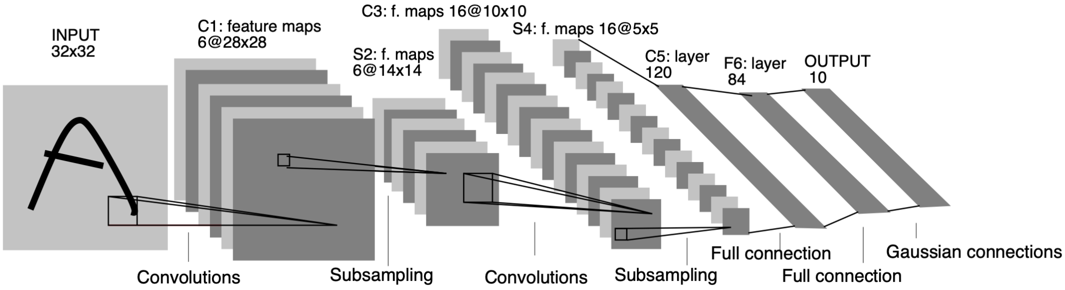


Figure 3.7: One of the early examples of a CNN: the LeNet-5 Lecun et al. (1998) architecture.

As shown in ”DCASE2018 Challenge Results” [6] and in ”DCASE2019 Challenge Results”, [7] if a large amount of labeled data is available, CNNs Liaqat et al. (2018) can be the best performing models in many audio classification problems, mainly when using Log-Mel spectrogram features. Nevertheless, other implementations such as Recurrrent Neural Networks (RNN) Bai et al. (2018), Non-negative-Matrix Factorization (NMF) Jamali et al. (2018), among others, achieved good results in the mentioned competition. This competition is one of the many examples Briggs et al. (2012), Grill and Schlüter (2017) that demonstrate the growth on research concerning bird detection. It is also important to remark that RFCx offers a solution for acoustic biodiversity monitoring [8], with the availability of a CNN model as an upcoming feature.

As mentioned previously, supervised learning algorithms need a considerable amount of labelled data, which might not be the case in the context of environmental data, limiting, therefore, this setting. *Data augmentation* emerges as a strategy to diminish this problem, as it synthetically generates new labeled samples from the existing ones, expanding the effectiveness of the training set. A more detailed overview of the techniques is given in Abeßer (2020), including techniques that apply various audio signal transformations, such as time stretching, pitch shifting, dynamic range compression, adding random noise, etc. Additionally, SpecAugment Park et al. (2019) is a simple data augmentation method for speech recognition, that contrasts with the most common ones, as it is directly applied on Log-Mel spectrograms instead of raw audios. In particular, Google's augmentation policy achieves state-of-the-art performance, outperforming all prior work attained in some speech recognition tasks.

Model performance and capability to capture the natural variability of data can be increased with the use of data augmentation techniques. Such signal transformations may include time shifting, volume control or adding additive noise to the acoustic data. The first concept consists of shifting a sound event in time and the second controls the volume of the acoustic signal. Additive noise consists of summing noise to the original signal, whether that represents Gaussian noise, uniform random noise, or a background recording, process further detailed in Eklund (2019). One can add *Gaussian or Pink noise*, with respect to the *Signal-to-Noise Ratio (SNR)*, technique that adaptively sets an appropriate noise level based on the amplitude of the original sound signal. Furthermore, Gaussian noise, often referenced as white noise, is a noise over the whole frequency range, oppositely to Pink which has a gradual decrease in noise intensity from low frequency to low frequency bands, approximating the characteristics of noise of the natural world.

[6] http://dcase.community/challenge2018/index
[7] http://dcase.community/challenge2019/index
[8] https://www.sieve-analytics.com/arbimon

3.7 Long Short-Term Memory Network

Recurrent Neural Networks (RNNs) Sherstinsky (2020) are a type of neural networks specifically designed for processing sequential data. They distinguish themselves from the other approaches by having a state which contains the information that the model has seen thus far. In particular, while traditional deep neural networks assume that inputs and outputs are independent of each other, the output of recurrent neural networks depends on the prior elements within a sequence. Another distinguishing characteristic of recurrent networks is that they share parameters across each layer of the network, with these weights parameters being adjusted through backpropagation and gradient descent. In detail, these neural networks leverage *backpropagation through time (BPTT)*, an algorithm which determines the gradients, being slightly different from the traditional method as it sums errors at each time step whereas feedforward networks do not need to sum errors as they do not share parameters across each layer. Nonetheless, RNNs tend to face the vanishing and exploding gradient phenomena, that arises from the difficulty to capture long term dependencies because of the multiplicative gradient, which can be exponentially decreasing/increasing with respect to the number of layers. This limitation results in a model which is no longer learning, or in an unstable model, respectively. *Gated Recurrent Units (GRU)* and *Long Short-Term Memory units (LSTM)* deal with the vanishing gradient problem encountered by traditional RNNs, with LSTMs being a generalization of GRUs.

The *Long Short-Term Memory (LSTM)* Hochreiter and Schmidhuber (1997) networks address the vanishing gradients problem, that is, succeed in keeping memory for a period of time. This is achieved by having a ”memory cell” in the hidden layers of the neural network, that has three gates, an input, output and forget gate. These gates control the flow of information which is needed to predict the output in the network, being this specific network is further detailed in Hochreiter and Schmidhuber (1997), Lezhenin et al. (2019).

As described in Lezhenin et al. (2019), LSTMs have been successfully applied to tasks such as speech recognition, speech synthesis, and video classification when in combination with CNNs. Moreover, these networks have also been introduced to sound event detection and classification Wang et al. (2016) and to urban sound classification Lezhenin et al. (2019). Additionally, one can find several implementations of these networks on the DCASE challenge [9], which explores and evaluates the LSTM’s performance when conducting sound event detection.

3.8 Weak Labeling

Multi-instance learning, originally proposed by Dietterich Dietterich et al. (1997) for drug activity detection, arises as a solution to tackle the common lack of labelled data.

[9] http://dcase.community/challenge2016/task-sound-event-detection-in-real-life-audio-results

In this setting, the training set is composed of several bags, each one with multiple instances, and if a bag contains at least one positive instance then it is labeled as a positive bag, otherwise, if it contains only negative instances is labeled as a negative one. Moreover, the only known labels are the ones belonging to each bag, being the training instances labels unknown.

The MIL of Neural Networks BP-MIP Kumar and Raj (2016) achieves good performance in comparison to well-established multi-instance learning methods, despite being a general algorithm, not optimized towards any data. It introduces a new error function, in relation to BP Zhou and Zhang (2002), in which the BP-MIP algorithm modifies the network's weights according to training bags, and not to training instances, therefore capturing the nature of multi-instance learning, a BP's shortcoming.

Particularly, audio event detection (AED) is conducted in Kumar and Raj (2016), by using the MFCCs and a Gaussian mixture model (GMM) to ensure a robust set of features, later fed to a detector model. Although the presented results showed reasonable performance for the BP-MIL framework, more exhaustive parameter tuning, concerning the neural networks's training, could have led to better results.

Additionally, the MIL framework can be extend to a *multi-instance multilabel (MIML)* framework, where a bag can be associated with multiple labels instead of only being linked to one. In Briggs et al. (2012), the data is transformed from its original representation into a suitable bag-of-instances representation, through the proposed MIML bag generator for audio. This makes possible the application of existing MIML classifiers, that in this article, perform the detection of birds species in audio files, namely in recordings collected from a forest.

The mentioned frameworks still required a set of manually annotated spectrograms, and the framework proposed in Ruiz-Muñoz et al. (2015) introduces a unsupervised segmentation method which does not requires training from manually segmented spectrograms. Nevertheless, the results show that there is no significant difference between the proposed method and its baseline Briggs et al. (2012).

3.9 Evaluation Environment

The approach developed in Branco (2020) is evaluated through the Numenta Anomaly Benchmark, which addresses anomaly detection in data streams. Nonetheless, nothing similar was found in the literature concerning motif detection, being Eammon Keogh's the major source of the work made in this field.

The matrix profile algorithm strongly relies on visual inspection, and as shown in Hao et al. (2013), Yeh et al. (2016), Zhu et al. (2016), one can carry out a posterior evaluation of the motifs found, by having labelled data at their disposal. Additionally, by correlating the observed motifs with other (internal or external) data, one can form hypotheses and open avenues for further research.

This section follows the definitions described in Branco (2020), as the problem at hand is similar, in the sense that motif discovery is binary, that is, we either identify a time point as being a motif or not. Furthermore, in the case of the ”Rainforest Connection Species Audio Detection” challenge, the objective is to classify bird and frog sounds. Consequently, multivariate evaluation metrics are necessary to capture the aggregation of the multiple classification scores for each attribute, combining them, afterwards, into a final result. In addition, the results obtained in this competition may serve as a benchmark to our developed solutions.

A *Confusion Matrix* is a table that accounts for the differences between the observed prediction and the true outcome. It comprises the number of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) classifications, widely used metrics for binary classification performance. Nonetheless, it holds reduced interpretation in cases where a class imbalance is present and, due to the nature of streaming data, does not reflect change detection in due time. Furthermore, several metrics can be extracted from it such as:

Definition 17. ***Accuracy:*** *Corresponds to the percentage of correct predictions over the total number of instances evaluated.*

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{3.2}$$

Definition 18. ***Sensitivity or Recall:*** *Measures the fraction of positive patterns correctly classified.*

$$Sensitivity = \frac{tp}{tp + fn} \tag{3.3}$$

Definition 19. ***Precision:*** *Measures the fraction of an identified event correctly classified.*

$$Precision = \frac{tp}{tp + fp} \tag{3.4}$$

Definition 20. ***Average-precision:*** *Summarizes the precision-recall curve as the weighted mean of precisions achieved at each threshold, with the increase in recall from the previous threshold used as the weight.* P_n *and* R_n *correspond to the precision and recall at the nth threshold.*

$$AP = \sum_{n} \left(R_n - R_{n-1}\right) P_n \tag{3.5}$$

Definition 21. ***F-Measure:*** *With* β *ranging in from* $[0, +\infty[$ *with higher* β *values putting more emphasis on false negatives. Default value is 1 where both false positives, and false negatives are weighted evenly. This may be particularly important in our case since anomalies are usually a minority instance and F-Measure is widely used in imbalanced situations.*

$$F - Measure = (1 + \beta^2)\frac{precision * recall}{(\beta^2 * precision) * recall} \tag{3.6}$$

Our preliminary work, revealed that the accuracy scores might be misleading, in the sense that they did not capture how well the model was generalizing. In depth, as the training set was unbalanced, the network was skewed towards the majority class. Thus, metrics such as Precision, Recall or the F-measure represent better the relevancy of the obtained results, being the last metric commonly used by Eammon Keogh Yeh et al. (2017a). It is also important to make sure such decisions are issued in due time and models run in resource aware environments, being the least amount of space used (limited memory).

Finally, the desire for a low false positive rate is worth mentioning as it might influence analysts into disregarding warnings if they are too common. Similarly, a low false negative rate is also desirable, when detecting chainsaw events, for instances, mainly because missing the detection of such events does not contributes to the prevention of illegal deforestation.

4

Methods

Contents

This section details the proposed methodologies that address the automation of species detection in noisy environments with limited training data.

The first approach is centered on the idea that sound signals can be represented by images. Thus, by extracting the spectral audio features, namely the Mel spectrograms, this methodology leverages deep neural networks, such as Convolutional Neural Networks, to perform the aforementioned task. Also, it takes advantage of transfer learning to reduce training requirements, both the amounts of data and time. The obtained results validate the proposed framework, as the proposed model is capable of differentiating the multiple events present in the image representations of sound.

Alternatively, the second methodology aims to reduce the procedures associated with an image-based approach. Thus, the second approach presents an alternative procedure, that explores different audio features and a distinct network, namely a Long Short-Term Memory (LSTM) network, to identify the given species in the multiple recordings. The obtained results reveal that the models trained on the cepstral features, namely the Mel-frequency cepstral coefficients (MFCCs), achieve better performance, nevertheless, the results are relatively worse than the ones attained by the spectral based one. In this sense, we complement this procedure with the motifs extracted by the matrix profile algorithm, as a way of improving the performance of the concerned network. Moreover, we explore the standard and the multidimensional implementation of the matrix profile algorithm, experimenting also different window sizes and predictive thresholds.

Finally, it is important to note that the classifications models that result from both approaches were trained and evaluated with a fixed training and test set. In particular, the results presented in section 5 reflect this condition, mainly because the definition of both methodologies results from numerous experiments, in which we considered different network architectures, training configurations and feature extraction and processing techniques. We only use cross-validation to train and test the developed classification models in section 5.2.5, when both methodologies are well defined and matured.

4.1 Spectral Based Classification Model

Our first proposal was a bioacoustic classification framework using transfer learning of deep neural networks. Thus, this section focuses on detailing each step of the suggested end-to-end pipeline, a process that results in a **classification model**, as represented in Fig. 4.1. The starting point consists of converting the sound sequences (**raw audio)**, that is, the time series, into audio features that can capture the distinctive properties of each event. Given the results obtained by deep neural networks in image classification problems our **feature extraction** step focuses on the extraction and processing of the Mel spectrograms, that are image representations of sound. So, we explore the learning ability of deep neural networks, namely Convolutional Neural Networks, describing their **training** process with the mentioned spectral shape features.

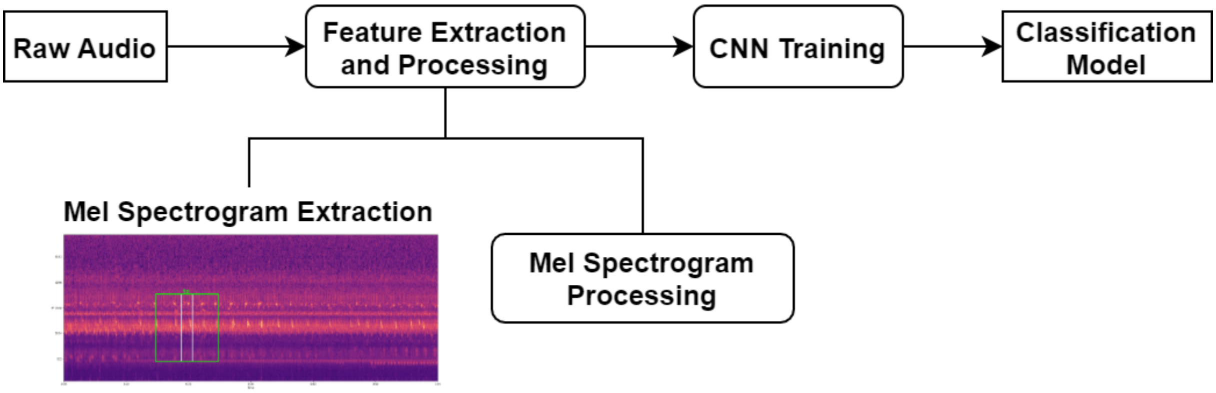

Raw Audio
Feature Extraction and Processing
CNN Training
Classification Model
Mel Spectrogram Extraction
Mel Spectrogram Processing

Figure 4.1: Spectral based approach flowchart.

