Audio Recognition in Incremental Open-set Environments

Hitham Jleed

A thesis submitted to the University of Ottawa
in Partial Fulfillment of the requirements for the
Doctor of Philosophy

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

Machine learning algorithms have shown their abilities to tackle difficult recognition problems, but they are still rife with challenges. Among these challenges is how to deal with problems where new categories constantly occur, and the datasets can dynamically grow. Most contemporary learning algorithms developed to this point are governed by the assumptions that all testing data classes must be the same as training data classes, often with equal distribution. Under these assumptions, machine-learning algorithms can perform very well, using their ability to handle large feature spaces and classify outliers. The systems under these assumptions are called Closed Set Recognition systems (CSR). However, these assumptions cannot reflect practical applications in which out-of-set data may be encountered. This adversely affects the recognition prediction performances. When samples from a new class occur, they will be classified as one of the known classes. Even if this sample is far from any of the training samples, the algorithm may classify it with a high probability, that is, the algorithm will not only be wrong, but it may also be very confident in its results. A more practical problem is Open Set Recognition (OSR), where samples of classes not seen during training may show up at testing time. Inherently, there is a problem how the system can identify the novel sound classes and how the system can update its models with new classes. This thesis highlights the problems of multi-class recognition for OSR of sounds as well as incremental model adaptation and proposes solutions towards addressing these problems. The proposed solutions are validated through extensive experiments and are shown to provide improved performance over a wide range of openness values for sound classification scenarios.

Dedication

TO

*The souls of **MY MOTHER**. You have supported me near the end of this work. You were eagerly awaiting this day, but you could not witness it. May God bless you!*

Acknowledgments

From the bottom of my heart, I would like to express my gratitude in a myriad of ways to following people, without whom I would not have made it through my doctoral degree.

I would like to express my gratitude to my supervisor Professor Martin Bouchard for his guidance on this journey and his patience and care. I am very grateful to the advisory committee members: Professor Jiying Zhao, Professor Ramy Gohary, for their time, constructive feedback, and encouragement during the thesis proposal. My thanks also go to anonymous reviewers of different papers that have arisen from this work, whose feedback played a significant role in shaping its current state.

To my family, I am eternally indebted to my father and my siblings' unwavering support, whom I look up to. It is from them that I gather my passion for higher education and the pursuit of knowledge. I am forever grateful for my wife, without whom I would not have crossed the finish line. Your love and encouragement are ever-present throughout the good times and the most stressful of times. To my children. You made me stronger, better, and fuller than I'd ever dreamed. I beg their pardon for missing out on some moments of their childhood while working towards my Ph.D. In remembrance of my mother, Mom, your wish came true. I made it.

There are always people who make success possible and rewarding. My friends who stayed with me on happy and sad days during this journey, the ones who helped me when I needed it the most. I would also like to express my special thanks to my colleagues and the faculty members at the University of Ottawa. Last but not least, this work would not have been possible without financial support and funding from the Libyan North America Scholarship program.

Hitham Jleed

Table of Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations/Acronyms

AEER	Acoustic event error rate
CAP	Compact abating probability
CNN	Convolutional neural networks
CSR	Closed set recognition
DCT	Discrete cosine transform
Del	Deletion portion
dB	Decibel
dBFS	Decibels-qualified-to-full scale
DHT	Discrete Hartley transform
Dist	Distance function
DNN	Deep neural network
EVT	Extreme value theorem
EVM	Extreme value machine
FFT	Fast Fourier transform
FN	False negative sample
FP	False positive sample
Hz	Hertz
Ins	Insertion portion
KCs	Known classes
KKT	Karush-KuhnTucker conditions
KLOS	Known labeled open space
k-NN	k-Nearest Neighbors
KUC	Known unknown classes
MAP	Maximum a posteriori
MFCC	Mel-frequency cepstral coefficients
ML	Maximum likelihood
MLE	Maximum likelihood estimation
MP3	MPEG audio layer-3

ODN	Open deep network
OSH	Optimal separating hyperplane
OSR	Open set recognition
OVA	One-versus-all
PLOS	Positively labeled open space
Pr	Precision
Re	Recall
PSR	Peak side ratio
PSR-SVM	Peak-side ratio support vector machine
RBF	Radial basis function
STFT	Short-time Fourier transform
SNR	Signal to noise ratio
SSVM	Specialized support vector machine
SVDD	Support vector data description
SVM	Support vector machine
Sub	Substitutions portion
TN	True negative sample
TP	True positive sample
VAD	Voice activity detection
W-SVM	Weibull-calibrated SVM
UUCs	Unknown unknown classes

List of Symbols

D	dataset
x_0	instant sample
δ	threshold
ES_{pq}	actual total spectrum energy
En_{pq}	estimated background spectrum energy
η	experiential threshold for VAD
z	empirical constant for VAD
Nk	number of classifiers
Y	class labels
t	target class
k	known class
u	unknown class
Y_t	label of target data
Y_k	label of known data
Y_u	label of unknown data
α	pre-emphasis parameter
N	segment duration (number of samples)
P	number of segments
A	accuracy

F	number of filter banks
F_s	sampling frequency
$H(m,k)$	Mel filter bank element
f_{HZ}	center frequency (Hz)
f_{Mel}	Mel frequency
Δf_{Mel}	Mel scale frequency resolution
$cep_i(p)$	i^{th} MFCC of p^{th} frame
d_p	delta MFCC coefficient
$\mathbb{R}^d$	real-valued space of d dimensions
L	cardinality
c	regularization parameter
γ	kernel parameter
b	SVM offset parameter
ψ	kernel function
H_{feat}	hyperplane
T_{det}	total number of detected events
$CM(i,j)$	confusion matrix
Φ	probability of inclusion
τ	tail size of EVM
ς	EVM coverage threshold
Nk	number of classes in training set

Chapter 1: Introduction

1.1. Background

Audio signals carry vast information about individual physical events and their surrounding environment. Researchers in both academic and industrial fields have proposed several machine-learning methods to recognize this information. However, traditional audio recognition methods usually have a limited scope. They only include a database of a few sound classes. Intuitively, they classify sounds among a set of known classes, where both training and testing sets include the same categories from the database. These traditional methods are called Closed Set Recognition (CSR), where they assign the input data in features domain into one or more predefined classes.

In the case of real-world recognition problems, a prospective algorithm can be trained with a few classes in a much larger space that has numerous classes. The algorithm must address unseen data and continuously update with additional categories. We emphasize below some challenges of real-world recognition that are considered the most serious obstacles facing the researchers. Some of them are not yet solved, and others may remain a problematic concern to be considered separately in each application.

- Representation and numerical description of the audio events. Even though several researchers have investigated the audio features representation, this problem is still application-dependent. In this thesis, we investigated most of the features commonly used in audio recognition. We selected the features that are more effective and can somehow separate database categories.
- Open set. The Open Set Recognition (OSR) problem assumes that not every class in testing time occurs in training time. The definition of the Open set concept has been introduced by [1]. Using the same definition, we modified a Support Vector Machine

(SVM) classifier to solve open-set audio domain problems by applying the peak-side-ratio (PSR) distribution on the posterior probabilities of the SVM, as described in Chapter 5. We expect that our solution for OSR can improve the event detection employed in surveillance or smart home applications.

- Incremental Learning. In this type of learning, a recognition model is continuously updated when new types of data arrive. Some researchers have used unsupervised learning [2], where data is delivered without the availability of class labels for training. Others have used semi-supervised learning [3]. The third group of researchers studied supervised learning [4] [5] [6] when data arrived with class labels for training. In this work, we have chosen supervised learning for our approach.
- Active learning. This process deals with the labeling of detected unknown samples. Researchers in the literature have tried to investigate how to label detected unknown samples, e.g. [7] used privileged information to re-identify images. This problem is not within the scope of this thesis. We assume that detected unknown samples are labeled by a user, human annotator, or "Oracle".
- Evaluation. This is how to measure the performance of the recognition system. It remains a challenge in terms of OSR and incremental learning. Even though the thesis does not aim for a significant contribution in this area, we investigate different evaluation statistics from the fields of audio recognition, image/face classification, speakers/speech identification, bioinformatics, and internet securities. We follow the state-of-art methods to measure the performance of our methods in the context of our target sound recognition application.

1.2. Thesis Aims

This research aims to develop novel approaches to solve the audio recognition problem when the audio data are obtained from classes existing during test time and not necessarily present at training time. Machine-learning algorithms must satisfy certain requirements to tackle such problems. We emphasize three requirements related to audio recognition which we aim to address:

1) The features extraction process needs to retain useful information about the audio sounds.
2) The classification systems should be capable of operating in an open set scenario.
3) Incremental learning should be allowed for the database to grow and for the model to be updated with new categories.

In this thesis, we are interested in addressing solutions to the mentioned problems. The fundamental problem when applying any system using probability is how to create a reliable probability model without overfitting. As a result, this research focuses on determining the classifier's performance under realistic conditions and figuring out how to improve recognition accuracy. In our attempt to help build a smart system for detecting and recognizing events in audio-based surveillance, we studied events that can occur in everyday life. Translating these events to a numerical meaning requires a good feature extractor, and we present different audio features in Chapter 4. From the three requirements of the previous paragraph, we investigated traditional pattern recognition techniques that can work in audio recognition applications, testing their capability for learning models of new sounds. Chapters 5, 6, and 7 describe our research contributions to propose solutions to the three requirements.

1.3. Research Contributions

The major contributions of this thesis are a consequence of the modeling approaches which are related to the three requirements for the task of audio classification specified in the previous section:

1. We reviewed the past and current efforts made for audio recognition. This multidisciplinary study presents the up-to-date literature related to audio classification and signal processing methods for commonly used machine learning algorithms and feature extraction techniques.
2. We formalized the problem of OSR where not all audio categories are known priors, and developed an algorithm for multi-classes recognition, with a rejection option for classes that are novel to the system. We utilized the probabilistic calibration of an SVM classifier and modified it under the OSR scenario. We proposed the use of the peak side ratio (PSR), a distribution technique to find thresholds. We used well-known metrics in our analysis to evaluate the proposed algorithm on different variations of open sets.
3. We addressed the issue of incremental learning, where the system is continuously updated with additional audio categories. We developed tools and techniques with a specific focus on audio recognition. We developed an algorithm for updating SVM models to improve the recognition performance of audio classification tasks. We combined the open-set framework with incremental learning procedure, so this allows new unknown classes to be recognized and tolerates the models learning constantly without the need for retraining the whole system.
4. We utilized the extreme value machine (EVM), which has successfully been used in recent recognition methods. The EVM distribution estimates the kernel distribution for each

positive training. It separates hyperplanes between positive training samples and the closest negative samples. We used the EVM distribution with the SVM classifier to perform an algorithm applied to the audio recognition task in open set scenario.

1.4. Thesis Publications

Journal Papers:

[1]. H. Jleed and M. Bouchard, "Open Set Audio Recognition for Multi-Class Classification with Rejection," *IEEE Access*, vol. 8, pp. 146523-146534, Aug. 2020, DOI: 10.1109/ACCESS.2020.3015227.

[2]. H. Jleed and M. Bouchard, "Incremental Multiclass Open-Set for Audio Recognition," in *Int. J. Adv. Intell. Inform*. 2022. (accepted)

Conference Papers:

[1]. H. Jleed and M. Bouchard, "Acoustic environment classification using discrete Hartley transform features," 2017 *IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, Windsor, Ontario, Apr. 30 - May 3, 2017, pp. 1-4, DOI: 10.1109/CCECE.2017.7946646.

[2]. H. Jleed and M. Bouchard, "CNN-Based Audio Recognition in Open-set Domain", *2nd International Conference on Signal Processing and Machine Learning (SPML 2022)*, online, May 12-18, 2022.

[3]. H. Jleed and M. Bouchard, "Audio event recognition in a smart office environment", *International Conference on Communication, Electronics and Electrical Engineering (ICCEEE)*, New York, Sept. 1, 2022.

1.5. Thesis Organization

The organization of the thesis is as shown in Figure 1.1 where the summary of each chapter is described.

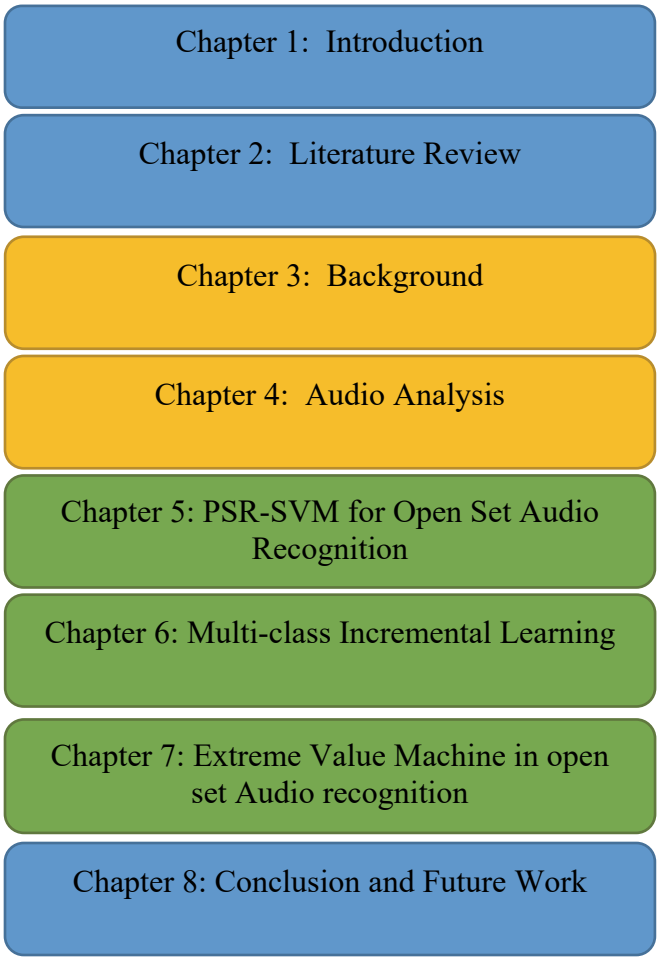


Figure 1.1 Structure of the Ph.D. thesis. Green chapters refer to the thesis contributions. Yellow chapters refer to the conceptual background

Chapter 1: Introduction

This chapter discusses the motivation, significance, problem formulation, and novelties of the proposed research.

Chapter 2: Literature Review

This chapter will outline the past and current study of audio recognition, and describe the most common features used for audio domain pattern recognition. This chapter will also carefully describe existing current audio recognition systems based on SVM or EVM.

Chapter 3: Background

This chapter provides background materials for audio recognition. It also discusses the learning assumptions used in literature. The evaluation metrics are explained, with their usage.

Chapter 4: Audio Analysis

This chapter shows the procedures for audio analysis, applied to the audio material before it is used by the recognition systems. These procedures include voice activity detection (VAD) and audio normalization. Further description of audio features tools is also provided.

Chapter 5: PSR-SVM for Open Set Audio Recognition

We present a proposed method to solve the OSR problem, where the training sets consist of a few known classes. We introduce the Peak-Side Ratio Support Vector Machine (PSR-SVM) classifier that can recognize the predefined classes and identify the novel classes.

Chapter 6: Multi-class Incremental Learning (MCIL)

We address the problem where there are new classes to be added to the system. We propose two methods that can incrementally update their modules as an extension of the multi-class recognition. The first method uses the PSR_SVM for OSR. The new class features are saved for incremental learning. The new models for saved classes are built according to the SVM procedure, where the new data are considered positive, and the rest of all training classes are considered negative. The second method is similar to the first one, but it uses only the nearest neighbor classes to build the new model.

Chapter 7: Extreme Value Machine in Open-set Audio Recognition

We present a new algorithm that combines a supervised learning algorithm for multi-class classification and extreme value distributions to handle new instances of known classes. The design of this algorithm is using extrema from known negative class distributions that lie near the positive sample distributions.

Chapter 8: Conclusion and Future Work

We conclude this thesis by presenting a conclusion of our approaches and contributions.

Chapter 2: Literature Review

This chapter reviews methods reported in the literature for audio recognition applications. Audio recognition involves the computation of various statistics to accomplish the best possible performance. The chapter is divided into three subsections. The features extraction process is investigated in the field of audio recognition. Support Vector Machine (SVM) algorithms that have proven to be very efficient in machine learning fields are described in the second subsection. Extreme Value Machine techniques are studied in terms of open set recognition in the third subsection.

2.1. Audio Features Extractions

Recognition of the received sounds is to be achieved using suitable machine learning algorithms. The first step of these algorithms is feature extraction processing. Features are distinguishing patterns extracted from a sound signal and extracting the most appropriate features improves the machine learning algorithm performance. Ideally, the features should have three aspects:

(1) They should emphasize the differences between the classes.

(2) The features' dimension should be adequate. If the features' dimension is too high, this will cause classifier confusion; whereas, a dimension too low does not provide enough information;

(3) Features should not have a high correlation with each other.

2.1.1. Mel-Frequency Features

Mel-Frequency features have proven to be particularly pertinent in the speech processing domain because of their proximity to the glottal excitation and the vocal tract [8]. These properties make them also suitable for general audio discrimination, as they capture key assets used in human

hearing. The Mel-frequency cepstral coefficients (MFCCs) features are obtained from the power spectrum, which is rescaled according to the Mel-frequency scaling. The resulting spectrum bins are framed into several critical bands that typically consist of overlapping triangular filter banks. In the last step, a discrete cosine transform (DCT) is applied to the triangular filter bank output magnitudes to obtain vectors of MFCC features.

In addition to being widely used in speech recognition research, the MFCCs are used in some non-speech research as well; for example, the music system developed by Pye [9] used MFCC features to classify audio signals compressed in MP3 format (MPEG audio Layer-3). Sharan and Moir [10] and Foggia et al. [11] published more general audio studies and concluded from their study that cepstral features outperform their comparative features. Heittola et al. [12] proposed a sound event detection system using MFCC features coefficients. They studied the benefits of using context-dependent information in audio events detection. Chu et al. [13] analyzed and recognized environmental sounds using MFCC features. They used a matching pursuit algorithm to extract time-frequency domain features to supplement the MFCC when needed.

2.1.2. Other Features

Besides MFCCs coefficients, different features have been used for audio classification. Mitrovic et al. [14] provided a survey of audio features used for other purposes (e.g. audio retrieval) as well as the challenges in features design. A collection of conventional features is described in the survey, bringing features from different domains together. Peeters [15] summarizes a collection of audio features and organizes the features between global and frame-based descriptions. In [16], the authors described a broad survey of features for multimedia retrieval that extracted both audio features and video features together from a group of TV programs. To demonstrate redundancy,

they focused on the link between these features. Other features have been reported, including but not limited to indexing-based features [17], MPEG-7 audio features [18], and bag-of-audio-words features [19]. Also, features based on the Discrete Hartley Transform [20] have been shown to be useful for audio scene classification applications.

2.2. Support Vector Machine

Support Vector Machines (SVMs) are tools built to classify data. They have the ability to handle large amounts of data quickly and effectively. The SVM has solid theoretical foundations for linear and non-linear machine learning algorithms. Huang et al. [21] examined the SVM's suitability for regression and classification. They outlined the main difference between SVM and other classification methods and explained how SVM algorithms achieve their advantage over different algorithms. The statistical learning theory of SVM was introduced by Vapnik [22]. The SVM can compute a hyperplane and minimize the empirical risk as well as the structure of risk minimization principles. The SVM was originally designed as a binary classifier, but it has been used most of the time for multi-class learning by considering the problem as a set of binary classification problems (one-vs-all approach) with the assistance of a kernel function (see Appendix A).

In general, SVM depends on three concepts to find a hyperplane for a certain class.

- Kernel functions to map the data to a high dimensional space and simplify the recognition problems, so the SVM can separate the classes.
- Support vectors that are located near the decision surface. They are very important to model classifiers in the training stage.

- Optimal Separating Hyperplane (OSH) that divides the data with the largest margin. The resulting maximum margin classifier will have good generalization properties.

The following subsections present the previous work using SVM for the application of audio recognition, in addition to the state-of-art techniques used to tackle open set recognition.

2.2.1. Audio Recognition with SVM

The literature for audio recognition has grown significantly in recent years. The published researches differ mainly in the extracted set of features used to present the audio signals and the applied classification algorithms to distinguish these signals from each other. We will present some of the research results that have used the SVM as a main classifier in their findings.

Lopatika et al. [23] employed an SVM algorithm to distinguish between different types of dangerous circumstances. They used a set of different audio features to classify and localize four classes. Likewise, Hilal et al. [24] used SVM and linear discriminate analysis (LDA) classifiers to identify and localize a set of environmental sound events. Huang et al. [25] used SVM for audio events classification. They utilized different features selections to recognize seven events, i.e., screaming, crying, speech (male), speech (female), laughing, knocking, and explosion. Mahana and Singh [26] presented a comparison among the audio classification algorithms. They found that SVM with a radial basis function kernel had a good performance while requiring less training data to achieve an outcome compared to other machine learning (ML) algorithms. In the same way, the problem of road surveillance was considered in [11], where two hazardous situations, car crashes and tire skidding, were detected.

