UNIVERSITY OF CRETE
SCHOOL OF SCIENCES AND ENGINEERING
COMPUTER SCIENCE DEPARTMENT

MASTER THESIS

MACHINE LEARNING TECHNIQUES FOR THE DETECTION OF ILLEGAL HUMAN ACTIVITY IN AUDIO RECORDINGS FROM PROTECTED AREAS

KONSTANTINOS PSAROULAKIS

THESIS ADVISOR: PROF. PANAGIOTIS TSAKALIDES

HERAKLION, OCTOBER 2020

Machine learning techniques for the detection of illegal human activity in audio recordings from protected areas

Konstantinos Psaroulakis

Thesis submitted in partial fulfillment of the requirements for the

Masters' of Science degree in Computer Science and Engineering

University of Crete
School of Sciences and Engineering
Computer Science Department
Voutes University Campus, 700 13 Heraklion, Crete, Greece

Thesis Advisor: Prof. *Panagiotis Tsakalides*

This work has been performed at the University of Crete, School of Sciences and Engineering, Computer Science Department.

The work has been supported by the Foundation for Research and Technology - Hellas (FORTH), Institute of Computer Science (ICS).

University of Crete
Computer Science Department

Machine learning techniques for the detection of illegal human activity in audio recordings from protected areas

Thesis submitted by
Konstantinos Psaroulakis
in partial fulfillment of the requirements for the
Masters' of Science degree in Computer Science

THESIS APPROVAL

Author: Konstantinos Psaroulakis

Committee approvals:

Panagiotis Tsakalides
Professor, Thesis Supervisor

Nikolaos Stefanakis
Assistant Professor, Committee Member

Yannis Stylianou
Professor, Committee Member

Departmental approval:

Polyvios Pratikakis
Assistant Professor, Director of Graduate Studies

Heraklion, October 7, 2020

MACHINE LEARNING TECHNIQUES FOR THE DETECTION OF ILLEGAL HUMAN ACTIVITY IN AUDIO RECORDINGS FROM PROTECTED AREAS

Abstract

Human activity is considered today as the primary reason for habitat loss for a large number of Earth's plant and animal species. This activity results to the permanent loss of species and to the weakening of the ecosystems that are of significant importance for the overall health of the planet and as a consequence, to the quality of the human life. One key measure to protect habitats is the establishment of protected areas where human activity is restricted. In these areas, systems employing multiple cameras and microphones may offer a significant assistance in monitoring the health of the ecosystem but also as the means to prevent human intervention that is harmful to the environment.

This Thesis concerns the application of signal processing and machine learning techniques to audio recordings acquired in protected areas in Greece, with the aim to automatically detect sound events that are indicative of illegal human activity such as illegal logging, grazing, hunting, etc. To collect and annotate the data that is required for training such a scheme, we illustrate the usefulness of a Voice Activity Detector (VAD) that is activated on the presence of harmonic structure in the audio content. The VAD is used in order to automatically segment hundreds of hours of audio recording into thousands of short duration audio clips that potentially carry the underlying pattern of interest. Continuing, we perform numerous experiments with the goal to find the optimal approach for (i) a binary classification problem that focuses on the case of chainsaw sound and (ii) a six class problem that includes additional patterns relating to illegal human activity. Experimental results illustrate the superiority of Deep Neural Networks (DNN) against other well-known conventional classifiers and furthermore, highlight choices that are advantageous for the intended task in terms of the DNN architecture and the type of acoustic features.

Χρήση τεχνικών μηχανικής μάθησης για τον εντοπισμό ανθρωπογενούς δραστηριότητας από ηχογραφήσεις σε προστατευόμενες περιοχές

Περίληψη

Η ανθρώπινη δραστηριότητα αποτελεί σήμερα τη σημαντικότερη αιτία εξαφάνισης βιοτόπων μεγάλου αριθμού φυτών και ζώων στον πλανήτη. Αυτή η δραστηριότητα οδηγεί αφενός στην ολική εξαφάνιση πολλών ειδών κι αφετέρου στη αποδυνάμωση των οικοσυστημάτων, γεγονός που διαταράσσει τις ισορροπίες στον πλανήτη και την ποιότητα ζωής του ανθρώπου. Ένα μέτρο προστασίας είναι η θέσπιση προστατευμένων περιοχών όπου η ανθρώπινη δραστηριότητα είναι περιορισμένη. Σε αυτές τις περιοχές, συστήματα ασφαλείας αποτελούμενα από πολλές κάμερες και μικρόφωνα μπορούν να συμβάλλουν σημαντικά στην παρακολούθηση της ισορροπίας του εκάστοτε οικοσυστήματος καθώς και στην αποτροπή ανθρώπινης δραστηριότητας που αποτελεί απειλή για το περιβάλλον.

Αυτή η εργασία επικεντρώνεται στην εφαρμογή τεχνικών μηχανικής μάθησης και επεξεργασίας σήματος σε δεδομένα ήχου από προστατευόμενες περιοχές ανά την Ελλάδα, με σκοπό τον αυτόματο εντοπισμό ηχητικών γεγονότων που σηματοδοτούν παράνομη ανθρώπινη δραστηριότητα, όπως παράνομη υλοτομία, βόσκηση, κυνήγι κ.α. Για τη συλλογή και κατηγοριοποίηση των δεδομένων που απαιτούνται για την εκπαίδευση των μοντέλων μηχανικής μάθησης, παρουσιάζουμε τη χρησιμότητα μιας μεθόδου εντοπισμού φωνής που ενεργοποιείται με την ύπαρξη αρμονικής δομής σε ένα σήμα ήχου. Η μέθοδος αξιοποιείται για την αυτόματη κατάτμηση εκατοντάδων ωρών ηχογράφησης σε χιλιάδες μικρής διάρκειας ηχητικά αποσπάσματα που εν δυνάμει φέρουν το υποκείμενο μοτίβο ενδιαφέροντος. Στη συνέχεια, εκτελούμε πολλαπλά πειράματα με στόχο την εύρεση της βέλτιστης προσέγγισης για (i) ένα πρόβλημα δυαδικής ταξινόμησης που επικεντρώνεται την περίπτωση του ήχου του αλυσοπρίονου και (ii) ένα πρόβλημα έξι κλάσεων που περιλαμβάνει πρόσθετα ηχητικά μοτίβα σχετικά με την παράνομη ανθρώπινη δραστηριότητα. Τα αποτελέσματα που παρουσιάζουμε αναδεικνύουν την υπεροχή των Βαθιών Νευρωνικών Δικτύων έναντι γνωστών συμβατικών προσεγγίσεων και αναδεικνύουν βέλτιστες επιλογές όσον αφορά τις αρχιτεκτονικές των δικτύων και τον τύπο των ακουστικών χαρακτηριστικών ανάλογα με το πρόβλημα ταξινόμησης που εξετάζεται.

Ευχαριστίες

Καταρχάς θα ήθελα να ευχαριστήσω τον επόπτη μου, Καθηγητή κ. Παναγιώτη Τσακαλίδη, για την στήριξή του και την ευκαιρία που μου προσέφερε να είμαι μέλος της ομάδας του, τον κ. Αθανάσιο Μουχτάρη μέσω του οποίου ξεκίνησε η συνεργασία μου με την ομάδα επεξεργασίας ήχου του εργαστηρίου, καθώς και το Ινστιτούτο πληροφορικής του ΙΤΕ ($FORTH - ICS$) για την οικονομική στήριξη και για την παροχή του απαιτούμενου εξοπλισμού.

Στη συνέχεια θα ήθελα να ευχαριστήσω τον κ. Ιωάννη Στυλιανού και τον κ. Νικόλαο Στεφανάκη που συμπλήρωσαν ως μέλη την εξεταστική επιτροπή της μεταπτυχιακής μου εργασίας.

Ιδιαίτερα στον κ. Νικόλαο Στεφανάκη οφείλω ένα μεγάλο ευχαριστώ για την τεράστια καθοδήγηση και βοήθεια που μου προσέφερε σε όλη τη διάρκεια εκπόνησης της εργασίας και τη στήριξή του σε κάθε δυσκολία που παρουσιάστηκε.

Ευχαριστώ επίσης όλους τους συναδέλφους και μέλη του εργαστηρίου στο ΙΤΕ τόσο για τη συμβολή τους στη δημιουργία μιας ευχάριστης ατμόσφαιρας στο χώρο εργασίας, όσο και για την προθυμία τους για βοήθεια κάθε στιγμή που χρειαζόταν.

Θέλω επίσης να εκφράσω την ευγνωμοσύνη και την αγάπη μου στους φίλους και την οικογένεια μου που πίστεψαν σε μένα και στάθηκαν δίπλα μου σε αυτό το διάστημα και ιδιαίτερα στην αγαπημένη μου για τη συμπαράσταση και τη στήριξη που μου προσέφερε τόσο σε εύκολες όσο και στις δυσκολότερες μέρες όλο αυτό το διάστημα.

Τέλος, θέλω να ευχαριστήσω από καρδιάς τους γονείς μου για την αμέριστη στήριξή τους σε όλες τις αποφάσεις και τα βήματά μου μέχρι και σήμερα.

Σας ευχαριστώ!

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Over the few past decades human have caused major destruction to the environment. Many species have been extincted, or are prone to be, due to illegal hunting, illegal logging and deforestation. One major example of the harm that humans cause to environment is deforestation. Estimates suggest the Earth has lost about half of its forests in 8,000 years of human activity, with much of this occurring in recent decades. However, because of the huge area that forests and protected natural environments cover, and the small manpower available, it is very difficult for governments to effectively monitor these areas to tackle illegal human activities. The solution for this problem is to develop surveillance technologies that automatically detect illegal human activity and in that way assist relevant authorities in monitoring protected natural environments and in taking measures for preventing such actions.

Systems based on both audio and video technology have been proposed for monitoring of protected environments. However, audio based systems can be less power consuming, produce less data, and have a steady performance despite of the different luminance levels during the day. Yet, a large number of acoustic sensors has to be installed in order to cover a spatially extended region and the final system should be able to detect acoustic patterns that have propagated long ranges. This is not so trivial because depending on the distance between sound source and sensor, sound patterns relating to illegal human activity can be submerged in excessive amounts of noise. Such examples of noise include adverse weather conditions, natural habitats of each natural environment or vehicles passing by close routes.

Many acoustic sensors, known as Autonomous Recording Units (ARUs), which are lowcost, and power efficient have been placed in many forests and natural environments all over the world to monitor some species and make observations. Data gathered from these sensors can be used to develop technologies that focus on the task previously described. However, an ARU is capable of recording continuously for days or weeks. This means that huge amounts of audio data can be collected

in a short period of time, making it infeasible for humans to manually inspect the entire collection of recordings. Therefore, apart from the research for an optimal intrusion detection algorithm, it is also important to automate the tools that are involved in the preparation of the data (e.g. collecting training data) and to develop unsupervised ways for extracting useful information for further processing and research. Clearly, this research introduces challenges that relate to the big data regime, or to the so called audio analytics in terms of audio processing [32].

1.1.1 Contribution of this work

The main contributions of this work are the following;

We propose the use of a known Voice Activity Detector (VAD) for detecting sounds with harmonic structure in an environmental audio recording. We exploit it in order to automatically segment the raw audio data into multiple short duration audio clips which can be easily annotated by a human. This way, we were able to annotate hundreds of hours of audio recordings and thus produce a dataset that is valuable for any research related to environmental sounds.

We highlight the usefulness of the so-called SRH-spectrogram, which is an intermediate product of the aforementioned VAD, as a novel acoustic feature for sound classification using DNN. To our knowledge, this acoustic feature is used for the first time for environmental sound classification, showing competing performance compared to other well known acoustic features.

Finally, we benchmark several choices and parameters relating to the use of DNNs for environmental sound classification, with particular focus on the chainsaw sound. To our knowledge, it is first time that such an extensive evaluation is performed on chainsaw sound based on real field recordings.

1.2 Background-Previous Work

The field of audio event detection and classification has been investigated by several researchers. In what follows, we present the previous work in the field with special focus on research related to detection and recognition of acoustic events in natural environments.

1.2.1 Applications

In [71] [57] [78] [92] researchers focused on the identification and classification of bird callings and bird sounds in general. Apart from birds, insects [70] or whales [87] are some additional targets in bioacoustics. Other researchers such as Piczak [69], Lopatka et al [55] and Jeong-Sik et al [67] focused on a civil audio surveillance task, that of recognizing hazardous situations in a civil environment, or emergency situations in a home environment [80]. In [29] [77] [65] the target events occur in urban, office or home environments. In similar environments, there have been works on smart-home applications [45], hands free speech event detection [82]

(home and residential), in speech-oriented event detection in real acoustic scenes [61] and many more.

Concerning wildlife environments, target patterns usually consist of a set of specific illegal human activities, depending on the needs of the location under surveillance. In [86] [63] [41] [76] [37] researchers focused on classification of patterns such as vehicles, gunshots, chainsaw, human speech etc, while in [15] Clavel et al focused specifically on the creation of a robust gunshot detection system in adverse conditions. Apart from gunshot detection, there has also been much research on detection of illegal logging and chainsaw detection in wildlife [2] [39] [5], where some researchers focused on light features and algorithms to create a system more suitable for wireless sensor networks [19] [16] [20].

In addition, it is worth to mention Rainforest Connection (RFCx) [17], a nonprofit organization that transforms recycled phones into autonomous, solar-powered listening devices to monitor and detect logging activities in order to guard rainforests. There is no published work of RFCx, but there are plans to make this data available to academic researches and government agencies, which will be a valuable contribution to the research community.

1.2.2 Standard classification techniques

Standard classification techniques on this field were predominantly Support Vector Machines (SVMs) and Gaussian Mixture Models (GMMs) in combination with a variety of spectral or temporal features [14]. When it comes to computing distances between sounds, Mel Frequency Cepstral Coefficients (MFCCs) have advantageous properties that contribute to this computations [62]. Thus, many researchers used SVMs and GMMs fed with Mel Frequency Cepstral Coefficients (MFCCs), in addition to statistical measures of the signal in time or frequency space such as in [39] [55]. Many works have used this approach as a baseline in addition to a proposed pipeline [80] [78] [61].

Another standard approach was to incorporate the use of Hidden Markov Models (HMMs) in combination with GMMs [65] [15] [67], or with Non-Negative Matrix Factorization (NMF) [93] [10]. An interesting approach to the detection step is to model two GMM distributions, one for the target and one for the background noise, building in this way multiple single-class detectors, one for each class. This approach showed to be efficient in combination with Gabor filterbank features [33]. Ntalampiras et al [64] [63], apart from temporal and spectral features along with MFCCs, they also used wavelet coefficients for acoustic surveillance in real world conditions, stating that parameters of different domains contribute to better performance. Another approach for insect detection and identification was to use Linear Frequency Cepstral Coefficients (LFCC) which are like MFCCs but in linear scale fed to a GMM [70]. In [3] and [6] MFCCs were compared to Linear Predictive Coefficients (LPCs) which were fed to an SVM and a GMM respectively. Apart from SVM and GMMs, LPCs were also used in combination with Random Forest classifier for detecting wildlife illegal human activities in [41] [40]. In [41] LPCs

were fed to Random Forest classifier, in [3] MFCCs and LPCs were fed to SVMs and in [6] MFCCs and LPCs were used in combination to a GMM. Finally, in [70] the authors fed LFCCs into a GMM for detecting and identifying insect sounds.

1.2.3 Low-cost algorithms and features

In the development of detection and classification algorithms for audio surveillance of protected environments, computational cost is of major concern. Sensors are usually battery powered, and hence, algorithms developed to run on such devices have to be as less computational demanding as possible, in order or consume the least possible energy.

Acoustic features relying to the so-called TESPAR analysis have been used by several scientists, mainly due to the low computational cost that they involve. Fourier Transform takes $O(n^2)$ (or $O(nlogn)$ in case of Fast Fourier Transform), autocorrelation method takes ($O(n^2)$), and TESPAR descriptors computation takes $O(n)$ [20]. TESPAR (Time Encoded Signal Processing and Recognition) is a simple way of describing a waveform in digital terms [47], usually as a 1D or 2D array, and its based on statistical measures over zero crossing points of the waveform [1]. One way TESPAR can be used is to be fed as a feature in classifiers. Ghiurcau et al has used TESPAR matrices in this way, with SVMs and GMMs [38] [37] in order to achieve low complexity on the feature extraction and classification system. Ghiurcau M. has also benchmarked the downsampling of raw audio before processing and the filtering of very low and very high frequencies to discard unwanted information, with positive results. As an attempt to further reduce computational complexity, the authors in [86], [36] used a few TESPAR descriptors as "archetypes" and relied on distance measures in order to perform classification. Another low complexity method is the use of autocorrelation (AuC) based features in combination with a simple thresholding detector, even though AuC complexity is relatively high in comparison with TESPAR or FFT. However TESPAR is shown to be less computationally demanding and contributes to better performance [20] [19].

Meeting similar requirements with respect to the computational cost, Ahmad et al [2] have benchmarked some low complexity algorithm such as K-means clustering, GMM and Principal Component Analysis (PCA) for classifying sounds related to tree cutting detection. In other works, such as in [72], the authors used simple thresholding to create a system able to perform chainsaw detection.