In addition, throughout the introduction of the framework we will apply the presented techniques to a toy problem, which consists of one recording labelled with a positive event, to better illustrate the suggested methodologies. The raw audio of the toy problem's recording is displayed in Fig. 4.2.

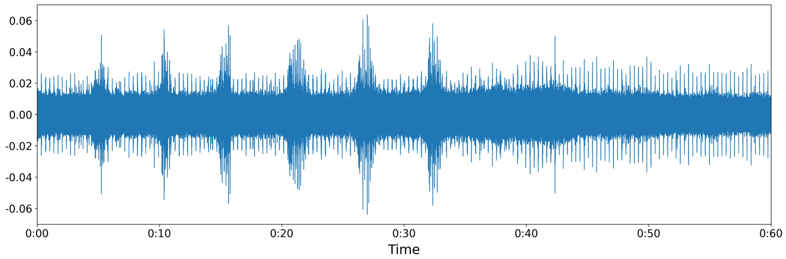


Figure 4.2: Toy problem: raw audio.

4.1.1 Feature Extraction and Processing

In audio processing and analysis, the frame length is critical to the neural network's performance, as it must be set long enough to preserve the meaningful events but not so long that temporal variations disappear. In this regard, this report proposes a window function and evaluates the effect of different window (frame) lengths on the model's results.

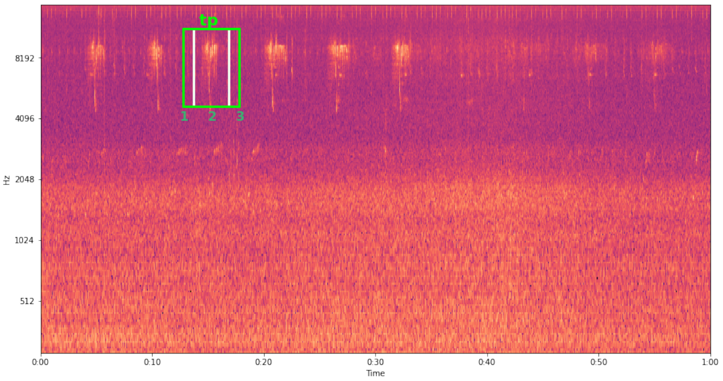


Figure 4.3: Toy problem window extraction - Mel spectrogram: window start (1) window center (2) window end (3).

The proposed window function defines the frame's center (window center) as the sum of half of the maximum label interval (maximum delta) of a given dataset, to the interval's beginning (interval start) of the concerned label. The start (window start) and end (window end) of the frame are the result of subtracting and adding to the center, respectively, the selected frame length (window duration) divided by two.

$$\text{window center} = \text{interval start} + (\text{maximum delta} / 2)$$

$$\text{window start} = \text{window center} - (\text{window duration} / 2)$$

$$\text{window end} = \text{window center} + (\text{window duration} / 2)$$

Additionally, it is important to note that the sampling rate by which the audio is extracted must be taken into consideration when extracting the mentioned window. All in all, the window function allows for a training set composed only by frames that are linked to a given event.

The extracted Mel spectrogram (**Mel spectrogram extraction**) is represented in Fig. 4.3 by the green box, being the white box the representation of the event's labelled time interval. Each Mel spectrogram is computed using the *librosa* Python package with the default settings (sampling rate = 48 kHz, NFFT = 2048, hop length = 512, window length = 2048, Hann window), specifying however the number of mel bands (n_mels = 224) and if available the minimum and maximum frequency. The frequency interval corresponds to the minimum and maximum value registered in the dataset, with a 10% margin to increase the considered interval.

The resulting Mel spectrograms, as a part of the **Mel spectrogram processing** step, are converted to units of decibel (dB), resized to the dimensions supported by the pre-trained model, that for the ResNet50 case correspond to 224x224 images, and normalized with the min-max scaling. Finally, the spectral features are converted to color images, that is, images with RGB channels and given the transfer learning setting, the spectrogram is processed to the adequate image format of the selected backbone model. For the ResNet50 model, for instances, the images are converted from RGB to BGR, then each color channel is zero-centered with respect to the ImageNet dataset, without scaling. We applied the mentioned techniques to the toy problem and the result of this procedure is showcased in Fig. 4.4.

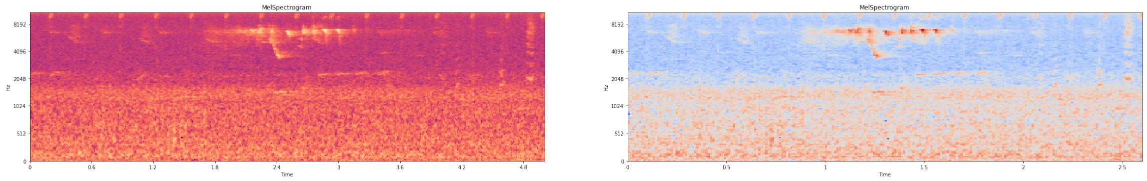


Figure 4.4: Toy problem: Mel spectrogram processing.

4.1.2 Model Training

The proposed model uses the pre-trained ResNet50 weights used for ImageNet classification, and includes only the feature extraction layers of this model, excluding the remaining layers, often referred as the network ”top”. Hence, the knowledge obtained in image classification, namely the detection of basic image features, can be transferred (**transfer learning**) to the task at hand by using the weights of the optimized model. In this sense, by freezing some layers of the pre-trained model and only training the last several layers, the model can be fine-tuned to our problem. In addition to ResNet50, our work also evaluates different backbone models, such as EfficientNetB0, InceptionResNetV2 and VGG19.

4.1.2.A Convolutional Neural Network

The proposed baseline model has as reference the networks introduced in Zhong et al. (2020), LeBien et al. (2020).

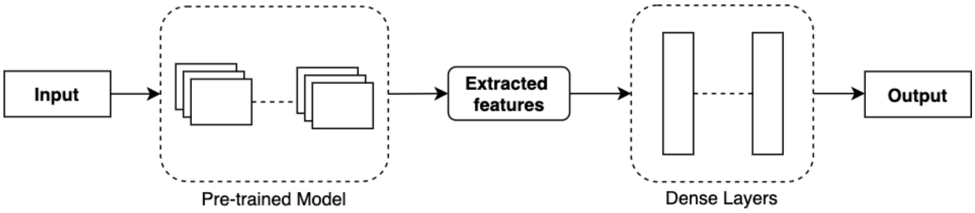


Figure 4.5: Model 1 Architecture: Transfer learning of a pre-trained CNN model.

The **model architecture** comprises the pre-trained model and two *fully connected (FC)* layers. The first consists of 512 nodes and uses the ”Relu” activation function that converts negative inputs to 0. This layer is followed by a batch normalization and drop-out layer, the latter with a drop-out rate of 50% in which each node is ignored with a 50% probability, helping prevent overfitting. The final layer, given the binary classification setting, consists of one node that passes through the sigmoid function.

4.1.2.B Convolutional Neural Network combined with a Long Short-Term Memory network

Additionally, we propose the addition of a LSTM layer to complement the above model, as a way of improving the general model's performance.

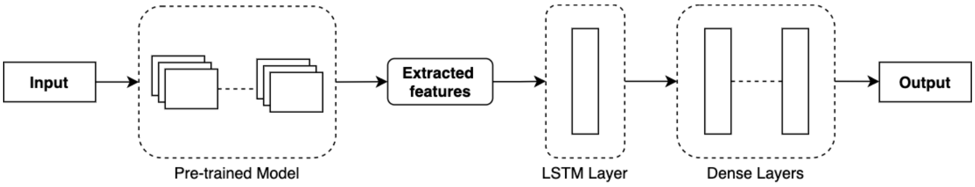


Figure 4.6: Model 2 Architecture: Transfer learning of a pre-trained CNN model combined a LSTM layer.

Hence, the **model architecture** includes the pre-trained network (ResNet50), and is followed by a Flatten and LSTM layer, being the latter composed by 512 neurons. Then, the subsequent layers follow the structure introduced in the section above (4.1.2.A), tuning however some parameters. In detail, we include two *fully connected (FC)* layers, the first with 1024 nodes and that uses the ”Relu” activation function and the last layer which comprises only one neuron. Likewise, between the aforementioned fully connected layers there are a batch normalization and a dropout layer to help prevent overfitting.

4.1.2.C Training

Given the binary classification setting, the **training** step consists of training the network on the spectral features to obtain a classification model. The optimizer uses the Adam optimization method with a learning rate of $1 * 10^{-4}$ and decay of $1 * 10^{-7}$. Moreover, the binary cross entropy loss function is utilized and 30 epochs are applied. These parameters result from a fine-tuning process in which we analysed the values who favored the model's performance. For instances, a higher number of epochs did not contribute to a significant improvement on the performance, ending up in an overfitting situation in the cases that did. Oppositely, a lower number of epochs usually did not lead to a convergence point, being the proposed value a trade-off between both scenarios.

Model performance and capability to capture the natural variability of data can be increased with the use of **data augmentation** techniques. Thus, two approaches are followed as a way of increasing the training set's effectiveness: the first randomly adds one of the two additive noises, Gaussian or

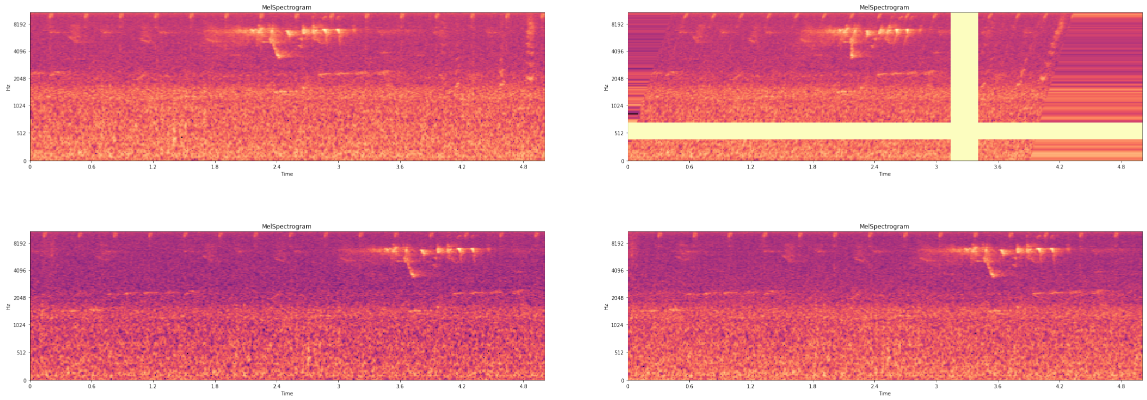


Figure 4.7: Toy problem data augmentations: a) original b) SpecAugment c) Gaussian Noise + Time Shift + Volume Control d) Pink Noise + Time Shift + Volume Control.

Pink, to the audio signal, time shifting and controlling its volume afterwards; the second applies the SpecAugment technique to the Mel spectrogram. Also, the mentioned techniques were applied to the toy problem and are showcased in Fig. 4.7.

4.1.2.D Overview

The presented end-to-end pipeline describes the feature extraction and processing steps required to transform the raw audios into the Mel spectrograms through the proposed window function. Furthermore, it details the training of two models that leverage transfer learning and data augmentation to improve their learning effectiveness. All in all, as explained in section 5, this framework is the best performing one, with the second model (4.1.2.B) improving the results attained by the first one (4.1.2.A).

4.2 Cepstral Based Classification Model

This section complements the research on the bioacoustic classification framework as it presents an alternative approach to the one proposed in section 4.1, the spectral based one. Despite having the same goal, as it also aims to obtain a model capable of learning the distinctive characteristics of the concerned events, it explores the use of different audio features, such as the Mel-frequency cepstral coefficients (MFCCs), the root mean square (RMS), the zero-crossing rate (ZCR) attributes, and even the raw audios. Nevertheless, it is important to remark that we focus our research on the cepstral ones.

Hence, the objective is to develop a simpler approach, in comparison to the previous one, in terms of the required feature extraction and processing steps. In this sense, the Convolutional Neural Network (CNN) was replaced by a Long Short-Term Memory network (LSTM), changing also the concerned features by the previously mentioned ones. In detail, we change the network to determine if the LSTM's remembering and forgetting nature contributes to the learning of the distinctive traits of the events present in our bioacoustic classification problem.

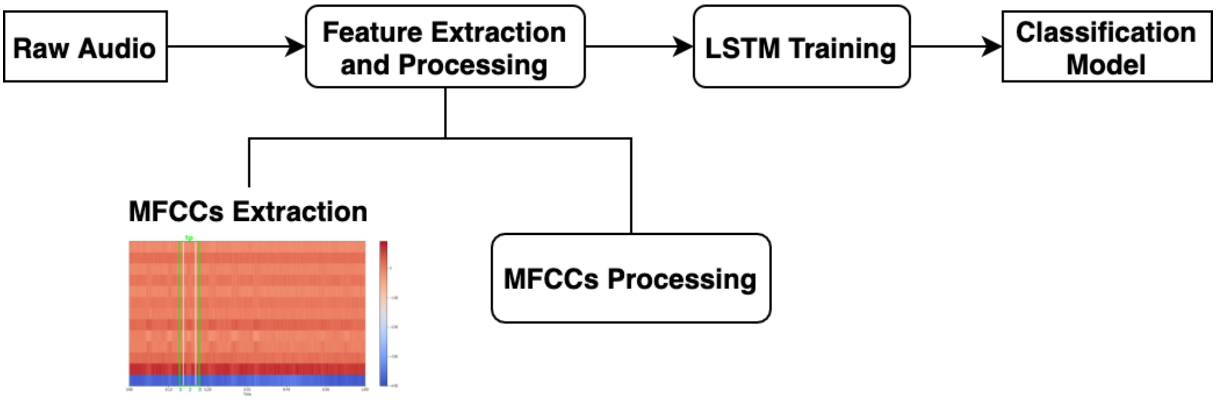


Figure 4.8: Cepstral based approach flowchart.

Analogously, the starting point of the proposed procedure consists of transforming the **raw audios**, displayed in Fig.4.2, into audio features that can be used to train the **classification model**. We describe the **extraction and processing** of the aforementioned features, as well as the **training** of the concerned classification model. Finally, we explore time series motif detection, facilitated by the matrix profile algorithm, as a resourceful manner of increasing the training set of each model. We hope that the larger training sets lead to better performance, as the data augmentation techniques did in the previous approach.

4.2.1 Feature Extraction and Processing

In this section, we cover the procedure that transforms the raw audios into the features used to train the developed model. We assume the window function proposed in section 4.1.1, as we will also have a training set composed only with frames associated with the presence or absence of a given event. So, the difference in this step lies on the extracted features and in their processing.