2.2.2. Open Set SVM Recognition

In real-life recognition problems, there are various classes received in the testing stage that have not been seen in the training/modeling stage. Therefore, we need to consider an open set scenario where an algorithm is capable to reject unknown classes and correctly classify all known classes. The open sets recognition problems received more attention when Scheirer et al. [25] defined the issue of open set image recognition. They introduced two terms; minimizing the positively labeled open space (PLOS) and the known labeled open space (KLOS). These terms defined an open space as a features space region that lies outside the support of the training trials. Researchers have utilized this idea and modified SVM classifiers to tackle the OSR problems. Junior et al. [27] proposed a Specialized Support Vector Machine (SSVM). The algorithm moves the hyperplanes in directions to balance between open-space risk and empirical risk. Similarly, Battaglino et al. [28] provided algorithms for audio scene classification in an open set. They used support vector data description (SVDD). Instead of the hypersphere, they used the minimum radius around the most positive training points.

One of the SVM designs [29] proposed a technique that reduces the number of support vectors without negotiating the classification performance. It requires only a subset of the support vectors when the Lagrange multiplier is greater than a certain threshold . Their finding provided a reduction in the support vectors number but it caused a small performance drops.

Scheirer et al. [30] introduced a Weibull-calibrated SVM (W-SVM) classifier, which uses two separate SVMs. The first is a calibrated one-class SVM fitted by the Weibull cumulative distribution function. The second SVM is fit via dual-calibrated, in-class training data and a reverse Weibull fit. For multi-class classification, they used a 1-vs-all framework. Jain et al. [31] modified

the Weibull-calibrated SVM classifier by introducing a Compact Abating Probability (CAP) model. It reduces the probability of a faraway test sample that can be misclassified as one of the predefined classes. They showed that the open space risk reduction is a balancing act between computing capabilities and support vector estimations. Cruz et al. [37] presented an open set intrusion recognition for recognizing malware samples. Scherreik and Riglin [32] modified the SVM and named it (POS-SVM) classifier. They used it for automatic target recognition systems of radar image classification. Similarly, Roos and Shaw [33] used open-set probability SVM and Eigen-template features for OSR on the field of radar images, thus, they formulated an automatic target recognition problem.

2.2.3. Incremental Learning via SVM

The concept of incremental learning differs from place to place in the literature review. Some researchers use the term incremental learning for systems that can correct themselves for better recognition. For SVM classifiers, Crammer et al.[34] published a discriminative large-margin algorithm that is based on online incremental learning algorithms that are capable of categorizing binary and multi-class data. Likewise, Xu et al [35] presented an incremental learning algorithm based on SVM. The paper performed a comparative analysis between randomly independent sampling and Markov resampling in incremental learning. These two methods and others proposed solutions for adaptive systems, when a classifier makes a prediction it receives feedback indicating whether it has a correct outcome or not.

However, our work is conducted to use the term incremental learning to describe multi-class incremental learning, i.e., an OSR with the ability to update the model as new data arrive. Liu et al. [36] and Rebuffi et al. [37] defined multi-class incremental learning as a process of teaching a

system new concepts while keeping old ones in the storage. The challenge of implementing this type of recognition is that we do not know prior knowledge about all the classes which may occur. The trivial solution is to train the model every time we add a new class, but we will face two obstacles if we solve it in this way: (1) it requires computational complexity, (2) we cannot re-train all models. In our chosen paradigm, we train only the new class data to build the new model. We propose two methods for this task. The first method utilizes the PSR-SVM classifier to apply OSR. It uses the new data as positive samples and all the previous classes data as negative samples, to build the new model. The second method does not use all the previous classes data as negative samples. It applies the distillation of the classes using a k-Nearest Neighbors (k-NN) classifier. Our work is also different from clustering [38] [39], where the authors used a hierarchical clustering algorithm to cluster the novel class samples. This way, they transfer the classification knowledge from supervised learning to unsupervised learning.

2.3. Extreme Value Machine

The Extreme Value Machine (EVM) was introduced in [33]. The model results from extreme value theory. The EVM applies the marginal distributions on the extreme value points. Margin distributions have been defined in different ways [40] [41], but they all agree in terms of maximizing the mean or median margin. In the literature, EVM has performed as a good nonlinear kernel-free classifier despite its limitations as reported in [42]. It has been used especially in computer vision and visual recognition [43]. The extreme value theory (EVT) focuses on the random variable behavior of the tails. The classifier fits an EVM distribution on a point to point basis from other classes over multiple fractional radial distances.

We assume that the margin distance of a point x_i to be half of the minimum distance between x_i and all the points belonging to another class in the training data set [42]. Then the margin distance of x_i will be,

$$M_j^{(i)} = \min_{j:y_j \neq y_i} Dist_j^{(i)} \tag{2.1}$$

Where $j \in Nk$ the number of classes in the training set. The EVT is applied to find the upper tail of the distribution $M_j^{(i)}$. To end this, the Fisher–Tippett–Gnedenko theorem (see Appendix B.3) is used to fit the distribution to estimate the largest observed $Dist_j^{(i)}$ for each point x_i. We use this estimate to label a new point x_0 as known if it is above the threshold δ. The threshold is chosen by a heuristic formula as in [44]. In the testing stage, an instant sample x_0 must be labeled as known or unknown, so the classifier needs to confirm if it comes from the training data distribution or not.

In this research, we will use EVM to support SVM classifiers to estimate a threshold for detecting new classes, which is the crucial process for open set recognition. In the literature, the threshold estimation has been studied in [45], who proposed a method to estimate a threshold using an adaptive method and extreme value distributions. They computed thresholds and removed the outlier samples to achieve better performance. Moreover, a combined algorithm of EVT and a Kalman filter has been published in [46] to detect new observations based on estimation. However, the EVM is essentially reliant on the geometry given by known classes. As a result, it may be unable to discriminate between known and unknown classes when this geometry provides false

information about unknown classes [42]. To solve this problem, we propose a method to combine the EVM and the SVM to improve the system's capability.

2.3.1. Open Set EVM Recognition

EVM provides the distribution for each of the known classes to detect new classes and to separate them from others. Therefore, EVM classifiers have been a recent research focus in OSR. A report paper was published in [43] to present an overview of how EVM becomes a good choice in visual recognition. In [30], the Weibull distribution was used to calibrate the posterior probability scores for the rejection threshold, and to enhance the classifier's ability to model decision boundaries. Luca et al. [47] used extreme value theory for detecting rare events in children's behavior by attaching an accelerometer to the body. A method was proposed in [48] that used posterior probability calibrations and Extreme Value Theory (EVT) based calibration. Moreover, the EVM was used in [5] for the security classification, where they used the EVM to obtain the signature probability for a certain class.

2.3.2. The EVM and Deep Learning

A deep neural network (DNN) is an artificial neural network with a multi-layer architecture that includes several layers. Several developments in the machine vision and speech processing communities have recently reignited interest in DNN designs.

A method called the Open Deep Network (ODN) was presented in [49]. They use multi-class triplet thresholding to detect unknown classes. Bendale et al. [50] introduced the OpenMAX classifier for image recognition, which is basically an algorithm that makes use of the EVM and Weibull-based calibration to boost the Softmax output function of a deep network. The tail of the

greatest sample's distance from the mean activation vector is fitted using a Weibull distribution. As a result, a maximum radius is fitted around the features of every class in the training process. In the testing stage, any activation vectors that exist outside of this radius are defined as unknown open set classes. As an extension to OpenMAX, another method called Generative OpenMAX (G-OpenMAX) was introduced in [51]. The difference is that OpenMAX uses calibrated scores from known classes to determine the pseudo probability of an unknown class, whereas G-OpenMAX estimates the probability of an unknown class directly.

As can be seen from this chapter, most of the previous methods for OSR have been conducted on image processing applications, such as face recognition, radar image recognition, and medical image classification. In this thesis, we propose to make use of these cutting-edge methods and metrics of performance that have been initially developed for image processing. We investigate how these methods are used and how well they perform for audio recognition.

Chapter 3: Background

3.1. Overview

There are many ways to define a recognition process, but we will use the general definition as in [52], where a recognition process is defined as a process that acquires sufficient statistical information taken from previous date representations. The classifier makes its models and future expectations based on the similarities of these representations. A recognition process was defined in [53] as a task that assigns new inputs to one of several discrete classes or categories. Several recognition methods found in the literature produce frame-by-frame outputs of event detection when decisions are taken either locally or over a longer observation buffer.

A number of tournaments and challenges were held for acoustic events detection and recognition, such as the SiSEC evaluation campaign for signal separation [54], the MIREX information retrieval competition for music [55], the CHiME speech separation and recognition challenge [56], and the DCASE 2013~2019 challenges [57]. The recognition process uses information from experimental data and relies on computations for decision making, i.e., decision rules are made by utilizing the robustness of the machine. The problem is to construct an algorithm that can recognize samples from previously unseen classes. Zhang et al. [58] reported a review toward robust pattern recognition. They described the unsolved problems in pattern recognition including both the open set recognition and incremental learning. Krstulovic [59] underlined the importance of open set recognition and illustrated the shortage of publications for audio domain applications.

3.2. Machine Learning Assumptions

Machine Learning can be described based on the human perception. The human auditory learning process is inherently incremental and depends on experience [60]. People can recognize certain sounds among different possible sound inputs, which can include sounds that have not been explicitly heard before. The central auditory system models sounds or groupings of sound events. Eventually, the sounds are classified and labeled based on prior knowledge (closed set) or labeled as a new kind of sound with an unknown class if never heard before (open set). The new sound is stored in the memory for future processing (incremental learning).

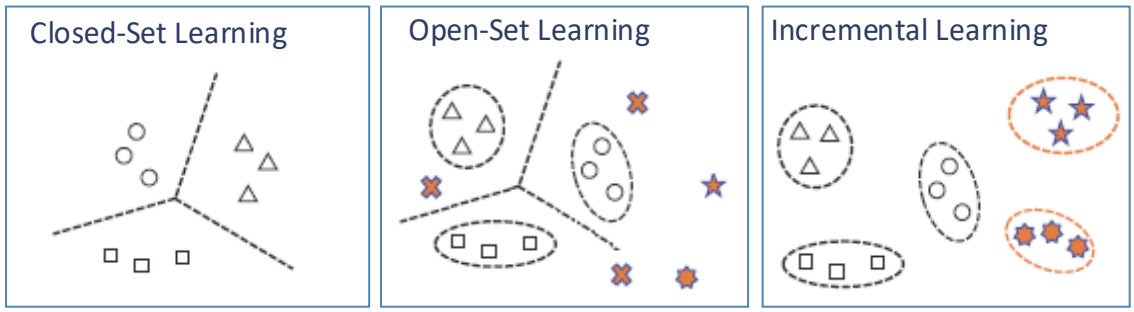


Figure 3.1 Machine learning assumptions

For machine learning, this is a notoriously difficult task. The field remains far from achieving human performance for real-world sound classification. Machine learning depends on annotated data. The labeled data are fed to the algorithms, modeling the data about certain categories. The algorithms become experts at recognizing what they have been provided. Therefore, algorithms that can only classify inputs that belong to one of some predefined classes are described as algorithms for closed set recognition. Closed set algorithms consider only a finite number of classes. There is another type of recognition in a more realistic scenario called open-set recognition, where audio classes in the test dataset have limited overlap with audio classes in the

training dataset. Under the open set recognition assumption, an algorithm may contain an option to reject inputs that do not meet certain criteria. A third type of algorithm does not discard the input sample. When the input sound is rejected, for example, they will transfer it to another process to introduce incremental learning. Figure 3.1 illustrates the three types of learning assumptions.

3.2.1. Closed Set Recognition

The basis of closed-set recognition is that classifiers are built using Nk pre-defined and fixed classes of training data, and the efficiency of the algorithm is measured using the same classes but with different samples. Let us assume that we have data $D_{train} = \{x_i, y_i\}_{i=1}^{n}$ with n examples, where x is the observed features in the instance space and y is the corresponding fixed number of Nk class labels. The aim of this task is to learn a model $f(x): X \to Y$ and to test the learned model on a different test set $D_{test} = \{\hat{x}_i, \hat{y}_i\}_{i=1}^{n}$. Under this assumption, the decision boundaries are clearly defined and partitioned into Nk regions. The task is to assign a class label to an input of audio sound. The class label indicates one of the Nk predefined classes. This typical technique of CSR is shown in Figure 3.2.

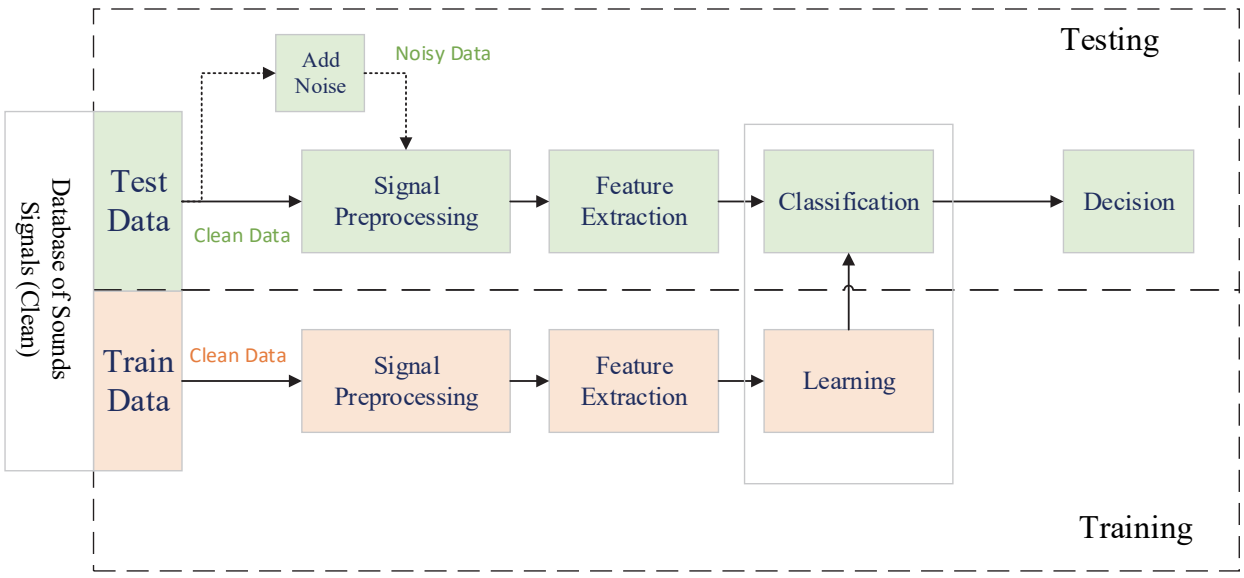


Figure 3.2 Typical closed set audio recognition technique

3.2.2. Open Set Assumption

Practically, unknown classes must be expected as inputs to a given classifier when we formulate recognition problems. The easy way to deal with this assumption is by expanding the Nk - class problem to $Nk+1$ classes by increasing the number of classes by one. The training data is given $D_{train}=\{x_i,y_i\}_{i=1}^{n}$ with n examples from Nk known classes. During testing, data from unknown classes can emerge at any time unexpectedly. we should consider rejection strategies to be added to the traditional machine learning algorithm. Therefore, the recognition task in the open set assumption rejects the new sample if the maximum of the posterior probabilities $\max_i P(i\,|\,x)$ is less than a certain threshold, whether the threshold is constant or adapted during the process.

3.2.3. Incremental Learning

In incremental learning, the existing model is expanded by adding the detected classes to the previously learned class set. To express the incremental process in practical operations, we assume that there are some incremental phases, in which the system updates its models. In each incremental phase t, the system holds the data $\{X^t, Y^t\}$ of new arriving classes C^t, in addition to storing some data $D^{o:t-1} = \{X^{o:t-1}, Y^{o:t-1}\}$ for old classes $C^{o:t-1}$. We denote the superscript $o:t-1$ to be old incremental phases and X, Y to be the data in the features domain and their labels, respectively. The letter o represents the phase at the first non-incremental state. The purpose of incremental learning is to continuously add new data and learn new classes without forgetting previous knowledge.

3.3. Validation

The incremental learning recognition paradigm is basically built on the idea of "teaching" algorithms on the training set to predict the right label for the tested input sound. A method such as cross-validation or bootstrapping with correct and strict procedures must be used to evaluate a developed paradigm's success [61]. Cross-validation is the most widely used method for evaluating prediction error, where a subset of the training set is kept aside. This subset is called the validation set [62]. The classification model uses a training set for learning, and the validation set is used as a test set for tuning the model and obtaining the parameters associated with the system.

3.3.1. Overfitting

Overfitting may occur if the model is inflated by short-run dynamics. Overfitting usually provides high performance at the training stage and bad performance at the testing stage [63]. Therefore, the test data should not be used in any manner to train the classifier or develop the classification algorithm. Cross-validation is a solution to this problem, as described next.

3.3.2. K-fold Cross-validation

K-fold cross-validation is very commonly used in audio classification. As recommended in [64], the evaluation of cross-validation folds should be averaged and viewed as a single experiment, so it guarantees that the metrics are calculated consistently over the same data. The procedure of *K*-fold cross-validation is as follows. The dataset is divided into *K* equal-sized parts. For every k^{th} part, the model is trained with the other $K-1$ parts of the data and calculates the performance of the model when it predicts the k^{th} part of the data, as shown in Figure 3.3. In this work we choose $K=5$, which means:

- The model is trained with 80% of labeled data.
- The model is evaluated on the remaining 20%.

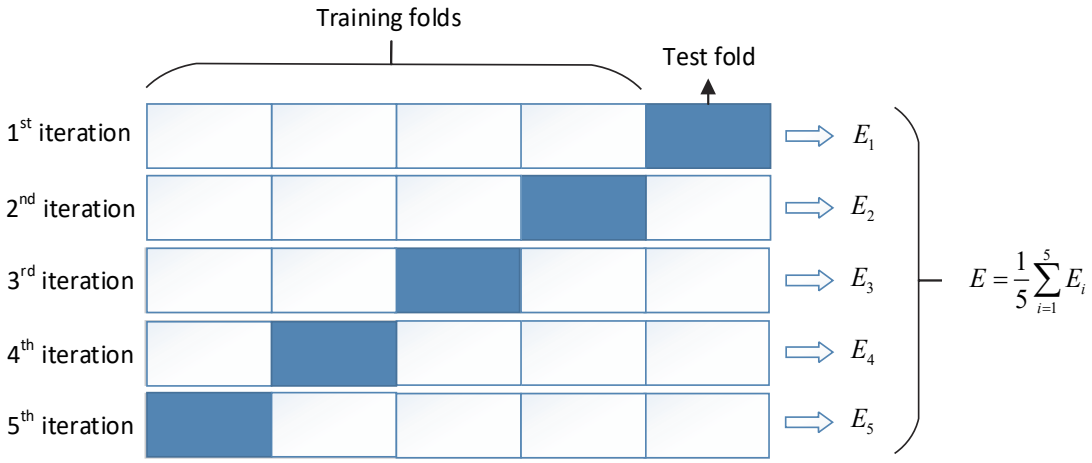


Figure 3.3 *K*-fold cross-validation

This process is repeated 5-times and combines estimates of the prediction errors. As a result, the sample mean of these five accuracies is used as the overall accuracy. An option for the cross-validation approach is stratified cross-validation, where each class in the dataset contains the same number of examples. This technique reduces the trial variance, which leads to better performance.

3.4. Measuring Performance and Averaging Options

The evaluation metrics are used to choose the best solution that can provide a more reliable classification. We approach the evaluation metrics as a multi-class recognition task. Good statistics depend on the domain of applications and their assumptions. Let us consider the binary classification, where the labels consist only of positives or negatives. Based on true labels and predicted labels, we define four intermediate statistics *TP*, *TN*, *FP*, and *FN*, which denote the true positive, true negative, false positive, and false negative counts, respectively. The definitions of these intermediate statistics are [65]:

- True-Positive (TP) is the count of positive predictions that are correctly classified.
- True-Negative (TN) is the count of negative predictions that are correctly classified.
- False-Positive (FP) is the count of positive predictions that are incorrectly classified.
- False-Negative (FN) is the count of negative predictions that are incorrectly classified.

We consider the multi-class classification as a set of many binary classification problems. Every single classifier produces a "positive" or "negative" prediction that can be "true" or "false" depending on the corresponding ground-truth label.

3.4.1. Ways of Measuring

The metrics compute a comparison between system output and reference ground truth after aligning them together. The comparison can be obtained in fixed frame length or at an event-instance level [66]. This result can thus be interpreted into two different ways of measuring performance: frame-based evaluation and event-based evaluation.

(1) Frame-based evaluation: The metrics compare system output and reference for each frame of the signal.

(2) Event-based evaluation: the system output is considered the same within the whole duration of an event. This means that metrics compare system output and corresponding reference event by event.