[1] More specific, some simple TESPAR coders make use of the duration of the signal between two real zeros and the local maxima/minima between two consecutive real zeros. This method is based on infinite clipping theory of Licklidder and Pollack, whose research focused on the effects of amplitude clipping on the intelligibility of speech. In fact, a mean random-word intelligibility score of 97% was achieved by removing all the amplitude information from a waveform, resulting in a binary transormation which preserved only the zero crossings of the initial signal (the so called infinite clipping format) [52]. This experiment demonstrated the potential power of the zero crossings, which led to the discovery of TESPAR matrix.

1.2.4 Use of Deep Neural Networks

On the other hand, when performance is more important than computational cost, Neural Networks and especially Deep Neural Networks (DNNs) outperform all standard classifiers with the use of suitable features. The last decades, DNNs have gained much ground due to the fast and great development of the hardware and the huge creation of data that is generated each moment. The features used when DNNs are employed are usually different from the classic features mentioned previously. Many variations and architectures have been tried and evaluated depending on the exact task.

Convolutional Neural Networks (CNNs) were among the first DNN architectures examined within the context of acoustic signals. Piczak [69] uses Convolutional Neural Networks (CNNs) fed with mel-spectrograms to classify urban sounds and Hyungui et al [53] use 1D Convolutional Neural Networks (CNNs) fed with mel-spectrograms to detect and classify rare events. Using a different dataset, Phan et al [68] trained a CNN model with 1D max pooling layers, fed with Spectrogram Image Features (SIF), which were initially introduced by Jonathan Dennis et al in [24] [25] [23] [26]. Zhang et al [91] use CNNs with Power Spectrogram and Mel-Spectrogram and they enhance features by applying some image processing techniques, in order to achieve a reasonable performance in noisy conditions. Other approaches with CNNs include MFCCs or mel energies [34] [93].

Apart from CNNs, Recurrent Neural Networks (RNNs) have been proposed for time-series data, due to their ability to model time dependencies in the data. Vanishing and exploding gradients is a well known issue related to the training of RNNs and for this reason, LSTMs have been proposed as an alternative architecture that overcomes this problem [44]. In [43] [89], researchers use an LSTM approach that showed to work better than standard approaches, such as SVMs, both in clip and frame level. An interesting approach was that of Lu et al [58], who proposed a multiscale RNN, that operates simultaneously on a fine and a coarse scale to model both fine-grained and long-term dependencies, achieving better performance than a single RNN approach. Hyungui et al use a mixed architecture of CNN and LSTM Units in order to capture temporal dependencies in training data as well [53].

In addition to the works discussed above, there has also been research in the detection and classification of concurrent events [93]. In [11] the proposed technique focuses on filterbank learning through CNNs using mel on polyphonic event detection. The same authors in [10] focus on the comparison between multiple single-label CNNs and a single multi-label CNN for polyphonic event detection, which were shown to have similar performance, giving more flexibility and choices for further work.

1.2.5 Noise and Enhancements

A big issue in wildlife audio event detection is the fact that sound propagates at long ranges before reaching the sensors and therefore the sound patterns of interest are observed at low SNR. In addition, losses due to air absorption [42] cause the high frequency components to be attenuated at a higher proportion compared to the lower frequency ones. Normalization techniques have thus been proposed as the means to achieve robustness to background noise and to balance the gain between low and high frequency bands. Wang et al proposed a frontend called Per Channel Energy Normalization (PCEN) as an alternative of log mel spectrogram in order to increase robustness to loudness variation [88]. PCEN combines a dynamic range compression and an adaptive gain control, contributing to robustness to both noise and channel distortions. While initially proposed for far field keyword detection [88], its usage has recently been expanded also to bioacoustics. Moreover, the positive impact of PCEN has been proven within a theoretical perspective in [56]. Further research of Lostanlen et al in PCEN, include marine bioacoustic patterns, such as whale calls [87] where the importance of PCEN was demonstrated on both near and far field recordings, and avian bioacoustic patterns, such as bird calls [87, 57].

Another idea proposed in the literature concerning the number of classes and the classification pipeline is the so-called hierarchical classification approach [15], [6]. In a hierarchical classification approach, the target class has been split in a few subclasses, and after training, the subclasses are merged into the one class of interest to make a more general prediction, or the inverse, such as in [63], where each test sample is first classified in a wider super-class, e.g., human/non human sound, and afterwards it goes through a second phase of classification where it is classified as a subclass of the initially selected / wider class.

Concerning the use of VAD algorithm is this field, Valenti et al [82] used a VAD algorithm to detect onset and offset times on events occurring in the signal and then fed each detection segment to a classifier. This two-step detection and classification approach allows the audio event detection in unstructured audio signals.

1.2.6 Data augmentation

It is a well known fact in deep learning that the more the data the better the performance. However data gathering is not always easy, thus data augmentation techniques have been proposed in order to reduce overfitting. Related applications include bird species classification [78, 57], urban sound classification [77] etc. In [77], [78] Salamon et al used a few types of data augmentation to widen the dataset, such as pitch shifting, time stretching, dynamic range compression etc, and contributed to a better performance compared to the model trained only on the original data, or a dictionary learning model with the augmented dataset. Lostanlen et al in [57], apart from pitch shifting and time stretching, used more sophisticated augmentation methods which take into account the whole training

data and occasionally the corresponding labels, to obtain augmented samples. For example, as Lostanlen et al state, mixing audio clips in a sensor with a noisy clip belonging to a different sensor leads to greater generalization of the model.

Chapter 2

Classification Approaches

This section is split into two main subsections. In the first section, we will attempt to present the general ideas behind DNNs, highlighting the facts that are relevant to the architectures and structures used in the experimental section of this Thesis. In the second section, the focus will be in more conventional machine learning techniques and especially on those that that were used as a baseline for comparison with DNNs in this Thesis.

2.1 Artificial Neural Networks

2.1.1 General

The idea of Artificial Neural Networks (ANNs) emerged from biology and in particular, from the brain neurons. They are computing systems consisting of neurons and connections between them. The initial idea goes back to 1940s-1950s when Warren McCulloch and Walter Pitts [59] (1943) opened the subject by creating a computational model for neural networks. A decade later, Rosenblatt created the perceptron [75], a system inspired by biological human brain neurons. At that time, due to hardware and data limitations, ANNs did not receive much attention and the field was frozen for years. Only the last decades ANNs have seen a major increase in their use and popularity and have been shown to outperform the classic machine learning approaches.

Two basic reasons contributed to this. The first is evolution of the hardware, enabling low cost computers to face the high demand on computational power that is required for training. The second reason is that today's scientists have easy access to massive amounts of all kinds of data which is required as input to the ANNs during the training stage.

An artificial neural network can be as simple as a linear function; having an input layer, an output layer and a hidden layer, or more complex having multiple layers between input and output layer. In the latter case it is called Deep Neural Network (DNN). DNNs are capable of learning abstract and hierarchical representations from raw data, unlike traditional methods that rely upon hand-crafted

features. Hence, DNNs are suited especially well for machine perception tasks, where the raw underlying features are not individually interpretable. A component that contributes to this, is the use of activation functions in each layer, which make the models learn complex non-linear relationships. In this work, the architectures that will be used for the classification purposes are Convolutional neural networks (CNN) and Recurrent Neural Networks (RNN).

2.1.1.1 Activation functions

An activation function is usually a non-linear mathematical function which transforms the output of each neuron given the input. Non-linear activation functions (or so called "non linearities") allow the network to model more complex relationships between input/output pairs and, thus, to compute nontrivial problems using only a small number of nodes [83]. Some examples of popular and widely used activation functions are softmax, hyperbolic tangent, rectified linear unit, sigmoid functions.

2.1.2 Convolutional Neural Networks (CNN)

Convolutional Neural Networks' (CNNs) [51] architecture is profoundly one of the most competitive architectures in the field of ANNs. CNNs were used mainly in computer vision and image classification tasks, because of their ability to extract distinguishing features from image data. However, their use has been extended to other domains such as audio and speech processing, object recognition in image data, recommendation systems e.t.c. Their power is based on stacking convolutional and pooling layers resulting in an hierarchical decomposition of the initial representation. In convolutional layers, the basic component of a CNN, the input is convolved with a filter, called kernel, and the result of this convolution is the output of the layer. This is similar to the response of a neuron in the visual cortex of animals to a specific stimulus [46].

Each convolution neuron processes only a restricted area called *receptive field*, in opposition to Feedforward Neural Networks (FFNN)(Fig. 2.1) where each neuron is connected with all neurons in previous and next layers (Fig. 2.2). In the latter case, a huge number of neurons would be required even for shallow architectures, especially for image data where the dimensionality of the input is very high. As a consequence feedforward neural networks lead to unnecessarily complex models (too many neurons/trainable parameters) especially when deeper architectures are deployed.

In addition, in CNNs, each filter is replicated across the entire visual field, allowing for patterns to be detected regardless of their position in the visual field. All these replicated units, which form a *feature map*, share the same parameterization (weights and bias) (Fig. 2.3), increasing efficiency by reducing the number of trainable parameters, and contributing to a better generalization of the model on vision problems.

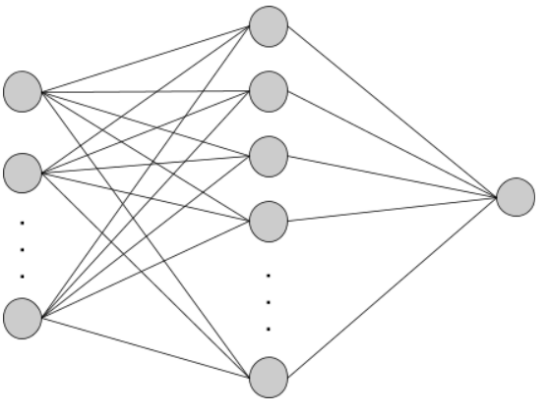

Figure 2.1: Feedforward Neural Network

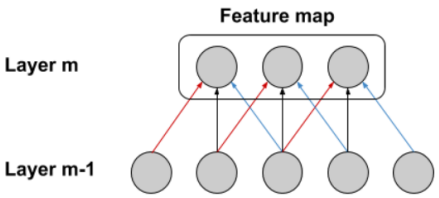

Figure 2.2: Shared weights

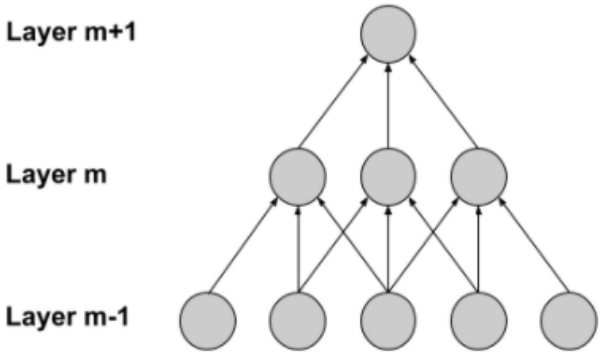

Figure 2.3: CNN Feature Map

Another component that was included in the CNN architectures after introduced by Yamaguchi et al in 1990 [77] was the pooling layer. In general, pooling layers apply a specified size reduction filter to the subregions of the initial representation. Specifically, maxpooling uses a max filter. In this way they lead to models with less trainable parameters while increasing the model's receptive field.

Two noteworthy architectures that made an impact in this field were *Neocognitron* proposed by Fukushima et al (Fig. 2.4) [31] which served as the inspiration for convolutional neural networks and AlexNet proposed by Krizhevsky et al (Fig. 2.5) [49] which is considered as one of the most influential papers in Computer Vision.

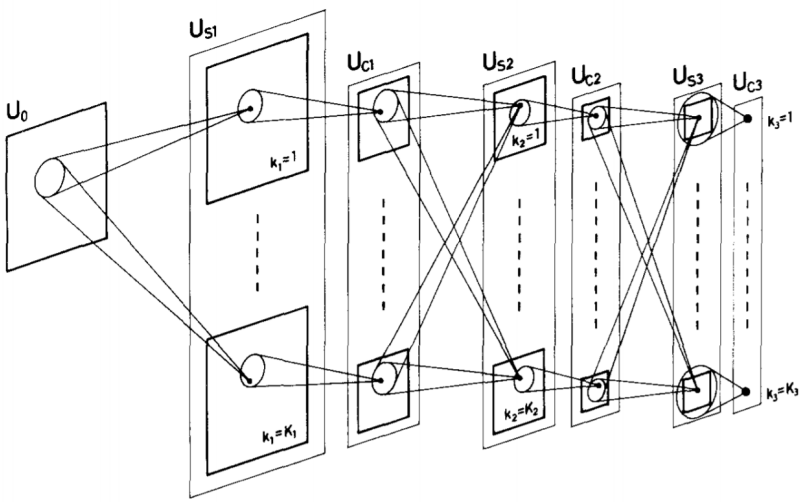

Figure 2.4: Neocognitron architecture proposed by Fukushima K. in 1980 [31]

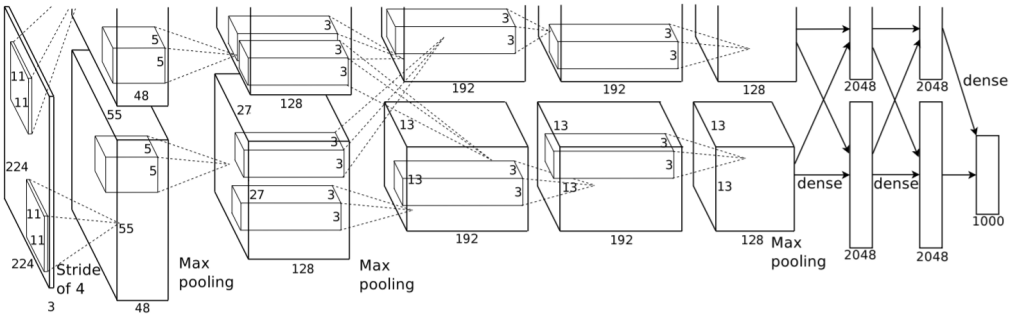

Figure 2.5: AlexNet architecture proposed by Krizhevsky et al in 2012 [49]

2.1.3 Recurrent Neural Networks (RNN)

A basic assumption of Feedforward Neural Networks (FFNNs), is that each sample is independent, an assumption that does not hold in the case of time series data. Hence, in spite of their importance, with this approach, after each example is fed to the network, any information describing the state of the model is lost.

Recurrent Neural Networks (RNNs) are capable of addressing this problem by processing one element at a time, like FFNNs, while passing information across adjacent sequence steps, capturing temporal dependencies at the same time. As Lipton et al stated, Recurrent neural networks (RNNs) are *feedforward neural networks augmented by the inclusion of edges that span adjacent time steps, introducing a notion of time to the model.* [54] Mathematically expressed, at a timestep t, nodes with recurrent edges receive input from the current data points $x(t)$ and from the network's previous state, which is captured in hidden node values $h(t-1)$. The output $\hat{y}(t)$ is calculated, given the hidden node values $h(t)$. This procedure can be summarized in two fundamental equations:

$$\mathbf{h(t)} = \sigma(\mathbf{W}^{xh}\mathbf{x}^{(t)} + \mathbf{W}^{hh}\mathbf{h}^{(t-1)} + \mathbf{b_h}) \tag{2.1}$$

and

$$\hat{\mathbf{y}}(\mathbf{t}) = softmax(\mathbf{W}^{hy}\mathbf{h}(t) + \mathbf{b_y}) \tag{2.2}$$

where $\mathbf{W}^{xh}$, $\mathbf{W}^{hh}$, $\mathbf{W}^{hy}$ are respectively the conventional weight matrices between the input x and the hidden state h, the matrix of recurrent weights between hidden states of the previous and the current time step, and the weight matrix which pairs the hidden state h with the output y. The vectors $\mathbf{b_h}$ and $\mathbf{b_y}$, are bias parameters which allow each node of the network to learn an offset.

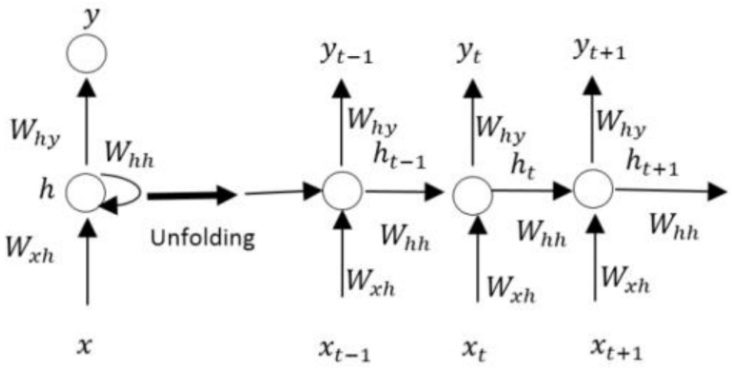

Figure 2.6: Folded and unfolded Recurrent Neural Network. This figure was adopted from [50]

The dynamics of the network across time steps can be visualized by unfolding it as in Fig. 2.6. Given the latter figure, the network can be interpreted as a deep network with one layer per time step and shared weights across time steps. RNN

models can be trained using backpropagation, but with an altered version of the algorithm introduced by Werbos, called backpropagation through time (BPTT). [90] The aim of the training process, is to determine the weight parameters illustrated in Eqs. (2.1-2.2), given the training examples. [54]

2.1.3.1 Long Short-Term Memory Units (LSTM)

Many variations of RNNs have been proposed over the years, with Long Short Term Memory cell (LSTM) being among the most succesful ones. LSTMs were introduced by Hochreiter & Schmidhuber in 1997 [44], primarily in order to tackle the issue of vanishing gradients[1]. The general idea of LSTMs is the replacement of a simple RNN node, with a unit that can model longer temporal dependencies between samples, compared to vanila RNNs. This is done by replacing each standard RNN node by a so called "memory cell" (Fig. 2.8). Memory cells, contain a self-connected recurrent edge of constant unit weight. This procedure aims to prevent gradients from vanishing or exploding.