The procedure introduced in this section focuses on the cepstral features (MFCCs), however, as previously noted, it also addresses other audio attributes such as the ZCR, the RMS, and the raw recordings. The *librosa* Python package once again enables the extraction of these features.

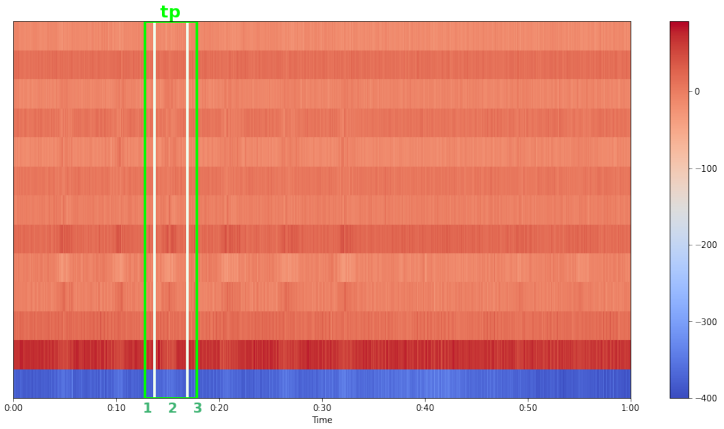


Figure 4.9: Toy problem window extraction - MFCC: window start (1) window center (2) window end (3).

As the lower order MFCCs contain most of the information present in the recordings we only extract 13 MFCCs. Also, although we initially tried normalizing this attribute, we ended up not performing this step as it did not benefit the model's performance. The toy problem's MFCCs attribute is represented in Fig.4.9. Lastly, the other features did not undergo through any additional processing.

4.2.2 Model Training

The **training** step is similar to the one described in section 4.1.2.C, differing only in the sense that it trains each model on the cepstral features, instead of the spectral ones. Similarly, the model is trained with the Adam optimization method, with a learning rate of $1 * 10^{-4}$ and decay of $1 * 10^{-7}$. Also, the binary cross entropy loss function is utilized, due to the binary classification setting, and 30 epochs are applied.

4.2.2.A Long Short-Term Memory network

The **model architecture** consists of one LSTM layer, that comprises 512 nodes and assumes the default activation function, the hyperbolic tangent (tanh). This layer is responsible for handling the input features and it is followed by three fully connected layers, the first with 256 neurons and the second with 128, both with a ”Relu” activation function. The final layer, given the binary classification setting, has one neuron that goes through the sigmoid activation function.

Also, it is important to note that the proposed architecture is the result of multiple experiments, in which we adjusted the configuration according to the attained results. The goal was to maintain the model as simple as possible without compromising its performance.

4.2.3 Motif Discovery using the Matrix Profile

The proposed methodology addresses an intrinsic problem of environmental data, that has to do with the limited amount of available training data. The spectral based approach, detailed in section 4.1, tackles this limiting factor with different data augmentations techniques.

On the other hand, this section explores the discovery of repeated patterns in the recordings, as a way of improving the general performance of the concerned classification model. In detail, we obtain the motifs with the matrix profile algorithm, exploring two variants of this method. The first uses the standard version of this technique and computes the matrix profile of one-dimensional features, of one of the MFCCs or of the ZCR attribute. The second explores the multidimensional version of the matrix profile, in which we conduct motif discovery on the whole MFCCs, as this variant is able to handle all this feature’s dimensions.

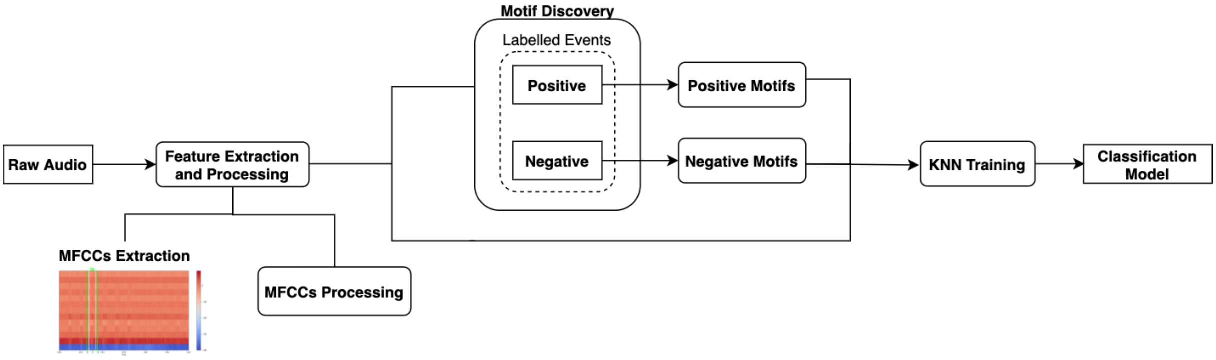


Figure 4.10: Motif based approach flowchart.

The motif extraction procedure, represented in Fig. 4.10, involves the transformation of the motifs extracted from the recordings into a training set that can serve as input to the classification model. In depth, we separate the obtained motifs into two groups that have a direct correlation with the two groups of labelled events. Thus, we end up with two datasets that contain the motifs related to these annotated groups, that is, associated with the presence (positive) or absence (negative) of an event. In the following sections, we detail the methodology used to perform motif discovery in both settings, the one-dimensional and the multidimensional one, explaining also the training process of the developed model. The feature extraction and processing step is the one described in section 4.2.1.

4.2.3.A One-dimensional Motif Discovery

Firstly, we focus on describing the procedure related with the discovery of motifs on the concerned one-dimensional features, as the original algorithm is not capable of handling multidimensional ones.

So, for each recording of the mentioned subsets we compute the matrix profile of the regarded sound attributes. Note that the input of the matrix profile is either the ZCR feature or one of the MFCCs. After a thorough analysis, we chose the first MFCC, out of the other 13, as input to this algorithm, which is a consequence of the aforementioned condition of this method. Once the annotated time series is computed, we carry out the motif discovery, concerning only the top motif or the top two repeated patterns. Finally, as each motif details its interval, we can extract the corresponding windowed feature, building both subsets with this methodology.

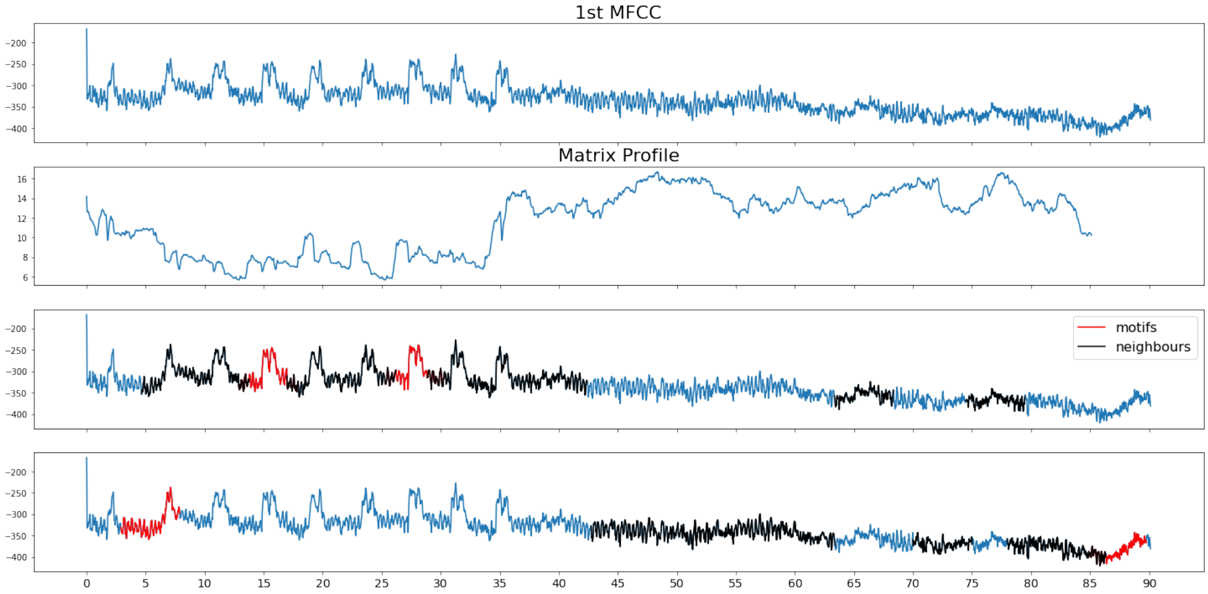


Figure 4.11: One-dimensional motif discovery example.

An example of this procedure is showcased in Fig. 4.11, where the first row corresponds to the regarded audio feature and the second represents its matrix profile. The final two rows match the top two motifs, found in the recording considered for this explanation. The red color portrays the motifs and the black color its neighbours, that is, the subsequences that are within a radius of 3 times the minimum distance (motif distance) using the regular matrix profile.

For each labelled record, the final result of the proposed methodology is a subset of the concerned audio feature, in which we only consider the part that corresponds to the discovered motif, having the same representation as the one presented in Fig. 4.9.

4.2.3.B Multidimensional Motif Discovery

As previously noted, the standard matrix profile algorithm does not support multidimensional features. In this regard, we leverage the multidimensional version Stumpy [1] to handle the MFCCs and Mel spectrogram features, and to compute their multidimensional matrix profile. This library implements methods such as the ”stumpy.mstump” function, which provide a highly efficient, accurate, and scalable multidimensional variant of the matrix profile definition found in Yeh et al. (2017b). Our work does not use the mSTAMP library concerned in the referenced article, as this implementation does not provide GPU oriented methods, for instances, as the Stumpy library does.

Firstly, it is important to demystify a common misconception concerning the multidimensional matrix profile, which stems from the idea that this technique might consist of one-dimensional matrix profiles stacked on top of each other. To do so, we introduce the following toy problem to better explain the algorithm's variant, having Yeh et al. (2017b) and the documentation of the Stumpy library as reference.

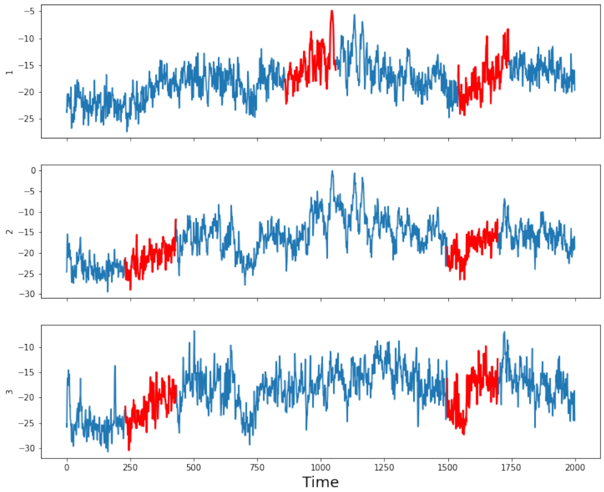


Figure 4.12: Multidimensional motif discovery toy problem.

[1] Stumpy

As represented in Fig. 4.12, it consists of 3 dimensions of the Mel spectrogram of a RFCx recording, that for simplicity was reduced to half of its duration. This multidimensional time series, $T = [D1,\ D2,\ D3]$, has $d = 3$ and $length = 2000$, having a shape of $d \times n$ (3×2000).

By choosing a window size of $m = 200$, for example, the i^{th} multidimensional subsequence can be defined as a continuous subset of values from T of length m, starting from position i and with an overall shape of $d \times m$ (3×200). Hence, one can incrementally slide the rectangular slice $l = 1801$ times $(l = n - m + 1)$, before reaching the end of T. In addition, for the i^{th} multidimensional subsequence, we can iterate over each of its dimensions independently and compute an aggregated multidimensional distance profile (i.e., three one-dimensional distance profiles stacked on top of each other). Essentially, the i^{th} multidimensional distance profile' shape is $d \times l$ (3×1801) and gives us the pairwise distances between the i^{th} multidimensional subsequence and all possible multidimensional subsequences within T.

$$i^{th}\, distance\, profile = \begin{bmatrix} 0.3 & 0.1 & 0.5 & ... & 0.2 & 0.7 & 0.9 \\ 0.8 & 0.3 & 0.2 & ... & 0.8 & 0.3 & 0.9 \\ 0.6 & 0.5 & 0.3 & ... & 0.2 & 0.1 & 0.4 \end{bmatrix} \tag{4.1}$$

Given that the values in the i^{th} column of the multidimensional matrix profile are directly derived from the i^{th} multi-dimensional distance profile, Equation 4.1 represents the common multidimensional distance profile for the i^{th} multidimensional subsequence. With this, we can now identify the set of d values that form the i^{th} column vector of the multidimensional matrix profile with shape $d \times 1$ (3×1). The value for the first dimension is found by extracting the smallest value in each column of the i^{th} distance profile and then returning the minimum value in the reduced set. Then, the value for the second dimension is found by extracting the two smallest values in each column of the i^{th} distance profile, averaging these two values and then returning the minimum averaged value in the reduced set. Finally, the value for the k^{th} out of d dimensions is found by extracting the k smallest values in each column of the i^{th} distance profile, averaging these k values and then returning the minimum averaged value in the reduced set. Consequently, by simply advancing the i^{th} multidimensional subsequence along the entire length of T and computing its corresponding i^{th} multidimensional matrix profile, we can easily populate the full multidimensional matrix profile. To sum up, this explanation supports the idea that the multidimensional matrix profile is not made up of one-dimensional matrix profiles stacked on top of each other.

Furthermore, it is important to remark that the lower dimensional repeated pattern(s) may or may not be a subset of the higher dimensional motif, since the lower dimensional motif pair can be closer than to any subset of dimensions in the higher dimensional motif pair.

Nevertheless, we have that the k-dimensional motif can be found by locating the two lowest values in the correspond k-dimensional matrix profile, which poses the problem of finding the ”correct” k value.

Consequently, this unconstrained search problem is reduced to selecting the best motif out of all possible k-dimensional motifs. As suggested in Yeh et al. (2017b), one can turn this into a classic elbow (or knee) finding problem, by visually or algorithmically locating the inflection point when plotting the minimum matrix profile value in each dimension for each k-dimensional motif.

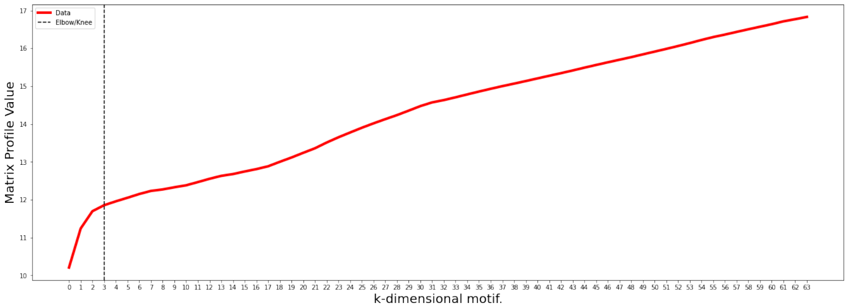


Figure 4.13: The matrix profile value for each k-dimensional motif.