For each of the above evaluation ways, we need to define what constitutes correct detection and what type of errors the system produces [67]. The event-based recognition process practically classifies each frame and then smooths the classification outputs of these frames using a majority voting technique or the mathematical mode.

3.4.2. Averaging Options in Calculating Metrics

The two most common averaging methods to compute the total number of the intermediate statistics of multiclass recognition are described as follows.

- Micro-averaging: This option gives equal weight to each individual decision. Each frame (frame-based evaluation) or event (event-based evaluation) has equal influence on the system outcome. A number of intermediate statistics *TP*, *TN*, *FP*, and *FN* are aggregated, and the metrics are calculated based on the overall values.
- Macro-averaging: This option weighs each class equally. The intermediate statistics are aggregated separately for every event class over the test data. This results in values that emphasize the system behavior on the smaller classes in the considered problem

Overall, the micro average is the study of the individual class, whereas the macro average deals with aggregates or totals [68], [69]. If classes all have the same amount of samples, the score will be the same for both macro- and micro-averaging.

3.5. Evaluation Metrics

The evaluation metrics are key factors in evaluating the classification performance and guiding the classifier modeling. We borrow the evaluation of systems from neighboring fields, such as face recognition [70], image classification [71] and speaker recognition [72]. Many metrics can test the ability of any multi-class classifier. A comprehensive review can be found in [73], but we use the most important metrics as follows.

3.5.1. Reduced Confusion Matrix

If we reduce the confusion matrix to evaluate only the known targets and unknown targets, we get a 2x2 matrix called the reduced confusion matrix. It is used to evaluate the open set performance [33]. It shows how well the algorithm distinguishes known targets from unknown targets, as in the following table.

Table 3.1 Reduced confusion matrix

		Predicted	
		Known	Unknown
Actual	Known	*TP*	*FN*
	Unknown	*FP*	*TN*

3.5.2. Confusion Matrix

The confusion matrix is an accuracy metric used to evaluate the recognition performance. It works well if the class labels are uniformly distributed [69]. It is a good choice for CSR tasks. The confusion matrix $CM(i, j)$ summarizes the performance of multi-class classification. Each row corresponds to a predicted class and each column of the matrix corresponds to an actual class. The diagonal of this matrix reveals the correct prediction when $(i = j)$ It is more convenient to normalize the matrix. A row-wise normalized confusion matrix $CM_n(i, j)$ is used in this work as defined in [74], where every element in the matrix is normalized by the sum of elements in each row:

$$CM_n(i,j) = \frac{CM(i,j)}{\sum_k CM(i,k)} \tag{3.1}$$

3.5.3. Accuracy

The accuracy of classification is a statistical metric that defines how well a recognition algorithm predicts the right decisions [67]. For multi-class recognition, we compute the average accuracy as:

$$A = \frac{1}{Nk} \sum_{i=1}^{Nk} \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \tag{3.2}$$

Where Nk is the number of training classes, and the intermediate statistics in the confusion matrix are as shown in Figure 3.4.

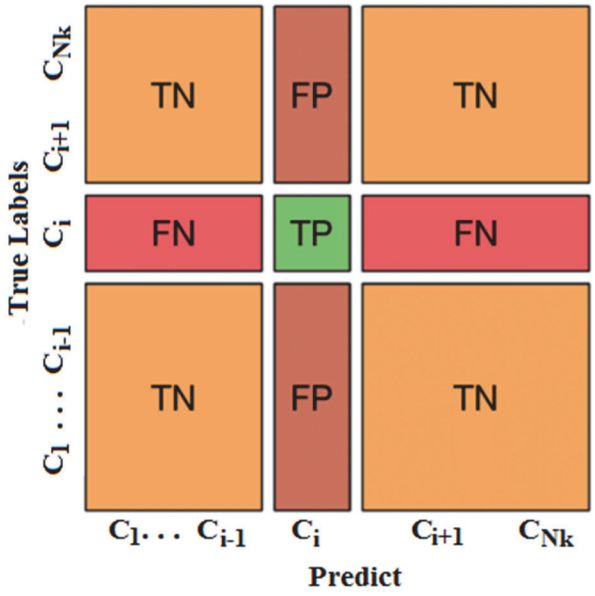


Figure 3.4 Interpreting multi-class confusion matrix when Class C_i is the reference.

3.5.4. Precision and Recall

Precision is the ratio of predicted positive samples that are computed correctly (true) divided by all predicted positive samples, while recall is the ratio of predicted positive samples correctly detected

from all ground truth positive samples labels [75]. For multi-class classification, there are two ways of computation, macro-averaging as in (3.3) and micro-averaging as in (3.4):

$$\mathrm{Pr}_{macro} = \frac{1}{Nk}\sum_{i=1}^{Nk}\frac{TP_i}{TP_i+FP_i}, \qquad \mathrm{Re}_{macro} = \frac{1}{Nk}\sum_{i=1}^{Nk}\frac{TP_i}{TP_i+FN_i} \tag{3.3}$$

$$\mathrm{Pr}_{micro} = \frac{\sum_{i=1}^{Nk} TP_i}{\sum_{i=1}^{Nk}(TP_i+FP)_i}, \qquad \mathrm{Re}_{micro} = \frac{\sum_{i=1}^{Nk} TP_i}{\sum_{i=1}^{Nk}(TP_i+FN_i)} \tag{3.4}.$$

3.5.5. F-Measure

The F-measure considers both the precision and the recall in a single score. The general form of F-measure is computed as:

$$F\text{-}measure = \frac{(\beta^2+1)\left(Pr\times Re\right)}{\left(\beta^2\right)Pr+Re} \tag{3.5}$$

where β is a parameter providing a weight for the precision (*Pr*) and recall (*Re*). When β=1, this measure is called the F1-measure. Even though the F1 score has been criticized [76], it remains one of the most widely used metrics in recognition algorithms. Therefore we used the F1-measure in this work.

3.5.6. Acoustic Event Error Rate

The Acoustic Event Error Rate (AEER) metric is defined in [77] as a measure to compute the classification errors in terms of substitutions (*Sub*), insertions (*Ins*) and deletions (*Del*). The three terms are defined as follows. Substitutions *Sub* are the number of true events that are incorrectly

detected. The remaining events are insertions Ins, defined as the number of detections with no corresponding event, and deletions Del expressed as the number of reference events with no corresponding detection [66]:

$$\begin{aligned} Sub &= \min(FN, FP) \\ Del &= \max(0, FN - FP) \\ Ins &= \max(0, FP - FN) \end{aligned} \tag{3.6}$$

where T_{det} is the number of all events. The AEER is then computed as:

$$AEER = \frac{Del + Ins + Sub}{T_{\text{det}}} \times 100 \tag{3.7}.$$

3.6. Summary

This chapter presented the background of audio recognition assumptions that include closed-set recognition, open-set recognition, and incremental learning. In closed-set recognition, the training and testing data are drawn from the same label and feature spaces. Open set recognition has samples from an unlimited number of classes during the testing stage, and some of them are unseen in training. The incremental learning process is adding new modules of new classes to the system in pre-determined ways.

The chapter also explained the validation methods that have been used in the literature to validate recognition performance and avoid biased results. It also described the suitable performance measures widely used in machine learning algorithms for different disciplines.

Chapter 4: Audio Analysis

This chapter presents some audio processing operations required for audio classification. Audio processing encompasses audio digital signal processing, linguistics, vocal music and speech signal processing techniques, and multimedia database technology, among other things [78]. We first explain how the system handles audio information, which is called pre-processing procedures. It then shows the steps involved in a typical audio features extraction process. Using a good set of features extracted from the audio data can play a substantial role in improving the audio recognition process.

4.1. Pre-processing

In this section, we describe some main steps of audio signals pre-processing, such as normalization, sampling, and silence removal. This is the first step to a successful system. If samples with poor signal quality are provided to a recognition system, the performance will also be poor.

4.1.1. Normalization

The original audio sounds may have different maximum values. Some files are quiet (low-level) audio recordings. For classification, it is often desirable to have recordings at 0 dB in the decibels-qualified-to-full scale (dBFS), which means a full scale without clipping. To achieve this, we normalize each audio file to a maximum unit amplitude, to have the amplitude of the entire file to its maximum value without clipping. After normalization, the dynamic range between the softest and loudest material in each sound file is still the same, but the overall level is uniform across audio files and the files are more audible overall when they are played back. In our system, stereo

audio channels are compacted into a single mono channel by averaging them. We consider the sampling frequency for each file to be at 44100 samples/sec. If any file has a different sampling frequency, it will be resampled to this value.

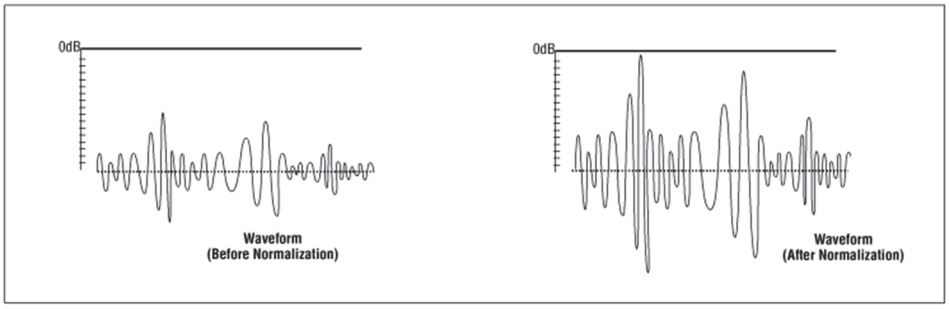


Figure 4.1 A waveform before and after normalization

4.1.2. Silence Removal

The purpose of this section is to precisely specify the way of selecting the most energetic parts of the audio signal and to discard the rest. We use a Voice Activity Detection (VAD) technique to remove the silence part of the sound. We implemented the VAD algorithm as in [79]. We computed the energies of signal and noise in the frequency domain since Fourier coefficients give better results for designing VAD algorithms [80]. We divided each recording into small frames of 20 ms with 50% overlapping. We assume that audio events and background noise are statistically independent.

Let ES_{pq} be the actual total spectrum energy and En_{pq} be some estimated background spectrum energy (e.g. what is heard during segments without a sound), where p and q denote frame index and frequency bin, respectively. We aim to estimate the noise power spectrum by smoothing and tracking spectral minima in each frequency band. We consider the first frame as pure noise, and the algorithm gets adapted for each frame. The estimated spectrum noise energy En_{pq} is updated when

the actual spectrum signal energy ES_{pq} is low. We determine these criteria by an experimentally determined threshold η, which is set to be $\eta = 2 \times En_{pq}$ from grid search experiments. If the actual spectrum signal is less than this constant in a specific frame, the estimated spectrum noise energy is independently adapted for that frame via a moving average function as:

$$En_{p+1} = \begin{cases} z \times En_p + (1-z) \times ES_p & \text{if} \quad ES_p < \eta \\ En_p & \text{if} \quad ES_p \geq \eta \end{cases} \tag{4.1}$$

where z is an empirical constant which controls the adaptation dynamics. We heuristically selected $z = 0.94$ in our work. The signal to noise ratio (SNR) of the p^{th} frame in dB is computed as:

$$SNR_p = \frac{1}{N} \sum_{q=0}^{N-1} 10 \log_{10} \frac{ES_{pq}}{En_{pq}} \tag{4.2}.$$

The VAD algorithm is computed by comparing the local SNR of the p^{th} frame (SNR_P) to a global SNR (SNR_{av}).

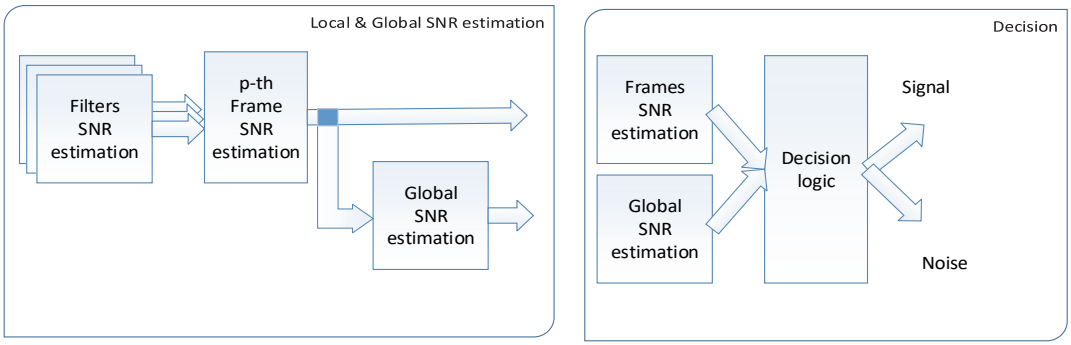


Figure 4.2 VAD paradigm. The decision is based on frames and global SNR estimation.

The global SNR is computed by taking the average local SNRs for the entire audio file. If the SNR of a certain frame is less than the global SNR, this frame is measured as silence, and it will be discarded. The outline of the VAD detector is shown in Figure 4.2. The decision process is made according to both the frame-SNR and the global-SNR estimations.:

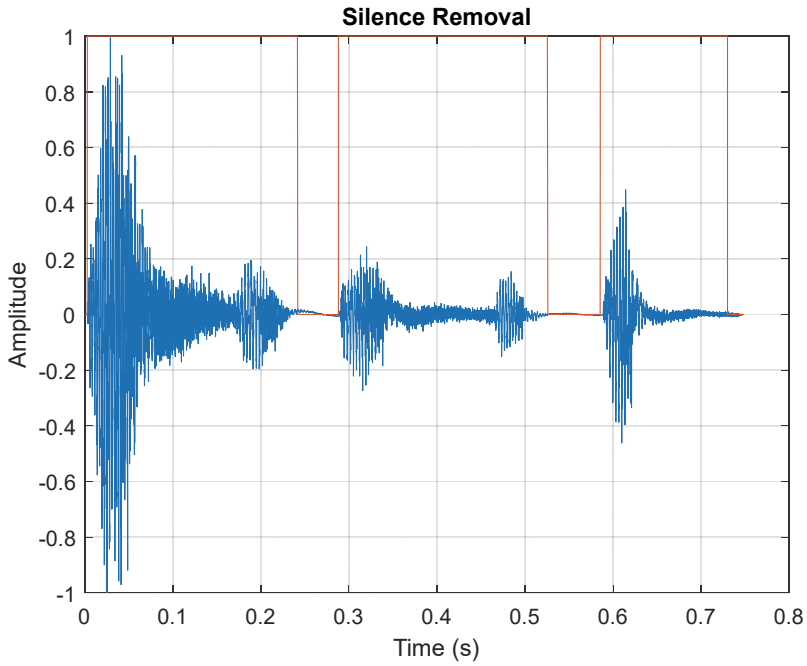


Figure 4.3 Silence removal. The red line indicates the presence of sound.

$$VAD(x) = \begin{cases} 0 & SNR_p < SNR_{av} \\ 1 & SNR_p \geq SNR_{av} \end{cases} \quad (4.3).$$

Figure 4.3 illustrates the silence detector example for an audio signal.

4.1.3. Short Time Processing of Signals

The goal here is to break the signal into possibly overlapping frames with moving window techniques. The reason is that the audio signals are statistically non-stationary. Dividing these

signals into a sequence of successive frames captures the signals in a quasi-stationary state [81]. Each frame consists of a fixed number of samples. Choosing the length of the frames is a problem-dependent task. Each frame must be long enough for an analysis method and short enough to ensure the approximate stationarity of the signal. The size of the frame is denoted by N samples shifted and overlapped by M samples, i.e., the number of samples that two successive frames have in common. We tried a few lengths from the literature and from the DCASE challenge, and heuristically selected a frame size of N=2048 samples with M=512 samples overlap (respectively 46.4 ms and 11.6 ms at 44.1 kHz sampling rate). Each frame of N samples is windowed to smooth out discontinuities. Windowing refers to a multiplying window function applied to each frame on a sample-by-sample basis. If we denote $x(n)$ the signal sequence, let the samples of the p^{th} frame be written as:

$$x_i(n) = x(n + m_i)w(n) \tag{4.4}$$

where m_i represents the window shift associated with the p^{th} frame. According to [57], the shortest length of accepted signal is 0.1 sec (sampling rate /10 samples), therefore audio signals that are too short, i.e., less than 4410 samples at 44.1 kHz sampling rate, are eliminated.

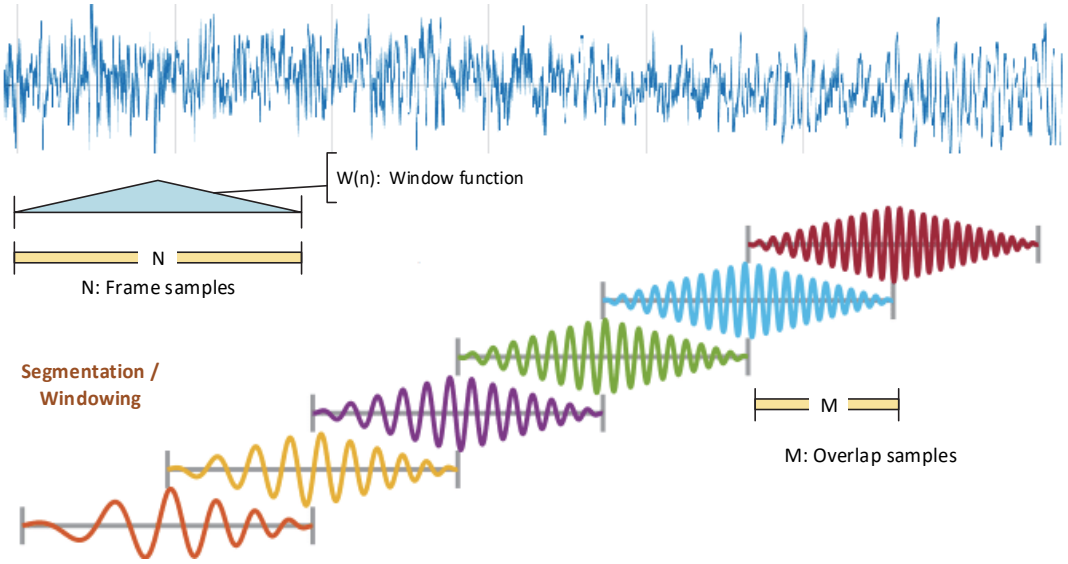


Figure 4.4 Segmentation and windowing process

For nonzero samples in the interval $\ell = 0,...,N-1$, typical analysis windows are the Hamming, the Hann, and the square-root Hann windows [80]. They are defined as follows:

$$\text{Hamming:} \quad w(\ell) = 0.54 - 0.46\cos\left(\frac{2\pi\ell}{N-1}\right) \quad \forall \ell \in (0,...,N-1) \tag{4.5}$$

$$\text{Hann:} \quad w(\ell) = 0.5\left(1 - \cos\left(\frac{2\pi\ell}{N-1}\right)\right) \quad \forall \ell \in (0,...,N-1) \tag{4.6}$$

$$\text{square-root Hann:} \quad w(\ell) = \sqrt{0.5\left(1 - \cos\left(\frac{2\pi\ell}{N-1}\right)\right)} \quad \forall \ell \in (0,...,N-1) \tag{4.7}.$$

For audio and speech processing, a common choice of window (almost exclusively) is a Hamming function [82].

4.2. Audio Features Extraction

Audio features aim to encode a large amount of audio samples into a small set of values that confines the information useful for the classification task. The audio features in the literature can be divided into three categories as shown in Figure 4.5: low-level features, mid-level features, and high-level features. The first portion of the features used in our work consists of low-level features because they can be easily computed. A second important feature set that is inherited from speech recognition consists of Mel-frequency cepstral coefficients (MFCC).

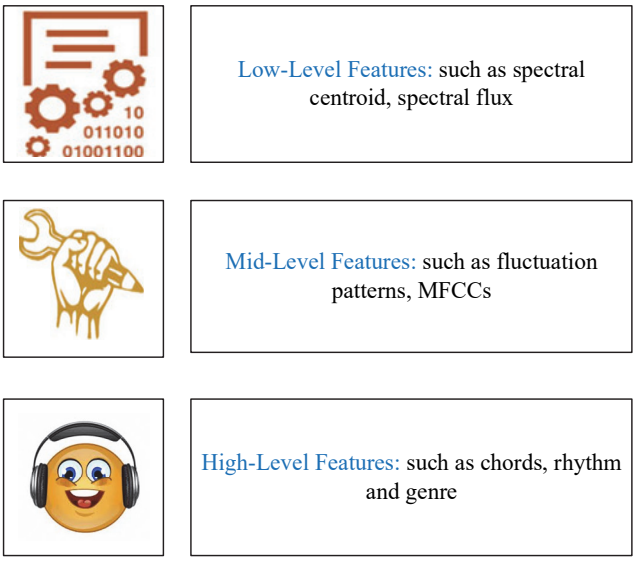


Figure 4.5 Audio features at different levels

4.2.1. Low-Level Features

These features are computed either in the time domain or in the frequency domain. The following features are the most common.