The intuition behind the term "long short-term memory" is well illustrated in [54], where Lipton states that *weights is the long-term memory of simple recurrent neural networks, by changing slowly during training and encoding general knowledge about the data. They also have short-term memory in the form of ephemeral activations, which pass from each node to successive nodes. The LSTM model introduces an intermediate type of storage via the memory cell.*

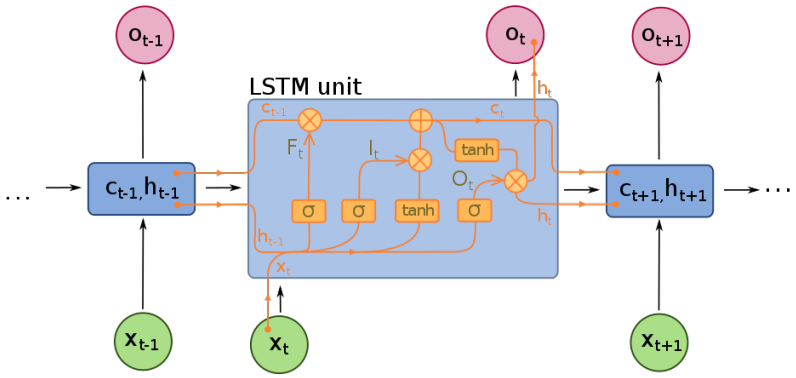

Figure 2.7: LSTM cell. (Adopted from Wikipedia [84])

[1] The so-called Vanishing & Exploding Gradients problem, first mentioned by Bengio et al [7] is referred to a common, yet important issue when training an RNN. The vanishing gradient problem occurs when a gradient is vanishingly small. Such gradients effectively prevent weights from updating their values and ultimately make the network unable to train. The exploding gradient problem describes the opposite case where the gradient increases progressively leading to very large weight updates during training. Again, this causes the model to be unstable and hence, unable to learn from training data.

A common LSTM unit/cell consists of the following components:

- *Input node*: In this unit, notated as $\mathbf{g_c}$, the weighted sum of the activation from input $\mathbf{x_t}$ at the current time step and from the hidden layer at the previous time step $\mathbf{h_{t-1}}$ is calculated, and passed through a non-linear activation function. This activation function is typically a hyperbolic tangent (tanh) or a sigmoid as presented in the original paper.

- *Input gate* (which information should enter the cell state) : Gates are a key component of the LSTM approach. A gate is a sigmoidal unit that controls the flow of information between past and following units by multiplying with its value, the value of the input node; whether to enable or cut-off the flow from a unit to another, by setting the value of the input gate equal to one or zero respectively.

- *Internal state*(the long-term memory): This unit at the heart of each memory cell, is the component that effectively tackles the vanishing gradients issue. The internal state h_c has a self-connected recurrent edge which spans adjacent time steps, with constant weight one, allowing the error to flow across time steps without having extreme values (vanishing or exploding). Mathematically formulated, the update for the internal state is:

$$\mathbf{s}(t) = \mathbf{g}(t) \odot \mathbf{i}(t) + \mathbf{s}(t-1) \tag{2.3}$$

- *Forget gate* (which decides which information should be forgotten from the previous cell state): These gates f_c, introduced by Gers et al in [35], provide a method by which the network can learn to flush the contents of the internal state. Adding forget gates, modified the update of the internal state as following:

$$\mathbf{s}(t) = \mathbf{g}(t) \odot \mathbf{i}(t) + \mathbf{f}(t) \odot \mathbf{s}(t-1) \tag{2.4}$$

- *Output gate*: The output value of a memory cell u_c is the value of the internal state, passed through a non-linear activation function and multiplied by the output gate o_c. Hence the output gate determines which information should be moving to the next hidden state.

In a nutshell, the equations that describe any LSTM architecture are the following:

$$\begin{aligned}
\mathbf{g}(t) &= \phi(\mathbf{W}^{xg}\mathbf{x}(t) + \mathbf{W}^{hg}\mathbf{h}(t-1)) + \mathbf{b}_g \\
\mathbf{i}(t) &= \sigma(\mathbf{W}^{xi}\mathbf{x}(t) + \mathbf{W}^{hi}\mathbf{h}(t-1) + \mathbf{b_i}) \\
\mathbf{f}(t) &= \sigma(\mathbf{W}^{xf}\mathbf{x}(t) + \mathbf{W}^{hf}\mathbf{h}(t-1) + \mathbf{b}_f) \\
\mathbf{o}(t) &= \sigma(\mathbf{W}^{xo}\mathbf{x}(t) + \mathbf{W}^{ho}\mathbf{h}(t-1) + \mathbf{b_o}) \\
\mathbf{s}(t) &= \mathbf{g}(t) \odot \mathbf{i}(t) + \mathbf{f}(t) \odot \mathbf{s}(t-1) \\
\mathbf{h}(t) &= tanh(\mathbf{s}(t)) \odot \mathbf{o}(t)
\end{aligned} \tag{2.5}$$

Instead of tanh, any non-linear activation function can be used. However, tanh is the typical activation used in this case. The following figure (Fig 2.8) gives an optical illustration of the computations done inside an LSTM cell.

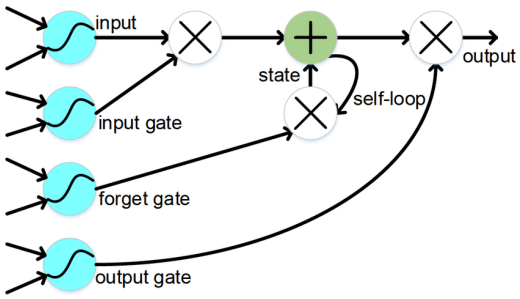

Figure 2.8: LSTM cell with optical explanation of the computations inside an LSTM cell. This figure was adopted from [30]

2.2 Baseline Algorithms

2.2.1 Gaussian Mixture Models

Gaussian mixture models (GMMs) are probabilistic models that assume all the data points are generated from a mixture of a finite number of Gaussians, whose parameters are to be learnt during the training process. The aforementioned mixture is a weighted sum of M component Gaussian densities as given by the equation:

$$p(\mathbf{x}|\lambda) = \sum_{i=1}^{M} w_i \; g(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i),$$

where $\mathbf{x} \in \mathbb{R}^D$ is the data vector, each $g(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, $i = 1, ..., M$ is a component Gaussian density and w_i, $i = 1, ..., M$, are the mixture weights which satisfy the following constraint:

$$\sum_{i=1}^{M} w_i = 1$$

Each component density $g \in \mathbb{R}^D$ is a function of the form,

$$g(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \; exp\left\{ -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)' \; \Sigma_i^{-1} \; (\boldsymbol{x} - \boldsymbol{\mu}_i)\right\} \tag{2.6}$$

with mean vector $\boldsymbol{\mu}_i$ and covariance matrix Σ_i.

The parameters needed for the description of a complete GMM, are mean vector, covariance matrices, and mixture weights for each component. All these parameters, are represented jointly by the notation:

$$\lambda = \{w_i, \boldsymbol{\mu}_i, \Sigma_i\} \quad i = 1, ..., M. \tag{2.7}$$

Each of the components included in equation 2.7 can vary, and can be estimated under some constraints. Parameters can be set to be shared or tied between all the Gaussian components, and covariance matrices can be constrained to be diagonal or spherical (diagonal with equal elements in the diagonal) instead of full rank Depending on the amount of available data and the specific use of the GMM, one can choose the model configuration, such as the aforementioned constraints and the number of the gaussian components.

The learning process of a GMM consists of the estimation of the parameters λ which best match to the distribution of a given set of training vectors. The two most popular and well-established techniques for the estimation of the parameters of a GMM are Maximum Likelihood (ML) Parameter estimation and Maximum A Posteriori (MAP) Parameter Estimation. In this work, the focus will be on the ML estimation, since this method is used in the following experiments. The goal of ML estimation is to find the model parameters which maximize the likelihood of the GMM given a set of training samples. In detail, given a sequence of M training samples, $x = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_M$, and assuming data samples are independent[2] of each other, GMM likelihood can be expressed as:

$$p(X|\lambda) = \prod_{t=1}^{T} p(\boldsymbol{x}_i|\lambda), \tag{2.8}$$

Since the expression is a non-linear function, the parameters of Gaussian mixtures cannot be estimated in closed form. A widely used iterative process is ML via the expectation-maximization (EM) algorithm [22]. The basic idea of the EMm algorithm is, beginning with an initial model λ, to estimate a new model $\hat{\lambda}$, such that $p(X|\hat{\lambda}) \geq p(X|\lambda)$. Then this step is repeated, by setting the initial model equal to the new one, until convergence according to a given threshold (Fig **??**). However, finite mixture modeling in general often suffers from convergence to locally optimal solutions. Therefore, EM algorithm is sensitive to initialization, since the initial model has to be given along with the data. Many initialization techniques have been examined, such as random initialization and initialization via vector quantization [79]. The formulas used on each iteration (Eqs. 2.9-2.11) ensure a monotonic increase in the model's likelihood values until convergence [22], thus, despite of the sensitivity to initialization, the algorithm is deterministic and given the same initialization setting and data, will produce the same results.

[2]The independence assumption is not always correct but it is necessary in order to make the problem manageable.

$$\textit{Mixture Weights} \qquad \hat{w}_i = \frac{1}{T}\sum_{t=1}^{T} Pr(i|\boldsymbol{x}_i, \lambda) \tag{2.9}$$

$$\textit{Means} \qquad \hat{\boldsymbol{\mu}}_i = \frac{\sum_{t=1}^{T} Pr(i|\boldsymbol{x}_i, \lambda)\ \boldsymbol{x}_t}{\sum_{t=1}^{T} Pr(i|\boldsymbol{x}_i, \lambda)} \tag{2.10}$$

$$\textit{Variances (diagonal covariance)} \qquad \hat{\sigma}_i^2 = \frac{\sum_{t=1}^{T} Pr(i|\boldsymbol{x}_i, \lambda)\ \boldsymbol{x}_t^2}{\sum_{t=1}^{T} Pr(i|\boldsymbol{x}_i, \lambda)} - \hat{\mu}_i^2 \tag{2.11}$$

where σ_i^2, x_i and μ_i refer to arbitrary elements of the vectors $\boldsymbol{\sigma}_i^2$, $\boldsymbol{x}_i$ and $\boldsymbol{\mu}_i$, respectively.

Furthermore, the *a posteriori* probability for component i is given by

$$Pr(i|\boldsymbol{x}_t, \lambda) = \frac{w_i\ g(\boldsymbol{x}_t|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{k=1}^{M} w_k\ g(\boldsymbol{x}_t|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)} \tag{2.12}$$

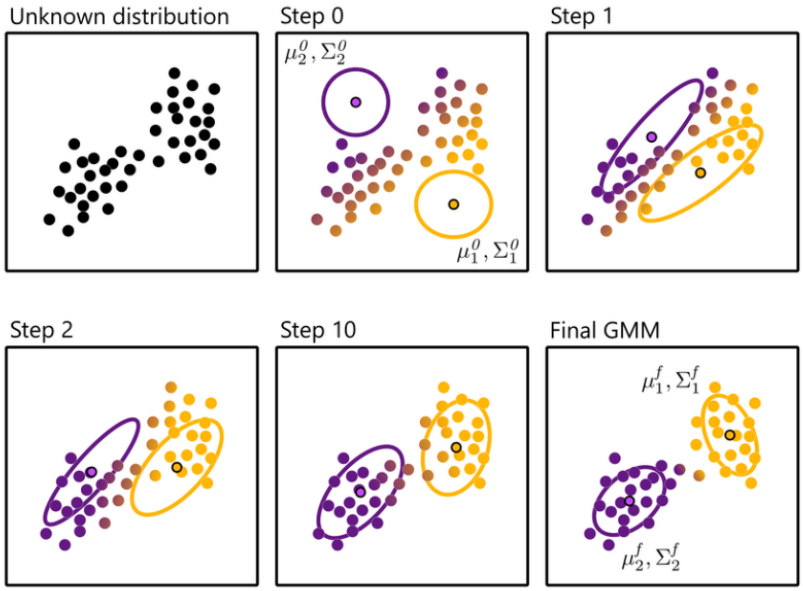

Figure 2.9: Illustrative example of EM algorithm run

2.2.2 Support Vector Machines

Support Vector Machines (SVMs), introduced by Vapnick [18], are supervised learning models used either for classification or regression analysis. Given a set of training data, the SVM training algorithm builds a model that assigns new examples to one of the two categories, making it a deterministic binary linear classifier. With an SVM model data points are represented as points in the feature space, classified so, that examples of separate categories are divided by a decision boundary and margins, forming a clear gap, as wide as possible. Then, in testing phase, new examples are predicted to belong to a category based on which side of the decision boundary they belong to.

The problem of binary SVMs can be stated as follows: Given the training samples (x_i, d_i), i=1, ..., N where $\mathbf{x}_i$ is an n-dimensional input feature vector and $d_i \in \{-1, +1\}$ represent the corresponding labels for the two classes of sounds, the decision function is given by:

$$f(\boldsymbol{x}, \boldsymbol{w}, b) \; = \; sign(\boldsymbol{w}^T \boldsymbol{x} + b)$$

where $\boldsymbol{w} \in \mathbb{R}^d$ is a weight vector and $b \in \mathbb{R}$ is a bias value. If the two classes are linearly separable, the optimal hyperplane can be determined from:

$$\begin{aligned} &minimize \; \frac{1}{2}||\boldsymbol{w}||^2 \\ &subject \; to \;\; d_i(\boldsymbol{w}_i \boldsymbol{x}_i + b) \geq 1, \qquad \text{i=1, ..., N} \end{aligned}$$

When the data samples are not linearly separable, the SVM can be transformed in a non-linear classifier by applying the so-called *kernel trick*[3] , to construct maximum-margin hyperplanes (Fig. 2.10) [9]. Kernel classifiers were first introduced by Aizerman et al, whith the invention of the kernel perceptron [4].

Suppose we have data $\boldsymbol{x}, \boldsymbol{y} \in X$ and a mapping $\phi \colon X \to \mathbb{R}^N$, a kernel k, where $k \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, is a function which takes as input two vectors $\boldsymbol{x}, \boldsymbol{y}$ from the original space and returns the dot product of the images of these vectors $\phi(\boldsymbol{x}), \phi(\boldsymbol{y})$ in the destination feature space (typically higher dimensional).

$$k(\boldsymbol{x}, \boldsymbol{y}) \; = \; \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle \tag{2.13}$$

The optimization problem of Eq. 2.12 -2.13 can then be reformulated as:

$$\begin{aligned} &minimize \;\; \frac{1}{2}||\boldsymbol{w}||^2 \; + \; C \sum_{i=1}^{N} {}_i \\ &subject \; to \;\; d_i(\boldsymbol{w}_i \boldsymbol{x}_i + b) \geq 1 - \xi_i, \xi_i \geq 0; i = 1, ..., N \end{aligned}$$

[3] *Kernel trick* is referred to the fast computation of kernel functions over data points, where instead of computing data coordinates in that space, it is sufficient to compute the inner product between images of all pairs of data in feature space. The only restriction on the kernel function is that $\langle \cdot, \cdot \rangle_\nu$ must be a proper inner product.

where C is known as the regularization parameter. The decision function is given by:

$$f(x) = sign([\sum_{i=1}^{N} \alpha_i d_i k(\boldsymbol{x}, x_i) + b]$$

where α_i are called Lagrange multipliers and $k(\boldsymbol{x}, x_i)$ are the Kernel functions.

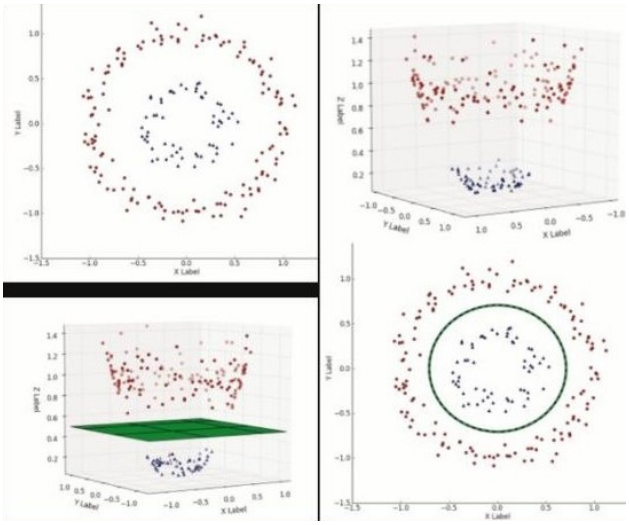

Figure 2.10: Illustrative example of decision boundary of an SVM model on a non linearly separable dataset, after applying an RBF (Gaussian) kernel. Adopted from [73].