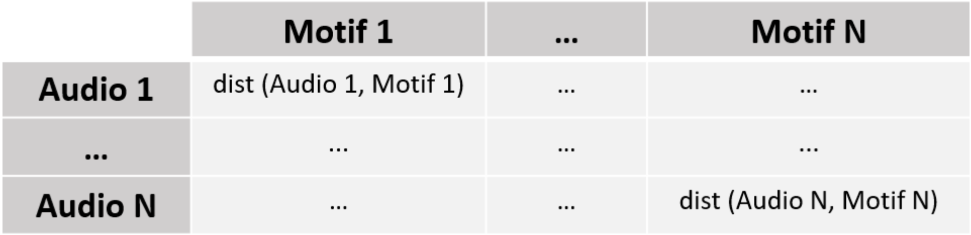
In line with the previous toy problem, Fig 4.13 is an example of the mentioned representation, as it showcases the matrix profile value in each dimension for each k-dimensional motif. The inflection point can be found automatically by using the *kneed* Python package.

4.2.3.C KNN Training

In this section we address the training of the K-Nearest Neighbors (KNN) classifier, from the motif discovery process to the construction of the training set that includes the extracted repeated patterns and serves as input to the developed model. In this paper, the concerned classifier considered the 3-nearest neighbors ($k = 3$).

Firstly, we extract the motifs subsets, following one of the previously introduced methodologies. Subsequently, for each recording we compute the respective audio feature, depending also on the chosen matrix profile version. The distance of each audio feature to each motif is measured according to a distance function which we present later in this section. Consequently, the training set assumes a tabular structure, as depicted in Fig. 4.14, which serves as input to the classification model

The premise of this approach lies in the fact that the classifier can learn the similarity, represented by the distance, between each recording and each motif, whether they are positive or negative, being able to classify each event accordingly. Thus, as represented in Fig. 4.14, the table's columns correspond to the motifs of the two subsets, which in practical terms end up merged, being the distinction between both a way to facilitate the problem's conceptualization. The table's rows comprise the recordings' features from the training or test set, depending on whether we are training or testing the developed model. The values in each cell correspond to the distance between each recording and each motif's feature.



	Motif 1	...	Motif N
Audio 1	dist (Audio 1, Motif 1)	...	...
...	...	...	...
Audio N	...	...	dist (Audio N, Motif N)

Figure 4.14: Motif based training set.

Before delving into the proposed distance functions, it is important to get acquainted with the particularities of the introduced approach. In depth, both the recording and the motif's features assume a multidimensional shape, because they both correspond to the MFCCs. Hence, a distance function capable of handling multidimensional features is required.

In the first proposed distance function, the distance is given by the difference between the norm of the recording's feature and the motif's attribute, as presented in the following equation:

$$distance = norm\ (Audio\ N) - norm\ (Motif\ N) \tag{4.2}$$

With the ”numpy.linalg.norm” method, of the ”NumPy” library, one can compute the norm of the recording's feature or the motif's attribute.

The second approach takes advantage of the ”Stumpy” library, used to compute the multidimensional matrix profile, to compute the z-normalized matrix profile distance measure between the recording's feature and the motif's attribute. Note that the only method of this library capable of handling the multidimensional features is the one that computes the matrix profile. In this regard, despite several attempts, we were not able to attain a meaningful distance between the recording's attribute and the motif's feature when using this method.

Also, after an extensive search we did not find more methods capable of obtaining the aforementioned distance, being a topic to address in further research. To conclude, it is important to stress the limited number of multidimensional matrix profile's implementations, as this study is rather recent, restricting the developed solution and being, alongside with the distance function, the bottleneck of this approach.

4.2.3.D Overview

Despite considering other audio attributes, the presented end-to-end pipeline describes the feature extraction and processing steps required to transform the raw audios into the MFCCs through the proposed window function. Moreover, it details the training of a LSTM network with the extracted cepstral features.

As the results obtained by the developed network were considerable worse than the ones attained by the spectral based one, we complemented this methodology with an additional step. In depth, we trained a KNN classifier with the motifs extracted by the matrix profile algorithm. It is also important to mention that we studied the influence of the motifs extracted by the two matrix profile implementations, namely the standard and the multidimensional one, on the developed classifier.

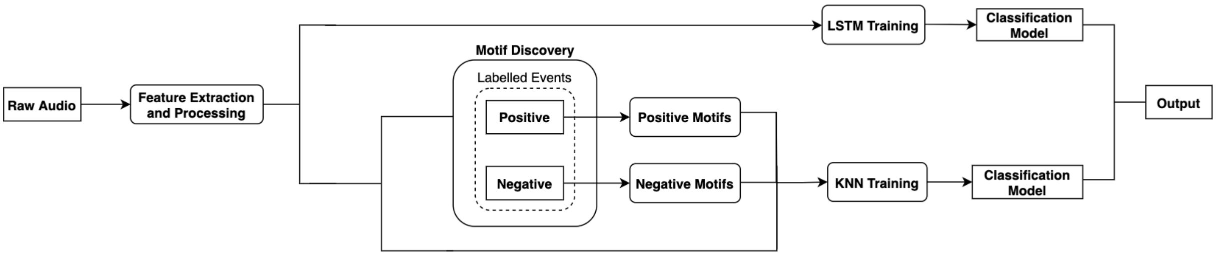


Figure 4.15: Ensemble approach flowchart.

Finally, as depicted in Fig. 4.15, we combine the output of the two models presented in this section to determine if the classifier trained with the motifs helps improving the attained results. Besides evaluating the performance of the proposed ensemble approach, we also assess the performance of the two proposed models, individually. All in all, the cepstral based approach needs further research and development as the classification models of this framework achieve worse performance than the ones concerned in the spectral based methodology.

5

Experimental Results

Contents

5.1 Case Studies

5.1.1 Kaggle Competition Dataset

The ”Rainforest Connection Species Audio Detection” [1] is a Kaggle competition that provides 6786 files: 64 TensorFlow records (.tfrec), 3 files which summarize the data (.csv) and 6719 audio files (.flac) that include sounds from numerous species.

The proposed bioacoustic classification framework is evaluated on these audio files, which were collected from about 700 sampling sites across the mountains of Puerto Rico at a sampling rate of 48 kHz with 24 kHz bandwidth, following a schedule of 1-minute audio recording every 10 min, as described in Zhong et al. (2020). In detail, the competition concerns the classification of 24 bird and amphibian species which inhabit the tropical mountains. It provides two distinct files to the competitors: the first has data about the true positive events registered in all recordings, having a labelled interval which refers to the specie call; the second has data about the false positive events, detailing by opposition the intervals where a certain specie does not appear. Furthermore, both files also provide data about the specie present in the audio sample, the sound´s song type as well as the frequency and time interval of the event.

Finally, it is important to note that the 6719 audio files are divided into two datasets, the first constitutes the training set, and whose information is summarized on the two mentioned above files, and the second encompasses the recordings that form the test set, which are meant to evaluate the developed solution.

5.1.2 Chainsaw Dataset

This dataset refers to the data provided by Hitachi Vantara through its partnership with Rainforest Connection. It also includes the VisBig project (PTDC/CCI-CIF/28939/2017), being important to remark that both connections enable more data to be considered. Despite that, our work only considers recordings from January 2020, having the concerned dataset 6567 audio files (.flac and .wav). These particular files have been preprocessed, a procedure in which files smaller than 1.1MB and with sampling rates lower than 12,000 Hz were filtered out. Also, the remaining audios are approximately 90 seconds-long.

Nonetheless, we consider a subset of these recordings, as only 1091 of these audio files possess annotations regarding chainsaw events. The labels were obtained by a manual confirmation process that validated the output of a model, developed by Huawei, that detects chainsaw events. In detail, each labelled recording can encompass multiple events, registering a total of 7885 confirmed and 3274 rejected chainsaw events, each one annotated with the corresponding event’s time interval.

[1] Rainforest Connection Species Audio Detection

5.2 Experimental Results

5.2.1 Spectral Based Model - Kaggle

There are 24 annotated species in the provided dataset, which would suggest a 24 multi-label classification setting. Nevertheless, two species have more than one song type, having both type 1 and 4, revealing the need of two additional labels. As a starting point, the created training set disregards the song type 4 for the mentioned species.

In this sense, our approach transforms the 24 multi-label classification setting into 24 distinct classification problems, where in each we train a model so that it can learn the presence or absence of a given specie. Moreover, the upcoming sections describe several experiments in which the concerned models follow the architecture described in section 4.1.2. In particular, note that the experiments' results represent the average of each specie related model's score, taking as an example the scores displayed in Fig. 5.1, that refer to the average of the accuracy scores across all 24 species. It is also important to remark that from these results, the ones presented in section 5.2.1.B refer to each specie related model as this analysis discriminates all species.

Also, the baseline training set includes the maximum number of true positive events for each specie, that for the majority of species corresponds to approximate 50 samples. Additionally, other variants may encompass different quantities of true positive augmented samples, as further described in section 5.2.1.A. Lastly, a subset of the available 350 false positive samples is extracted, for each specie, in the same quantity as the true positive subset, that may contain augmented instances. For example, if we complement the baseline approach with data augmentation, one specie that has 50 true positive samples, will also have 50 augmented samples and 100 false positive samples, resulting in a balanced training set for each specie.

5.2.1.A Window Size and Data Augmentation

The first approach aims to assess the effect of different window sizes and data augmentation techniques on the performance of each model. Thus, the considered frame lengths were 2, 5 and 10-second-long, as more than 80 percent of the recorded events have intervals smaller or equal to 4 seconds. Also, by including the 2 second window one can verify if smaller frames can capture enough image traits to conduct automate species detection. In addition, for each window size, we trained a model with a training set that did not include augmented samples (baseline) and compared it to two models whose training set contained samples augmented by the two techniques described in Section 4.1.2.C.

As detailed in the previous section, the training set that does not takes advantage of data augmentation techniques includes the maximum number of available true positive samples, having the same number of negative samples. Conversely, both training sets with augmented instances, from the two aforementioned augmentation techniques, differ from the latter by having augmented samples in the same number as the true positive calls, thus enabling the use of more negative instances. Finally, the followed evaluation metrics were *accuracy* 17, *precision* 19, and *recall* 18, being the test set classified with a threshold score of 0.6. Once again, it is also relevant to stress that the evaluation metrics represent the average of the scores of each individual model, excluding those who fail to learn the distinguishing characteristics of the audio features.

As depicted in Fig. 5.1, by including the augmented samples in the training set we increased the accuracy scores across all windows sizes. The model trained on the 10-second-long window failed to capture the data's variability, leading to the worse results in terms of precision and recall. The 5 second window obtained a significant accuracy increase, registering the best precision and recall score (0.77 and 0.78) with the SpecAugmented spectrograms. Furthermore, the smallest concerned frame obtained similar results in comparison to the 5 second window in terms of accuracy, achieving, nevertheless, lower precision and recall scores.

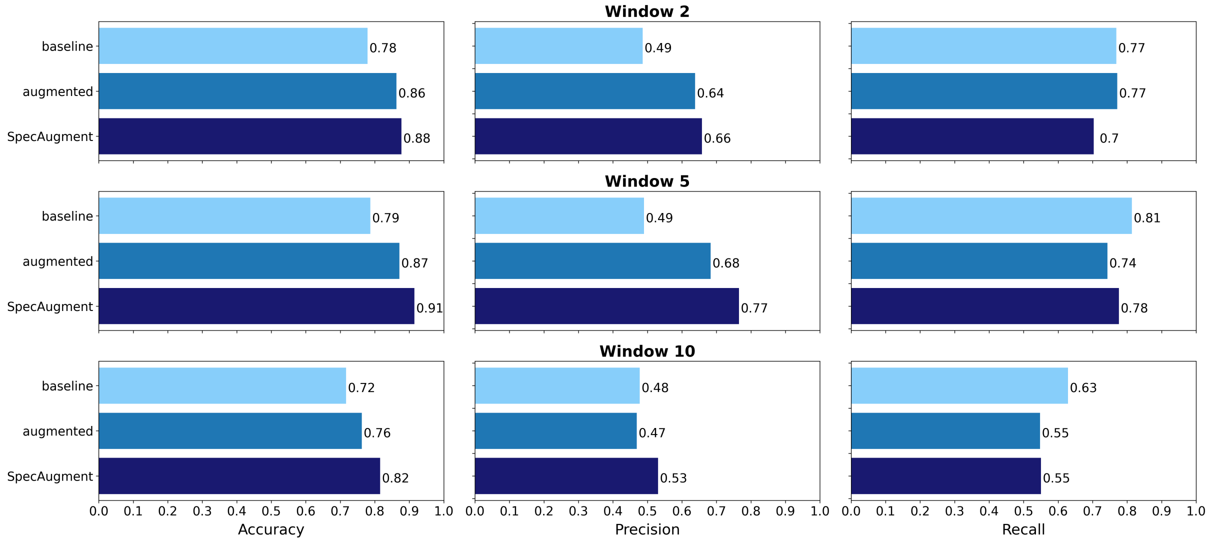


Figure 5.1: Effect of different window sizes and data augmentation techniques on accuracy, precision and recall.

All in all, the results confirm the well-known precision-recall relation, in which generally an increase in precision leads to a decrease in recall, and vice-versa. Consequently, a balance is desired if false positives and false negatives are equally significant, which is not the case in our problem's spectrum as recall is slightly more important because false negatives are more costly. From this experiment, both the 2 and 5-second long frames seem to be able to capture the distinctive traits of each Mel spectrogram. Nonetheless, as the model trained on the 5-second-long windows performed slightly better, this is the frame length concerned from this point forward.

5.2.1.B Dynamic Window Sizes

The results obtained in the previous section are strongly marked by the models that fail to differentiate both classes, difficulty amplified with the 10 second frame. So, in order to assess if each model would perform better with a tailored window size, a different approach was experimented. More concretely, each specie related spectrogram was obtained by taking into consideration the mean time interval of each specie call with a one second margin, which implied that, for instances, a specie with an average interval call of 2 seconds would have a 3-second-long Mel spectrogram.

Hence, the average-precision, presented in the definition 20, was the metric used to compare the model trained on the dynamic windows with the one trained on the fixed window size (5 seconds). Also, both models had recordings augmented with the SpecAugment method.