4.2.1.1. Zero-crossing rate

The zero-crossing rate computes the number of times the signal changes its sign in every frame. It describes statistics within the frame about the dominant frequency. The zero-crossing rate is defined as [83],

$$ZCR(i) = \frac{1}{N}\sum_{n=0}^{N-1}\left|\text{sgn}\left[x_i(n)\right] - \text{sgn}\left[x_i(n-1)\right]\right| \tag{4.8}$$

where,

$$\text{sgn}\left[x_i(n)\right] = \begin{cases} 1 & \text{x}_i(n) \geq 0 \\ -1 & \text{x}_i(n) < 0 \end{cases} \tag{4.9}.$$

The term $1/N$ is sometimes removed in textbooks because the frame length is fixed. However, the equation compares the crossing count in this case rather than the crossing rate.

4.2.1.2. Short-time Fourier Transform (STFT)

Features in audio classification are very often based on the short-time magnitude spectrum of the signal because the signal's frequency contents change over time. Therefore, the appropriate way is to analyze them in the time-frequency domain. The STFT characterizes the time-frequency distributions by segmenting a non-stationary signal into many frames. Each time frame is analyzed by Fourier transform to ascertain the frequencies that exist in that frame. The STFT uses a window function to reduce the side lobes in the spectra. Let $x(n)$ be the samples of the audio signal. The standard formula of Short-Time Fourier Transform (STFT) is described by [84]:

$$X(mS,\omega) = \sum_{n=-\infty}^{\infty} x(n)w(n-mS)e^{-j\omega n} \tag{4.10}.$$

Where w is an analysis window, S is a step size (window shift), and m is the index of the frames of the STFT.

4.2.1.3. Discrete Hartley Transform (DHT)

The Discrete Hartley Transform delivers a combination of the real part and the imaginary part of the Fourier transform, which embeds magnitude and phase information. The DHT is an orthogonal transformation that saves complexity over several implementations of the FFT. The standard formula of *N*-point forward and inverse discrete Hartley transforms are identical. If we have a sequence $x(n) \quad n = 0,1,\ldots,N-1$, the DHT is defined as:

$$\begin{aligned} X(k) &= \sqrt{\frac{1}{N}}\sum_{n=0}^{N-1} x(n)cas\left(\tfrac{2\pi}{N}kn\right) \;\; 0 \le k \le N-1 \\ x(n) &= \sqrt{\frac{1}{N}}\sum_{k=0}^{N-1} X(k)cas\left(\tfrac{2\pi}{N}kn\right) \;\; 0 \le n \le N-1 \end{aligned} \tag{4.11}$$

where $cas(x) = \cos(x) + \sin(x)$. The DHT has been found to provide more information than the Fourier Transform (FFT) magnitude in audio classification applications [20].

4.2.1.4. Spectral Centroid

The spread centroid measures the average of frequencies weighted by spectrum amplitude. It provides information about where most of the spectral information (power) is located. High values of the centroid mean "brighter" acoustic structures, i.e., with more energy in the high frequencies. For each p^{th} frame, it is computed as in [74]:

$$Centroid(p) = \frac{\sum_{k=0}^{N-1}(k+1)\left|X_p(k)\right|}{\sum_{k=0}^{N-1}X_p(k)} \tag{4.12}.$$

4.2.1.5. Spectral roll-off

Spectral roll-off is defined as the frequency sample under which a predefined percentage *c*% (e.g., *c* =85 or 90) of the total spectral energy is found. It computes the skewness of the spectral shape:

$$\sum_{m=0}^{m_c^R(i)}\left|X_i(m)\right| = \frac{c}{100}\sum_{m=0}^{N-1}\left|X_i(m)\right| \tag{4.13}.$$

4.2.1.6. Spectral flux

Spectral flux characterizes the spectral information variation:

$$flux(p) = \sum_{k=0}^{N-1} \left(\left| X_p(k) \right| - \left| X_{p-1}(k) \right| \right)^2 \quad (4.14).$$

It is also normalized by the maximum magnitude value of the frequency coefficients of the p^{th} frame.

4.2.1.7. Spectral sparsity

Spectral sparsity is the frequency domain feature computed by the division of the maximum spectral magnitude over the spectrum energy:

$$Sparsity(p) = \frac{\max\{\left| X_p(0) \right|, ..., \left| X_p(N-1) \right|\}}{\sum_{k=0}^{N-1} \left| X_p(k) \right|} \quad (4.15).$$

4.2.2. Mid-Level Features

The most common mid-level features are the cepstral features, which are defined as frequency-smoothed representations of the spectrum using some homomorphic filtering.

The Mel scale, short for Melody, is a subjective scale, judged by different listeners to have equal distances given a reference tone. Their coefficients are the most widely used features, especially in speech recognition, because they represent the timbral information of the audio signal and can model a good estimate of the human auditory system.

4.2.2.1. Mel frequency cepstral coefficients (MFCC)

These are obtained from the inverse Fourier transform of the logarithm of the magnitude spectrum of the signal. It uses the Mel scale to simulate the variations of the human ear's critical frequencies

bandwidth. The MFCC coefficients are computed for each frame. The frame $x_p(n)$ with length N is pre-emphasized with a high-pass filter whose transfer function is described in (4.16):

$$F_{HP}(z) = 1 - \alpha z^{-1} \qquad \forall\ \alpha \in [0.9\ ,\ 1.0] \tag{4.16}.$$

Each frame is multiplied by a window followed by an STFT to produce $X_p(k)$ [85], where p is the frame index. Then the spectral result is an input to a Mel-scale filter bank of F filters. The filter bank center frequencies are converted to Mel scales as in (4.17) [74]:

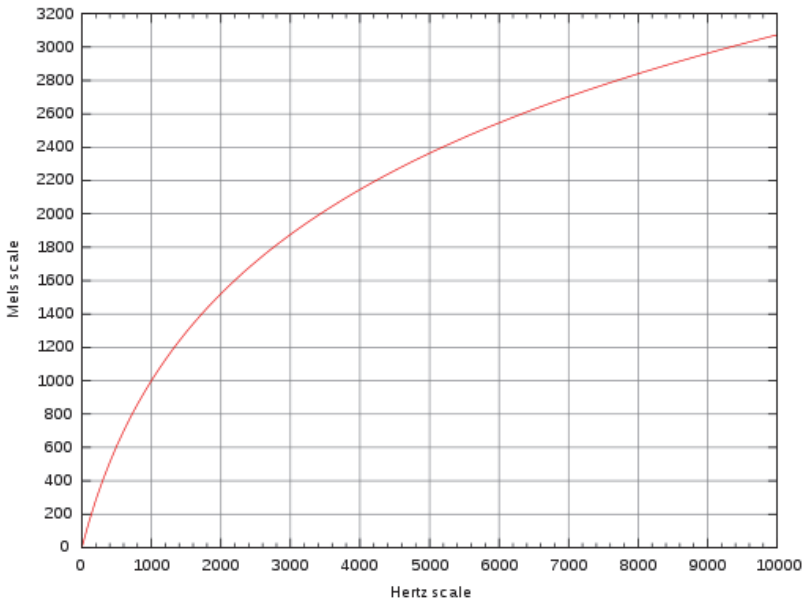


Figure 4.6 Mel-scale vs Hertz scale

$$f_{Mel} = 2595.0\ \log\left(1 + \frac{f_{HZ}}{700}\right) \tag{4.17}.$$

A fixed Mel scale frequency resolution is then calculated using a logarithmic scaling of the frequency:

$$\Delta f_{Mel} = \left(\max(f_{Mel}) - \min(f_{Mel})\right) / (F+1) \tag{4.18}$$

The inner product of the magnitude spectrum $\left|X_p(k)\right|$ and the Mel filter bank $H(m,k)$ produces the filter bank output $X'(m)$. The Mel filter bank $H(m,k)$ is computed in terms of the center frequencies $f_c(m)$ as:

$$H(m,k) = \begin{cases} 0 & \text{for} \quad f(k) < f_c(m-1) \\ \dfrac{f(k) - f_c(m-1)}{f_c(m) - f_c(m-1)} & \text{for} \quad f_c(m-1) \le f(k) < f_c(m) \\ \dfrac{f(k) - f_c(m+1)}{f_c(m) - f_c(m+1)} & \text{for} \quad f_c(m) \le f(k) < f_c(m+1) \\ 0 & \text{for} \quad f(k) \ge f_c(m+1) \end{cases} \tag{4.19}$$

The Mel scale center frequencies are equal to $f_{c.Mel}(m) = m\,\Delta f_{Mel}$ for m = 1, 2 ... F. The filter-bank energies are correlated with each other because of their overlapping. Therefore, the DCT de-correlates the energies, which means that diagonal covariance matrices $X_c(p)$ can be used for MFCC based classification:

$$cep_i(p) = \sqrt{\tfrac{2}{F}} \sum_{c=0}^{F-1} X_c(p) \cos\left[\tfrac{i\pi}{2F}\left(2c+1\right)\right] \tag{4.20}$$

Where p is the frame index, cep_i is the i^{th} coefficient computed via the F triangular filters. Since the average power in the spectrum is represented by the first DCT coefficient, the second coefficient estimates the spectrum broad shape and the higher-order components describe finer spectral details. Practically in speech recognition, the first 8–13 MFCC coefficients are enough to

represent the shape of the spectrum [86]. But for audio recognition, the audio spectral structure can be more diversified.

4.2.2.2. MFCC derivatives

It can also be useful to include as features variations or derivatives of MFCCs across adjacent temporal frames. We compute delta "differential coefficients" using the following formula:

$$\Delta cep_p = \alpha \sum_{i=1}^{2} i\left(cep_{p+i} - cep_{p-i}\right) \tag{4.21}$$

Where Δcep_p is a delta coefficient for the frame p, and α is a weight constant set to 0.2.

4.2.2.3. Gammatone Cepstral Coefficients (GTCCs)

The GTCCs are part of the cepstral features family. The processing for these features involves the Gammatone filters, which are derived from the psychophysical observations of the auditory periphery [87].

The impulse response of the gammatone filter bank is given by the equation:

$$g(f,t) = t^{n-1} e^{-2\pi Bt} \cos(2\pi ft) \tag{4.22}.$$

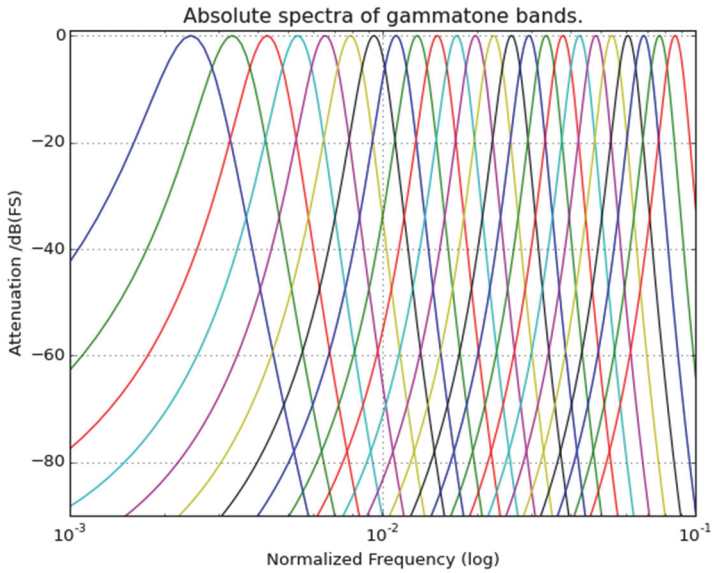


Figure 4.7 Spectra of gammatone bands

Where t is time, f is the center frequency, and n is the filter order. In this work, a 4th-order linear Gammatone filter is used. The factor B is a rectangular bandwidth defined as:

$$B = 25.17\left(\frac{4.35f}{1000}+1\right) \tag{4.23}.$$

The center frequencies are equally distributed on the equivalent rectangular bandwidth (ERB) scale. The computational details about the GTCC were explained in [88][89]. The general equation of GTCC computation is:

$$Gcep_i(p) = \sqrt{\tfrac{2}{G}}\sum_{c=0}^{G-1} G_c(p)\cos\left[\tfrac{\pi i}{2G}(2c+1)\right] \tag{4.24}$$

Where G is the number of Gammatone filters; $G_c(p)$ the energy of the signal in the p^{th} frame, and $Gcep$ is the m^{th} Gammatone Cepstral coefficient.

4.2.2.4. GTCC derivatives

As previously for the MFCCs, the first-order derivative is called delta coefficients, and the Gammatone coefficients are augmented with their derivatives:

$$\Delta Gcep_p = \alpha \sum_{i=1}^{2} i \left(Gcep_{p+i} - Gcep_{p-i} \right) \tag{4.25}$$

Where $\Delta Gcep_p$ is a delta coefficient for the frame p, and α is weight constant set to 0.2.

4.3. Summary

This chapter has presented audio analysis techniques which are the primary step for any audio recognition system. The audio analysis includes the preprocessing procedure and feature extraction. Many useful features have been explained that can be used at the front end of an audio recognition system. Audio features are contextual information obtained from an audio signal. Better features lead to better audio recognition performance overall.

Chapter 5: Multi-class Open Set Audio Recognition

5.1. Background

This chapter investigates open set recognition (OSR) appropriate for multi-class classification and recognition of audio, with a rejection option for classes that never appeared in the model's training. A probabilistic calibration is used and formulated under the open set scenario of a support vector machine (SVM) classifier. The SVM has a good reputation as a classifier for its optimization in separating hyperplanes. The decision function using a Platt-calibrated SVM offers the posterior probability of each classifier. We use a one-vs-rest fashion to conduct multi-class recognition. The SVM normally works for the closed set problem, where the testing set contains the same classes as the training set database. However, in OSR, probability estimation of unknown classes cannot be computed, so the maximum posterior probability (MAP) estimate is not enough to perform the recognition job. In addition, when novel class data are far away from any training data, the probability estimation does not give an adequate scale. Therefore, there should be a way to formulate new assumptions that can balance and minimize both empirical risks: the risk of a tested sample misclassification and the risk of labeling unknown samples [1]. All the classes of an open set classification scenario are under one of the following categories [30]:

1. Known classes (KCs): the classes when data samples are positive for training and testing. They are seen in both training and testing.
2. Known unknown classes (KUCs): classes that exist at training time, but are considered as unknown at validation time. They are used for validation to build unknown models. These are used for tuning parameters.
3. Unknown unknown classes (UUCs): classes that have not been in the training or the validation stages. They show up only in the testing stage.

A practical solution is the probability of classifier confidence as a function proportional to the distance from the decision boundary. The OSR classification system needs a special setup, methodology, and experiments. While multi-class classification for the closed set system is assessed by tracing the correct and incorrect classification, the open set classification evaluation must keep track of the errors between unknown and known categories in addition to counting incorrect multi-classifying over known categories.

A simple rejection rule for multi-class recognition with a rejection option for image recognition was presented by Zhang et al. [90] and Chow [91]. Their approaches incorporated a posterior probability estimator $P(y \mid x)$. They included a decision threshold into an existing multi-class algorithm. The samples are discarded if the maximum posterior probabilities are less than a certain threshold, i.e., an instance x is rejected if:

$$y^* = \begin{cases} \arg\max_{y_i \in Nk} P(y_i \mid x) & \text{if } P(y^* \mid x) \geq \delta \\ \text{"Unknown"} & \text{otherwise} \end{cases} \tag{5.1}$$

where $\delta \in [0,1]$ is a fixed threshold that sets the error-rejection trade-off. Chow's criteria somehow simplify the Bayesian decision rule [92]. However, these rules cannot be applied directly to calibration scores. The decision score calibration process of SVM is a weak scale in OSR. The problem is that samples that are not from known negative classes do not necessarily imply that it is from the positive class. Therefore, we introduce in this work a distribution scale that provides a good way to distinguish unknown classes and recognize classes that were trained by the model.

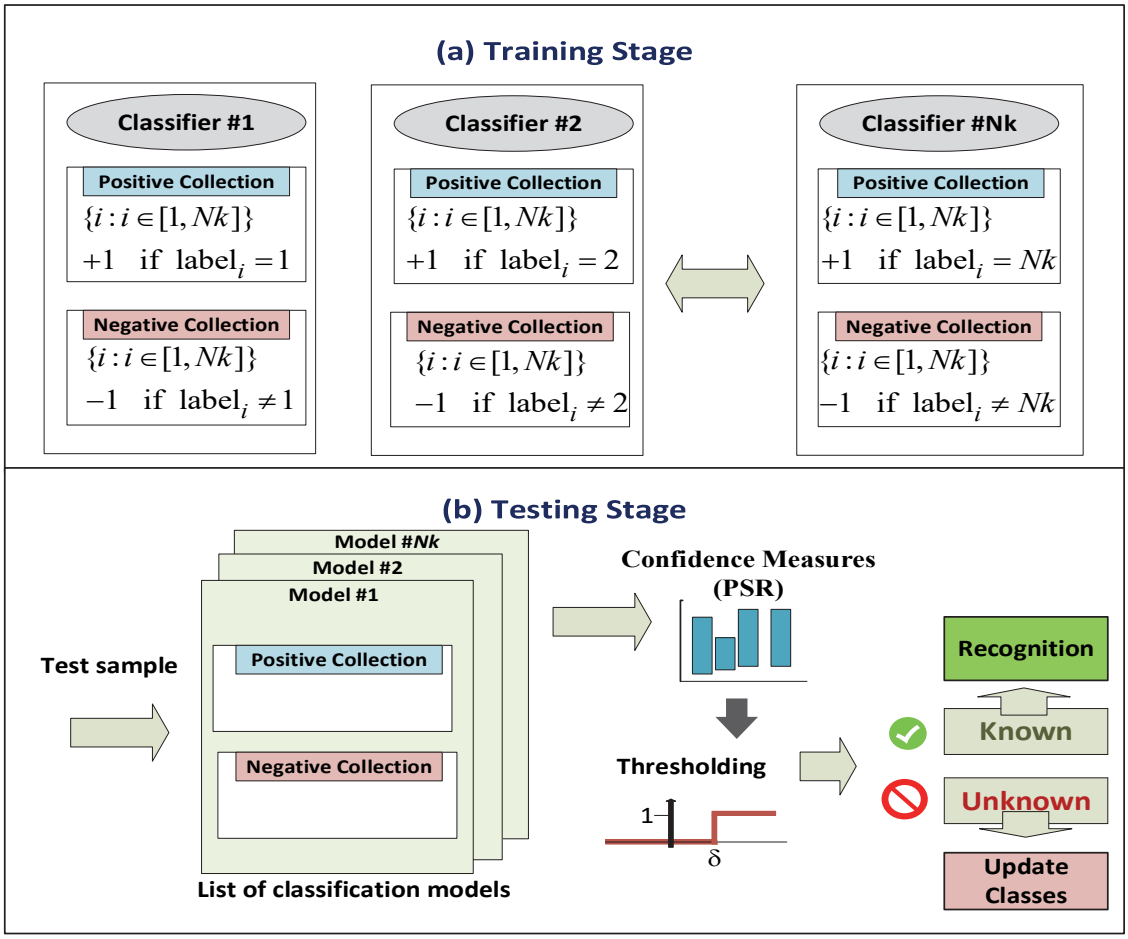


Figure 5.1 The system design of the proposed solution.

The theory paradigm in Figure 5.1 represents the proposed method of solving OSR. We propose a method using PSR measurement for audio classification to make a rejection decision. It defines the relationship between the maximum posterior probability value and other values. It expresses how the predicted posterior probabilities are structured. If the PSR is higher than a certain threshold, the tested sample will be rejected and labeled as an unknown class. If the PSR value is high, the recognition is questionable due to uncertainty and yields rejection. We compute the threshold a priori during a validation step, where we examine the system under a variety of thresholds. When we find the best threshold, we consider it as constant during the entire experiment. The methodology section explains these steps in detail.

5.2. Notation and Definition

Scheirer et al. [1] defined a term called "openness." It measures how open a system is. Let us define Y as the set covering all class labels in a given data and use the subscripts t, k and u for the target, known, and unknown sets. Openness is defined (5.2) as:

$$Openness = 1 - \sqrt{\frac{2 \times |Y_t|}{|Y_k| + |Y_u|}} \qquad \forall Openness = [0, .., 1) \tag{5.2}$$

where 0 represents an entirely closed set problem. As the number of training classes increases, the openness ratio drops.