Multi-class classification with SVM classifier is not straightforward, as it is natively a binary classifier. One approach is to split the mutli-class dataset into multiple binary datasets, and train a binary classification model on each sub-problem. Two examples of this approach is the One-vs-One and the One-vs-Rest strategy. In the One-vs-Rest strategy, a classifier is trained for each class, by merging all the rest classes into one. In this strategy, base classifiers need to produce a real-valued confidence score, as discrete labels alone can lead to ambiguities, where multiple classes are predicted for a single sample [8]. In the one-vs-one (OvO) reduction, the training requires $\frac{K \cdot (K-1)}{2}$ binary classifiers for a multiclass problem of K classes. Each of these classifiers, is trained to make predictions between a pair of classes from the original training set. Then, at prediction time, a voting scheme is applied, where all trained classifiers are deployed and the class with the highest number of positive predictions gets predicted by the combined classifier [8].

2.2.3 Random Forests

First introduced by Ho [81] in 1995, Random Forests or Random Decision Forests is a supervised learning algorithm, which can be applied both in classification and

regression tasks. During training, an ensemble of decision trees is built, with the use of the *bagging*[4] method, and in order to make predictions, they output the most popular class among the output of the trees consisting the forest. Decision trees are prone to overfitting and underperforming in unseen data, but with Random Forests, this issue is tackled up to a point, when a sufficent number of estimators is used. Their disadvantage compared to decision trees is the lack of interpretability.

A decision tree is a tree which is built in such a way that each internal node is labeled with an input feature, and represents a test on this attribute (e.g. whether a coin flip comes up heads or tails) . Each of these nodes has a number of children equal to the number of possible outcomes of this test (e.g. the possible values of the input feature). In this way, each branch separates the dataset into smaller sets. Finally, each *leaf node*[5] is labeled with a class or a probability distribution over the classes, meaning the data points included in current leaf have been classified into a specific class.

The splitting of each node, is based on a the importance of each feature which is derived from a chosen criterion. The free choice of that criterion makes it also a parameter of the training. A widely used criterion is the information gain (or mutual information), which is a special case of Kullback-Leibler (KL) divergence (Eq. 2.14). Each split aims to maximize the mutual information between a feature distribution and the label distribution, which is equivalent to minimizing entropy. In other words, IG measures how much information a feature gives about the class of a sample, and hence, the feature with highest information gain will be included first in the tree.

$$IG(X;Y) = \sum_{x,y} p(x,y) log_2\big(\frac{p(x,y)}{p(x)p(y)}\big) = D_{KL}(P(X,Y)||P(X)P(Y)) \quad (2.14)$$

Using an importance criterion for the features, a feature ranking can be obtained from the decision trees training (and hence random forests training), based on their correlations and their relevance with the problem. This importance can be obtained both as a scores vector, or directly from the tree graph; the closer the split to the root of the tree, the higher the importance of the corresponding feature. However, the impurity based ranking has some flaws. When the dataset has correlated features, then the classifier chooses any of them as predictor with no preference, resulting to the significant reduction of the rest correlated features importance. This fact is not an issue when the target of feature selection is to reduce overfitting, but it can lead to misinterpretation of the data if it is used in exploratory research.

[4]The idea of the bagging method is that with a combination of learning models the overall performance increases.

[5]A leaf node is a node without children

Chapter 3

Dataset

The evaluation results shown in this Thesis are based on real acoustic data, captured in various protected areas in Greece. In this chapter our intention is to describe how the original data was acquired and to provide a qualitative description of the background noise and the underlying sound patterns that exist in the dataset. Moreover, approaches to augment the dataset are described as the means to improve the classification performance.

3.1 Autonomous Recording Units (ARUs)

The dataset was developed by processing raw 24/7 recordings obtained from SWIFT ARUs (Fig. 3.1a) developed by Cornell University's Lab of Ornithology (`www.birds.cornell.edu/brp/swift/`). Each ARU consists of an omnidirectional analog microphone (PUI Audio Inc., Part Nr: POW-1644L-B-LW100-R, see Fig. 3.1b) protected by a military high density windscreen (WindTech 10380, see Fig. 3.1c). The above battery powered microphones have a signal to noise ratio (SNR) of > 58 dB and a frequency response of 50Hz to 16kHz. The recorded signals are in PCM format using 8kHz sampling rate and 16 bits of resolution. Each ARU is connected to a specific location in Greece and to a specific period of time when its data was acquired. More details can be seen in Table 3.1.

(a) SWIFT ARU

(b) Microphone cap

(c) Windscreen cap

Figure 3.1: ARU setup

ARU	season	year	location
EVR	spring	2019	Evros
PR	winter	2016	Prespes
RP1	autumn	2019	Rhodope
RP2	summer	2019	Rhodope
RP3	summer	2019	Rhodope
RP4	summer	2019	Rhodope
RP6	summer	2019	Rhodope
RP10	summer	2019	Rhodope
RP11	summer	2019	Rhodope
RP14	summer, autumn	2019	Rhodope
RP15	autumn	2019	Rhodope
SW1	autumn	2017	Maroneia
SW2	winter	2018	Evagelistria
DS	autumn	2018	Crete

Table 3.1: Location, season and year for each one of the ARUs.

3.2 Preprocessing and dataset creation

Recordings obtained from each ARU have duration of several hours and the sound events of interest are sparsely distributed within them. This means that a human subject would need to spend hundreds of hours to listen to the audio content in order to discriminate sound portions with audio events that are meaningful for the purpose of this research. To facilitate data collection, we propose the use of an automatic segmentation process, described in more detail in Section 4.1.2. Using this automated process, a long duration recording is reduced to a manageable number of utterances, where each utterance last a few seconds. Then each of these utterances was labeled manually into the 9 categories: "saw", "vehicle", "aeroplane", "mammal", "insect", "bird", "dog", "bell", "other". The first 6 categories were found to be representative of the acoustic content that mostly triggered the voice activity detection algorithm in the automatic segmentation process. The number and the categories of audio samples gathered from each ARU can be seen in Table 3.2.

After the data gathering, each extracted sample was labelled manually into one of the mentioned classes, in order to form a weakly labelled dataset.

Apart from the chainsaw sounds, several "bell" instances were captured due to grazing animals such as goats and sheep. Several detections were triggered by aeroplanes and also by cars and tracks that happened to pass within the acoustic range of each ARU. These instances were categorized generally as "aeroplanes" and "vehicles" respectively. Finally, there were several cases of human voice and also events that it was not possible to distinguish if it was an animal, a human, a bird or a dog. These cases were categorized as "other". In addition to the aforementioned categories, one additional category "empty" was added, characterized by the lack of any foreground sound source and consisting mainly of background noise.

As we search for illegal human activity, and especially for illegal logging, the choice of the above class labels was made based on the information that each class provides for our task and on the plethora of samples of each class;

- saws → illegal logging,
- vehicle → possible illegal logging,
- mammal → if it's human voice → human presence
- bell → illegal grazing
- dogs (+gunshot) → illegal hunting
- insect → harmonic structure much alike saw, so its important to include it in order to guide the algorithm to distinguish between saw and insect.
- aeroplane → Many aeroplane occurences were segmented by the algorithm that could form a single class.

- bird → Many segments include bird sounds, that could form a single class
- other → All segments resulted from the VAD algorithm that had indistinguishable content, or seemed to have only background noise

3.2.1 Difficulties

Analyzing the dataset and comparing the number of samples for each ARU, the amount of data is not equally distributed among ARUs neither among classes. Hence, an imbalancement is introduced to the dataset. This may result to data overfitting during the training of a classifier and lead to a model that does not generalize well. In the latter case, the classifier may be biased towards the classes with the most occurences.

In should be also noted that in several cases, sounds from more than two categories could be heard simultaneously. In most cases, this was due to the sound produced by airplanes, giving rise to long utterances. To handle cases of simultaneously occurring patterns, we followed the rule that each utterance with more than one categories should be categorized as the category with the highest priority based on our detection task. The label priorities from high to low are the following:

- 9 class case "saw", "vehicle", "aeroplane", "mammal", "dog", "bell", "insect", "bird", "other+empty"
- 6 classes case "saw", "vehicle+aeroplane", "mammal+dog+bell", "insect", "bird", "other+empty"
- 3 classes case "saw", "insect", "other"
- 2 classes case (Binary) "saw", "other"

3.3 Data augmentation

Data augmentation is a technique to widen a dataset and increase the number of samples. Especially when training Deep Neural Networks, the number of training samples plays an important role in the performance, since it contributes to better generalization. In the current dataset, the number of the original audio clips, i.e. those without data augmentation, is 21782 (Table 3.2). This data was artificially enriched using Matlab's built-in function *resample* in order to downsample and upsample each audio segment by 5%. Through this process, both the duration and the pitch of the input audio segment is altered. This was preferred against classical pitch shifting operation as it was observed that conventional pitch shift introduced audible artifacts in the produced signals, a fact that can be possibly explained by the low SNR in the audio recordings. Following the resampling operation, two additional utterances are produced for each available audio segment

(except utterances of class "empty"), totaling in 62844 samples in the augmented dataset. (Table 3.3)

ARU	saw	vehicle	aeroplane	mammal	dog	insect	bird	other	bell	empty	All classes
DS	433	0	0	0	0	0	0	0	0	0	433
EVR	11	78	14	308	70	0	251	269	4	0	1005
PR	154	203	50	521	156	2	60	170	41	216	1573
RP1	0	0	129	0	0	344	2	6	6	0	487
RP2	0	0	75	1	0	51	1	0	0	0	128
RP3	78	30	496	36	3	961	617	141	0	0	2362
RP4	0	2	22	195	0	14	23	12	0	0	268
RP6	1747	0	190	436	8	67	43	0	388	0	2879
RP10	129	0	421	0	0	320	10	38	0	469	1387
RP14	126	0	40	3	5	24	40	17	8	374	637
RP15	46	110	23	0	60	2	149	49	14	192	645
SW1	0	42	660	125	62	102	48	0	683	0	1722
SW2	3921	94	1305	1548	795	141	82	123	247	0	8256
SUM	6645	559	3425	3173	1159	2028	1326	825	1391	1251	21782

Table 3.2: Original dataset

ARU	saw	vehicle	aeroplane	mammal	dog	insect	bird	other	bell	empty	All classes
DS	1299	0	0	0	0	0	0	0	0	0	1299
EVR	33	234	42	924	210	0	753	807	12	0	3015
PR	462	609	150	1563	468	6	180	510	123	216	4287
RP1	0	0	387	0	0	1032	6	18	18	0	1461
RP2	0	0	225	3	0	153	3	0	0	0	384
RP3	234	90	1488	108	9	2883	1851	423	0	0	7086
RP4	0	6	66	585	0	42	69	36	0	0	804
RP6	5241	0	570	1308	24	201	129	0	1164	0	8637
RP10	387	0	1263	0	0	960	30	114	0	469	3223
RP14	378	0	120	9	15	72	120	51	24	374	1163
RP15	138	330	69	0	180	6	447	147	42	192	1551
SW1	0	126	1980	375	186	306	144	0	2049	0	5166
SW2	11763	282	3915	4644	2385	423	246	369	741	0	24768
SUM	19935	1677	10275	9519	3477	6084	3978	2475	4173	1251	62844

Table 3.3: Augmented dataset

Chapter 4

Methodology

4.1 Voice Activity Detection (VAD) in chainsaw detection

In the course of this research, we realized that there are several acoustic patterns of interest with harmonic structure that would likely trigger the proposed VAD. Chainsaw, insects, mamals, vehicles, birds and of course human voice are some examples of such sounds. The VAD is exploited in order to create a tool to automatically segment a long duration audio recording into a large number of short duration audio clips that potentially carry an acoustic pattern of interest. This has a huge impact in the time required for collecting training data, since we can directly assign a label to each extracted segment and we don't need to listen to the entire duration of the recording in order to spot interesting events. Moreover, the automatic segmentation process can be used in the implementation phase in order to filter our background noise and uninteresting content, thus reducing the amount of data that has to be presented to the classifiers.

Our VAD is based on the Summation of Residual Harmonics method (SRH method) [27], which is known for it's ability to provide reliable voicing decisions in noisy conditions. The outcome of the SRH approach that we utilize is a time-varying metric of the voicing activity that is called Voicing Strength (VS). Voicing activity is returned as a time-series in the form $v(\tau)$, where τ is the time-frame index. This metric is calculated using the relevant function provided in the COVAREP toolbox [21]. VS can give very accurate results for voiced/unvoiced segments, using thresholding. Thus, we make use of some thresholds to get harmonic/non harmonic decisions (in accordance to voiced/unvoiced). Furthermore, in an intermediate step of the SRH algorithm, the algorithm calculates a spectro-temporal representation of the pre-whitened signal, that may be directly used as an acoustic feature for classification. We will refer to this feature as "*SRH spectrogram*".

4.1.1 Summation of Residual Harmonics (SRH)

The residual harmonics method [27] is a well known approach in the context of pitch tracking and voice activity detection. This method focuses on the residual harmonicity and more specific on the spectrum of the residual signal, instead of auto-correlation that other methods use. The procedure for calculating SRH is the following:

1. Perform auto-regressive modeling of the spectral envelope for a specified number of time-lags.

2. Perform inverse filtering with the auto regressive model in order to obtain the residual signal $e(t)$. This whitening process has the advantage of removing the main contributions of both noise and possibly the ARU response (which may be compared to the vocal tract resonances).

3. At each time-frame a hanning window is applied and the amplitude spectrum $E(f)$ is computed. An important notice on $E(f)$ is its relatively flat envelope. For segments that have harmonic structure (for voiced segments in terms of voice activity detection), peaks at the harmonics of the fundamental frequency F_0 will appear. From this spectrum, and for each frequency in the range $|F_{0,min}, F_{0,max}|$ the SRH is computed as: $SRH(f) = E(f) + \sum_{k=2}^{N_{harm}}[E(k \cdot f) - E((k - \frac{1}{2} \cdot f)]$. Considering only the term $E(k \cdot f)$ in the equation, only the contribution of the first N_{harm} harmonics are taken into account, and is expected to reach a maximum for $f = F0$. However, this also holds for each of the harmonics present in the range $[F_{0min}, F_{0max}]$. Consequently, the subtraction by $E(k\frac{1}{2} \cdot f)$ is important to reduce the relative importance of the maxima of SRH at the even harmonics. Then, an estimated pitch value F^0 is extracted for each residual frame is the frequency that maximizes $SRH(f)$ at that time.

The proposed criterion for detecting harmonicity is acknowledged for its robustness to noisy conditions, which makes it a promising candidate for the intended task, since sources with harmonic content are submerged in high levels of background noise. In what follows, we explain how the SRH method is exploited in order to automatically segment the audio recordings.

4.1.2 VAD-based segment selection procedure

Our dataset consists of 24 hr recordings that span many different days and many different places (Table. 3.1). Thus the manual cropping and annotation of the recordings is inefficient. The idea here us to use VS as a metric to detect audio portions with harmonic structure in each recordings and present these short duration segments directly to a human listener for annotation. The patterns of interest that relate to illegal human activity and that exhibit a harmonic structure are chainsaw (illegal logging), vehicles (illegal trespassing), sheep bells (illegal

grazing), human voice and dog barking (possibly illegal hunting). Due to the way that the VAD algorithm works, other patterns that don't imply human activity but do have a harmonic structure are also expected to produce high VS values. Such patterns are birds, insects, aeroplanes and various mammals that happen to pass at a close distance from the ARU.

An important element of the process is that patterns like chainsaw or vehicles or dog barking produce high VS values not in single isolated time-frames but along continuous time frames, in contrast to e.g. gunshot sound that usually occurs in single isolated time frames. Following this idea, audio clips are kept only if there are at least N_{ca} consecutive time frames with high VS values that are over a threshold T_{srh}. The procedure that is followed for the segment extraction in more detail is;

1. Calculate VS in terms of $v(\tau)$ across the entire duration of the recording

2. Construct the collection of all the active time-frame indexes, where an active time-frame is defined as any time-frame where condition $v(\tau) \geq T_{srh}$ holds. In what follows, let us use the term *utterance* in order to refer to a collection of consecutive active time-frames starting at time τ_u^{start} and ending at time τ_u^{end}, where u is the time-frame index.

3. Keep only the utterances which are formed by more than N_{ca} consecutive time-frames, i.e., $\tau_u^{end} - \tau_u^{start} > N_{ca}$ holds.

4. If there are utterances with lengths smaller than N_d, extend their lengths to become equal to N_d. Perform this operation by simply reducing τ_u^{start} and increasing τ_u^{end} an equal number of time-frames.

Following this operation, several utterances of different lengths are produced from each audio recording and the length of each utterance is at least N_d time-frames. In general $N_{ca} < N_d$ holds, meaning that the number of active time-frames observed from a certain event can be much less that the total number of time-frames in the final extracted segment. Also, this condition will allow temporal overlap between consecutive utterances. The reason for using a different value for N_d and N_{ca} is that a chainsaw revving may last several multiples of N_{ca}, but due to the high level of noise or due to the non-stationary nature of the particular chainsaw instance, only a small portion of the event passes the $v(\tau) \geq T_{srh}$ criterion. Additionally, the tactic to extend the acoustic representation along time makes it easier for the listeners to annotate the data, possibly by allowing them to better perceive the transitions at the beginning or end of the extracted event.