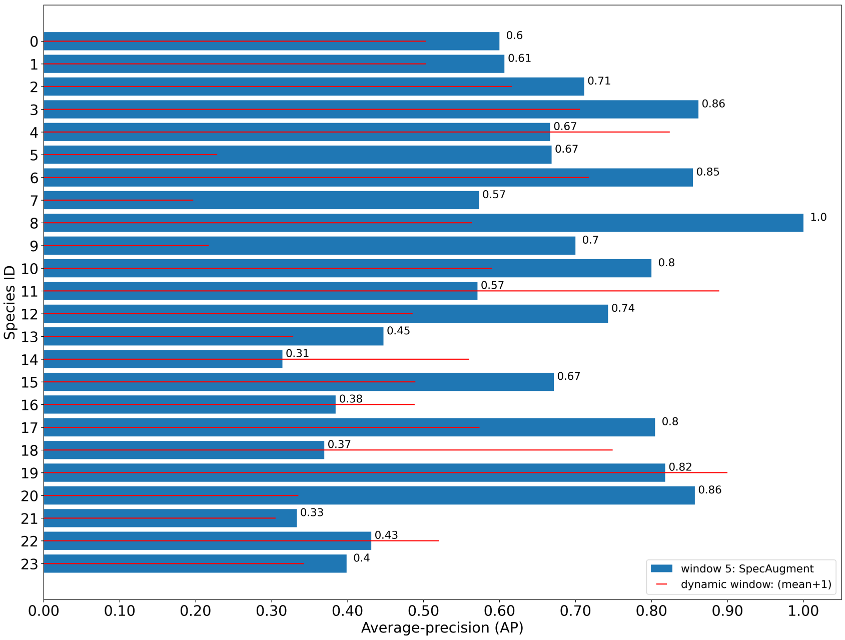


Figure 5.2: Average-precision of dynamic and fixed window size approaches.

In particular, Fig. 5.2 demonstrates the difference in average-precision between each model trained on the dynamic window size (red) and those trained on the 5 second window (blue). The mean average-precision scores for the dynamic and fixed size approach are 0.52 and 0.63, respectively. Furthermore, species with a mean window size smaller than 5, such as 11 and 18, for instances, are the ones who benefit the most from the dynamic approach. Also, it is possible to understand the impact that models with lower scores have on the metrics depicted in Fig. 5.1. Lastly, it is important to note that the difference in the training size, that for species 2, 9, 17, 20 and 22, is significantly smaller due to the available positive samples, is not the only cause for a poorer model's performance, as the average-precision of specie 17 is higher than the one of specie 23, for example. To sum up, the goal of this approach was to understand if a small combination of window sizes, as a large one would be extremely costly in the prediction step, would favor the model's results. Despite the aforementioned improvement on the species that register smaller calls, the overall performance was not sufficient to justify the cost that a windowed approach would require.

5.2.1.C Predictive threshold

The predictive threshold represents the probability value by which a given sample is classified, that is, if the probability returned by the model is superior to the defined threshold the sample will be classified as belonging to the class, and vice-versa. On that account, the previous experiments considered a predictive threshold of 0.6, achieving a precision of 0.77 and a recall of 0.78 with the best performing model. Nonetheless, one can try to improve the precision score by increasing the predictive threshold.

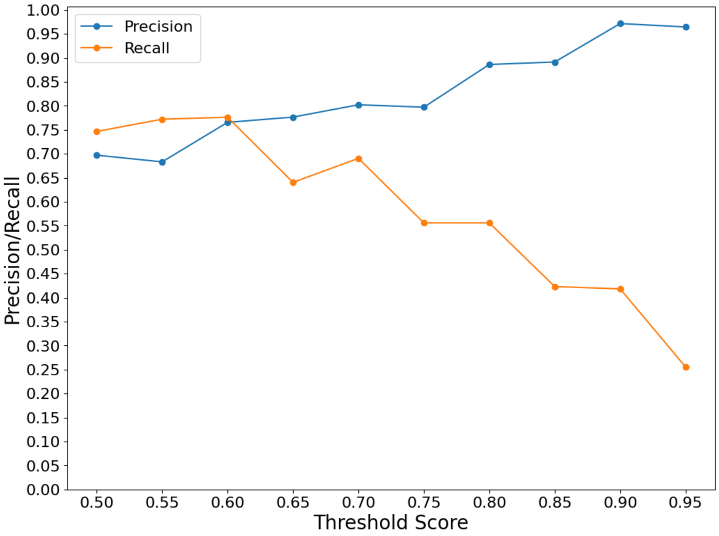


Figure 5.3: Precision/Recall threshold curve of the model trained on a 5-second-long window with SpecAugmented samples.

Hence, Fig. 5.3 displays the mean precision and recall variation with different thresholds, so that the influence of the threshold value on the obtained results could be determined. The increase in the threshold value leads to higher precision scores, nevertheless, this increment also results in a significant decrease in recall. The precision-recall balance is achieved somewhere between the 0.50 and the 0.65 predictive threshold value, with the 0.60 threshold registering the optimal value for the develop approach, with a precision of 0.77 and a recall of 0.78.

Following the prior analysis, we analyzed the confusion matrices of 3 different thresholds (60%, 75% and 90%) to better comprehend the increase in precision and the decrease in recall, as we incremented the predictive threshold value. These matrices are represented in Fig. 5.4 and explain the mentioned behaviour. Firstly, the reduction in the true positive explains why the balance between these metrics is not found in the higher threshold values, as a stricter threshold value reduces the ability of correctly classifying the positives events of a given class. Regarding recall, the growth of the number of false negative samples explains the abrupt reduction of this score. Oppositely, the drop in the false positives samples justifies the rise of the precision score.

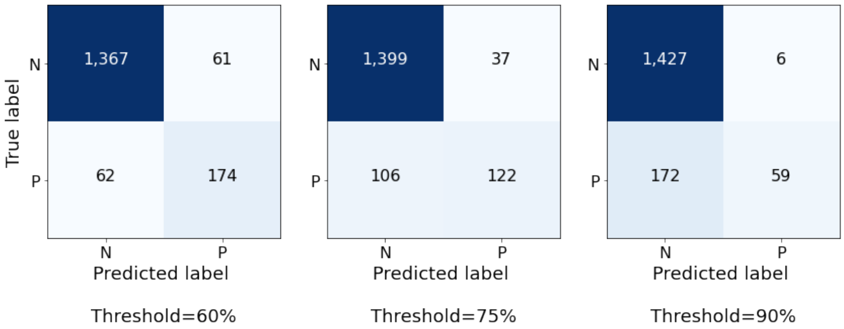


Figure 5.4: Confusion Matrices of 3 predictive thresholds (60%, 75% and 90%).

5.2.1.D Convolutional Neural Network combined with a Long Short-Term Memory network

As referenced in section 3.7, similar classification problems were addressed with a hybrid architecture, that combined Convolutional Neural Networks with Long Short-Term Memory networks. In this sense, we complement the previous architecture with an LSTM layer, as an attempt to improve the general performance of the developed model.

The network architecture described in 4.1.2.B stems from several experiments in which we tried to establish the optimal combination to the problem at hand. In depth, we started by adding an LSTM layer with 512 neurons between the pretrained model and the fully connected layer with 512 neurons. Despite being the initial experiment, it remained the best performing one, achieving an accuracy score of **94%** and a precision and recall score of **83%** and **84%**, respectively.

According to our experiments, an increase in the number of LSTM's neurons led to a scenario where we ended up with higher recall scores and slightly lower precision scores, such as 86% and 80%, for example. Conversely, an increment in the number of neurons of the fully connected layer resulted in lower precision (82%) and recall scores (79%). Finally, we also tested multiple settings where we tried several combinations of LSTM and fully connected layers, nonetheless none of them improved the results from the best performing one.

5.2.2 Spectral Based Model - Chainsaw

In view of the results attained in the previous sections, we evaluated the proposed framework on the dataset, previously introduced in section 5.1.2, that includes the recordings labelled with the chainsaw events. The main difference to the annotations concerned in the previous dataset (5.1.1) lies on the information regarding the event's frequency interval, as the labels from this dataset do not detail the mentioned interval. Thus, as we analysed the Mel spectrograms of the different recordings we noticed that chainsaw events, for the most part, took place in the lower frequencies. Despite being possible to find other animal sounds in this frequency range, we also observed that events such as bird sounds, would generally assume higher frequencies in comparison to the chainsaw sounds. In this sense, as our goal is to detect chainsaw events, we reduced the previous sampling rate of 48 kHz to 22kHz since there was no need to concern such high frequencies when training our model, imposing also a minimum and maximum frequency of 0.08 and 3kHz, respectively, for the extracted Mel Spectrogram.

Furthermore, the amount of available labelled recordings is considerable larger, in comparison to the previous dataset, favoring the model greatly as it allows for a bigger training set. However, due to hardware limitations we restricted the training set to 1600 positive and 1600 negatives samples. So, apart from the aforementioned modifications, the training process of this particular classification model was similar to the one described in section 4.1.

5.2.2.A Transfer Learning and Fine-Tuning

The transfer learning setting enables the use of multiples models which can be used for prediction, feature extraction, and fine-tuning. The ResNet50 network was the selected to complement the developed baseline architecture, as it is often utilized in similar image classification problems, being the model used as a start point in numerous solutions. Nevertheless, this section aims to compare the performance of the previously mentioned model with other networks that were also trained on the ImageNet dataset.

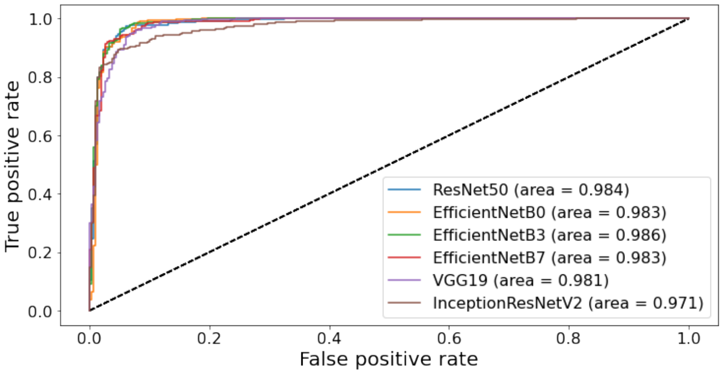


Figure 5.5: Comparison between the performance of different pretrained models.

Consequently, we experimented 3 other pretrained networks, EfficientNetB0, InceptionResNetV2 and VGG19, as the backbone of our proposed architecture. We then compared the obtained results with the ones attained with the initial architecture, which comprised the ResNet50 model. The used training set was the one described in the above section (5.2.2), being showcased in Fig 5.5 the comparison between the aforementioned models' performance.

The different networks present similar AUC (area under the ROC curve), being the Resnet50 model the best performing one, along with the EfficientNetB3. Even though the ResNet50 network (as in 50 weight layers) is much deeper than VGG19, its model size is substantially smaller due to the usage of the global average pooling layers rather than the fully-connected ones, which makes it preferable to the latter.

Furthermore, it is also possible to compare all models against the family of EfficientNets (EfficientNetB0 to EfficientNetB7), considering only, for simplicity, the B0, B3 and B7 variants. This network introduces a new Scaling method called Compound Scaling, as opposed to the one used by models such as the ResNet50, that follow the conventional approach of scaling the dimensions arbitrarily and adding up more and more layers. This method proposes that if we scale the dimensions by a fixed amount at the same time and do so uniformly, we achieve much better performance. The performance of the EfficientNetB0 network is very similar to the ResNet50 one, being also possible to notice that the use of other EfficientNet variants does not increase significantly the obtained results.

Also, the combination of the Inception architecture with residual connections, present in the InceptionResNetV2 network, does not justify the use of this specific architecture as it does not register a better performance when comparing with the ResNet50 network.

Hitherto, the layers from the pretrained models were frozen, that is, they were not trained during the training process to avoid destroying any of the information they contained. Nevertheless, in this setting one can take one last optional step, referred to as fine-tuning, that consists of unfreezing the entire model, or a part of it, and retraining it on the new data, with a very low learning rate. Despite multiples attempts, we were not able to attain better results when fine-tuning our model, as all our experiments ended up with poorer performance.

5.2.3 Cepstral Based Model - Chainsaw

As stated in section 4.2, this approach aims to provide an alternative to the spectral based classification model (4.1), by exploring a different network architecture, namely the Long Short-Term Memory network, and different audio features. In particular, it focuses on features such as the root mean square (RMS), a reliable indicator for silence detection, the zero-crossing rate (ZCR), useful for discriminating periodic signals from those marked by noise, to understand if it is possible to perform biacoustic classification without all the processing related with an image-based approach. Moreover, we also explore the MFCCs as this feature is widely used in similar problems, as previously referenced.

The first experiment sets the baseline for the concerned approach, as it compares the performance of the LSTM network when trained with the different referenced audio features. Additionally, it follows the windowed approach introduced in section 4.1.1. Given the results presented in Fig.5.6, it is possible to conclude that the multidimensional feature (MFCCs) outperforms the one-dimensional ones, obtaining a similar performance to the spectral based approach, in this particular setting.

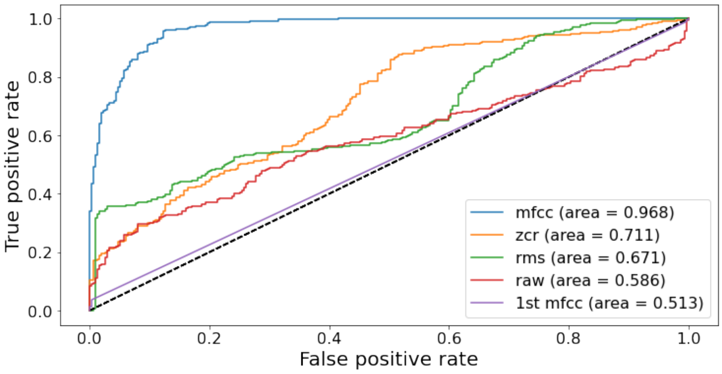


Figure 5.6: LSTM network's performance with different audio features.

In addition, the networks trained with the raw recordings and with one of the MFCCs failed to learn the unique properties of the labelled events, being the worse performing models. Also, the models trained with the RMS and the ZCR features registered a significant improvement in performance when comparing to the ones trained on the aforementioned attributes, being, nonetheless, quite far from achieving similar results as the ones from the network trained on the cepstral features.

All in all, given the results, from this point forward we will only concern the MFCCs and the ZCR attribute, being the latter considered as a way of confirming the above results, since we also evaluate the framework on the Kaggle dataset.

5.2.3.A Motif Discovery using the Matrix Profile

This section expands the research referenced in sections 3.3 and 3.4, that describes the use of the matrix profile algorithm to conduct motif discovery on audio features. In this regard, we carry out several experiments that test different approaches regarding the discovery of the repeated patterns. Thus, the following analysis explores the methodology that best suits the motif discovery process in environmental audios, namely with the recordings which encompass the labelled chainsaw events (5.1.2). Our goal is to improve the performance of the developed model by using the extracted motifs to augment and complement the available training set.