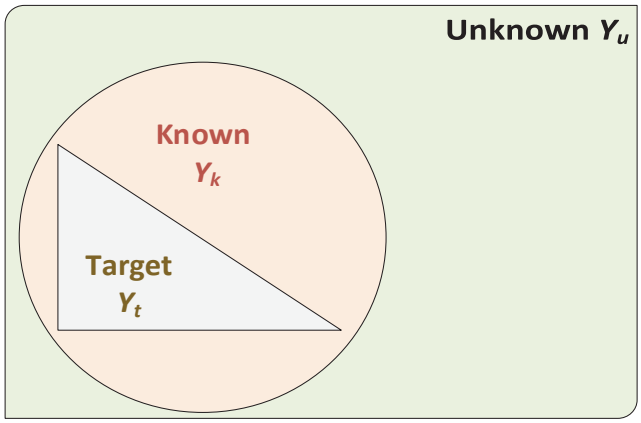


Figure 5.2 Venn diagram of acoustic classes. Y_t is the set of known target classes to be recognized, Y_k is the set of all known classes used in training, Y_t, Y_k and Y_u is the set of classes used during testing. $Y_{train} = Y_t \cup Y_k$ and $Y_{test} = Y_t \cup Y_k \cup Y_u$

The square root reduces the variance of the openness when moving toward the one value. The unknown data are considered negative classes to be used in training $Y_{train} = Y_t \cup Y_k$ and only known

classes are used in training; whereas, all classes of data are used in the testing $Y_{test} = Y_t \cup Y_k \cup Y_u$, as described by the Venn diagram in Figure 5.2. If we define the sets of training and testing classes as $Y_{train} \subset Y$ and $Y_{test} \subset Y$, the novel classes will be $Y_u = \{y \mid y \in Y_{test} \text{ and } y \notin Y_{train}\}$.

5.3. Features Generation

The objective of this process is to generate representations that are useful in the audio recognition task. Unlike video signals, there are fast variations in an audio signal within a short time. The framing and normalization of the audio data were described in Section 4.1, with frames of length 2048 samples and 512 samples overlap, at a sampling frequency of 44100 samples/sec. Normalization of the audio signal is also performed as described in Section 4.1.

5.3.1. Mel Frequency Cepstral Features

The Mel frequency cepstral coefficients MFCCs, described in Chapter 4, are well-known features for audio classification. On the Mel-scale, the frequency bands are similarly spaced to mimic how humans hear a sound. 13 MFCCs are adequate for speech or speaker recognition to encode vocal track information [93]. However, for audio recognition, the spectrum can be more diversified or complex, and the use of more coefficients is recommended. For each frame, 39 MFCC coefficients are computed in this work, including the coefficient at zero frequency. 9 differential coefficients of delta MFCC are also computed, which captures the dynamic properties of the cepstrum. For the pre-emphasis filter applied to each frame $x_p(n)$, the transfer function of the filter is $H(z) = 1 - \alpha z^{-1}$, where we used $\alpha = 0.97$.

5.3.2. Frequency Domain Features

The spectrum of the signal is computed by a Short-Time Fourier transform (STFT) $X_p(k) \quad k = 0,1,\ldots,N-1$, and three of the frequency domain features described in Section 4.2.1 are implemented: spectral sparsity, spectral flux, and spectral centroid. Their computations were described in chapter 4.

5.4. Methodology

A classification task usually involves training and testing data. The data consist of features instances that have been extracted. The training set consists of many instances. Each instance in the training set encloses one label and several features. Closed set audio classification has been investigated for decades, so it has well-known methods for experimentation. The open set audio recognition, however, needs a special setup, methodology, and experiments, which can distinguish audio events of interest from frequent events. The pseudocode in Algorithm 5.1 describes the PSR-SVM probability estimation process for a new test sample.

5.4.1. Database

We conducted our proposed experiments in an open set audio recognition using a dataset that consists of audio events recorded in an office-like environment retrieved from DCASE2013 [94] and from the Freesound online database [95]. The recorded events vary in size and noise levels. Event classes used were: short alert, clearing throat, cough, door slam, drawer, keyboard clicks, keys putting on the table, knocking the door, laughter, click a mouse, page-turning, pen, pencil touching table surfaces, phone, printer, speech, and switch. The challenge when we work with these sounds is the strong similarities between some of these sounds. To conduct our experiments in OSR, we randomly

selected six classes to model our classifiers and kept the remaining ten classes as unknown classes. For each trial, 2-5 classes from the known classes are selected to be target classes to create different levels of openness. From the unknown classes as well, in each set 3-9 classes are chosen. Experiments in multiple trials are performed with 28 combinations.

Algorithm 5.1 Multi-class PSR-SVM prediction

Required: **Test vector** x

Required: **Threshold** δ **computed during validation.**

Required: **Class labels** $y_i,\ i = 1,...,Nk$ $\Leftrightarrow$ Nk **is the number of training classes**

Required: **1-vs-all SVM decision functions** $f_i,\ i = 1,...,Nk$

1. **Function predict(** x, y, f, δ, A, B **)**
2. $i' = \arg\max(f_i(x))$
3. $p = P(y_{i'} \setminus f_{i'}(x))$ $\Leftrightarrow$ **via equation (5.6)**
4. **PSR** $\Leftrightarrow$ **via equation (5.7)**
5. **If (PSR >** δ **) then return** $y_{i'}$
6. **Else return** $y_{unknown}$
7. **Endif**
8. **End function**

5.4.2. Signal Processing

The input audio files are preprocessed with frame segmentation, normalization, and windowing. The sound file has silent frames in the recording. To avoid ambiguity in sound event labels, we used the silence removal technique described in chapter 3 to select the frames that have information and discard the rest. We use Voice Activity Detection (VAD) as described in Section 4.1.2. After silence removal and annotation/labeling, the audio clips are sliced into frames of 2048 samples with 512 samples overlap, as described earlier. Any audio event is defined as having at least four-consecutive frames that are not silent.

5.4.3. PSR-SVM Classifier

We propose our algorithm to solve OSR using the SVM classifier with peak-side-ratio (PSR). The SVM optimizes the hyperplanes and margins to separate data spaces of different classes. The SVM is originally a binary classifier. When we use it in multi-class recognition, it requires several SVM classifiers operating in parallel. We use the one-vs-all (OVA) structure to build these classifiers. This structure needs the number of classifiers to be the same as the number of classes Nk . The OVA structure applies a winner-takes-all strategy to the classifiers' outputs. Each classifier will consider one class positive and the rest as negative. For the j^{th} binary SVM classifier, the classifier will label the class j a positive class (+1), whereas the remaining classes are labeled as negative (-1). That is true for other classifiers as well. The decision function of each SVM classifier is given by:

$$f_{\omega,b}(x) = \operatorname{sgn}\left(\langle \omega, \psi(x) \rangle + b\right) \tag{5.3}$$

where b and ω are an offset parameter and a multi-dimensional weight vector, respectively, and $\psi : R^d \rightarrow H_{feat}$ is a kernel function. It transfers the input data into a high-dimensional feature space H_{feat}. Practically, the transformation ψ is indirectly defined by a kernel function, so that,

$$K(x_i, x_j) = \left\langle \psi(x_i)\psi(x_j) \right\rangle \tag{5.4}.$$

The Radial Basis Function (RBF) Kernel is used here, i.e., $K(x_i, x_j) = \exp(-\gamma \left\| x_i - x_j \right\|^2)$ where $\gamma > 0$. Each model is designed to find maximum separation boundaries of the j^{th} class from the rest of the other classes. A hyperplane is defined for the j^{th} class as:

$$H_{feat,i}(x) = \operatorname{sgn}\left(f_{\omega,b}(x) \right) = \begin{cases} +1 & \text{if } f_{\omega,b}(x) > 0 \\ -1 & \text{if not} \end{cases} \tag{5.5}.$$

Let us define $\hat{x}$ as a test feature sample, and the SVM computes the probability of $\hat{x}$ from the separating hyperplane and assigns it to a certain class Y_i. We want to determine the conditional posterior probability $P\left(Y_i \mid \hat{x}\right)$ with which $\hat{x}$ belongs to its class without sacrificing the discriminative ability. Platt calibration has proven to be well-adapted for calibrating SVM [96]. With the probabilities based on the decision function f, they are obtained by fitting a sigmoid:

$$P(Y_i = 1 \mid f(x_i)) = \frac{1}{1 + \exp(Af(x_i) + B)} \tag{5.6}.$$

The A and B parameters are determined by Maximum Likelihood Estimation (MLE). In the regular One-versus-All (OVA) method for multi-class recognition, the winner-take-all strategy is used. The final predicted label $\hat{y}$ is given by the classifier that has an index-linked to the largest, argmax, probability value. For OSR, the classifier decides either that the tested sample belongs to a class seen

during training or it labels it as unknown. We cannot assign the new sample to a certain class Y_j that has an index linked to the largest probability value, because we cannot estimate the probability of unknown classes $P(Y_{unknown} \mid \hat{x})$, and argmax of the probability estimation does not provide an adequate scale when novel class data are far away from any training data. Our proposed method for the detection aspect involves using the PSR (peak-side-ratio). It is similar to the p-value construction that has been proposed by Li et al. [97] and Proedrou et al.[98]. PSR is a confidence measurement that computes the difference between the maximum posterior probability value and other values. Visualization of hyperplane SVM models is presented in Figure 5.3, where the tested data lie far away from hyperplane models of training classes.

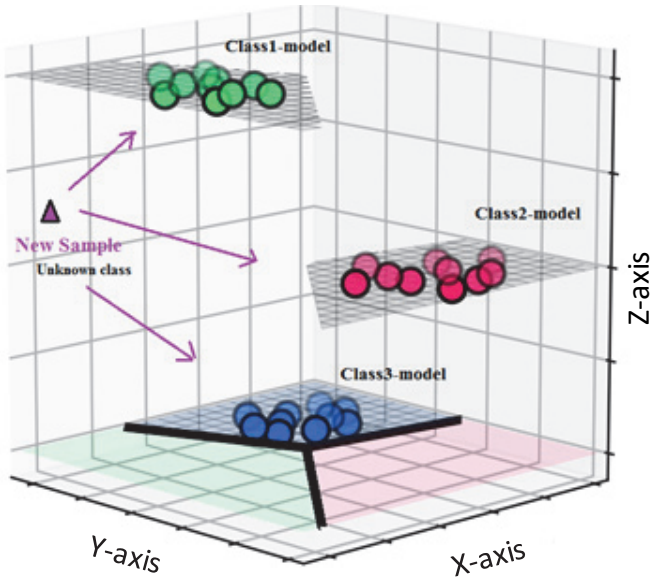


Figure 5.3 Visualization of hyperplane SVM models. When the input lies beyond any class seen during training, it does not have a strong relationship with any model.

Let us arrange in descending order posterior probability values P_j, where $j = 1,...,Nk$ and Nk is the number of classifiers, with P_1 being the largest value and P_{Nk} the smallest value. The PSR is then:

$$PSR = \left| P_1 - \frac{P_2 - P_{Nk}}{\sqrt{\frac{\sum_{j=2}^{Nk}(P_j - \overline{P})^2}{Nk-1}}} \right| \tag{5.7}$$

We rely on the PSR to make the rejection decision. If the PSR is higher than a certain threshold, we reject this sample and label it as an unknown class. Otherwise, we assign this sample to the class that has the maximum posterior probability:

$$\hat{y} = \begin{cases} \underset{i}{\text{argmax}}\, P(Y_i \mid \hat{x}) & \text{if PSR} \le \delta \\ \text{"Unknown"} & \text{if PSR} > \delta \end{cases} \qquad \forall i \in (1,...,Nk) \tag{5.8}.$$

The PSR characterizes the distribution of posterior probability values. It helps to determine the threshold for rejecting or accepting a particular class. A small PSR score gives credibility to the classifier. A high score means that the posterior probabilities are randomly distributed, which leads to a questionable classification decision, and the tested signal will be rejected. To visualize the effect of the PSR, Figure 5.4 shows the histogram of PSR and posterior probabilities. The use of the PSR boosts the accuracy for distinguishing new classes from predefined classes.

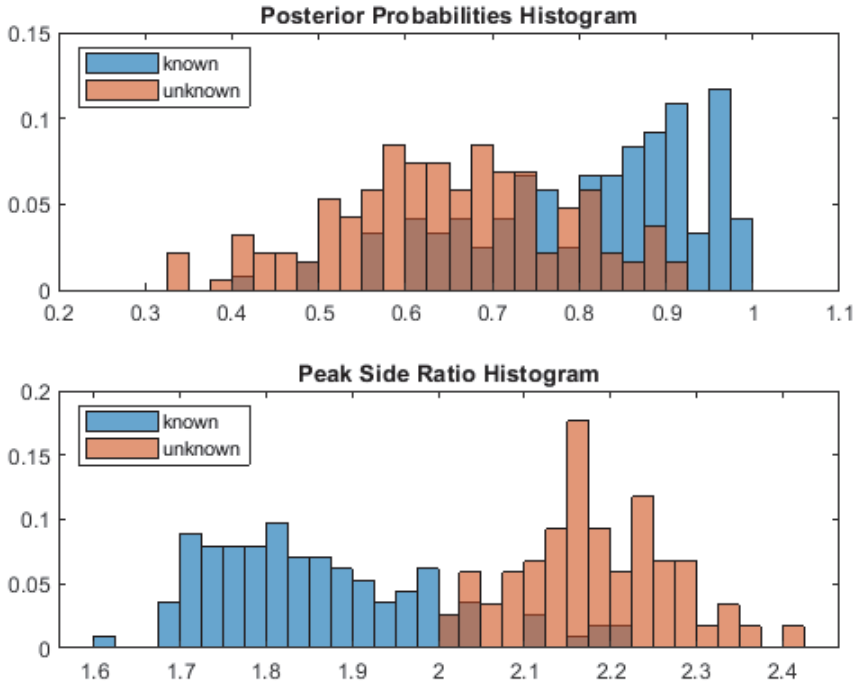


Figure 5.4 Histograms of matching rates distribution. The blue color denotes data from known classes. The brown color represents data from new classes.

5.4.4. Thresholding Criteria

To address the potentially high computational requirements, we adopt a classification with a threshold tuned using a validation set (not a part of the training set nor the test set). We adjust the decision threshold to determine the optimal value. We select 10% of the audio samples from all classes for validation, i.e., for tuning parameters based on a grid search. The remaining data are used later for the classification experiments. The optimal threshold is determined over different threshold values in the range $\delta \in [1,4]$, with a resolution of $\Delta\delta = 0.01$.

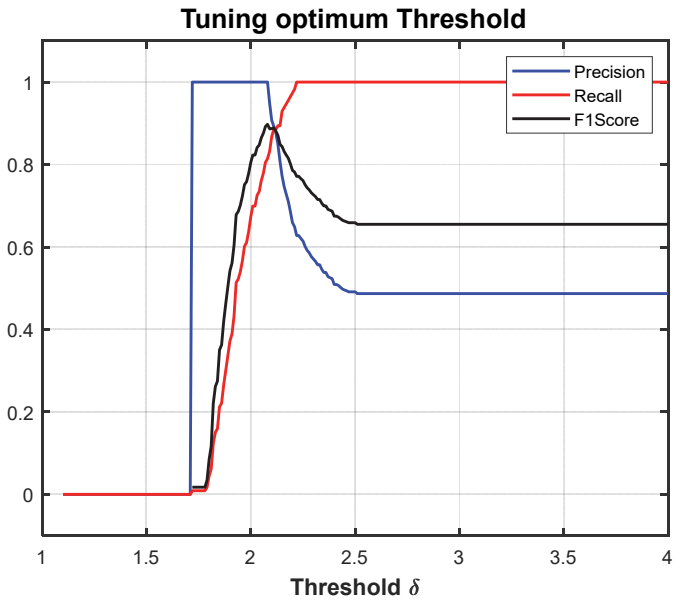


Figure 5.5 Tuning threshold values using five-fold stratified cross-validation

The F1 measure is computed for each threshold value. To improve the robustness, we use five-fold stratified cross-validation, and the F1-measure values are computed as a mean ensemble. As shown in Figure 5.5, we found that the threshold $\delta = 2.1$ leads to the best performance. That is, the threshold value optimizing the performance of the classifier on the validation set is fixed when applying the classifier on new samples in a test set.

5.4.5. Setup Protocols

The features of the validation set are used to tune the SVM training parameters. A wide range of c and γ values were tested, resulting in a total of 225 pairs of (c, γ) in each fold. The 15 different values of c were 0.001, 0.01, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100, 1000, 10000 and the 15 different values of γ were 0.0001, 0.001, 0.01, 0.1, 0.2, 0.4, 0.5, 0.8, 1, 2, 5, 10, 20, 100 and 1000. Using 5-fold cross-validation and the toolbox in [99], we plotted the grid search surface, as shown

in the following figure. We found that the optimum hyperparameters that give the best prediction score were $c = 100$ and $\gamma = 20$.

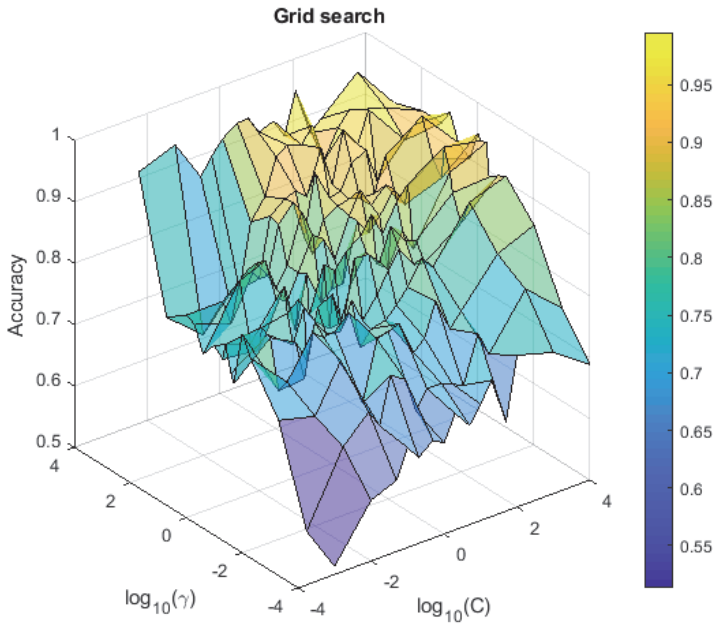


Figure 5.6 Grid search of combinations of SVM hyper-parameter values using 5-fold cross-validation to find optimum values.

For the closed set scenario, the system is evaluated by tracking the correct and incorrect classification results. The audio dataset is first split into the training and evaluation datasets. We use 5-folding cross-validations. In other words, from the data unused by the validation step, 20% of the unused data are used for testing, and 80% of the unused data are used as the training dataset to model the classifiers. The classifiers are modeled using a training dataset. The multi-class SVM is conducted with the one-vs-all method, where each event class is modeling one classifier separately. In the testing stage, a classifier fusion step is used to merge the results of multiple classifiers. The final predicted class is the one that has the maximum vote. The audio classification is performed at both frame-level and event-level. In general, event-based performance is expected to be better than

frame-based performance, because the detected event class is obtained as the most frequent frame-based class detected over that whole event, which removes some local errors.

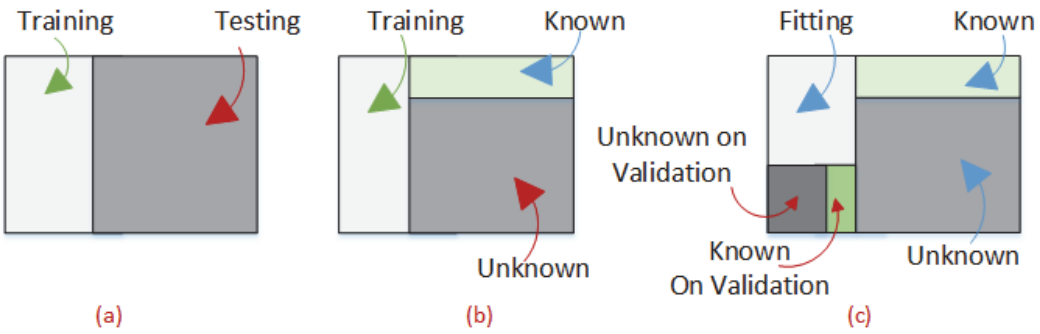


Figure 5.7 Different audio identification problem formulations. (a) Closed set problem: audio samples in training and testing stages are from the same classes. (b) Open set detection problem, audio samples in the testing stage, including new classes. (c) Open set classification problem, where a portion of training data is used for validation

For OSR, the evaluation must keep track of incorrect multi-classifying over known categories and the errors between unknown and known categories. While multi-class classification for the closed set system is evaluated by tracking the correct and incorrect classification, the open set classification evaluation must keep track of incorrect multi-class classification over known categories and errors between unknown and known categories. Therefore, two types of open set experiments are conducted: reduced OSR, which computes errors between known and unknown categories, and multi-class OSR, which evaluates the whole system.

5.5. Results and Discussion

This section evaluates the efficiency of the proposed PSR-SVM architecture. Throughout the experiments, we randomly sorted classes to be the target, known, and unknown classes, creating a gradual transition from closed set configurations to a progressively open set. We conducted our

experiments, as shown in Figure 5.7 on a closed set scenario and an open set scenario. We follow a cross-validation testing protocol to verify system reliability over multiple trials. The classification error can be one of the following:

a) Misclassification: Audio sample is misclassified with a wrong label, and it originally belongs to one of the predefined classes.

b) False unknown: The audio sample is rejected as unknown, but, it belongs to one of the predefined classes.

c) False known: The audio sample is truly unknown, but it is assigned to one of the predefined classes.