4.2 Feature extraction for DNN training

4.2.1 Selection of frequency range

The frequency range of the acoustic features is an important choice since it affects the final features' dimension and thus affects both the performance and the computational cost of the implementation. The sampling rate of 8kHz, allows for frequency analysis up to 4kHz, thus we examined several cases of frequency ranges, depending on the feature. The analysis frequency range for the SRH method was 40Hz-2kHz, whereas in the extracted SRH-spectrogram the frequency range was cropped to 40Hz-540Hz. Considering Mel-spectrogram, in the case of limited frequency range, and especially up to 1kHz, mel filter would be approximate linear and result in a slightly modified linear frequency axis. Thus, Mel-spectrogram was tested in frequency range 0-4kHz, so mel filter would span in a wide frequency range to result in a spectrogram with logarithmic mapping in frequency axis. Power spectrograms were examined with frequency ranges 80Hz-4kHz and 150-2kHz. The following facts suggest that it might be advantageous not to consider the entire frequency range but to limit the highest frequency in the analysis well below the Nyquist rate;

- Chainsaw units rely mainly on an internal combustion engine which operates at 6000-18000 rpm, which results in an acoustic signal with fundamental frequency f_0=100-300Hz and period of T_0=3-10ms (26-80 samples for the current dataset sampled at 8kHz) [66]. Thus, the frequencies with highest energy are at the low and middle frequency range.

- Sound sources are most of the times located far from the ARU. Due to long range acoustic propagation, the characteristic sound patterns are detected within each recording at very low signal levels relative to background noise. As a consequence, the higher frequency harmonics - which usually are weaker than the lower harmonics and the fundamental frequency - are masked by background noise and thus not detectable within the observed audio signal.

- During long range propagation, sound waves attenuate in quadratic proportion to their frequencies [42]. As a consequence, high frequency components often vanish and are not observable within the audio recordings.

4.2.2 Normalization of raw audio

Before feature extraction, normalization was used for each utternace for all features (except SRH spectrogram) as $s_{norm} = \frac{\sqrt{N}}{\|\mathbf{s}\|_2}\mathbf{s}$, where $\mathbf{s}$ is the PCM signal observed in the utterance and N denotes the number of samples in each utterance. This is a variation of the widely used normalization of dividing signal by its norm. The square root of N was added in the nominator to adjust the normalization depending on the length of the utterance. We note that that this process is not

required for SRH feature thanks to the prewhitening process, resulting to a feature that is neutralized with respect to the signal level.

4.2.3 Power Spectrogram and Mel-spectrogram

Power spectrogram and mel-spectrogram are two of the most common spectro-temporal representations used as features for DNN. For constructing the power spectrogram, the temporal parameters of frame-size and hopsize were empirically tuned to values of 60 and 30 ms respectively. Each time-frame is windowed using a Hann window, while the Fast Fourier Transform (FFT) is used with 2048 points, resulting in a 1024 points spectrum. The squared magnitude of the spectrogram was used a the final feature while the phase was disregarded. For the needs of this Thesis, mel-spectrogram was constructed by multiplying the power spectrogram with a mel filterbank that directly maps the linear frequency axis onto the mel scale.

4.2.4 Per Channel Energy Normalization (PCEN)

PCEN has been proposed as a transformation of spectrotemporal features with the goal to derive acoustic features that lead to better generalization during DNN training. It is proposed as an alternative to log mel spectrogram or log spectrogram with the aim to increase robustness to channel distortion. PCEN combines a dynamic range compression (DRC), which reduces the variance of foreground loudness and an adaptive gain control (AGC) which is used to suppress stationary background noise. In contrast with log mel spectrogram which uses a static log or root compression, a key component in PCEN is the replacement of this compression with a dynamic compression. The PCEN operation can be written as

$$PCEN(t,f) = \left(\frac{E(t,f)}{(\epsilon + M(t,f))^a} + \delta\right)^r - \delta^r, \tag{4.1}$$

$$M(t,f) = (1-s)M(t-1,f) + sE(t,f), \tag{4.2}$$

where t and f are the time and frequency indices respectively, $E(t,f)$ indicates filterbank energy , $M(t,f)$ is a smoothed version of $E(t,f)$ traditionally computed using a first-order auto regressive process defined in Eq. (4.2), with parameter s used as a smoothing coefficient and ϵ used as a small constant for division stability. Equation (4.1) can be broken up in two parts, where the first part implements a form of feed-forward AGC with parameter $\alpha \in [0,1]$ used for gain normalization strength (Eq. 4.3) and the second part implements a stabilized root compression with parameters δ (offset) and r(exponent), which further reduces dynamic range (Eq. 4.4).

$$T_{gn}(t,f) = \frac{E(t,f)}{(\epsilon + M(t,f))^\alpha} \tag{4.3}$$

$$T_{comp}(t,f) = (T_{gn}(t,f)+\delta)^r - \delta^r \tag{4.4}$$

PCEN is applied on already extracted power spectrograms , in order to compare its performance to the rest of the features. As parameters, we used a fixed set of parameters, matching with the ones presented in the original paper, where s = 0.025, $\alpha = 0.98$, $\delta = 2$ and r = 0.5.

4.2.5 SRH spectrogram

The SRH method has been already described in Section 4.1.1 as an unsupervised approach to detect patterns with harmonic structure in an audio recording. The procedure followed in order to extract $v(\tau)$ at each time frame involves calculation of a matrix that will be called from now on as the SRH spectrogram. This intermediate product of the SRH method is the result of obtaining the spectrogram on the prewhitened signal and then applying the summation of residual harmonics. To our knowledge, SRH spectrogram is used for the first time as a potential acoustic feature for DNN classifiers in this Thesis. An illustrative example of the SRH spectrogram is presented in Fig. 4.1, adopted from the original SRH paper [28]. It can be observed that due to the subtraction that takes place in the calculation, SRH spectrogram consists of negative values apart from positive ones. We finally note that the paremeters of 60 ms frame-length and 30 ms hop-size also hold for the SRH spectrogram. A final limit concerns the upper frequency limit used for construction of the SRH spectrogram which is set to 500 Hz. Due to the way that higher frequencies are reflected upon the lower frequencies (see Section 4.1.1), it should be realized that there is no one-to-one relation between frequency index and natural frequency, rather each frequency index in the SRH spectrogram is also representative of the energy contained in higher harmonics of that frequency index.

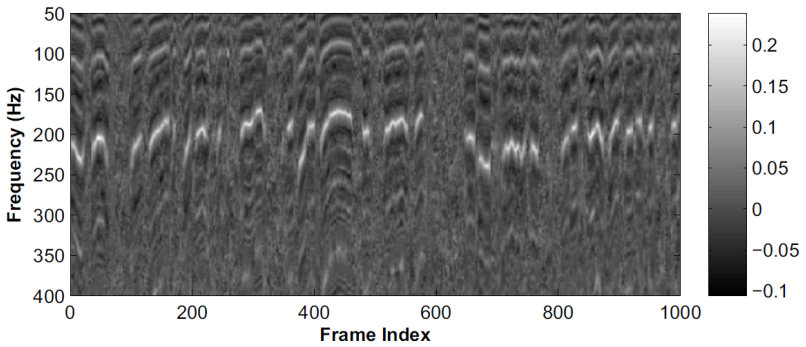

Figure 4.1: SRH spectrogram for a segment of clean speech of a female speaker. Adopted from the original paper of SRH [28]

4.2.6 Final feature dimension

The acoustic features intended for DNN classification can be presented as two-dimensional matrices where one axis is the time dimension and the other axis is the frequency dimension. The frequency dimension is set to 500 bins for power spectrograms and SRH-spectrogram. For the mel spectrogram the frequency dimension is equal to the number of mels (e.g. 128 bins for 128 mel spectrogram). However, we need a fixed size also along the temporal dimension (especially for the CNNs), and the value that was decided was 48 time-frames for all features. This number of frames corresponds also to the minimum utterance duration produced using the automatic segmentation process of Section 4.1. In order to achieve a fixed feature dimension for utterances of longer duration, a downsampling procedure was implemented. Specifically, assuming that N_{tf} is the number of time-frames extracted from a specific utterance, a child feature is obtained by downsampling along the temporal dimension with a factor equal to $\lfloor \frac{N_{tf}}{48} \rfloor$. From the downsampled spectrogram 48 frames around the center are kept, in this way, we can make a spectrogram of any length fit into the desirable number of frames without zero padding.

To provide an example, a 3 seconds utterance will result in a spectrogram consisting of 100 frames. This spectrogram will be downsampled by a factor of ([100/48] =)2 to 50 frames, and 48 frames around center of the spectrogram will be kept as feature, discarding 2 frames. Two extreme cases is the one that the spectrogram consists of a number of samples which is an exact multiple of 48 and the one that the spectrogram consists of a number of samples which is multiple of 48 plus 47. In the first case, all the (downsampled) information available will fit in the desired number of frames, without discarding any excessive frames. In the latter case, more frames will be discarded in the last step. e.g. A spectrogram consisting of 191 samples would be downsampled to 63 frames (by a factor of 3), discarding 15 frames. In all cases, more than 50% of the information around the center of the clip will be retained.

4.3 DNN architectures

Two main types of architectures where tested, namely :

- CNN - based.
- CNN+LSTM combined.

In both cases experiments where conducted using different number of layers and units per layer, and 4 architectures were chosen to be presented. The optimization algorithm used was Adam [48] and learning rate followed Early Stopping and Reduce LR on plateau schemes. Reduce LR on plateau scheme reduces learning rate during training by a factor of 0.2 when validation loss has stopped improving

for five consecutive epochs[1]. Early stopping, terminates training procedure when validation loss has stopped improving for 10 consecutive epochs. The procedure of training and testing was completed using the Keras Deep Learning framework [13] and the Tensorflow [1] framework as backend. To boost training time, we also used the NVIDIA CUDA DNN library cuDNN library [12], which is a GPU-accelerated library for Deep Learning. The hardware specifications of the underlying system were an Intel i7-8550U CPU, an NVIDIA GeForce MX150 (2GB VRAM) GPU and a 20GB RAM.

In all cases, the last activation layer is a softmax layer. Softmax layers in general take N inputs and normalize them into a probability distribution consisting of N probabilities. Expressed mathematically :

$$\sigma(\mathbf{z})_i = \frac{e^{\mathbf{z}_i}}{\sum_{j=1}^{N} e^{\mathbf{z}_i}}$$

for $i = 1 \dots N$ and $\mathbf{z} = (z_1, z_2 \dots z_N) \in \mathbb{R}^N$ [85]. Since we conduct experiments for varying number of classes, the softmax layer consists of 2, 3, 6, or 9 inputs / outputs, matching the number of classes. In this context, each softmax layer produces as many probabilities as the number of classes that sum to 1.

4.3.1 CNN based architectures

Convolutional layers in this Thesis are used with one-dimensional kernels that are applied in the time domain. Multiple layers are stacked together, in some cases combined with pooling and max pooling layers in between. Two alternatives are examined in this Thesis, a so-called small and a large CNN architecture. Details can be seen in the two Tables that follow.

Small CNN Architecture			
Layer Type	**# Units**	**# Trainable Parameters**	**Output Shape**
Conv1d	32	48032	(batch_size, 46, 32)
Conv1d	64	6208	(batch_size, 44, 64)
MaxPooling 1D	–	0	(batch_size, 22, 64)
Dropout	–	–	(batch_size, 22, 64)
Flattten	–	–	(batch_size, 1408)
Dense	–	8454	(batch_size, 6)
Sum of trainable params		62694	

Table 4.1: Small CNN architecture (Input size 48x500)

[1]The train set fed to each DNN model, is split into a train (90%) and a validation set (10%). The loss obtained based on the latter, is the validation loss.

Large CNN Architecture			
Layer Type	**# Units**	**# Trainable Parameters**	**Output Shape**
Conv1d	32	80032	(batch_size, 44, 32)
Conv1d	64	10304	(batch_size, 40, 64)
Conv1d	128	24704	(batch_size, 38, 128)
Conv1d	256	98560	(batch_size, 36, 256)
Flattten	–	–	(batch_size, 9216)
Dense	–	294944	(batch_size, 32)
Dropout	–	–	(batch_size, 32)
Dense	–	198	(batch_size, 6)
Sum of trainable params		508742	

Table 4.2: Large CNN architecture (Input size 48x500)

4.3.2 Mixed architectures (CNN+LSTM)

LSTM networks are significantly more computationally demanding than CNNs even with the same number of trainable parameters. This is due to the unfolding in time that is executed during training / testing of RNN based cells. However, since audio data are time sequences, RNN based architectures may be more capable of capturing the dependencies between different time intervals. In the tested architectures, LSTM layers follow different combinations of convolutional and pooling layers with the hope to capture such long-term time dependencies. The idea of using LSTM layers after CNNs is based on the ability of CNN layers to learn high level visual features from each input sample in the hidden layers, at the same time reducing the dimension of features passed to next layers. Thus, this combination of CNN and LSTM layers leads to models with less trainable parameters.

Following these concepts, two mixed architectures (Mixed 1 and 2) are proposed, detailed in the two Tables that follow.

Mixed 1 Architecture			
Layer Type	**# Units**	**# Trainable Parameters**	**Output Shape**
Conv1d	256	640256	(batch_size, 44, 256)
Batch Normalization	–	1024	(batch_size, 44, 256)
Max Pooling 1D	–	–	(batch_size, 22, 256)
Conv1d	256	131328	(batch_size, 21, 256)
LSTM	64	82176	(batch_size, 10, 64)
Dense	–	4160	(batch_size, 10, 64)
Dropout	–	–	(batch_size, 10, 64)
LSTM	32	12416	(batch_size, 32)
Dense	–	198	(batch_size, 6)
Sum of trainable params		871558	

Table 4.3: Mixed 1 architecture (Input size 48x500)

Mixed 2 Architecture (Similar to Large CNN + LSTM Layer)			
Layer Type	**# Units**	**# Trainable Parameters**	**Output Shape**
Conv1d	32	80032	(batch_size, 44, 32)
Conv1d	64	10304	(batch_size, 40, 64)
Conv1d	128	24704	(batch_size, 38, 128)
Conv1d	256	98560	(batch_size, 36, 256)
LSTM	64	82176	(batch_size, 64)
Dense	–	8320	(batch_size, 128)
Dropout	–	–	(batch_size, 128)
Dense	–	774	(batch_size, 6)
Sum of trainable params		304870	

Table 4.4: Mixed 2 architecture (Input size 48x500)

4.4 Number of classes

As it is obvious, the number of classes has an important role on the classification accuracy. In this work, experiments were performed with a varying number of classes and also with a different association between sound patterns to classes. In particular, the problems considered are presented in the following figure: (Fig. 4.2).

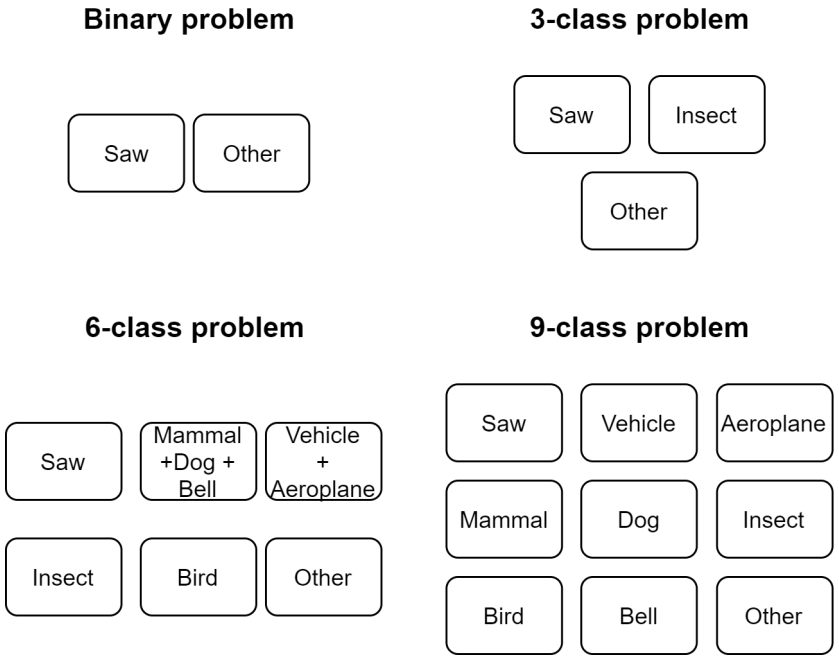

Figure 4.2: Class labels for the considered binary, 3-class, 6-class and 9-class problems.

Given a collection with multiple sound categories, it is obvious that these categories can be grouped in a different manner in order to construct a problem with different number of classes. In the attempt to "reduce" the number of classes the explicit way is to assign the same label to two or more different sound patterns and proceed by considering a number of classes equal to the number of labels. The implicit way, is to train the classifier on a large number of classes and during the implementation phase, to merge decisions for some of the classes together. One example is the 3-class problem shown above, which is considered with the intention of detecting chainsaw events. The distinction between "insect" and "other" is meaningless from a practical point of view, however, by considering a three class problem we hope that our classification model will better learn to discriminate insect from chainsaw sounds. These ideas are similar to the hierarchical classification problem referenced in [15, 6]. Our expectation is that training a model with that hierarchical scheme, will conclude to higher performance than a simple binary classification scheme.