5.2.3.B One-Dimensional Motif Discovery

Firstly, we evaluate the algorithm's ability to find motifs with different audio features. So, we start by considering a subset of 100 files from the dataset described in section 5.1.2, that only concerns confirmed chainsaw events to better comprehend the features that favor the discovery of the repeated patterns. The algorithm's method responsible for the motif discovery by default finds the top 3 motifs and up to 10 of their neighbours, that is, the subsequences that are within a radius of 3 times the minimum distance (motif distance) using the regular matrix profile. Nonetheless, we limited this analysis to the top 1 and 2 motifs as we only want to consider the most distinctive patterns of the concerned class.

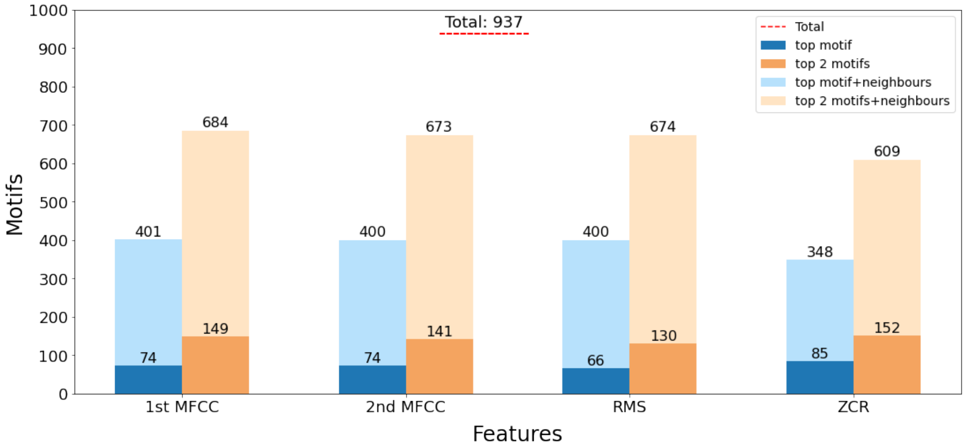


Figure 5.7: One-dimensional motif discovery: top 1 and 2 motifs with different audio features.

As depicted in Fig. 5.7, all attributes present a similar number of found patterns, stressing however the slightly smaller value for the RMS feature when considering the top motifs. The results suggest that all four are capable of identifying the labelled motifs, nevertheless, from this point forward we will perform motif discovery with the MFCCs, as they seem to achieve the more balanced results in both analysis. When analysing the top 1 and 2 motifs, the ZCR attribute registers the highest number of found patterns in both cases, and, by opposition, the RMS feature obtains the lowest value in both analysis.

Furthermore, when we concern the motif's neighbours, the feature that finds the most repeated patterns is the first MFCC. Also, we analysed the distribution of the found motifs to determine if at least one repeated pattern was detected in each recording. With the top motif, all 4 attributes manage to find motifs in approximate 50 files, whereas with the top 2 motifs, the number increases to 71. In both cases, when we consider the motif's neighbours we are able to find motifs in almost all files.

Afterwards, we applied the methodology described in section 4.2.3 to the training set used to train the developed models. In detail, we divided the labelled events into two groups, the one that refers to positive events and the one related with the negative ones. For the two groups, we extracted the top motif from each of their recordings, ending up with the motifs that describe the positive events and the ones that refer to the negative ones. Despite being possible, in this setting, to extract a large amount of motifs to complement the training data, we only consider a subset of 100 repeated patterns per group, mainly because our goal is to improve the model's performance when the available training data is limited, as in the case of the Kaggle dataset (5.1.1).

With the two motif subsets, the training set is created by following the previously referenced procedure. In particular, the training set assumes a tabular structure, having in the columns the features of the two motif groups, that in practical terms end up merged, being the distinction between both subsets (positive and negative) done to make the problem's conceptualization clearer for us. The rows include the recordings' features from the training or test set, depending on whether we are training or testing the developed model. The values in each cell correspond to the distance between the respective motif and the recording's feature, where the distance function is one of the introduced in section 4.2.3.C. However, in this section, we only assess this approach with the first proposed distance function, that computes the difference in the motif and recording feature's norm.

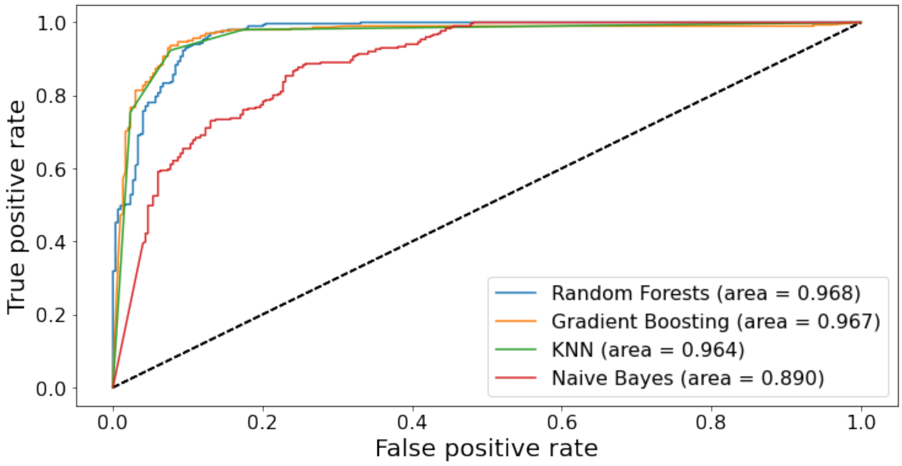


Figure 5.8: Comparison between the performance of different classifiers trained on the motif based approach.

Hence, we trained several classifiers, namely Random Forests, Gradient Boosting, K-Nearest Neighbors (KNN) and Naive Bayes, on the mentioned training set and compared their performance on the test set, as Fig. 5.8 showcases. The Random Forests, Gradient Boosting and the KNN classifier stand out in comparison to the Naive Bayes, being the latter the worse performing one. Given the attained results, we decided to choose the KNN classifier as the model concerned in our framework due the nature of our problem. In detail, as we want to classify each recording according to its closest motif, in a sense, we are overlapping the concept of the closest neighbour. Consequently, further research will consider this classifier.

Finally, we compared the KNN classifier, whose training process included the motifs, with the LSTM network trained on the MFCCs. Additionally, we compared both classifiers with a third approach where we consider an ensemble of the two developed models, that is, in the prediction step this solution takes into consideration the output of both methods. Moreover, the ensemble only concerns 50% of each classifier's prediction, despite being possible to use other combinations.

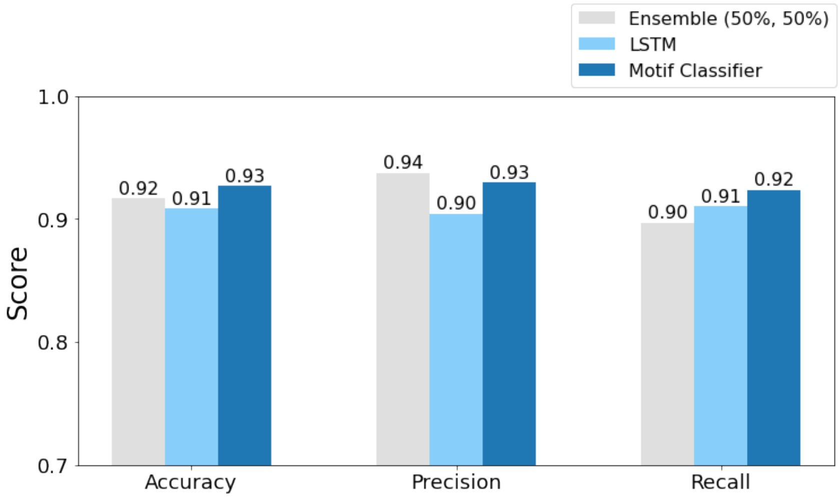


Figure 5.9: Comparison between the KNN (Motif Classifier), the LSTM network and the Ensemble classifier.

As Fig. 5.9 demonstrates, the difference between the 3 procedures is relatively small, being the best performing model the KNN classifier that was trained on the motifs. Moreover, in the carried out analysis we introduce the baseline ensemble approach, that can be the focus of further research. In this particular setting, this classifier was able to attain similar results as the ones from the two other models.

5.2.3.C Multidimensional Motif Discovery

Apart from the previously introduced procedure, that concerns the one-dimensional motif discovery, we also tested the proposed multidimensional methodology. In this setting, the matrix profile algorithm is capable of searching the repeated patterns in the whole set of feature's dimensions, instead of considering just one of them.

Thus, we compute the multidimensional matrix profile for each recording feature's, identifying the k-dimensional motif as explained in section 4.2.3.B. In depth, the k-dimensional motif is found by transforming the problem into a classic elbow (or knee) finding one, where we locate the inflection point, when considering the minimum matrix profile value in each dimension for each k-dimensional motif. Further research can focus on different approaches to address this problem, nevertheless, our work only explores the mentioned one.

Once the k-dimensional motif is revealed, we consider only the MFCCs' subset that encompasses the repeated pattern's interval, as in the one-dimensional approach. We repeat this process for each recording, creating the subsets introduced in section 4.2.3, used to build the training set that enables the training of the developed model.

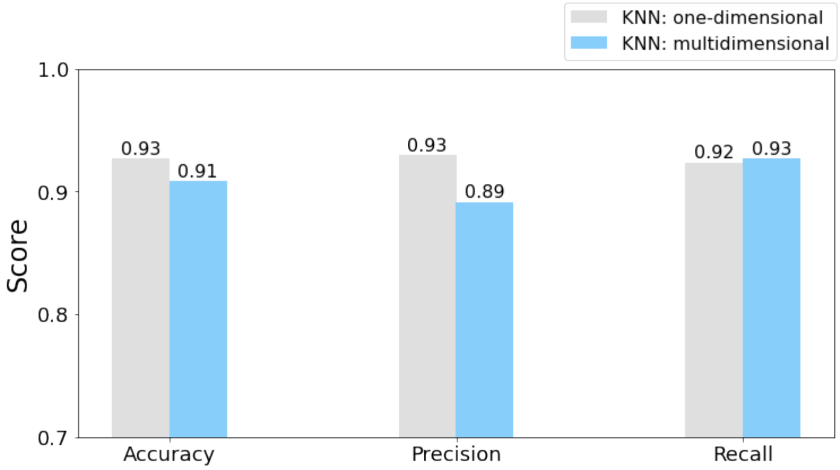


Figure 5.10: Effect of the one-dimensional and multidimensional motif discovery on the KNN classifier's performance.

In Fig 5.10 we compare the effect of the one-dimensional and multidimensional motif discovery on the classifier's results. The attained results favour the one-dimensional motif discovery as it performed better than the multidimensional one. In detail, it achieved higher accuracy and precision scores, having, however, a worse recall score, despite not having a really significant difference. All in all, the chainsaw dataset benefits the most from the KNN classifier that is complemented by the one-dimensional motif discovery.

5.2.4 Cepstral Based Model - Kaggle

In light of the results described above, the classification model presented in this section, was obtained by following two different approaches. The first trains each specie related LSTM network with the MFCCs and the second uses the ZCR attributes instead.

5.2.4.A Window Size

LSTM's networks can keep track of arbitrary long-term dependencies in the input sequences, thus, in this sense, the time component can play a major part on the outcome of the developed solution. So, we complemented our research with the study of the window size's effect on the model's performance, as we did in section 5.2.1.A. In particular, we want to determine if this type of network benefits more from longer frames or smaller ones.

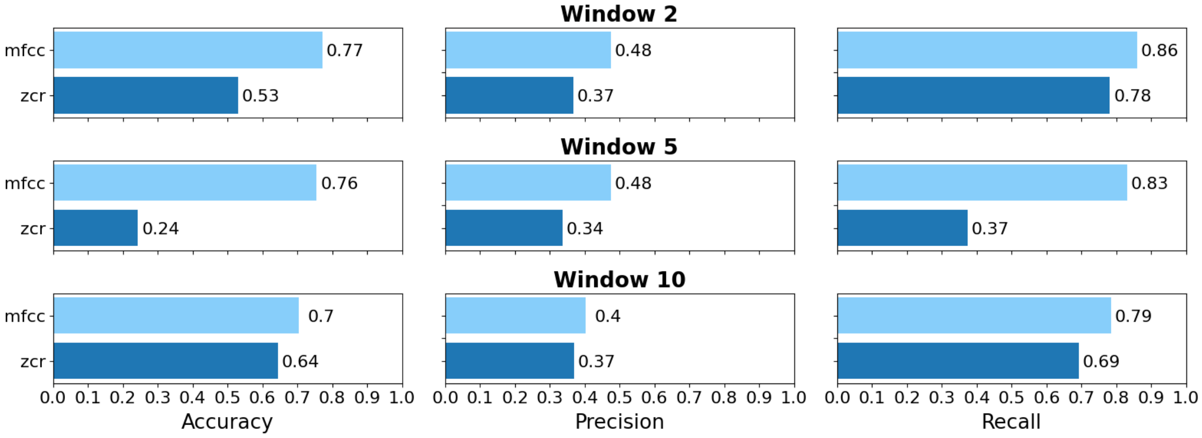


Figure 5.11: Effect of different window sizes on the LSTM's accuracy, precision and recall scores.

The results, displayed in Fig. 5.11, reveal the discrepancies in relation to the previous experiment, as the networks trained on this dataset have lower scores in comparison to the ones obtained by the models trained on the chainsaw one. Also, it is important to remark that the starting point of this analysis is significantly worse than the one described in section 5.2.1.A.

Moreover, from all the networks trained with the different frame lengths, it is possible to conclude that the ones trained with the MFCCs achieved better results. When concerning only this feature, the accuracy scores were very homogeneous, with a slight decrease in the one attained by the network trained with the 10-second-long window. Oppositely, when regarding the ZCR feature, the model who stands out in terms of accuracy is the one which considered the 10-second-long frame, as it achieved the highest score.

In terms of precision, all models got similar results when regarding the same feature. However it is important to mention that we registered multiple specie related networks that failed to distinguish both classes, when considering the models trained with the 5-second-long ZCR features, as their output was made only of negative instances, a case which we do not consider.

In relation to recall, it is possible to observe that all models that concern the ZCR feature achieve lower scores in comparison to the ones trained with the MFCCs, noting also the considerable low result of the network trained with the 5-second-long ZCR attribute. In addition, the recall scores of the models trained with the MFCCs were very similar, with the 2 and 5 frame lengths standing out in relation to the other.

To sum up, when concerning the cepstral features, the 2 and 5-second-long window sizes are the frame lengths which favour the learning capability of each model, similarly to in section 5.2.1.A. Nevertheless, there is a significant gap between the performance of the cepstral based approach and the spectral one, as the first attained worse results. As a consequence, further research will attempt to improve this approach, focusing on the network trained with the 5-second MFCCs, as the difference to model trained with the 2-second-long frame is relatively small.