For CSR, there is only the first type of error possible. This part is important because it explains how well an algorithm learns the training data. Whereas for OSR, experiments are conducted on two different open set scenarios:

(1) Reduced OSR evaluates the classifiers' ability to distinguish unknown from known targets, as described in [32]. It evaluates new class detection, but it does not show if the sample is assigned to one of the known classes. Therefore, the first type of error is not possible in this scenario.

(2) Multi-class OSR evaluates the classifiers' ability to detect novel classes and recognize known classes, as described in [1]. If a test sample belongs to a sound class not trained by the system, it is labeled as unknown; otherwise, it is classified as belonging to the most probable class. In this scenario, the three types of errors are possible.

5.5.1. Comparison Methods

To determine the accuracy and effectiveness of our proposed process, we compare our work with the following methods for OSR described in the prior art:

a. LP-SVM classifier [96]: SVM classifier with a linear kernel and the posterior probabilities are calibrated using Platt scaling.

b. RBF-SVM classifier [31]: One-vs-all multi-class SVM classifier with Radial Basis Function (RBF) kernel, and posterior probabilities calibrated using Platt scaling.

c. 1-vs-Set Machine [1]: a linear classifier with a One-vs-All approach. It optimizes the recognition of empirical and open space risk. We used the new version of the C code implementation provided on the website [100].

5.5.2. Closed Set Recognition

When the number of unknown acoustic classes is zero $Y_k = Y_u$, the system is a full closed set. The training/test protocol is 5-fold cross-validation. The model of each class is trained on four sets of the class, and the model is tested using the remaining 5th fold. Since the purpose of the CSR experiment is to validate the ability of our proposed algorithm to discriminate among known classes, we did not conduct comparisons with other algorithms in this part.

Table 5.1 Overall accuracy and F1 metrics for closed set recognition

Measurement	Frame-based		Event-based	
	Macro-avg	Micro-avg	Macro-avg	Micro-avg
Precision	0.843	0.843	0.877	0.877
Recall	0.867	0.845	0.908	0.876
F1-score	0.854	0.844	0.892	0.876

Table 5.2 Evaluation metrics of each class for closed set recognition

	Frame-based			Event-based		
	Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
knock	**0.778**	**0.906**	**0.837**	**0.792**	**0.920**	**0.851**
printer	**0.796**	**0.963**	**0.872**	**0.800**	**1.000**	**0.889**
keys	**0.753**	**0.894**	**0.817**	**0.730**	**1.000**	**0.844**
drawer	**0.798**	**0.919**	**0.854**	**0.800**	**1.000**	**0.889**
speech	**0.840**	**0.849**	**0.844**	**0.930**	**1.000**	**0.964**
keyboard	**0.816**	**0.800**	**0.808**	**0.870**	**0.861**	**0.866**
clearthroat	**0.861**	**0.936**	**0.897**	**1.000**	**0.935**	**0.966**
pendrop	**0.878**	**0.978**	**0.926**	**1.000**	**1.000**	**1.000**
doorslam	**0.710**	**0.888**	**0.789**	**0.737**	**1.000**	**0.849**
mouse	**0.849**	**0.955**	**0.898**	**0.861**	**1.000**	**0.926**
laughter	**0.820**	**0.774**	**0.796**	**0.861**	**0.813**	**0.837**
alert	**0.900**	**0.957**	**0.928**	**0.870**	**0.916**	**0.892**
pageturn	**0.899**	**0.839**	**0.868**	**0.861**	**0.926**	**0.892**
switch	**0.990**	**0.541**	**0.700**	**1.000**	**0.516**	**0.680**
cough	**0.919**	**0.740**	**0.820**	**1.000**	**0.834**	**0.909**
phone	**0.880**	**0.936**	**0.907**	**0.920**	**0.821**	**0.868**

The evaluation statistics of CSR are summarized in Table 5.1. The overall averaged metrics (across all classes) can be computed with either macro-averaging or micro-averaging. As can be seen, there are 11% misclassifications due to the similarities among some classes. To compare results from the two figures in a simpler way, we compute the F1-score of each class and obtain a single measure for the whole model. The F1-score conveys a balance between the precision and the recall as shown in Table 5.2

The confusion matrices performance for both event-based and frame-based are computed. The matrix rows and columns refer to the ground truth and predicted labels, respectively. The matrix is normalized row-wise. All classes are considered as equal size, and the dataset becomes class-balanced, as explained in Chapter 3. In the confusion matrix, the elements in the diagonal are correctly classified, while the elements out of the diagonal are misclassified. Fig. 5.8 and Fig. 5.9 show confusion matrices for event-based and frame-based, respectively.

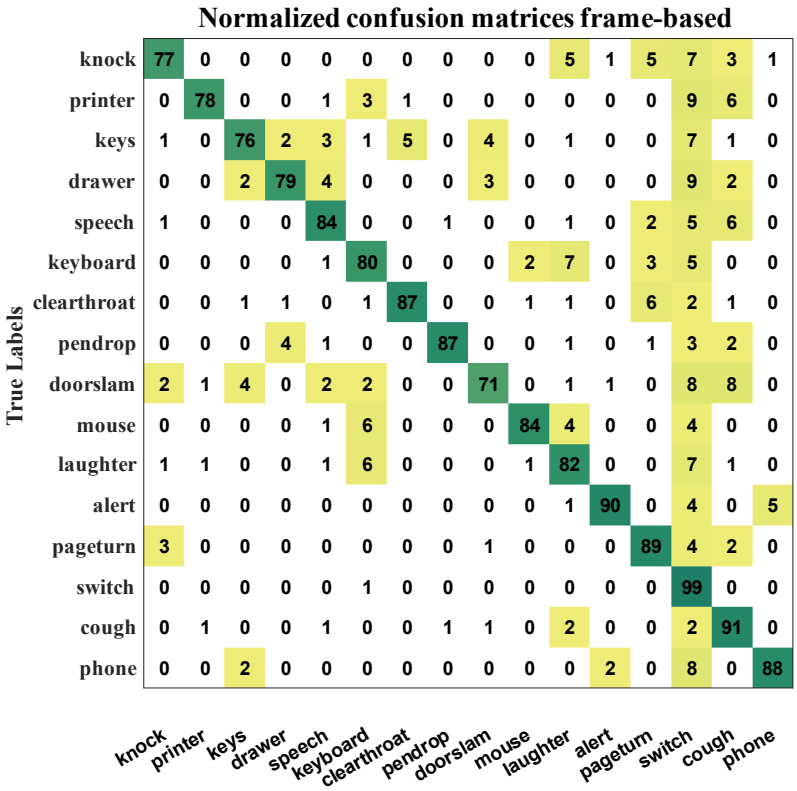


Figure 5.8 Normalized confusion matrix for frame-based CSR. Sixteen classes are tested in 5-fold cross-validation.

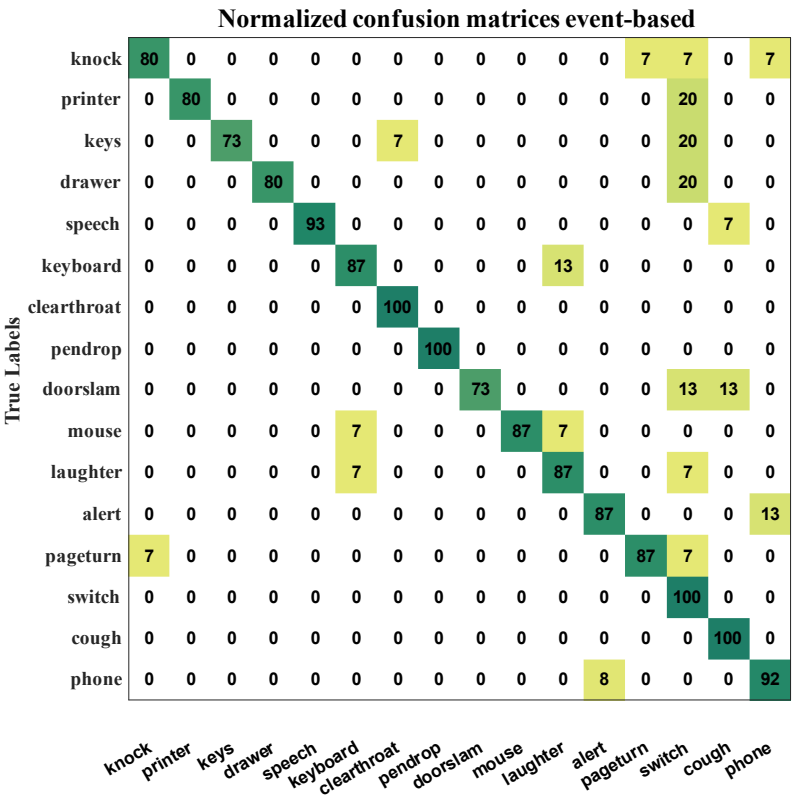


Figure 5.9 Normalized confusion matrix for event-based CSR. Sixteen classes are tested in 5-fold cross-validation.

The macro-averaged F1 scores for the experiments of these figures are respectively 0.854 and 0.892, while the micro-averaged F1 scores are 0.844 and 0.876, respectively. As can be seen, most of the signals are correctly classified. However, there have been occasions where the system struggled to differentiate between them because of their close similarity. It can be noticed in both figures that among 16 classes, 'door slam' was the most difficult sound to identify. It can be shown that it has been misclassified with other short-duration event classes, such as 'switch' and 'keys.'

5.5.3. Reduced Open Set Recognition

The reduced OSR describes how an algorithm differentiates between known and unknown data. We chose a portion of the classes to be considered as known classes for both training and testing to

simulate this scenario. This experiment is conceived solely to reject or accept a new sample. The classification decision describes whether this sample belongs to the identified groups (known classes) or not, but it does not assign it specifically to one of the known classes. We use F1-precision and recall measures to evaluate the performance of the system. We conduct experiments with different numbers of the target, known, and unknown classes. The selected trials are shown in Table 5.3 as well as our proposed algorithm responses. A macro-average treats all classes equally, as it independently measures each class and then takes their average. A micro-average combines all classes' contributions to calculate their average. Since we have more instances than others for certain groups, the micro-average here is preferable.

We evaluate the results of the reduced OSR experiments, for the proposed method as well as other methods. They are summarized and discussed in the following. We start with frame-based metrics evaluation. The results of the detection experiment of our system are shown in Fig. 5.10. Regarding other methods, rejection with Linear kernel and Platt probabilities with threshold provided the worst results. A likely explanation for this is that the Linear-Platt classifier made the calibration model weak for unknown classes, so they were falling in between separated hyperplanes. Similarly, SVM produced high F1-measurement statistics with RBF kernel and Platt probabilities at lower rates of openness, but as openness increased, it produced poor results. The problem of classification becomes more complex as the number of unknown groups increases.

Table 5.3 The openness as a function of number of target classes Y_t , known classes Y_k and unknown classes Y_u, with corresponding performance metrics.

Y_t	Y_k	Y_u	Openness		F1	AEER
5	6	9	0.310	Frame-Based	0.693	0.254
				Event-based	0.697	0.215
4	6	4	0.293	Frame-Based	0.705	0.253
				Event-based	0.760	0.146
6	6	4	0.134	Frame-Based	0.735	0.219
				Event-based	0.788	0.224
3	6	9	0.466	Frame-Based	0.560	0.360
				Event-based	0.618	0.210

In Fig. 5.11, our proposed method generally yielded better results compared with other classifiers over a wide range of openness values, except for the extreme case with an openness of 0.46, where the 1-vs-Set Machine method yielded better results. In the same way, the reduced OSR results for event-based metrics are reported in Figure 5.11. As to be predicted, the classifiers are more accurate as more classes are available during the training (less openness). We note that, for all classifiers, the performance of event-based metrics is better than that of frame-based metrics shown in the previous figure. All the classifiers tested in our experiments provided probability scores, which have a threshold choice for rejection. Thus, they all perform well during CSR. Nevertheless, as openness increases, RBF linear performance drops rapidly, and again the 1-vs-Set Machine preserved a consistent performance for very open scenarios. In general, our proposed method again produced better results compared to other classifiers over a wide range of openness values.

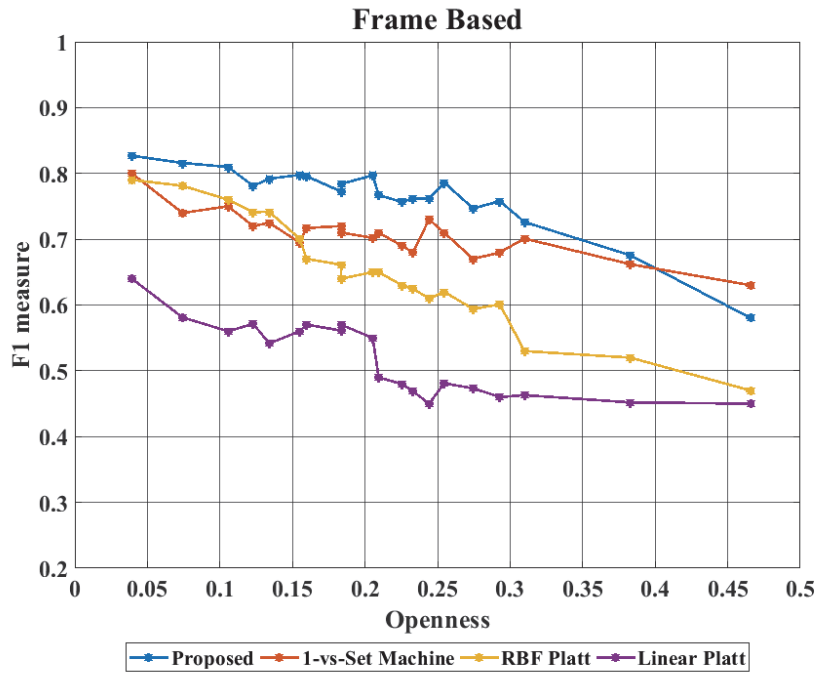


Figure 5.10 Reduced OSR results as a function of openness, increasing from left to right. Results are computed under frame-based metrics.

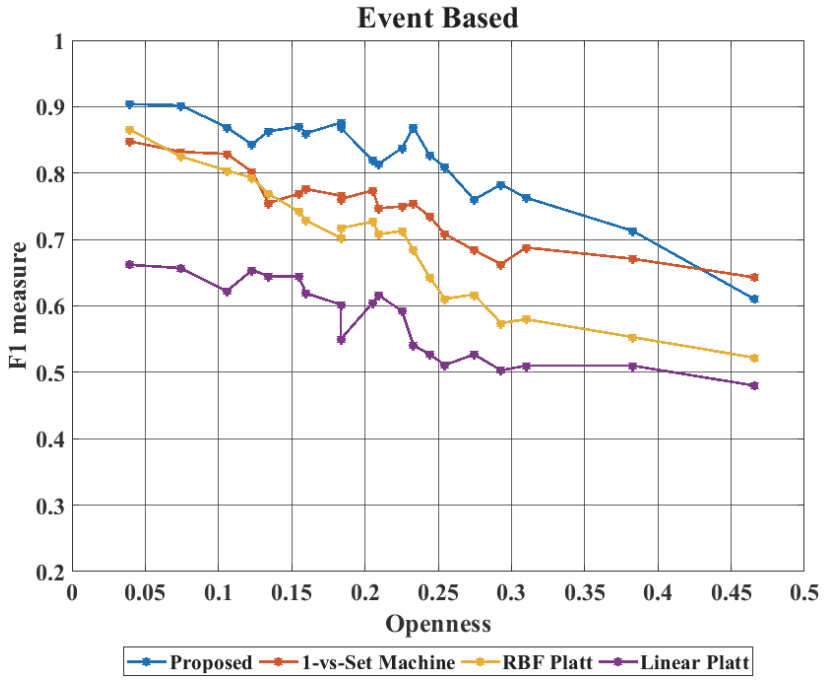


Figure 5.11 F1-measure for reduced OSR results as a function of openness, growing from left to right. Computed with event-based metrics

5.5.4. Multi-class Open Set Recognition

In the previous subsection, the results showed the ability of our proposed method to identify and reject unknown classes and its ability to accept the known classes without labeling them. The performance of the classifiers in this part is evaluated for multi-class OSR with a rejection option. Our experiments were performed by randomly selecting 6 classes for training. We consider the remaining ten classes as unknown data. To generate different amounts of openness, each trial selects from 1 to 10 classes of the available unknown classes and considers from 2 to 6 classes as target classes. To sustain a fair comparison, all the algorithms use the same data with the exact same negative and positive examples. During the testing stage, we used the PSR score to consider whether the tested samples are new classes or not, rejecting the new classes and accepting the known classes. Furthermore, the algorithms assign accepted samples to one of the known classes. The class with the maximum score, probability, or votes is the predicted class. Any algorithm that does not produce a good rejection will have very poor precision as the simulation setup becomes more open because if they miss accepting or rejecting a sample at the threshold stage, it will appear as misclassified in the metrics used.

We see from Fig 5.12 that our proposed method performs either with the best performance or near the best performance over a wide range of openness values, for the task of separating known classes from unknown classes and distinguishing among known classes. On the other hand, the Linear-Platt is again the weakest method for this experiment. The decrease of performance as the openness of the dataset increases is again very clear for most methods.

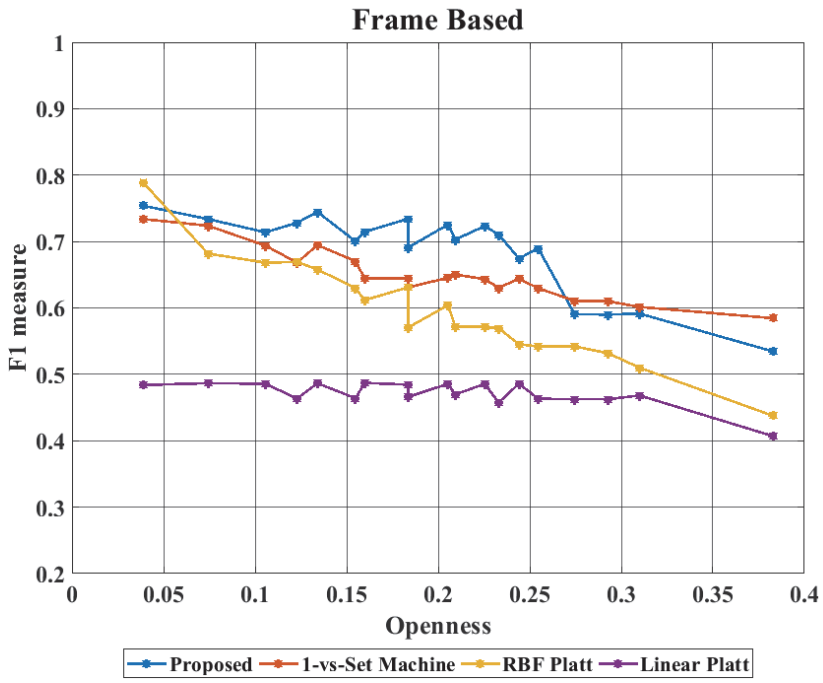


Figure 5.12 F1-measure for multi-class OSR as a function of openness, growing from left to right. Computed with frame-based metrics.

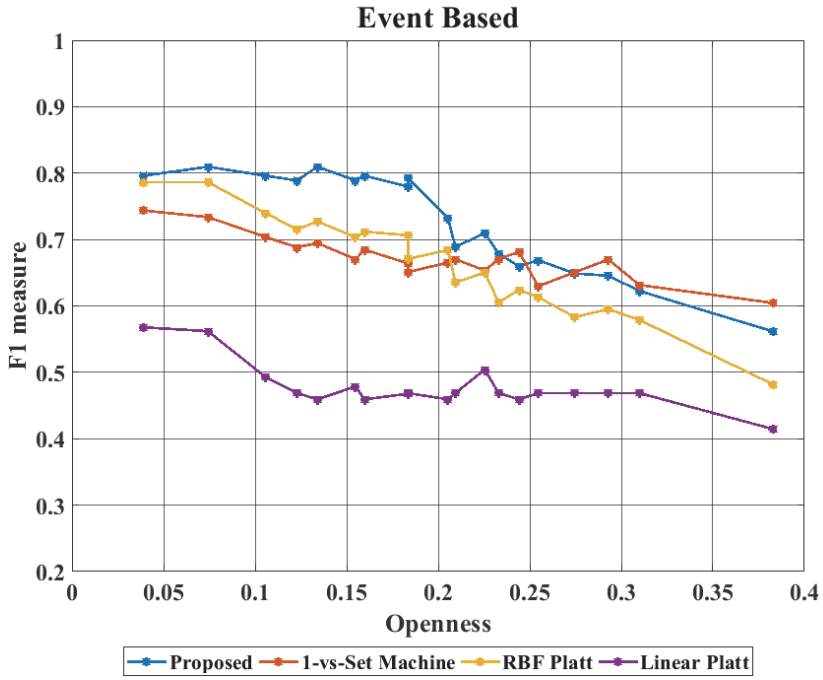


Figure 5.13 F1-measure for multi-class OSR as a function of openness, growing from left to right. Computed with event-based metrics.