4.5 Feature extraction for the baseline approaches

While the focus of this Thesis is mainly on deep learning, we cannot neglect some of the most successful traditional machine learning approaches, namely, Support Vector Machines (SVMs), Random Forests (RFs) and Gaussian Mixture Models (GMMs). In what follows, we describe the acoustic features that were used in combination with these classifiers.

For the baseline algorithms, some temporal and spectral features were extracted and combined so as to form a single feature vector per utterance. In order to use a 1D vector for each utterance, each clip is segmented into frames on which the spectral and temporal features are computed. Then the mean and the variance along all the frames of a clip are extracted to form two features of the 1D feature vector. The features are mentioned below:

- Zero Crossing Rate (ZCR): it is a measure of the number of times the signal value cross the zero axis. It is often an indication about the periodicity of the signal. Periodic signals tend to have a small ZCR value, while noisy sounds tend to have a high ZCR value.

- RMS: is the root mean square magnitude of the audio samples in each time frame.

- Mel frequency cepstral coefficients (MFCCs): They are derived from a type of cepstral representation of the audio clip using mel scale. They represent the shape of the spectrum with very few coefficients.

- Spectral centroid: it is the center of gravity of the linear spectrum

- Roll-off frequency (or spectral roll-off): is the frequency so that a specified percentage (85 % in current experiments) of the signal energy is contained below that frequency. It is correlated to the harmonic/noise cutting frequency.

- Spectral flatness (or Wiener entropy or tonality coefficient): is a measure of the noisiness of a spectrum. A high value of spectral flatness (close to 1) indicate flat spectrum which corresponds to a noise-like signal, similar to white noise, whereas low values of spectral flatness indicates sinusoidality and tonality of a spectrum.

- Spectral Contrast: it considers the spectral peak, the spectral valley, and their difference in each frequency subband.

- F_0: is the estimated fundamental frequency for each time frame sample obtained from the SRH method, by observing the frequency that exhibits the maximum value at each time frame.

- Voicing Strength (VS): it is the same metric used for the VAD (see Section 4.1).

All features mentioned above, except from F_0 and VS, are calculated using the librosa library [60]. By computing the mean and the variance along frames for all the aforementioned features, a 272×1 vector is obtained for each utterance. Such a high-dimensional feature vector could result in unstable training or not achieve convergence. Thus, apart from the original 272-feature vector, three additional reduced-size feature vectors were examined. A feature vector of size 111×1 was obtained by performing feature selection. More particularly, a recursive feature elimination approach was performed using cross validation with a random forest classifier. One of the additional feature vectors was of size 40×1, consisted of 13 MFC coefficients (from 128 extracted) and the rest temporal and spectral features. The smallest feature vector was of size 8×1, consisted only by the temporal and spectral features. Standarization of the features, before proceeding to the training, was also considered as an option, as the means to achieve a more stable performance. The results presented for each baseline case in what follows are the best ones achieved with respect to the aforementioned parameterization and feature selection.

4.6 Cross Validation and Evaluation metrics

4.6.1 Cross Validation

Performance metrics presented in a later section are based on an outer Cross Validation (CV) scheme that follows a "leave-one-sensor-out" approach. More particularly, all the utterances extracted from one ARU are put into test, while

the data extracted from the rest ARUs is considered to be available for training. As stated in [57], this validation approach will better reflect the system's ability to adapt to variations of background noise in time (e.g. dawn vs. dusk) and space (i.e., different sensor location), as well as to variations in the characteristics of different ARUs (e.g. frequency response). The test is repeated for 7 from the 14 ARUs (namely 'PR', 'RP3', 'RP6', 'RP10', 'RP14', 'RP15', 'SW2'), since it was decided that ARUs which did not contain any chainsaw events would be used only for training.

For training purposes, an additional inner CV loop is considered, in which case the training set is further split. For DNN training, the data used for training is split in two parts after being shuffled; a training set consisting of 90% of the data, and a validation set consisting of 10% of the data. This inner split is repeated 5 times for each outer validation loop, and the scores reported for the specific ARU are concatenated in a vector which is used to extract each evaluation metric. This was done in order to compensate for the variability that characterizes the produced DNN models due to the random selection of data samples in the training and in the testing set. For the baseline machine learning approaches which are characterized by some degree of randomness, such as Random Forests (selection of random sub set of features on each predictor) or GMMs (random initialization on Kmeans at the start of the training), we used a similar CV scheme. The data left for training fed to the model as training set where again, the algorithm is trained 5 times for each outer validation CV, obtaining a vector of all predicted scores for each of test set samples. The SVM model was fed the whole training set once, and prediction scores were extracted for each outer validation CV. Thus, by following this procedure we obtain an (1x7) vector for each model consisting of a metric for each ARU.

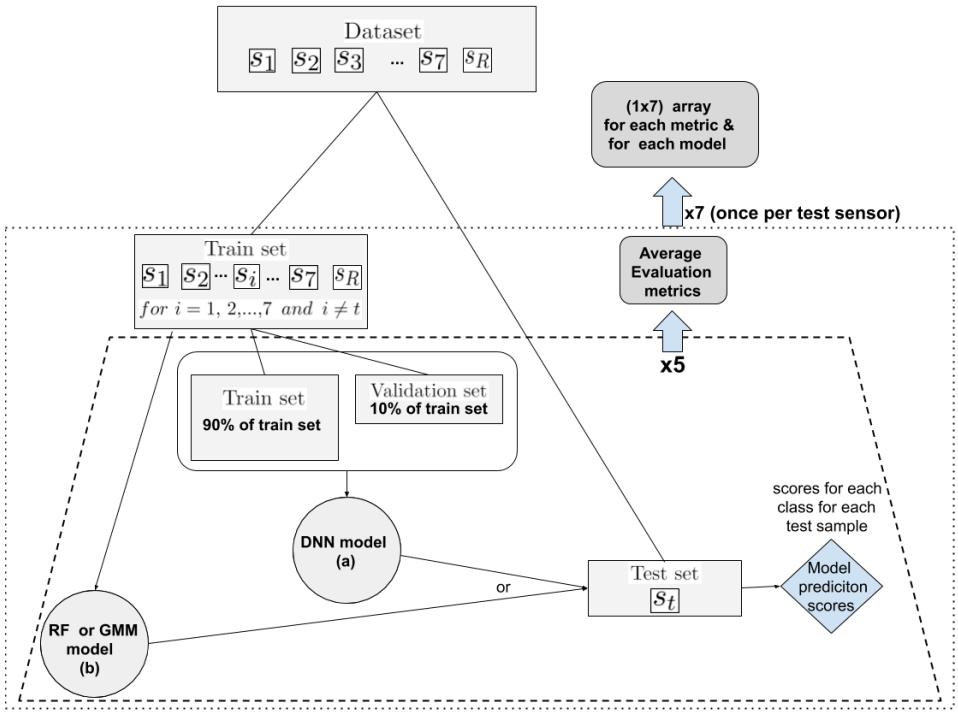

Figure 4.3: Illustrative graph of the Cross-Validation scheme, where s_i is one of the 7 sensors, s_t the sensor left out for testing and s_R denotes the remaining sensors that are used only for training.

It should be also clarified (i) that testing is performed only for the original audio samples extracted from the test-ARU and not on any audio samples that were produced from data augmentation and (ii) the augmented dataset produced from a specific ARU is not allowed to participate in the training set when that ARU is put into test.

The proposed cross-validation scheme is expected to lead to performance metrics that are realistic and representative for the classification performance when applied on data from new ARUs. Moreover, it is expected to reflect the impact on variations of the background noise due to geographical location and due to season.

4.6.2 Evaluation Metrics

In the evaluation section, an average class accuracy will be used as the most relative metric to accuracy because it incorporates the performance of each model for all the classes. Since not all test ARUs have examples from every training class and each test ARU has different number of examples, metrics such as balanced accuracy or simple accuracy would be biased.

A widely used metric for binary classifications is the Receiver operating characteristic (ROC) curve, which illustrates in a plot the classification performance

of a binary classifier. Typically, the ROC curve is created by plotting the True Positive Rate (TPR) against the false positive rate (FPR) using various decision threshold settings. The Area Under the ROC Curve (ROC AUC) is the second metric used in this Thesis. AUC reduces the information of ROC curve in a single number with possible values between [0,1], where 0 indicates worst performance and 1 indicates perfect performance. AUC scores will be used for evaluating in chainsaw class, which is the main target class on this Thesis.

Chapter 5

Experiments and Results

5.1 Evaluation of VAD-based segment selection

This subsection presents results for the VAD-based segment selection algorithm, with the intention to illustrate the algorithm's sensitivity to the presence of harmonic events. Two different types of evaluation were performed. The first focuses solely on chainsaw events, while the second showcases the algorithms sensitivity to harmonic sounds in general, as a function of some parameter values.

For this VAD implementation we employed the SRH method with a frame-size of 90 ms rather than 60 ms that was used for calculation of the SRH spectrogram. Moreover, we set $N_{ca} = 3$ and $N_d = 23$ while threshold T_{srh} was set to values of 0.08 in a first experiment and equal to 0.078 in a second experiment. Additional parameters that should be reported concern those used to invoke the "pitch_srh" function in COVAREP toolbox, that was necessary for calculation of VS. Note that the proposed parameter values were empirically tuned and differ from the default ones that are intended for speech detection; the frame-length and hop-size for calculation of the SRH was 180 and 60 ms respectively, which means that the length of an extracted utterance is ensured to be at least 1.5 s. The number of harmonics was set to 4 and the minimum and maximum $F0$ values were set to 40 and 760 Hz respectively.

In the first experiment that was used in order to quantify the algorithms sensitivity to chainsaw event, we used Praat in order to manually annotate audio segments with chainsaw events in randomly selected portions from all 7 ARUs that contained chainsaw events. Specifically, Praat was used in order to mark the beginning and end of several chainsaw events in each recording and the total duration of the marked regions in each recording. Each test recording was then presented as input to the system and the audio portion detected by the algorithm was compared against the ground truth (the manually annotated events). This way, we were able to compare the total duration of chainsaw events detected by the algorithm to those detected by the human listener. The results are shown in the second row of 5.1. It can be seen that for $T_{srh} = 0.08$ the algorithm detects 80.1% of the actual chainsaw events, while for $T_{srh} = 0.078$, the chainsaw detection rate increases to 86.6%. To our opinion this proves that VS, although initially designed for speech signals, is also appropriate for detecting the chainsaw pattern.

In a second experiment, we fed several randomly selected 12-hour recordings from eight different ARUs and we counted the duration of the audio segments extracted by the algorithm, for the aforementioned values of T_{srh}. The general detection rate is calculated by dividing the total duration of extracted segments to the duration of the input recordings (96 hours) and the results can be seen in the third row of Table 5.1. The results are indicative about the specificity of the algorithm. It can be seen that even for the lowest threshold value of $T_{srh} = 0.078$, VAD-based segment selection lets less than 9% of the original data duration to pass through. To our opinion this is a positive attribute for the algorithm, highlighting it's validity for the pre-processing of environmental sound recordings.

T_{srh}	0.08	0.078
Chainsaw detection rate	80.1%	86.6%
General detection rate	147.2 s/hour	313.9s/hour

Table 5.1: Chainsaw and general detection rate for the VAD-based selection algorithm.

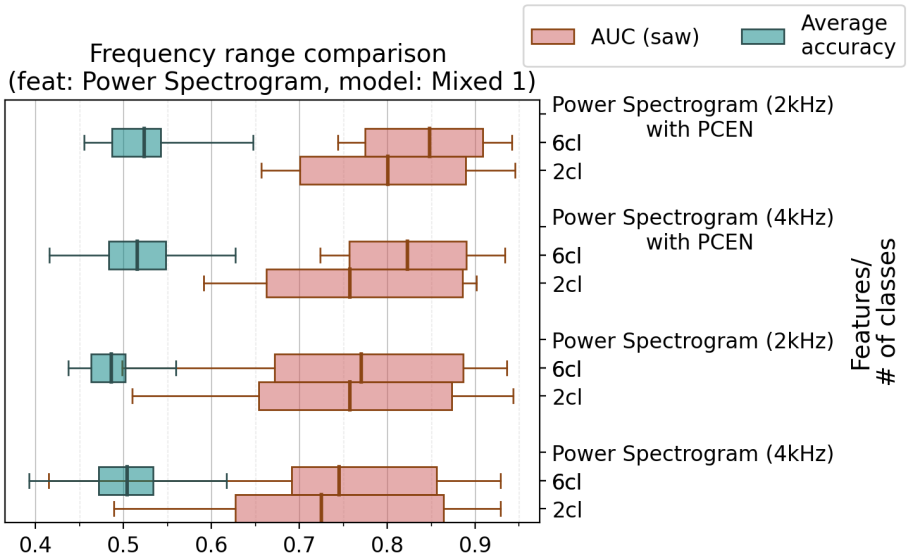

Figure 5.1: Classification performance for power spectrogram and PCEN spectrogram as a function of the frequency range.

5.2 Effect of Frequency Range

In Section 4.2.1 the discussion was whether the acoustic features should be designed so as to include information from the entire frequency range or whether an upper frequency limit should be applied. In Fig. 5.1 the classification performance is illustrated when using power spectrogram and PCEN spectrogram for full frequency range (80-4000Hz) and limited frequency range (150-2000Hz). The red box plots represent the average AUC scores, considering the binary problem were "chainsaw" is discriminated from the "non-chainsaw" class, while the green boxes represent the average accuracy results for the explicit 6-class problem (see Section 4.2). The box plots were chosen here as a more informative type of illustration, since they reveal at the same time an indication of the variance[1] (the red/green bar), the mean (red/green vertical line) and the extremes (minimum/maximum)

[1]Specifically, each colored box reveals the interquartile range, where the middle 50% of the data lies.

of the distribution (thin red/green horizontal lines). The initials 2cl represent the 2-class problem where training was performed with only two labels, "chainsaw" for the Positive class and "non-chainsaw" for the Negative class. On the other hand, the initials 6cl represent the performance for the case where the Negative class is comprised of 5 sub-classes.

In general, PCEN spectrogram achieves higher AUC scores compared to power spectrogram and the same holds for 6cl against 2cl, however these comparisons are investigated in more detail in the Sections that follow. For the moment, it is worth to observe that limiting the highest frequency at 2 kHz results to an increase in the performance for the binary problem, both for the case of power spectrogram and PCEN spectrogram. This is probably an indication that higher harmonics of the chainsaw sound are missing and therefore, higher frequencies do not add significant information. Concerning the effect of frequency range for the 6-class problem, it can be seen that this has a limited impact for the case of PCEN spectrogram, but wider frequency range seems to slightly improve performance for the case of power spectrogram. In what follows, results will be presented considering the limited frequency range for power and PCEN spectrogram.

5.3 Comparison of features and DNN architectures

The four proposed DNN architectures are tested in this section for power spectrogram in Fig. 5.2, PCEN spectrogram in Fig. 5.3 and SRH-spectrogram in Fig. 5.4. Results are presented for chainsaw as the target pattern for an explicit two-class problem (2cl) and an implicit two-class problem (6cl) after merging the decisions taken upon five "non-chainsaw" classes into a single class. The average accuracy for the 6-class problem is also plotted in corresponding plots.

Concerning the use of the different DNN models for the binary problem, it can be seen that the mixed DNNs provide a significant improvement compared to CNNs, with Mixed 1 slightly surpassing Mixed 2 in terms of average AUC. Especially for the Mixed 1 and Mixed 2 models it may also be observed that the 6cl approach improves significantly compared to the 2cl, providing an increment in average AUC up to 5% in some cases. Power spectrogram appears to be the weakest acoustic feature, providing the lowest average AUC and highest variance among ARUs compared to the other features. Inspection of Figs. 5.3 and 5.4 reveals that, in terms of average AUC, the best performance is achieved when combining SRH spectrogram with model Mixed 1. Specifically, SRH spectrogram achieves 0.859 and 0.878 average AUC for 2cl and 6cl respectively while for PCEN spectrogram the corresponding values are 0.801 and 0.848. Also, SRH not only achieves higher AUC scores than PCEN spectrogram (and of course than power spectrogram) but smaller variance across the different ARUs as well.

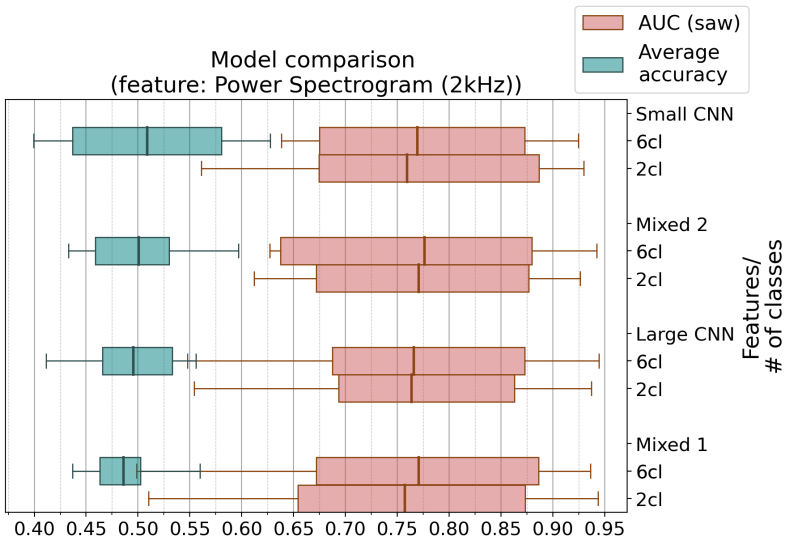

Figure 5.2: Classification performance of power spectrogram for different DNN models.