5.2.4.B Predictive Threshold

Following the results of the above section, it is possible to notice the considerable difference between the recall scores and the precision ones, being the first substantially higher. Thus, as introduced in section 5.2.1.C, one can attempt to diminish this gap by changing the predictive threshold value, that for the previous analysis held a value of 60%.

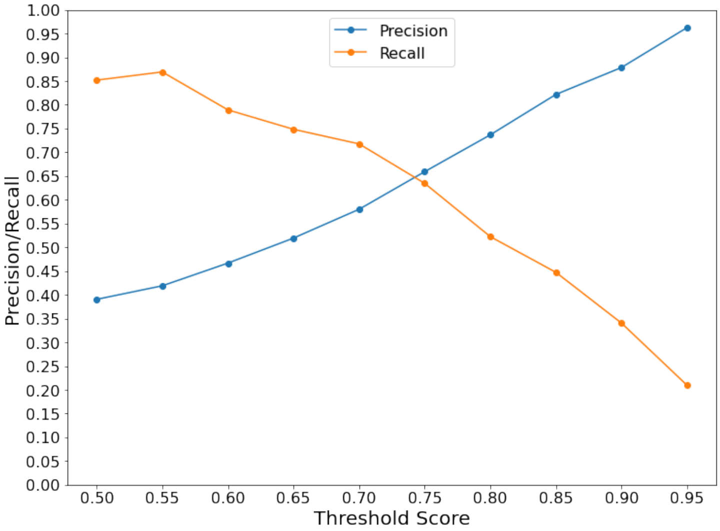


Figure 5.12: Precision/Recall threshold curve of the LSTM network trained with the 5-second-long MFCCs.

From Fig. 5.12, it is possible to observe the effect that the predictive threshold holds on the precision and recall scores. As previously noted, our experiments reveal that an increase in the threshold value leads to higher precision values and to lower recall ones. Moreover, as opposed to section 5.2.1.C, the balance in both scores is not attained with a 60% threshold value but with a 75% one, achieving a precision of **66%** and a recall of **63%**. Thus, this is the predictive threshold considered from this point forward, in this approach, as it is the one that favours the developed model, as it increases the precision score without compromising immensely the recall one.

5.2.4.C Motif Discovery using the Matrix Profile

As previously noted, this section complements the research made in section 5.2.3.A and applies the methodology described in section 4.2.3 to the Kaggle dataset. In detail, we seek to improve the results obtained in the previous section (5.2.4.B) by improving the effectiveness of the models trained with the small training sets sizes.

Given the results obtained with the chainsaw dataset (5.1.2), we started by applying the procedure used in that particular problem. In depth, for each specie we extracted the two subsets of motifs, associated with the presence and absence of a given specie. Analogously, the training set of each model, composed with the 5-second MFCCs of each recording, was transformed so that it could include the insights present in the repeated patterns. So, it assumed a tabular shape, as depicted in Fig. 4.14, where each cell stored the distance of a given recording's feature to a certain motif.

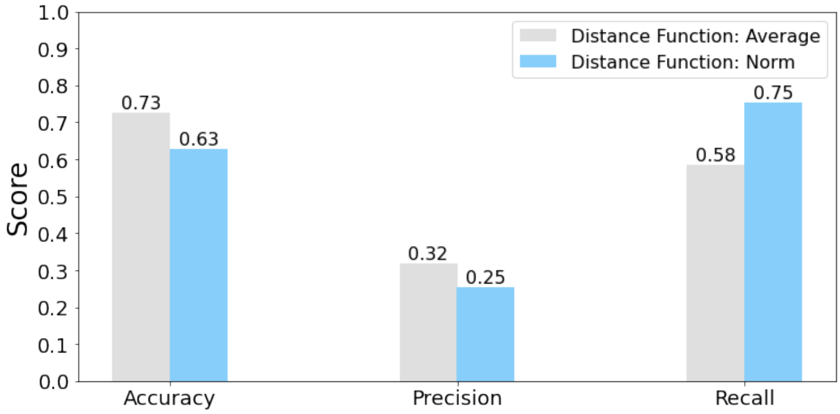


Figure 5.13: Distance function's effect on the KNN classifier.

Initially, the followed distance function was the one also used with the chainsaw dataset and introduced in section 4.2.3.C, that computes the difference in the norm of the motif and the recording's feature. Nonetheless, this approach did not improve the model's performance, suggesting that the concerned distance function was not able to capture correctly the similarities (or dissimilarities) between a given feature and a given motif. This learning difficulty stems from the fact that the introduced method must be capable of computing the distance between two multidimensional features, the recording's and the motif's one. As reported in the referenced section, the limited amount of methods capable of computing such distance poses a bottleneck to the developed solution, so, in this sense, our work encompasses the two distance functions that we were able to apply to the problem at hand.

In particular, Fig. 5.13 represents the average of the scores obtained by all the specie related models, while using the two different distance functions. When we compare the scores achieved by the KNN network trained with norm distance function in both datasets (chainsaw and Kaggle), it is possible to conclude that the ones presented in this section are substantially worse. The precision score is the most affected metric, achieving a score of 25%, contrasting with the recall one which registered a value of 75%.

Due to the obtained results, we tested a different distance function that relies on the ”Stumpy” method to compute the z-normalized matrix profile distance measure between the recording’s and the motif’s feature, being both one-dimensional. Note that the only available method, of the researched multidimensional matrix profile implementations, capable of handling the multidimensional features is the one responsible for the multidimensional matrix profile’s computation. Despite several approaches, we were not able to obtain a meaningful distance between the recording’s attribute and the motif’s feature when using the mentioned matrix profile variant. Consequently, as described in section 4.2.3.C, the second proposed distance function uses the mentioned method to compute the z-normalized matrix profile distance between each audio’s attribute and each motif’s feature dimension, being the final value the average of distances of all the dimensions. The introduced distance function helps the network achieving higher accuracy and precision scores, reducing however the recall one, as displayed in Fig. 5.13.

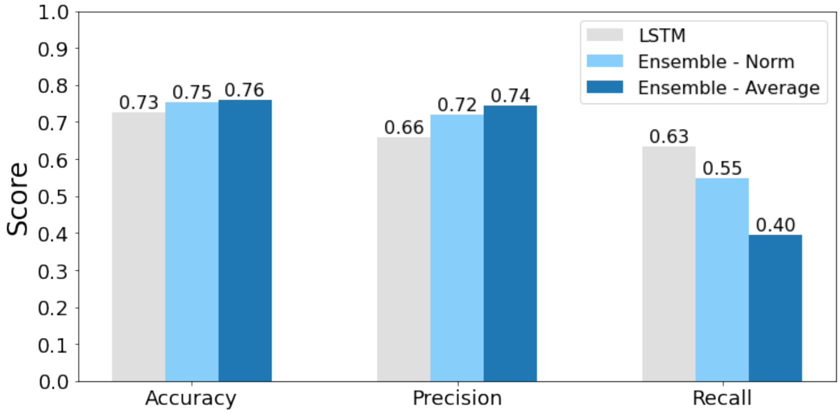


Figure 5.14: Comparison between the LSTM network and two ensembles, that merge the output of the LSTM and the KNN classifier, while using two distance functions.

Finally, we compared the learning ability of the LSTM network against the learning capability of two ensembles, the first which considered 50% of the LSTM’s output and 50% of the KNN classifier, and that used the norm distance function to build the training set. The second also encompassed 50% of the LSTM’s output and 50% of the KNN classifier but used the average distance function to form the training set. The results show that both the ensembles do not improve the model's performance as they reduce the recall score, despite achieving higher precision values. Nevertheless, it is important to note that, in this particular experiment, the norm distance function achieved better results than the average one.

All in all, the results do not justify the use of an ensemble approach as the individual LSTM network attained the most balanced results across all the 3 metrics.

5.2.4.D Multidimensional Motif Discovery

Similarly to section 5.2.3.C, we explore the multidimensional variant of the matrix profile to enable the discovery of the motifs, later used in the training of the developed model. In particular, we evaluate the performance of a model trained on the motifs obtained by this multidimensional algorithm's version, as the one-dimensional approach did not improve the attained results.

In this sense, Fig. 5.15 compares the results obtained by the KNN classifier when following the multidimensional motif discovery approach and the one-dimensional one. Both techniques are analysed with the two proposed distance functions, the one that computes the difference between the norm of the recording's feature and the motif's attribute; and the one which concerns the difference between the average of the z-normalized matrix profile distance of each dimension from the audio's and from the motif's feature. In both cases, we can conclude that the use of the multidimensional motifs does not improve the classifier's performance, as the attained results are similar to the ones from the network trained with the one-dimensional repeated patterns.

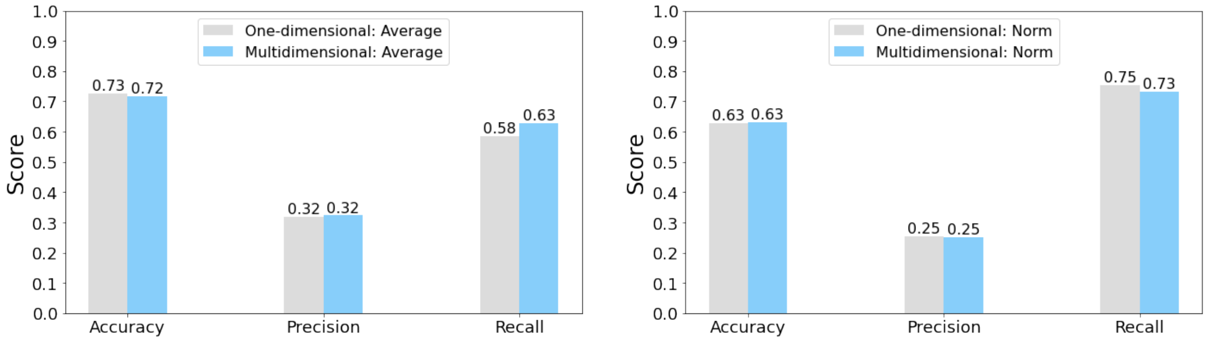


Figure 5.15: Effect of the one-dimensional and multidimensional motifs on the KNN classifier's scores, while adopting two different distance functions.

Given that the presented approach did not improve the model's performance, further research can focus on the bottlenecks related with this procedure. In depth, as previously mentioned, the lack of available distance functions, capable of computing the distance of two multidimensional features limits the developed solution. The limitations of the proposed distance functions may contribute to the absence of better results, as both may not give a correct representation of the similarity (dissimilarity) between the concerned audio's feature and the extracted motifs.

5.2.5 Bioacoustic Framework Appreciation and Best Configuration Results

So far, we have only estimated the models' performance, as the concerned metrics were measured in a set of known records. Despite the insight given by those performance measures there is still uncertainty regarding the models' behaviour when facing unseen objects. So, this section focuses on determining the confidence bounds which detail how much the attained estimate may deviate from the true value. The mentioned process will only concern the best configurations of each approach, as the goal of this work is to establish the best bioacoustic framework possible.

In this regard, to compute the aforementioned intervals we apply the stratified k-fold cross-validation technique to both datasets, in which each dataset is divided in k equal-sized parts (folds), that preserve the percentage of samples for each class. Afterwards, we train each model under the multiple proposed configurations on the different folds. The concerned metrics are obtained regarding also their respective confidence bound. We considered 5 folds for both datasets ($k = 5$) and we used the T-student (95%) distribution to compute the confidence intervals. Note that the mentioned computation considers the average of each metric across the 5 folds.

5.2.5.A Chainsaw Dataset

With the chainsaw dataset, both approaches performed well due to the considerable amount of available labelled recordings. In this sense, regarding the spectral based approach, we only complement the configuration described in section 5.2.2 with the use of cross-validation.

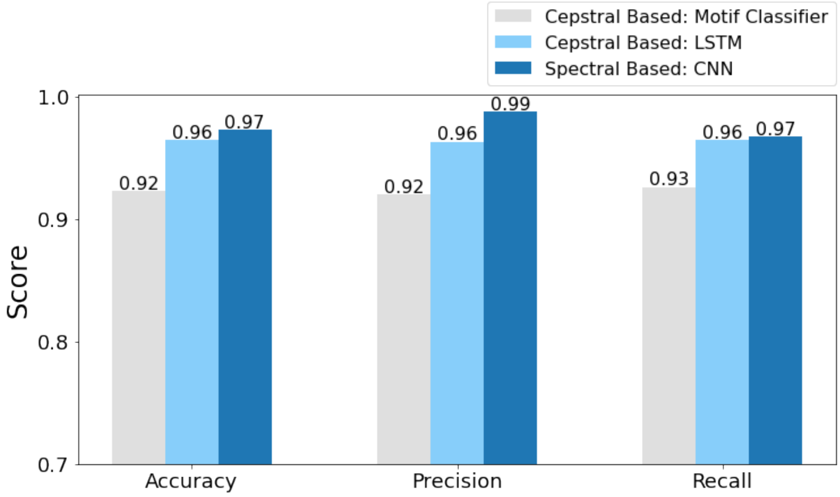


Figure 5.16: Cepstral and spectral based classification models performance (Chainsaw dataset).

The Convolutional Neural network is trained with 5-second-long Mel spectrograms and leverages the use of the SpecAugment data augmentation technique to increase its training set. In relation to the cepstral based approach, we expand the work developed in section 5.2.3, with the introduction of the cross-validation technique. In detail, the Long Short-Term Memory network is trained with the 5-second-long MFCCs. Also, both procedures use a predictive threshold of 60%.

In Fig 5.16, it is possible to compare the accuracy, precision, and recall scores from both approaches. From this figure, we can conclude that the spectral based classification model is the one that achieves the best performance. Nevertheless, in this particular setting, the cepstral based classification network registers similar results, stressing the scores obtained by the motif classifier as this alternative approach was able to approximate the performance of the other two.

Model	Accuracy Lower	Accuracy Upper	Precision Lower	Precision Upper	Recall Lower	Recall Upper
Spectral Based (CNN)	0.95	0.99	0.96	0.99	0.97	0.99
Cepstral Based (LSTM)	0.90	1.00	0.89	1.00	0.89	1.00
Cepstral Based (Motif Classifier)	0.92	0.93	0.92	0.93	0.92	0.93

Table 5.1: Spectral and cepstral based classification model accuracy, precision and recall T-student (95%) confidence intervals (Chainsaw dataset).

Additionally, in Table. 5.1 it is possible to observe the confidence bounds for each metric (accuracy, precision and recall), obtained at the final step of this analysis. Note that across all metrics, the spectral based approach is the one with higher confidence bounds. Not only it attains better results as the higher confidence intervals support our performance estimation.

All in all, the considerable amount of available labelled recordings is the key factor that contributes to the good performance of the developed models.