Fig. 5.13 shows the performance of the same simulation setups but with event-based metrics. We note that the performance of event-based metrics is again better than the performance of frame-based metrics for all classifiers. And again, our proposed method with peak side ratio (PSR) of calibrated posterior probabilities has either the best performance or near the best performance over a wide range of openness values.

5.6. Summary

In general, this chapter investigated the use of a supervised classification strategy for sound event detection and identification in an open set scenario. Extensive experiments were conducted using sound features proposed in the literature for closed set audio identification. For challenging open set scenarios, experiments using SVM classifiers were performed for recognizing known versus unknown audio events, using a threshold and a rejection function. The rejection function proposed in this work for audio classification is a confidence measurement called the peak side ratio (PSR). It computes the distribution of posterior probabilities for all classifier outputs, to determine whether a particular measured event belongs to a certain group of known events or not. The experiments in the chapter were performed on data from the DCASE challenge. Compared to previous work, our proposed method delivered the best or nearly the best performance over a wide range of openness values. However, for very large openness values, our proposed method was outperformed by the 1-vs-Set Machine method.

Chapter 6: Multi-class Incremental Learning

6.1. Introduction

In the previous chapter, the proposed approach detected the novel classes, but it did not update the system with the new finding. This chapter presents a recognition system that can handle novel classes of data. The system that incrementally includes known and unknown classes is called open set classification or open-world recognition [43]. Some previous research has been conducted to treat unknown samples in an incremental scenario. Polikar et al. [101] introduced an approach that can accommodate an increasing number of classes but includes repetitive training data for all classes. Tax and Laskov [102] have used support vector data description classifier to perform incremental learning, which is a one-class classification method.

In real-world audio recognition, the sounds that are not recognized are often denoted as outliers. The incremental process can be decomposed into three steps as in Figure 6.1. The first step is recognizing samples from known classes and rejecting the novel ones. The rejected sample will be saved in a buffer. The second step is labeling saved samples in the buffer into new categories. The last step is then updating the classifier with augmented categories and samples. Step 1 has already been discussed in Chapter 5. Step 2 should be ideally automated, but herein we presume supervised learning seeking help from human beings. Some sophisticated processes are using active learning, e.g. [103] [104], but it is beyond the scope of this thesis. The contribution of this chapter is for Step 3.

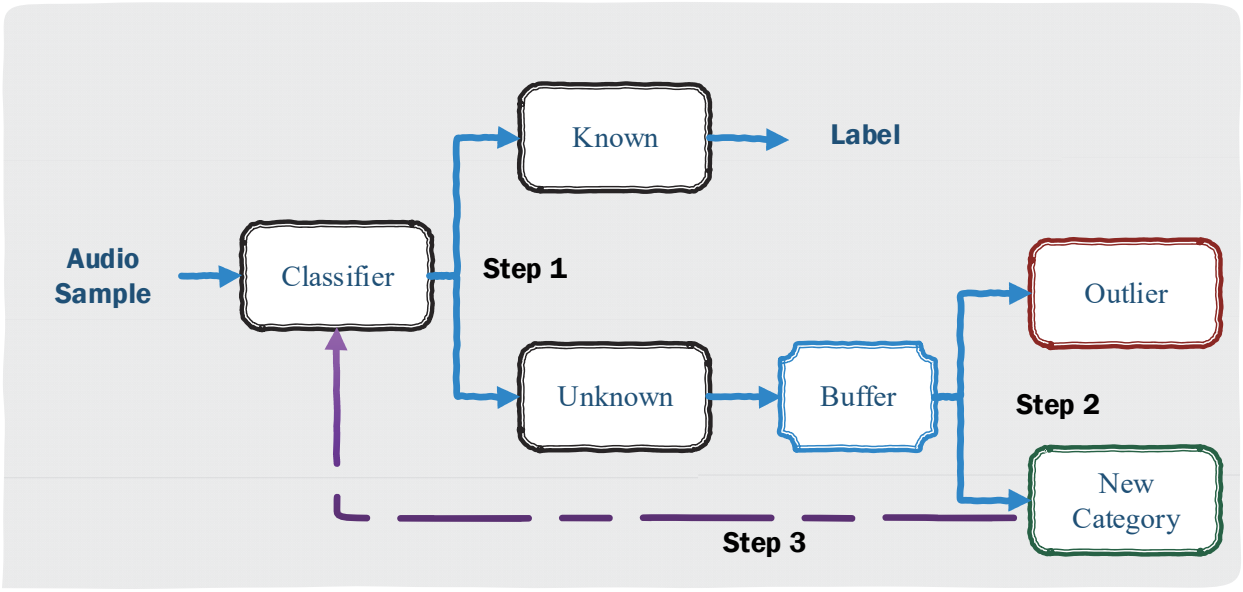


Figure 6.1 OSR with incremental learning.

In this chapter, we propose two methods for multi-class incremental learning (MCIL). The first method utilizes the OSR of the PSR-SVM classifier, which detects the unknown classes. The method does not exclude the unknown data. When the users label a group of data with a specific class, it feeds back to the classifier. A new model is trained by these data. The model is built according to the SVM procedure, where the new data are considered positive, and the rest of all training classes are negative. The classifier determines hyperplane boundaries for the new class. The second method differs from the first one on modeling matters. This method does not use all the training classes to allocate the hyperplane boundaries of the new class. It selects the closest classes boundaries to define the new class hyperplane with help from the k-nearest neighbors algorithm (k-NN).

6.2. Machine Learning Classifier

Audio classification aims to discriminate between features representing different groups of interest. This classification involves learning sounds with a pre-determined number of labels and tries to predict similar sounds based on learned knowledge.

6.2.1. Support Vector Machine

Support vector machine (SVM) methods have achieved great success in classification applications. The SVM is considered the state of the art for kernel-based classification in incremental learning [105]. In general, the SVM determines the optimal hyperplanes that divide two or more classes, given training data pairs (x_i, y_i) for $i = 1,...,N$, with observations $x_i \in \mathbb{R}^d$ and their associated labels $y_i \in \{1,...,Nk\}$ from Nk-category classification problems.

6.2.1.1. Binary SVM

For binary recognition, the output labels are $y_i = \pm 1$. An observation x is classified into certain classes according to,

$$y = \mathrm{sgn}\left[f(x)\right] \tag{6.1}$$

where $\mathrm{sgn}(.)$ is the signum function, and the decision function $f(.)$ of SVM kernel classifier is,

$$f(x) = \sum_j \alpha_j y_j K\left(x_j, x\right) + b \tag{6.2}$$

Where b is the offset, and α is the Lagrangian multiplier. According to the Karush–Kuhn–Tucker (KKT) optimality condition [106], the non-zero multipliers $\alpha \neq 0$ are called support vectors (SVs)

whose examples lie closest to the optimal hyperplane. $K(.)$ is kernel function that transforms the input non-linearly into a high dimensional feature space. The commonly used kernels are linear, polynomial, quadratic, and radial basis functions [107]. We use the Radial Basis Function (RBF) kernel $K(x_i, x_j) = \exp(-\gamma \left\| x_i - x_j \right\|^2)$, where $\gamma > 0$.

6.2.1.2. Multi-class SVM

Since SVM is a binary classifier, it is required to use several SVM classifiers operating in parallel to solve multiple binary classification problems. We use the one-vs-all (OVA) structure to build these classifiers. This structure needs the number of classifiers to be the same as the number of classes Nk. As a new sample x arrives, each classifier gives a continuous classification score $f_j(x) \in R$ used to perform classification decisions according to a winner-takes-all strategy,

$$\hat{y} = \underset{j=1,\ldots,Nk}{\operatorname{argmax}} f_j(x) \tag{6.3}.$$

In the open-set scenario, the classification decision requires the system to not only assign a certain class but also predict whether it is a class type.

6.2.2. k-Nearest Neighbors

The idea behind the nearest neighbors method is to find a predetermined number of training samples that are closest to the new point. The simplicity of the k-nearest neighbors (abbreviated as k-NN) problem is stated as follows: given a query point $qu \in \mathbb{R}^d$, a set of reference points Pn, and a distance $Dist : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, calculate the closest point Pn^* to the query point. The problem can be stated more formally as follows

$$Pn^{*} = \arg\min \mathrm{D}ist\left(Pn_{i}, qu\right) \tag{6.4}$$

Where $Dist$ is a distance function. The distance between two data points is decided by a similarity measure. As k-NN does not require the off-line training stage, the idea behind the nearest neighbors method is to find a predetermined number of training samples that are closest to the new point.

Algorithm 6.1 The k-NN Classifier

Req: Training feature vectors and their labels $\left(X, Y\right) = \left\langle x^{(i)}, y^{(i)} \right\rangle, i = 1,.., N$
Req: Distance $d : X \times X \to \mathbb{R}$
Req: Parameter k
Req: Query instance q
Ensure: Predicted class label $\hat{y}$
Find the k closest feature vectors $X_k = \{x_{i1}, ..., x_{ik}\}$
Count the number of feature vectors in X_k, k_j that support the class label y_j
Predict $\hat{y} = y_{j^*}$ where $j^{*} = \arg\max_j \{k_j\}$

The method depends on the choice of the value k and a suitable distance function. The k-points determine how many data points are selected as neighbors. The algorithm predicts a class label $\hat{y}$ for a given query input qu. The k-NN classifier is summarized in Algorithm 6.1.

6.3. Methodology

This section describes our methodologies that are applied for multi-class incremental learning. We propose two methods for the MCIL. The first method (IOmSVM) utilizes open-set recognition, which detects the unknown classes and recognizes the familiar ones. The unknown data are saved in the buffer in order to be identified. When the users label a group of data with a new class, it feeds back to the classifier for incrementation.

Algorithm 6.2 IOmSVM

Req: $D = \left\langle x^{(i)}, y^{(i)} \right\rangle, i = 1,..,N$ **samples from different classes**

$C_{all} \leftarrow \{C_1, C_2, ...\}$, **set of all classes**

$C_{current}^{train} \leftarrow$ **(randomly) pick** N_k **classes from** C_{all}

$D_{current}^{train} \leftarrow \left\{ \left\langle x^{(i)}, y^{(i)} \right\rangle \mid y^{(i)} \in C_{current}^{train} \right\}$ **, samples from classes**

Training SVMs on $D_{current}^{train} \rightarrow \quad Models = (M_1, ..., M_{N_k})$

Loop :

$C_{current}^{unknown} \leftarrow$ **(randomly) pick classes from** $\left\{ C_{all} - C_{current}^{train} \right\}$

$D_{current}^{test} \leftarrow D_{current}^{train} \bigcup \left\{ \left\langle x^{(i)}, y^{(i)} \right\rangle \mid y^{(i)} \in C_{current}^{unknown} \right\}$

$Novel \leftarrow$ **outlier detected via PSR-SVM tested on** $D_{current}^{test}$

$\Gamma user \rightarrow$ **label new class from** $(Novel) \Rightarrow C_{\text{dominant}}$

$M_{new} \leftarrow SVM\left[\left\{ C_{current}^{train} = negative \right\} \left\{ C_{\text{dominant}} = positive \right\} \right]$

$C_{current}^{train} \leftarrow C_{current}^{train} \bigcup C_{\text{dominant}}$ **, increase "known" classes**

$D_{current}^{train} \leftarrow D_{current}^{train} \bigcup \left\{ \left\langle x^{(i)}, y^{(i)} \right\rangle \mid y^{(i)} \in C_{\text{dominant}} \right\}$

$Models \leftarrow Models \bigcup M_{new}$ **, update the SVM model**

End loop

The system optimizes the margin of the distance from previous hyperplanes and builds a new model. The pseudo-code for this method is shown in Algorithm 6.1. and the full source code is available on GitHub [108].

Algorithm 6.3 IOmSVM + k-NN

Req: $D = \left\langle x^{(i)}, y^{(i)} \right\rangle, i = 1,..,N$ **samples from different classes**

$C_{all} \leftarrow \{C_1, C_2, ...\}$, **set of all classes**

$C_{current}^{train} \leftarrow$ **(randomly) pick** N_k **classes from** C_{all}

$D_{current}^{train} \leftarrow \left\{ \left\langle x^{(i)}, y^{(i)} \right\rangle \mid y^{(i)} \in C_{current}^{train} \right\}$ **, samples from classes**

Training SVMs on $D_{current}^{train} \rightarrow \quad Models = (M_1, ..., M_{N_k})$

Loop :

Neighbor classes: $K_{all}^{nearest} \leftarrow kNN\left\{C_{current}^{train}\right\}$

$C_{current}^{unknown} \leftarrow$ **(randomly) pick classes from** $\left\{C_{all} - C_{current}^{train}\right\}$

$D_{current}^{test} \leftarrow D_{current}^{train} \bigcup \left\{ \left\langle x^{(i)}, y^{(i)} \right\rangle \mid y^{(i)} \in C_{current}^{unknown} \right\}$

$Novel \leftarrow$ **outlier detected via PSR-SVM tested on** $D_{current}^{test}$

$\Gamma user \rightarrow$ **label new class from** $(Novel) \Rightarrow C_{\text{dominant}}$

$C_{dk}^{nearest} \leftarrow kNN\left\{K_{all}^{nearest} \Leftrightarrow C_{\text{dominant}}\right\}$, dk **=k neighbors**

$M_{new} \leftarrow SVM\left[\left\{C_{dk}^{nearest} = \text{negative}\right\}\left\{C_{\text{dominant}} = \text{positive}\right\}\right]$

$C_{current}^{train} \leftarrow C_{current}^{train} \bigcup C_{\text{dominant}}$ **, increase "known" classes**

$D_{current}^{train} \leftarrow D_{current}^{train} \bigcup \left\{ \left\langle x^{(i)}, y^{(i)} \right\rangle \mid y^{(i)} \in C_{\text{dominant}} \right\}$

$Models \leftarrow Models \bigcup M_{new}$ **, update the SVM model**

End loop

The second method (IOmSVM +k-NN) does not include all the known classes features to build the new model. Instead, it determines the closest known classes according to a k-NN algorithm. The method computes the hyperplane from its tangent points. The pseudo-code for this is described in the box Algorithm 2, and the full source code is also available on GitHub [108]

6.3.1. Preprocessing

The input audio signals are preprocessed with normalization, pre-emphasis, segmentation, and silence removal. We use a database that contains audio events recorded in an office-like environment retrieved from the DCASE 2013, 2016 challenges [94] and the Freesound online database [95]. As stated on their websites, the datasets contain various sounds over different background noises at different levels (high, medium, and low). All the recordings used have a WAV format (i.e., uniform 16 bits quantization) to avoid transformation/coding artifacts (e.g., from MP3 or AAC encoding).

During the preprocessing, we resample all files at a fixed sampling frequency, 44100 samples/sec, and store them in the library with the mono format, and each recording is normalized to maximum unit amplitude, to bring the gain of the entire track to its maximum without clipping. We implemented a VAD algorithm to remove silence, where each recording is divided into small frames of 20 ms with 50% overlapping. Let ES_{pq} be the actual total signal energy and En_{pq} be some estimated background noise energy, where p and q denote the frame index and the frequency bin, respectively. The estimated noise energy is updated when the actual signal energy is low, as described in Chapter.4. The VAD algorithm is computed by comparing the local SNR of the p^{th} frame to a global SNR. The global SNR is computed by the average of the local SNRs

for the current recording. If the SNR of a certain frame is less than the global SNR, this frame is measured as silence. Table I shows the number of recorded frames used for each class.

Table 6.1 Audio dataset: number of frames per class

	DCASE2016	DCASE2013	FreeSound	Total
alert	-	1162	420	1582
clearthroat	437	650	74	1161
cough	596	670	284	1550
doorslam	277	1257	-	1534
drawer	612	951	-	1563
keyboard	1094	2178	714	3986
keys	-	1163	-	1163
knock	437	743	530	1710
laughter	949	866	-	1815
mouse	-	840	286	1126
pageturn	635	1823	-	2458
pendrop	-	472	-	472
phone	1119	5307	950	7376
printer	-	12099	810	12909
speech	1107	1709	-	2816
switch	-	285	110	395

After silence removal, the audio clips are segmented to partially overlapped frames, with frames of 2048 samples and 512 samples overlap. Each frame is windowed to smooth out discontinuities using a Hamming window: $w(n) = 0.54 - 0.46\cos\left(\pi n / N\right) \quad 0 \le n \le N-1$.

6.3.2. Features Extractions

Each audio signal is composed of different frequencies and different energy amplitudes, with quick variations within a short time. The frame $x_p(n)$ has 2048 samples with 512 samples overlap. The frame is first pre-emphasized with a high-pass filter $H(z) = 1 - \alpha z^{-1}$ to compensate for a spectral slope, where $\alpha = 0.97$. The segments are multiplied with a window to reduce the variance for spectrum estimation. We compute Mel frequency cepstral coefficients and Gammatone cepstral coefficients. In addition, the spectral flux and spectral centroid features are also computed.

6.3.3. Multi-class Incremental Learning

The framework is initialized by training the algorithm with examples limited to be from Nk classes. The training samples contain feature vectors $x_i \in \mathbb{R}^d$ with corresponding labels $y_i \in Y = \{1,...,Nk\}$, where d is the features' dimension. The one-versus-all multi-class SVM classification is used for this recognition, where Nk classifiers are conducted to build the models. Each classifier is trained using samples from a certain class. The samples from that class are considered positive and samples from all the other $Nk-1$ classes are considered as the negative class. The models of trained classes $\{M_1, M_2, ..., M_{Nk}\}$ are stored to be used in different stages. In the testing stage, for a signal sample the Platt probabilities estimate [96] are computed based on the decision function using a sigmoid:

$$P(Y_i = 1 / f(x_i)) = \frac{1}{1 + \exp(Af(x_i) + B)} \tag{6.5}$$

where the A and B parameters are determined by Maximum Likelihood Estimation (MLE). As the new sample $\hat{x}$ arrives in the system, the SVM classifiers compute the posterior probability, $P(Y_j \mid \hat{x})$, but the probability estimation of unknown classes $P(Y_{unknown} \mid \hat{x})$ is not possible. Therefore, our method uses a confidence measurement called peak-side-ratio (PSR) introduced in Chapter 5. It helps to determine if the new sound belongs to one of the predefined classes or not. The posterior probability values P_j are arranged in descending order, where P_1 is the largest value and P_{Nk} the smallest value. The PSR is then:

$$PSR = \left| P_1 - \frac{P_2 - P_{Nk}}{\sqrt{\frac{\sum_{c=2}^{Nk}(P_c - \bar{P})^2}{Nk-1}}} \right| \tag{6.6}$$

If the PSR exceeds a certain threshold or a trigger value, the new sample is labeled as an unknown class. Otherwise, the sample is assigned to the class that has the maximum posterior probability using the following formula:

$$\hat{y} = \begin{cases} \underset{j}{\text{argmax}}\, P(Y_j \mid \hat{x}) & \text{if PSR} \le \delta \\ \text{Unknown} & \text{if PSR} > \delta \end{cases} \quad \forall j \in (1, ..., N_k) \tag{6.7}$$

where δ is a threshold determined during the validation process. The threshold is determined over different possible threshold values in the range $\delta \in [0,4]$, with a resolution of $\Delta\delta = 0.01$. Using five-fold stratified cross-validation, the F1-measure values are computed as an ensemble mean. The threshold value $\delta = 2.1$ produces better performance as shown in Figure 6.2. This threshold is considered constant throughout all experiments.

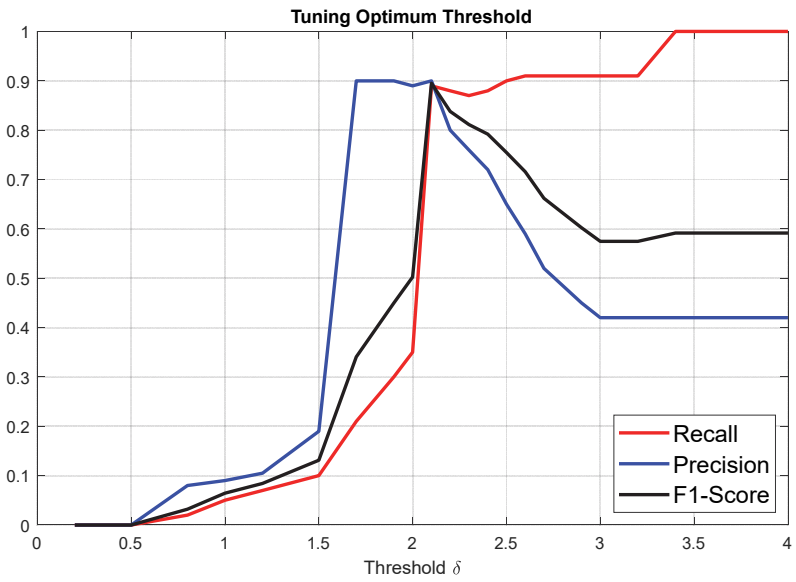


Figure 6.2 Optimizing threshold through the validation process

The decision function for classifying the Nk known classes is set based on the predefined threshold. When the system receives a sample for recognition, it will be compared to the stored models to determine the posterior probabilities. The tested sample is assigned to the most similar class, or it is declared as being originated from an unknown class. The unknown class samples are stored to produce a new M_{Nk+1} model and an updated set of labels $y_j \in Y = \{1,...,Nk, Nk+1\}$, where $j = Nk+1$ is a new class.