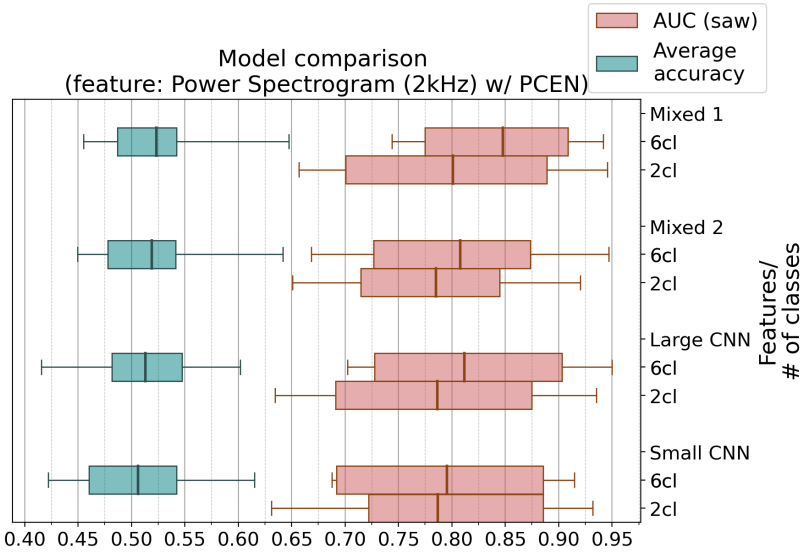

Figure 5.3: Classification performance of PCEN spectrogram for different DNN models.

In terms of the 6-class problem, the average accuracy varies much less as a function of the DNN model and acoustic feature and is concentrated in all cases around 50%. The best accuracy scores are observed for PCEN spectrogram and especially by combining it with model Mixed 1, in which case the average accuracy

is around 52.5%. In Fig. 5.5, the confusion matrix illustrates the classification performance for the best case, i.e., PCEN spectrogram combined with model Mixed 1. This matrix is created by concatenating the predictions made with all seven ARUs and all five repetitions of the training process. The matrix is illustrative of the most typical errors that the classifiers perform, however it should be kept in mind that there is a large variance in the number of samples in each class. One interesting observation is that "insects" represent the most common source for confusion with "chainsaw" and vice versa. The difference in the value of the average accuracy (68.95% compared to 52.5%) can be explained by the fact that when building the confusion matrix, ARUs with more utterances play a greater role in the classification score. On the other hand, average accuracy shown in the plots is averaged with respect to the ARUs, and therefore each ARU has an equal significance in the final score.

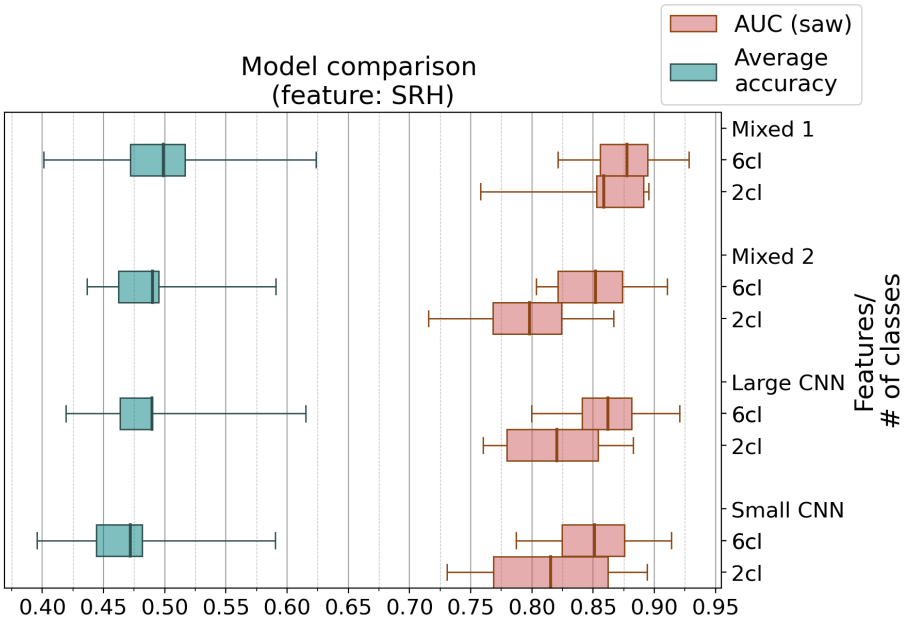

Figure 5.4: Classification performance of SRH spectrogram for different DNN models.

Figure 5.5: Confusion matrix for six class problem using PCEN spectrogram combined with model Mixed 1.

5.4 Comparison of DNNs and Baseline Algorithms

In Fig. 5.6 we illustrate the classification performance for the two best combinations of features and DNN models and for three different baseline approaches, SVM, RF and GMM. In terms of average AUC in the binary classification problem, it can be seen that the baseline approaches also achieve higher scores when following the 6cl approach compared to the 2cl one. However, none of the baseline approaches is able to surpass the two best DNN approaches consisting of model Mixed 1 combined with PCEN spectrogram and SRH spectrogram. The best baseline classifier appears to be the SVM, achieving an average AUC approximately equal to 0.8 for the 6cl binary problem. However, in the case of the 6-class problem, RF achieves better average accuracy compared to SVM and it appears able even to compete PCEN spectrogram and SRH spectrogram.

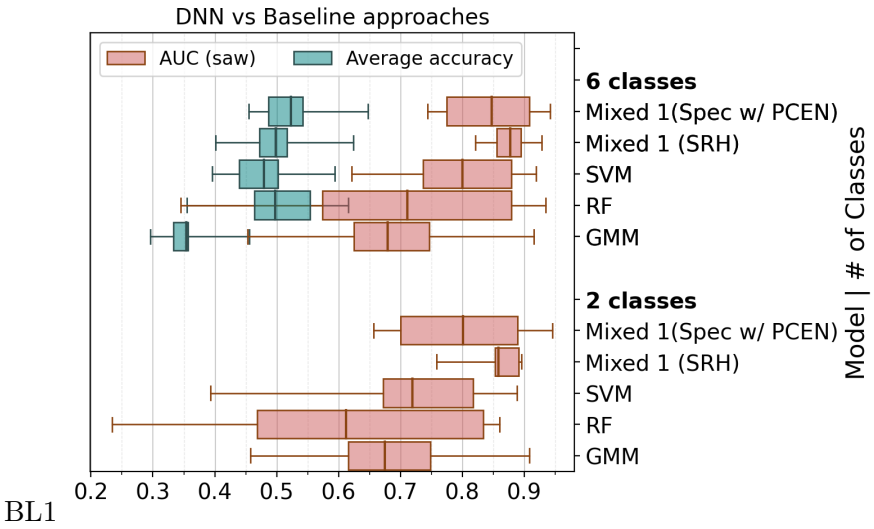

Figure 5.6: Classification performance for selected DNNs and baseline approaches.

5.5 Number of Classes

While the previous results have already demonstrated the advantage gained by following a hierarchical classification scheme (2cl versus 6cl), in this Section we perform a more detailed comparison by considering a 3-class and a 9-class in addition to the 6-class approach. The 3-class approach represents a binary problem with "chainsaw" as the Positive class and "insect" as an additional representative of the Negative ("non-chainsaw") class. On the other hand, the 9-class problem includes eight different subclasses in the Negative class, as they were defined a previous Section in Fig. 4.2. Results are shown for the DNN classifiers and particularly, for power spectrogram in Fig. 5.7, for PCEN spectrogram in Fig. 5.8 and for SRH spectrogram in Fig. 5.9. Apart from average AUC scores, the average accuracy is also plotted for the explicit 3-class, 6-class and 9-class problems.

Considering the effect of the number of subclasses for power spectrogram in Fig. 5.7, it can be observed that they have a small effect in performance, with the 3-class providing a slight advantage compared to the other cases. For the case of PCEN spectrogram in Fig. 5.8, the 6-class formulation provides the best performance for model Mixed 1, however the 3-class approach is the optimal one for the case of the small CNN. Finally, for SRH spectrogram in Fig. 5.9, the 3-classes performs slightly better than the 2-classes for the case of model Mixed 1 while 6-class and 9-class approaches compete with very slight difference for the first place when the small CNN is used. Overall, the results suggest than accounting for more than one subclasses in the Negative class is beneficial for chainsaw detection, with 2 and 5 sub-classes (for a 3-class and 6-class problem respectively) representing the best choice, depending on the DNN architecture.

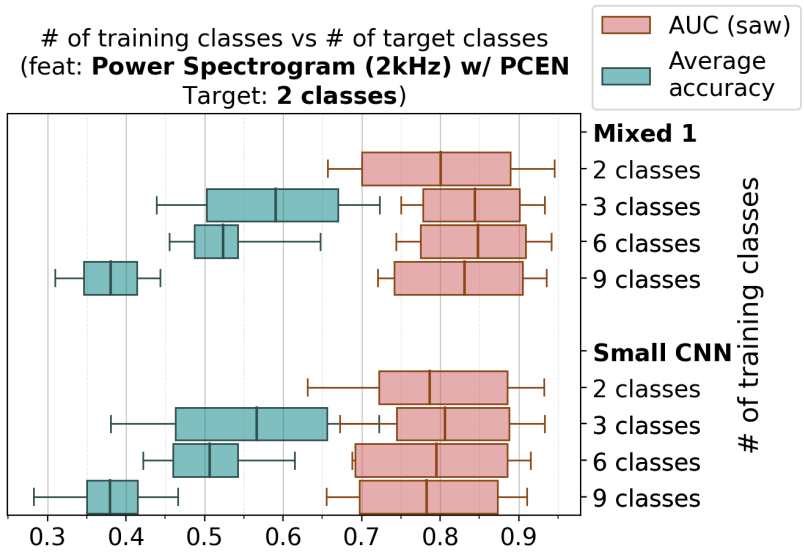

Figure 5.8: Effect of the number of training classes for PCEN spectrogram.

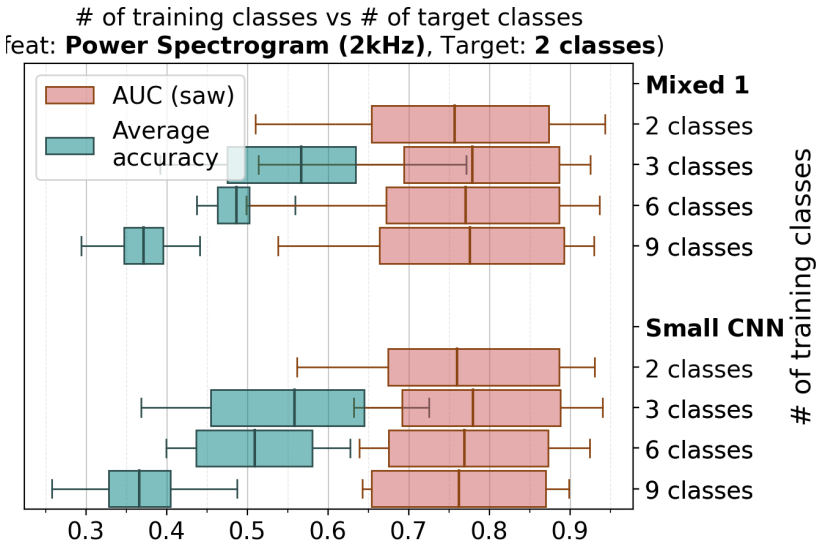

Figure 5.7: Effect of the number of training classes for power spectrogram.

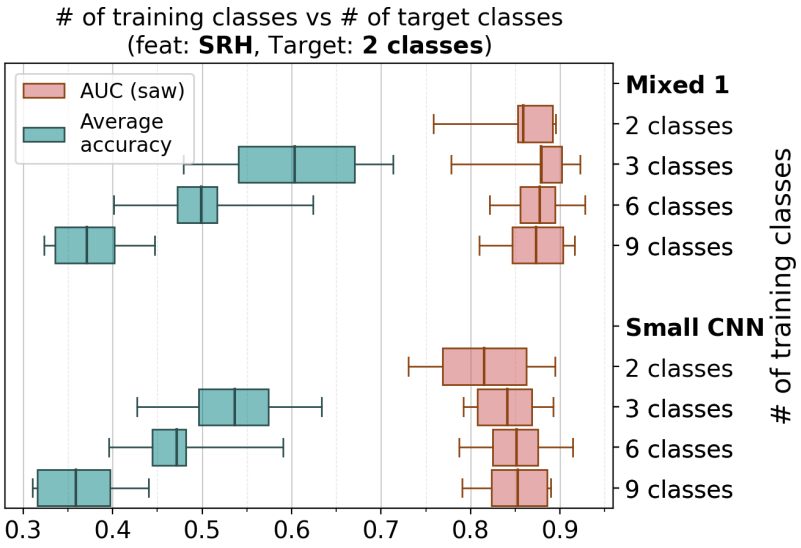

Figure 5.9: Effect of the number of training classes for SRH spectrogram.

5.6 Data Augmentation

To illustrate the effect of data augmentation, we plot in Fig. 5.10 the classification performance with and without data augmentation for PCEN and SRH spectrogram in combination with model Mixed 1. Results for mel spectrogram are also shown, considering a filterbank with 128 frequency centers spanning the entire available frequency range (up to 4 kHz). Concerning the binary problem, data augmentation appears to have a positive impact in performance. One exception is the case of PCEN spectrogram with 6 classes, where the model trained with the augmented data performs slightly worse in terms of average AUC. In all cases, the improvement with the augmented dataset is more evident in the 2cl than in the 6cl approach. Moreover, for 2cl and SRH spectrogram, data augmentation provides an average AUC equal to 0.8835, which is the best result in terms of average AUC in this Thesis. The improvement of data augmentation is quite significant for mel spectrogram in the case of binary problem. However, mel spectrogram has much inferior performance compared to PCEN and SRH spectrogram, which is one of the reasons why it has been excluded from the previous analysis.

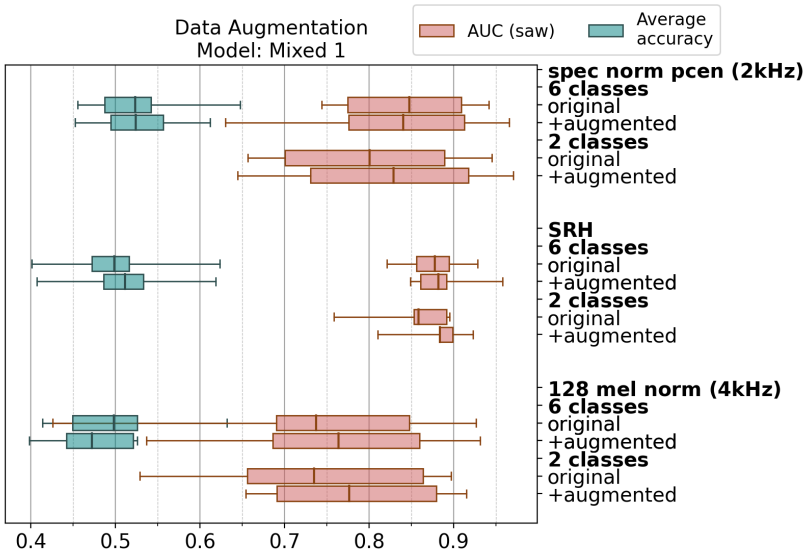

Figure 5.10: Effect of data augmentation in classification performance

Chapter 6

Conclusions & Future Work

6.1 Conclusions

Acoustic sensors in protected environments may offer a significant assistance in monitoring the health of the ecosystem but also as the means to detect and prevent human intervention that is harmful to the environment. In this Thesis, machine learning techniques have been studied as the means to automatically detect sound events that are indicative of illegal human activity with special focus on the chainsaw sound which cam be related to illegal logging.

Preparation of the data required for training was one of the main difficulties that have to be handled. On one hand, DNNs are particularly demanding in terms of training data and on the other hand, when dealing with environmental recordings, the patterns of interest are sparsely distributed in the audio recording. This means that one should spend enormous amounts of time listening to audio recordings in order to prepare the training data. To assist in this task, we proposed the use of a VAD-based segmentation process that automatically detects audio portions with harmonic content within a long audio recording. Using this approach to automatically segment a long audio recording, acoustic events that have a high relevance to illegal human activity can be detected (e.g. chainsaw, vehicles, bells, grazing animals) as well as acoustic events that relate to the natural inhabitants of the environment. VAD-based segment selection provided a significant assistance in collecting and annotating data from hundreds of hours of real-field recordings in Greece, making the research presented in this Thesis possible.

Using the extracted data, this research examined the formulation of a single-label classification problem with main focus on detecting sounds relating to illegal human activity and with special focus on the chainsaw sound. In the Thesis, several different DNN classifiers, as well as conventional classifiers were put into test, alongside with different types of acoustic features. A novel acoustic feature named SRH-spectrogram was also introduced, emerging as an intermediate product of the SRH method that was used for constructing the VAD-based segment selection process.