5.2.5.B Kaggle Dataset

In this particular dataset, the number of labelled recordings is very small, as a consequence, both approaches present different techniques to address this problem. As in the previous section, we expand the analysis concerned up until this point, with the introduction of the cross-validation technique.

In regard to the spectral based approach, our analysis includes two different architectures, both presented in section 4.1.2.A and 4.1.2.B. The main difference between them lies in the introduction of an LSTM layer in the second architecture. Moreover, both networks are trained with 5-second-long Mel spectrograms, the training set of the two is increased with the ”SpecAugment” technique, and the used predictive threshold value is 60%. Oppositely, the cepstral approach is trained with the 5-second-long MFCCs and according to the previous results, this procedure uses a predictive threshold of 75%.

The attained results are depicted in Fig. 5.17, and from a general point of view the spectral based classification model performed better than the cepstral one, supporting the idea that this approach is the more suitable to a bioacoustic classification task.

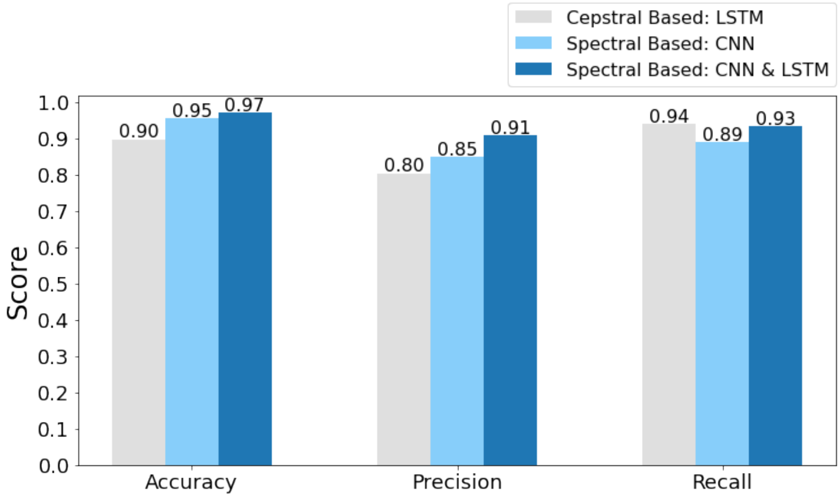


Figure 5.17: Cepstral and Spectral based classification models' performance comparison (Kaggle dataset).

In depth, apart from recall, it achieved the highest scores, with the network that includes the LSTM layer standing out from the other one, and justifying the introduction of this layer. The cepstral based classification model benefited from the cross-validation technique, as it registered a significant improvement in all scores, nevertheless, despite having a higher recall score, all the other metrics are lower than the ones obtained by the spectral based classification model.

Model	Accuracy Lower	Accuracy Upper	Precision Lower	Precision Upper	Recall Lower	Recall Upper
Spectral Based (CNN)	0.89	1.00	0.64	1.00	0.70	1.00
Spectral Based (CNN & LSTM)	0.90	0.99	0.74	1.00	0.75	1.00
Cepstral Based (LSTM)	0.74	1.00	0.56	1.00	0.76	1.00

Table 5.2: Spectral and cepstral based classification model accuracy, precision and recall T-student (95%) confidence intervals (Kaggle dataset).

Moreover, in Table. 5.2 we showcase the confidence bounds for the previously presented results. Once again, our performance estimation is much stronger for the spectral based classification model. Nonetheless, in both approaches, the lower confidence bound value is much smaller in comparison to the ones obtained in the previous setting. In a sense, the attained confidence intervals align with the learning difficulties faced by the developed networks, since their training relied on a limited set of labelled recordings. The spectral based classification model that includes the LSTM layer is the one that provides more certainty regarding our performance estimation, being the cepstral based classification model the one with the lower confidence bounds.

To sum up, once again the spectral based approach seems to be the more adequate approach to a bioacoustic classification task, however, further research needs to focus on trying to attain higher confidence bounds when using the proposed methodology with datasets that have limited training data.

6

Conclusion

Contents

6.1 Conclusions

The field of bioacoustics is key to ensure the conservation of rainforests and their wildlife, as it helps reducing human impact on the environment. In this sense, Rainforest Connection emerges as a prominent source of environmental audio data, contributing to this cause by encouraging the development of bioacoustic monitoring systems. Deep learning methods have been successful on automating the process of species identification in environmental recordings, requiring nonetheless a large number of training samples per species. Thus, recent research focused on developing solutions capable of automate high-accuracy species detection in noisy soundscapes with limited training data.

Our work proposes a bioacoustic classification framework that achieved encouraging results, presenting capable solutions for the problem at hand. In depth, it details two different approaches to address this task, and it evaluates different concepts and procedures to determine the most suitable one. The first leverages off the transfer learning setting to reduce the training requirements, both the amounts of data and time, and relies on the Mel spectrograms to train the developed classification model (CNN). Conversely, the second uses the MFCCs to train the developed classification model (LSTM), proposing also an additional network trained on the matrix profile motifs to complement the proposed methodology.

We have demonstrated that both approaches are able to automate this process and can be included in bioacoustic monitoring systems. The spectral based approach performed better than the cepstral one, in both datasets. In particular, it achieved an accuracy of 0.97, a mean precision of 0.99 and a mean recall of 0.97, with the chainsaw dataset. With the Kaggle dataset, it registered an accuracy of 0.97, a mean precision of 0.91 and a mean recall of 0.93. The cepstral based approach aimed to present an alternative to the previous methodology, as it concerned other audio features and other network type. Additionally, it attempted to improve the results obtained by this procedure, by exploring a setting in which a classifier was trained with the motifs extracted by the matrix profile algorithm.

All in all, we can state that all the goals set for this work were fully met, namely the definition of a capable bioacoustic classification framework.

6.2 System Limitations and Future Work

Concerning the spectral based approach, future work can address the used data augmentation techniques, as this procedure can be optimized to further improve the effectiveness of the training set. Despite several attempts, we were not able to fine-tune the developed network to our task, being also a subject to be addressed in posterior work.

In regard to the cepstral based approach, subsequent research can focus on the reasons that limited the performance of this solution. Moreover, as previously mentioned, we attempted to complement this approach by including a classifier that considered the motifs obtained by the matrix profile algorithm.

This procedure presented some bottlenecks, namely the introduced distance functions and the limited amount of available methods capable of handling the matrix profile related matters and the concerned multidimensional features.

All in all, this document encompasses an in-depth research on numerous audio related subjects (audio features, processing techniques,...), that are key to the development of the proposed end-to-end pipelines. In this regard, a natural subsequent step of our work would be to take advantage of the proposed frameworks to built a model capable of performing sound classification in audio data streams. In addition, as we address motif discovery with the matrix profile algorithm, future solutions can expand the carried out experiments to achieve the aforementioned purpose.

Bibliography

Abeßer, J. (2020). A review of deep learning based methods for acoustic scene classification. *Applied Sciences*, 10(6).

Aucouturier, J.-J. and Sandler, M. (2002). Finding repeating patterns in acoustic musical signals: Applications for audio thumbnailing. In *Audio Engineering Society Conference: 22nd International Conference: Virtual, Synthetic, and Entertainment Audio*. Audio Engineering Society.

Bai, J., Wu, R., Wang, M., Li, D., Li, D., Han, X., Wang, Q., Liu, Q., Wang, B., and Fu, Z. (2018). CIAIC-BAD system for DCASE2018 challenge task 3. Technical report, DCASE2018 Challenge.

Branco, C. M. M. (2020). Online anomaly detection in univariate data streams. Master's thesis, Instituto Superior Técnico.

Briggs, F., Lakshminarayanan, B., Neal, L., Fern, X., Raich, R., Frey, S., Hadley, A., and Betts, M. (2012). Acoustic classification of multiple simultaneous bird species: A multi-instance multi-label approach. *The Journal of the Acoustical Society of America*, 131:4640–50.

Campana, B. and Keogh, E. (2010). A compression based distance measure for texture. In *A Compression Based Distance Measure for Texture*, volume 3, pages 381 – 398.

Dai, W., Dai, C., Qu, S., Li, J., and Das, S. (2017). Very deep convolutional neural networks for raw waveforms. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 421–425. IEEE.

Dau, H. and Keogh, E. J. (2017). Matrix profile v: A generic technique to incorporate domain knowledge into motif discovery. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Dietterich, T. G., Lathrop, R. H., and Lozano-Pérez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1):31 – 71.

Eklund, V.-V. (2019). Data augmentation techniques for robust audio analysis. Master's thesis, Tampere University.

Glaze, C. M. and Troyer, T. W. (2007). Behavioral measurements of a temporally precise motor code for birdsong. *Journal of Neuroscience*, 27(29):7631–7639.

Grill, T. and Schlüter, J. (2017). Two convolutional neural networks for bird detection in audio signals. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 1764–1768.

Hao, Y., Shokoohi-Yekta, M., Papageorgiou, G., and Keogh, E. (2013). Parameter-free audio motif discovery in large data archives. In *2013 IEEE 13th International Conference on Data Mining*, pages 261–270.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Hershey, S., Chaudhuri, S., Ellis, D. P., Gemmeke, J. F., Jansen, A., Moore, R. C., Plakal, M., Platt, D., Saurous, R. A., Seybold, B., et al. (2017). Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 131–135. IEEE.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9:1735–80.

Hsu, J.-L., Liu, C.-C., and Chen, A. (2001). Discovering nontrivial repeating patterns in music data. *Multimedia, IEEE Transactions on*, 3:311 – 325.

Imoto, K. (2018). Introduction to acoustic event and scene analysis. *Acoustical Science and Technology*, 39.

Jamali, S., Ahmadpanah, J., and Alipoor, G. (2018). Bird audio detection using supervised weighted nmf. Technical report, DCASE2018 Challenge.

Krizhevsky, A., Sutskever, I., and Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

Kumar, A. and Raj, B. (2016). Audio event detection using weakly labeled data. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1038–1047.

LeBien, J., Zhong, M., Campos-Cerqueira, M., Velev, J. P., Dodhia, R., Ferres, J. L., and Aide, T. M. (2020). A pipeline for identification of bird and frog species in tropical soundscape recordings using a convolutional neural network. *Ecological Informatics*, 59:101113.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lezhenin, I., Bogach, N., and Pyshkin, E. (2019). Urban sound classification using long short-term memory neural network. In *2019 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 57–60.

Liaqat, S., Bozorg, N., Jose, N., Conrey, P., Tamasi, A., and Johnson, M. T. (2018). Domain tuning methods for bird audio detection. Technical report, DCASE2018 Challenge.

Liu, S. and Deng, W. (2015). Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 730–734.

Liu, Y., Cheng, Z., Liu, J., Yassin, B., Nan, Z., and Luo, J. (2019). Ai for earth: Rainforest conservation by acoustic surveillance. *arXiv preprint arXiv:1908.07517*.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440.

Maccagno, A., Mastropietro, A., Mazziotta, U., Scarpiniti, M., Lee, Y.-C., and Uncini, A. (2017). A cnn approach for audio classification in construction sites. In *IIH-MSP*.

Minnen, D., Isbell, C., Essa, I., and Starner, T. (2007). Detecting subdimensional motifs: An efficient algorithm for generalized multivariate pattern discovery. *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 601–606.

Nordby, J. (2019). Environmental sound classification on microcontrollers using convolutional neural networks. Master's thesis, Norwegian University of Life Sciences.

Nyquist, H. (1928). Certain topics in telegraph transmission theory. *Transactions of the American Institute of Electrical Engineers*, 47(2):617–644.

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D., and Le, Q. V. (2019). Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*.

Rocchesso, D. (1995). Sound processing. *Computer Music Journal*, 19.

Ronneberger, O., P.Fischer, and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer. (available on arXiv:1505.04597 [cs.CV]).

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Ruiz-Muñoz, J. F., Orozco-Alzate, M., and Castellanos-Domínguez, G. (2015). Multiple instance learning-based birdsong classification using unsupervised recording segmentation. In *IJCAI*.

Serizel, R., Bisot, V., Essid, S., and Richard, G. (2018). Acoustic features for environmental sound analysis. In *Computational analysis of sound scenes and events*, pages 71–101. Springer.

Sherstinsky, A. (2020). Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306.

Stowell, D., Giannoulis, D., Benetos, E., Lagrange, M., and Plumbley, M. (2015). Detection and classification of acoustic scenes and events. *IEEE Transactions on Multimedia*, 17:1733–1746.

Tanaka, Y., Iwamoto, K., and Uehara, K. (2005). Discovery of time-series motif from multi-dimensional data based on mdl principle. *Machine Learning*, 58(2-3):269–300.

Wang, Y., Neves, L., and Metze, F. (2016). Audio-based multimedia event detection using deep recurrent neural networks. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2742–2746.

Weik, M. H. (2001). *Nyquist theorem*, pages 1127–1127. Springer US, Boston, MA.

Wen, T. and Keyes, R. (2019). Time series anomaly detection using convolutional neural networks and transfer learning. *arXiv preprint arXiv:1905.13628*.

Yeh, C.-C. M., Kavantzas, N., and Keogh, E. (2017a). Matrix profile iv: Using weakly labeled time series to predict outcomes. *Proc. VLDB Endow.*, 10:1802–1812.

Yeh, C.-C. M., Kavantzas, N., and Keogh, E. (2017b). Matrix profile vi: meaningful multidimensional motif discovery. In *2017 IEEE international conference on data mining (ICDM)*, pages 565–574. IEEE.

Yeh, C.-C. M., Van Herle, H., and Keogh, E. (2016). Matrix profile iii: the matrix profile allows visualization of salient subsequences in massive time series. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 579–588. IEEE.

Yeh, C. M., Zhu, Y., Ulanova, L., Begum, N., Ding, Y., Dau, H. A., Silva, D. F., Mueen, A., and Keogh, E. (2016). Matrix profile i: All pairs similarity joins for time series: A unifying view that includes motifs, discords and shapelets. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 1317–1322.

Zhong, M., LeBien, J., Campos-Cerqueira, M., Dodhia, R., Lavista Ferres, J., Velev, J. P., and Aide, T. M. (2020). Multispecies bioacoustic classification using transfer learning of deep convolutional neural networks with pseudo-labeling. *Applied Acoustics*, 166:107375.

Zhou, Z.-H. and Zhang, M.-L. (2002). Neural networks for multi-instance learning. *Proceedings of the International Conference on Intelligent Information Technology*.

Zhu, Y., Yeh, C. M., Zimmerman, Z., Kamgar, K., and Keogh, E. (2018). Matrix profile xi: Scrimp++: Time series motif discovery at interactive speeds. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 837–846.

Zhu, Y., Zimmerman, Z., Senobari, N. S., Yeh, C. M., Funning, G., Mueen, A., Brisk, P., and Keogh, E. (2016). Matrix profile ii: Exploiting a novel algorithm and gpus to break the one hundred million barrier for time series motifs and joins. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 739–748.