Concerning the detection of chainsaw sound, in terms of a binary classification problem, PCEN spectrogram and SRH spectrogram were shown to provide the best classification performance, surpassing classical power spectrogram and mel-spectrogram. Among different DNN architectures, combinations of the best two features with the mixed DNN architectures, consisting of both convolutional and LSTM layers, were shown to provide an advantage compared to the two CNN architectures. An additional tactic that provided a benefit in terms of classification performance was the formulation of an implicit 2-class problem (as opposed to an explicit 2-class problem) that is based on training the Negative class on multiple sub-classes and then merging the decisions of the Negative class together.

Results were also presented considering an explicit 6-class problem, however the performance was relatively poor, nearly exceeding 50% in terms of average accuracy in the best case. Probably this is an indication that additional data is required in order to capture the variability within the additional classes. One way for achieving this is with the annotation of additional data, especially if this can be applied on recordings from new ARUs that have not participated in the dataset yet, but also by directly inserting annotated data that can be found from other open-access datasets. Moreover, additional data argumentation techniques can be examined as the means for enriching the training set.

6.2 Future Work

The research presented in this Thesis can be extended along two main axes, one concerning the improvement of the performance in the same problem as the one examined, and one along the discovery of new applications and problems.

One major bottleneck in the research associated to this Thesis is the tuning of parameter values associated to the different types of features and the different types of architectures. For obvious reasons it is not possible to perform an exhaustive research on all parameters combinations and one has to rely on trial and error on a limited number of examples, possibly with the help of some personal intuition. One approach to reduce the effort required for parameter tuning would be to integrate PCEN and possibly SRH as a trainable frontend in the architecture of the neural network, in order to eliminate heuristic parameters, and make as many parameters as possible trainable. Furthermore, the research was limited to only one type of data augmentation (resampling of the data) and it would be worth to investigate additional types of data augmentation so as to better capture variability of not only the underlying patterns of interest but also of the background noise which has a dominant presence in the observed audio signals. Concerning the latter, audio signal enhancement techniques might be also worth to investigating as the means to derive "cleaner" version of the audio signal before feeding it to the DNN.

The research presented in this Thesis focused on the chainsaw pattern and most decisions taken were driven by the performance on that pattern and moreover, on short duration audio clips extracted after the use of the VAD-based segmentation

process. In the future, a more complete investigation should be implemented for the performance of an end-to-end system that includes a VAD-based segmentation process at the front-end and the DNN classifier(s) at the back end. The work could then of course extend to additional sound patterns relating to illegal human activity, as for example illegal grazing, taking into account classes such as bells and mammals. Finally, it would be worth to investigate the transition from single-class to a multi-label problem, in which case the classifiers would output decisions concerning multiple classes that are active at the same time instant.

Bibliography

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[2] Sheikh Fahad Ahmad and Deepak Singh. Automatic detection of tree cutting in forests using acoustic properties. *Journal of King Saud University - Computer and Information Sciences*, 02 2019.

[3] T. Ahmed, M. Uppal, and A. Muhammad. Improving efficiency and reliability of gunshot detection systems. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 513–517, 2013.

[4] M. A. Aizerman, E. A. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. In *Automation and Remote Control,*, number 25 in Automation and Remote Control,, pages 821–837, 1964.

[5] V. Andrei, H. Cucu, and L. Petrică. Considerations on developing a chainsaw intrusion detection and localization system for preventing unauthorized logging. *Journal of Electrical and Electronic Engineering*, 3(6):202–207, 2015.

[6] P. K. Atrey, N. C. Maddage, and M. S. Kankanhalli. Audio based event detection for multimedia surveillance. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 5, pages V–V, 2006.

[7] Y. Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, 5:157–66, 02 1994.

[8] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[9] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual*

Workshop on Computational Learning Theory, COLT '92, page 144–152, New York, NY, USA, 1992. Association for Computing Machinery.

[10] E. Cakir, T. Heittola, H. Huttunen, and T. Virtanen. Multi-label vs. combined single-label sound event detection with deep neural networks. In *2015 23rd European Signal Processing Conference (EUSIPCO)*, pages 2551–2555, 2015.

[11] E. Cakir, E. Ozan, and T. Virtanen. Filterbank learning for deep neural network based polyphonic sound event detection. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 3399–3406. IEEE, 2016.

[12] Sharan Chetlur, Cliff Woolley, Philippe Vandermersch, Jonathan Cohen, John Tran, Bryan Catanzaro, and Evan Shelhamer. cudnn: Efficient primitives for deep learning, 2014.

[13] Francois Chollet et al. Keras, 2015.

[14] S. Chu, S. Narayanan, and C. . J. Kuo. Environmental sound recognition with time–frequency audio features. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(6):1142–1158, 2009.

[15] Chloé Clavel, T. Ehrette, and Gaël Richard. Events detection for an audio-based surveillance system. *2012 IEEE International Conference on Multimedia and Expo*, 0:1306–1309, 01 2005.

[16] J. G. Colonna, B. Gatto, E. Miranda Dos Santos, and E. F. Nakamura. A framework for chainsaw detection using one-class kernel and wireless acoustic sensor networks into the amazon rainforest. In *2016 17th IEEE International Conference on Mobile Data Management (MDM)*, volume 2, pages 34–36, 2016.

[17] Rainforest Connection. https://www.rfcx.org/. (accessed Sept. 12, 2020).

[18] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, September 1995.

[19] László Czúni and Péter Varga. Lightweight acoustic detection of logging in wireless sensor networks. 2014.

[20] L. Czúni and P. Z. Varga. Time domain audio features for chainsaw noise detection using wsns. *IEEE Sensors Journal*, 17(9):2917–2924, 2017.

[21] G. Degottex, J. Kane, T. Drugman, T. Raitio, and S. Scherer. COVAREP—A collaborative voice analysis repository for speech technologies. In *International conference on acoustics, speech and signal processing (ICASSP)*, pages 960–964. IEEE, 2014.

[22] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.

[23] J. Dennis, H. D. Tran, and E. S. Chng. Image feature representation of the subband power distribution for robust sound event classification. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(2):367–377, 2013.

[24] J. Dennis, H. D. Tran, and H. Li. Spectrogram image feature for sound event classification in mismatched conditions. *IEEE Signal Processing Letters*, 18(2):130–133, 2011.

[25] J. Dennis, H.D. Tran, and E.S. Chng. Overlapping sound event recognition using local spectrogram features and the generalised hough transform. *Pattern Recognition Letters*, 34(9):1085 – 1093, 2013.

[26] J. W. Dennis. Sound event recognition in unstructured environments using spectrogram image processing, 2014.

[27] T. Drugman and A. Alwan. Joint robust voicing detection and pitch estimation based on residual harmonics. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[28] Thomas Drugman and Abeer Alwan. Joint robust voicing detection and pitch estimation based on residual harmonics. pages 1973–1976, 01 2011.

[29] M. Espi, M. Fujimoto, K. Kinoshita, and T. Nakatani. Exploiting spectro-temporal locality in deep learning based acoustic event detection. *EURASIP Journal on Audio, Speech, and Music Processing*, 2015:1–12, 2015.

[30] Weijiang Feng, Naiyang Guan, Yuan Li, Xiang Zhang, and Zhigang Luo. Audio visual speech recognition with multimodal recurrent neural networks. pages 681–688, 05 2017.

[31] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, Apr 1980.

[32] Amir Gandomi and Murtaza Haider. Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 35(2):137 – 144, 2015.

[33] J. T. Geiger and K. Helwani. Improving event detection for audio surveillance using gabor filterbank features. In *2015 23rd European Signal Processing Conference (EUSIPCO)*, pages 714–718, 2015.

[34] O. Gencoglu, T. Virtanen, and H. Huttunen. Recognition of acoustic events using deep neural networks. In *2014 22nd European Signal Processing Conference (EUSIPCO)*, pages 506–510, 2014.

[35] Felix Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. *Neural computation*, 12:2451–71, 10 2000.

[36] M. V. Ghiurcau and C. Rusu. Vehicle sound classification. application and low pass filtering influence. In *2009 International Symposium on Signals, Circuits and Systems*, pages 1–4, 2009.

[37] Marius Ghiurcau and Corneliu Rusu. About classifying sounds in protected environments. pages 84 – 87, 10 2010.

[38] Marius Ghiurcau, Corneliu Rusu, Radu Bilcu, and J. Astola. Audio based solutions for detecting intruders in wild areas. *Signal Processing*, 92:829–840, 03 2012.

[39] N'tcho GNAMÉLÉ, Yelakan OUATTARA, Tokpa KOBEA, Geneviève Baudoin, and J.-M Laheurte. Knn and svm classification for chainsaw identification in the forest areas. *International Journal of Advanced Computer Science and Applications*, 10, 01 2019.

[40] L. Grama, E. Buhuş, and C. Rusu. Acoustic classification using linear predictive coding for wildlife detection systems. In *2017 International Symposium on Signals, Circuits and Systems (ISSCS)*, pages 1–4. IEEE, 2017.

[41] Lacrimioara Grama, Elena Buhus, and Corneliu Rusu. Acoustic classification using linear predictive coding for wildlife detection systems. pages 1–4, 07 2017.

[42] C. Harris. Absorption of sound in air versus humidity and temperature. *The Journal of the Acoustical Society of America*, 40(1):148–159, 1966.

[43] Tomoki Hayashi, Shinji Watanabe, Tomoki Toda, Takaaki Hori, Jonathan Le Roux, and Kazuya Takeda. Duration-controlled lstm for polyphonic sound event detection. *IEEE/ACM Transactions on Audio Speech and Language Processing*, 25, 08 2017.

[44] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[45] Danilo Hollosi, J. Schröder, Stefan Goetze, and Jens Appell. Voice activity detection driven acoustic event classification for monitoring in smart homes. pages 1 – 5, 12 2010.

[46] D. H. Hubel and T. N. Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, 195(1):215–243, 1968.

[47] R. King and T. C. Phipps. Shannon, tespar and approximation strategies. *Comput. Secur.*, 18:445–453, 1999.

[48] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[49] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017.

[50] Thi-Thu-Huong Le, Jihyun Kim, and Howon Kim. Analyzing effective of activation functions on recurrent neural networks for intrusion detection. *Journal of Multimedia Information System*, 3(3):91–96, 2016.

[51] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, Contour and Grouping in Computer Vision*, pages 319–, London, UK, UK, 1999. Springer-Verlag.

[52] J. Licklider and I. Pollack. Effects of differentiation, integration, and infinite peak clipping upon the intelligibility of speech. 1948.

[53] Hyungui Lim, Jeongsoo Park, Kyogu Lee, and Yoonchang Han. Rare sound event detection using 1d convolutional recurrent neural networks. 05 2018.

[54] Zachary Lipton. A critical review of recurrent neural networks for sequence learning. 05 2015.

[55] K. Lopatka, J. Kotus, and A. Czyzewski. Detection, classification and localization of acoustic events in the presence of background noise for acoustic surveillance of hazardous situations. *Multimedia Tools Appl.*, 75(17):10407–10439, September 2016.

[56] V. Lostanlen, J. Salamon, M. Cartwright, B. McFee, A. Farnsworth, S. Kelling, and J. P. Bello. Per-channel energy normalization: Why and how. *IEEE Signal Processing Letters*, 26(1):39–43, 2019.

[57] Vincent Lostanlen, Justin Salamon, Andrew Farnsworth, Steve Kelling, and Juan Bello. Robust sound event detection in bioacoustic sensor networks. *PLOS ONE*, 14:e0214168, 10 2019.

[58] Rui Lu, Zhiyao Duan, and Changshui Zhang. Multi-scale recurrent neural network for sound event detection. pages 131–135, 04 2018.

[59] Warren Mcculloch and Walter Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:127–147, 1943.

[60] Brian McFee, Vincent Lostanlen, Alexandros Metsai, Matt McVicar, Stefan Balke, Carl Thomé, Colin Raffel, Frank Zalkow, Ayoub Malek, Dana, Kyungyun Lee, Oriol Nieto, Jack Mason, Dan Ellis, Eric Battenberg, Scott Seyfarth, Ryuichi Yamamoto, Keunwoo Choi, viktorandreevichmorozov, Josh Moore, Rachel Bittner, Shunsuke Hidaka, Ziyao Wei, nullmightybofo, Darío

Hereñú, Fabian-Robert Stöter, Pius Friesch, Adam Weiss, Matt Vollrath, and Taewoon Kim. librosa/librosa: 0.8.0, July 2020.

[61] I. McLoughlin, H. Zhang, Z. Xie, Y. Song, and W. Xiao. Robust sound event classification using deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):540–552, 2015.

[62] P. Mermelstein. Distance measures for speech recognition, psychological and instrumental. *Pattern Recognition and Artificial Intelligence*, pages 374–388, 1976.

[63] S. Ntalampiras and I. Potamitis. Detection of human activities in natural environments based on their acoustic emissions. In *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, pages 1469–1473. IEEE, 2012.

[64] S. Ntalampiras, I. Potamitis, and N. Fakotakis. Probabilistic novelty detection for acoustic surveillance under real-world conditions. *IEEE Transactions on Multimedia*, 13(4):713–719, 2011.

[65] Stavros Ntalampiras, Ilyas Potamitis, and Nikos Fakotakis. Automatic recognition of urban environmental sounds events. 06 2010.

[66] Jozef Papan, M. Jurecka, and J. Puchyova. Wsn for forest monitoring to prevent illegal logging. pages 809–812, 01 2012.

[67] Jeong-Sik Park and Seok-Hoon Kim. Sound learning–based event detection for acoustic surveillance sensors. *Multimedia Tools and Applications*, pages 1–13, 2019.

[68] Huy Phan, Lars Hertel, Marco Maaß, and Alfred Mertins. Robust audio event recognition with 1-max pooling convolutional neural networks. 09 2016.

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

[70] Ilyas Potamitis, Todor Ganchev, and Nikos Fakotakis. Automatic acoustic identification of insects inspired by the speaker recognition paradigm. 01 2006.

[71] Ilyas Potamitis, Stavros Ntalampiras, Olaf Jahn, and Klaus Riede. Automatic bird sound detection in long real-field recordings: Applications and tools. *Applied Acoustics*, 80:1–9, 06 2014.

[72] D. C. Prasetyo, G. A. Mutiara, and R. Handayani. Chainsaw sound and vibration detector system for illegal logging. In *2018 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*, pages 93–98, 2018.

[73] Shini Renjith. Detection of fraudulent sellers in online marketplaces using support vector machine approach. *International Journal of Engineering Trends and Technology*, 57:48–53, 03 2018.

[74] Douglas Reynolds. *Gaussian Mixture Models*, pages 659–663. Springer US, Boston, MA, 2009.

[75] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958.

[76] C. Rusu and L. Grama. Spectrograms, sparsograms and spectral signatures for wildlife intruder detection. In *2015 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, pages 1–4, 2015.

[77] J. Salamon and J. P. Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24(3):279–283, 2017.

[78] J. Salamon, J. P. Bello, A. Farnsworth, and S. Kelling. Fusing shallow and deep learning for bioacoustic bird species classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 141–145, 2017.

[79] Emilie Shireman, Douglas Steinley, and Michael J. Brusco. Examining the effect of initialization strategies on the performance of gaussian mixture modeling. *Behavior Research Methods*, 49(1):282–293, Feb 2017.

[80] S. Sigtia, A. M. Stark, S. Krstulović, and M. D. Plumbley. Automatic environmental sound recognition: Performance versus computational cost. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(11):2096–2107, 2016.

[81] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pages 278–282 vol.1, 1995.

[82] M. Valenti, D. Tonelli, F. Vesperini, E. Principi, and S. Squartini. A neural network approach for sound event detection in real life audio. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 2754–2758, 2017.

[83] Various. Activation function, wikipedia the free encyclopedia. (accessed Sept. 21, 2020).

[84] Various. Long short-term memory, wikipedia the free encyclopedia. (accessed Sept. 21, 2020).

[85] Various. Softmax function, wikipedia the free encyclopedia, 2019.

[86] M. Vasile Ghiurcau, C. Rusu, and R. Ciprian Bilcu. Wildlife intruder detection using sounds captured by acoustic sensors. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 297–300, 2010.

[87] Kaitlin Palmer Vincent Lostanlen. Long-distance detection of bioacoustic events with per-channel energy normalization. *Detection and Classification of Acoustic Scenes and Events (DCASE) workshop*, Oct 2019.

[88] Y. Wang, P. Getreuer, T. Hughes, R. F. Lyon, and R. A. Saurous. Trainable frontend for robust and far-field keyword spotting. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5670–5674, 2017.

[89] Y. Wang, L. Neves, and F. Metze. 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, 2016.

[90] P. J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[91] Haomin Zhang, Ian McLoughlin, and Yan Song. Robust sound event recognition using convolutional neural networks. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 559–563. IEEE, 2015.

[92] Yan Zhang and Dan Lv. Selected features for classifying environmental audio data with random forest. *The Open Automation and Control Systems Journal*, 7:135–142, 03 2015.

[93] Emre Çakır, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen. Polyphonic sound event detection using multi label deep neural networks. pages 1–7, 07 2015.