6.4. Experiments & Evaluations

Scheirer et al. proposed an evaluation protocol for open-set recognition, where training uses known classes, and testing uses both known and unknown classes. Ristin et al.[4] provided a scenario in which new item categories are added incrementally to a closed set learning scenario. We incorporated principles from both open-set recognition and incremental learning approaches to create a protocol for open-set recognition, in which new categories are regularly introduced to the

system while it is also tested with unknown categories. We approach the performance evaluation with metrics that compute similarities after aligning the recognition outputs with a reference ground truth. Two fundamental assumptions have been used in the DCASE/ AASP challenges [66],[94] to evaluate how individual audio sounds are classified:

1. Segment-based evaluation: the system output and ground truth are compared for each segment length.
2. Event-based evaluation: the system output is considered the same within all ranges (duration) of the event. This means that event labels in the recognition output will be compared to the ground truth event.

Our experiments are conducted for several evaluations in which we compare the performance of our proposed algorithm to the performance of representative previous algorithms: W-SVM [30], OSNN [110], and OSmIL [111]. The parameters of all previous algorithms were set according to the corresponding paper.

6.4.1. Experimental Settings

We start our initial training with four classes. These classes are considered known classes to the system. We train the classifiers with 70% of the known classes samples, and 30% of these samples will be used in the test stage. We randomly choose two classes to be unknown. The number of unknown classes is fixed for all the experiments. We use our OSR algorithm to reject unknown classes. The SVM with peak-side ratio determines the novel class. For known classes, the final

predicted class is the one that has the maximum vote, where one-vs-rest fusion is used to merge the results of multiple classifiers.

Our incremental learning simulation setup is as follows. The test is run by adding classes for multiple iterations and it records the changes in performance. We increment one of the unknown classes in each iteration and select another class to be unknown, which means there are always two unknown classes, and the number of known classes grows by one, ranging from 4 to 11. The new model for the incremented class is built at every iteration, as described earlier.

6.4.2. Audio Recognition and Rejection

This section evaluates the performance of our proposed algorithms in a closed-set scenario and an open-set scenario but without incremental updates. We evaluate the performance of the system for both segment-based and event-based metrics, using standard evaluation setups provided in [30]. The systems are evaluated for the following three types of errors:

a) Misclassification: Test samples misclassified with a wrong label, belonging to one of the predefined classes.
b) False unknown: Test samples rejected as unknown, but belonging to one of the predefined classes.
c) False known: Test samples truly unknown but assigned to one of the predefined classes.

This task examines the classifier's accuracy in closed-set recognition, where only the first type of error is possible (misclassification). This task is important because it explains how well an algorithm learns the training data. Since the purpose of the closed set recognition experiment is to validate the ability of our proposed algorithm to discriminate among known classes, we did not conduct comparisons with other algorithms in this part.

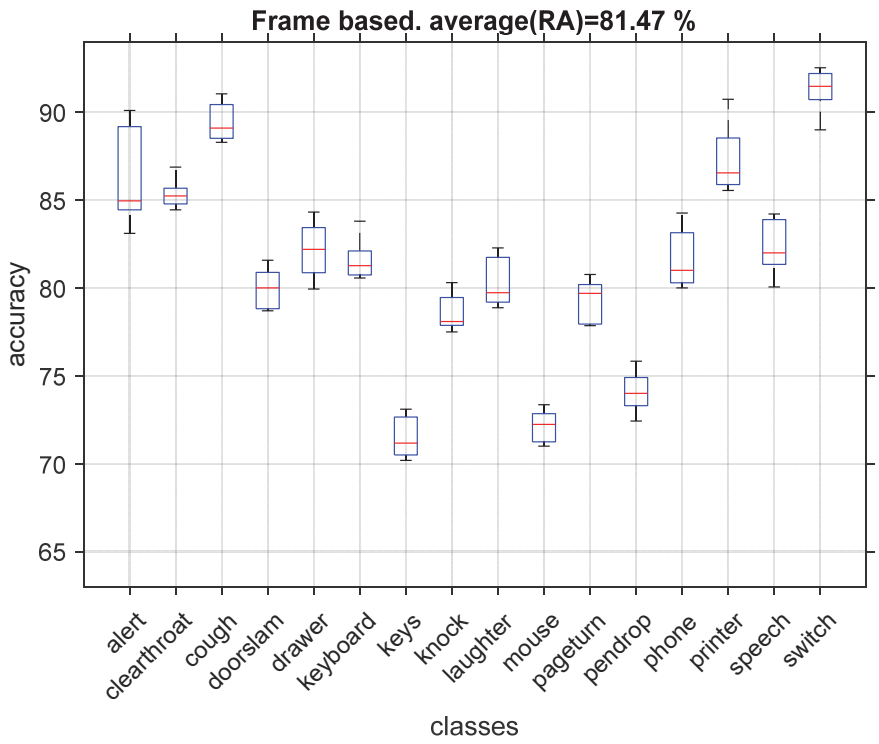


Figure 6.3 Box-plot distributions of recognition accuracy of closed-set recognition for frame-based metrics

To have robust evaluation metrics, the experiments were performed in a stratified five-fold cross-validation setup. Figure 6.3 and Figure 6.4 use box plots [112] to show the experimental observations. The red middle bar in the boxes shows the median, and the interior of this box indicates the upper and lower quartiles. Points outside of this range are possible outliers. It is noticeable that the system has good classification accuracy even though some classes are not easily distinguishable due to the strong correlation among them. This can be noticed in closed set recognition. The experiments were repeated according to a stratified five-fold cross-validation process. The horizontal red line inside the box is the median value. The upper and lower ends of the box show the upper and lower quartiles.

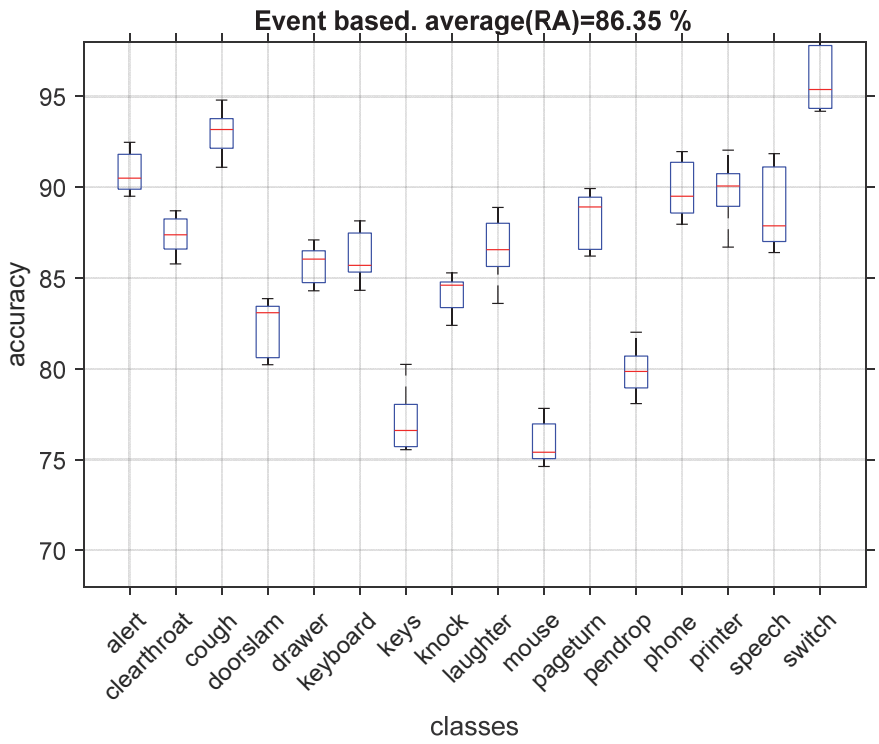


Figure 6.4 Box-plot distributions of recognition accuracy of closed-set recognition for event-based metrics

It is noticeable that the system has good classification accuracy even though some classes are not easily distinguishable due to the strong correlation among them. This can be noticed in both frame-based and event-based metrics. Among the 16 classes, the 'keys' and 'mouse' sounds were the most difficult classes to identify. The average event-based accuracy was 86.35%, while the average frame-based accuracy was 81.47%, which shows that the event-based performance is better than the frame-based performance (as expected).

6.4.3. Open-set Recognition

The experiments in this section were performed to recognize audio sounds where the testing set may not include the same categories as in the training set. These experiments measure the capability to

discriminate known classes from unknown classes and to discriminate known classes from one another. We use the same measure in the previous chapter, which is,

$$Openness = 1 - \sqrt{\frac{2 \times |Y_t|}{|Y_k| + |Y_u|}} \tag{6.8}$$

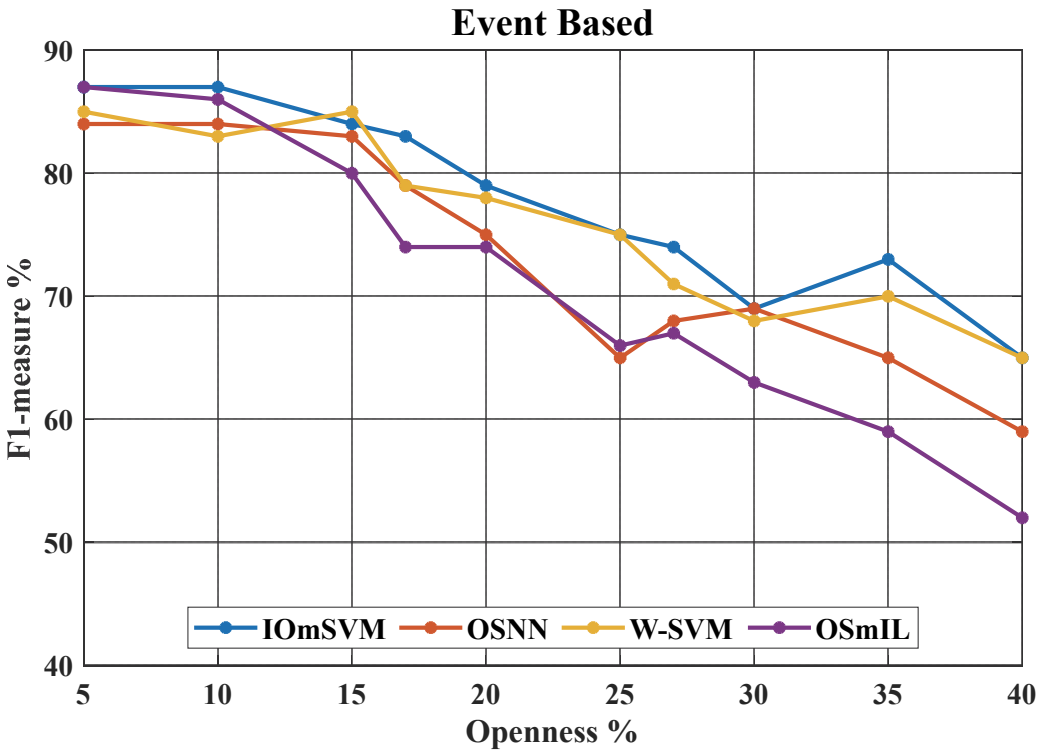


Figure 6.5 F1-measure for open-set recognition as a function of openness. Results computed for event-based metrics.

Where the subscripts t, k and u are for the target, known and unknown label sets. The testing dataset is a combination of all classes $Y_{test} = Y_t \cup Y_k \cup Y_u$, while the training dataset is the combination $Y_{train} = Y_t \cup Y_k$ and the unknown classes are its complement $Y_u = \{y \mid y \in Y_{test} \text{ and } y \notin Y_{train}\}$. If we set $Y_t = Y_k$, this yields $Y_{train} = Y_k$. We used varying degrees of openness and followed k-fold cross-validation to obtain robust evaluation metrics. The experiments were performed by selecting six

classes for training. The remaining ten classes were used as unknown data. To generate different amounts of openness, each trial selected from 1 to 10 classes of the available unknown classes, and from 2 to 6 classes to be considered as target classes. All the algorithms were executed using the same data, with the same negative and positive examples to sustain a fair comparison.

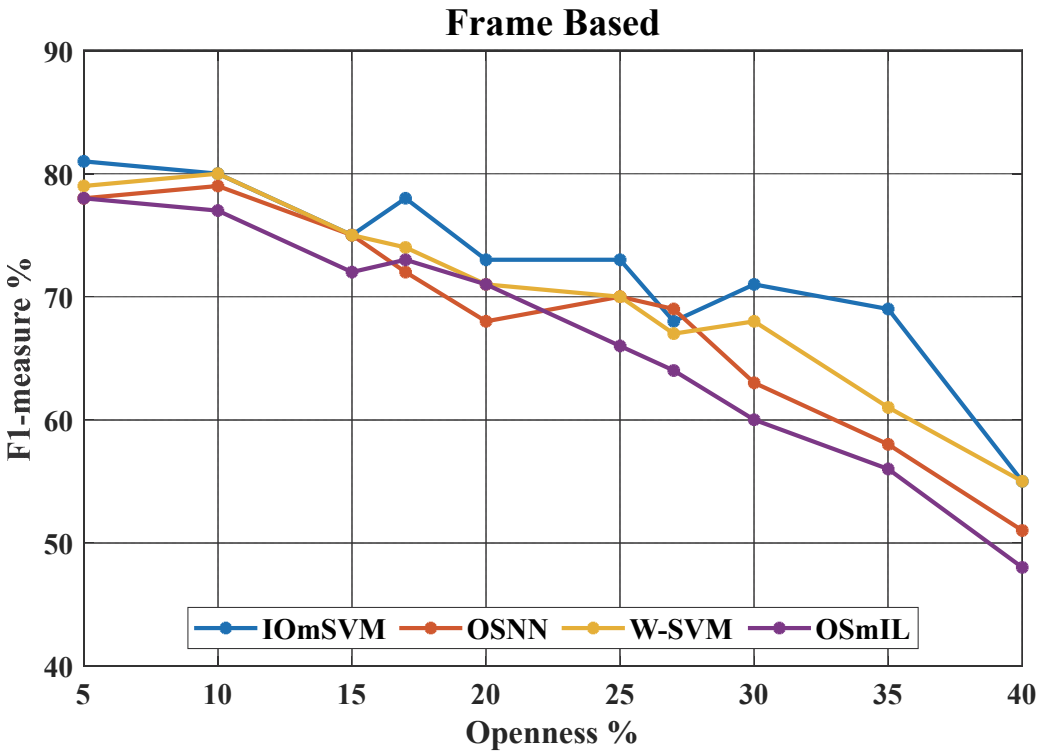


Figure 6.6 F1-measure for open-set recognition. Results computed for frame-based metrics.

The experimental results are shown in Figure 6.5 and Figure 6.6. It can be observed that our proposed method provides either the best performance or near-best performance over a wide range of openness values for the task of separating known classes from unknown classes and for distinguishing among the known classes. We note that the performance of event-based metrics is again better than the performance of frame-based metrics.

We then used 10% and 35% openness as examples to show additional metrics for the considered methods, in Figure 6.7. The results were computed based on micro-average metrics. As described in chapter 3, a macro-average treats all classes equally since it computes each class independently and then takes the average of them. A micro-average combines the contribution of all classes to compute their average. Since we have more examples for some classes than others, the micro-average is preferable here. It can be observed again from Figure 6.7 that our proposed algorithm provides overall the best OSR performance for the considered openness values.

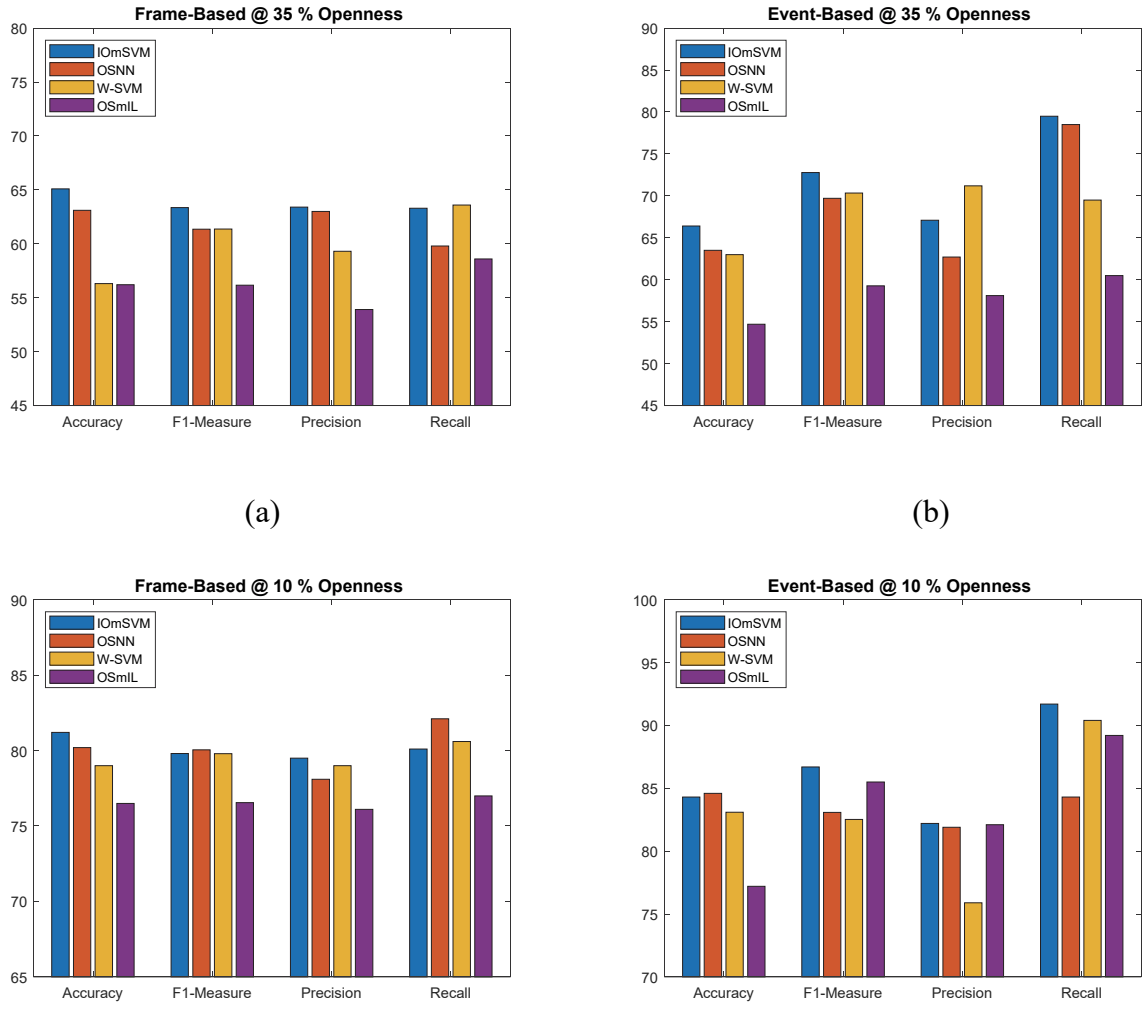


Figure 6.7 Accuracy, precision, recall, and F1-measure for open-set recognition at different openness values.

6.4.4. Incremental Learning

For incremental learning, we train the system on a pre-defined number Nk of initial classes and then incrementally add the additional classes one by one, to evaluate the incremental learning performance. As we start using incremental learning, we use both of our proposed methods (IOmSVM and IOmSVM+ k-NN) for these experiments. In both cases, the training is divided into two parts: an initial learning phase and an incremental learning phase. The initial phase models a

certain number of classes. Once the incremental learning phase starts, new classes are added to the system one by one. The system is evaluated for the incremental learning performance in both closed set and open set scenarios, where the data are split into two sets, the known set and the unknown set, to simulate closed set and open set performance. These procedures are repeated as the classes are added to the system continuously.

6.4.4.1. Closed-set Incremental Learning Performance

We conduct this experiment to demonstrate the incremental learning performance in a closed set scenario. The closed set scenario is chosen for this experiment because misclassification may occur in the calibration process of open set rejection. The results in Figure 6.8 and Figure 6.9 show that the performance of the IOmSVM+ k-NN algorithm provides the best performance overall. Again, it can be noticed that the performance of event-based metrics is better than the performance of frame-based metrics.

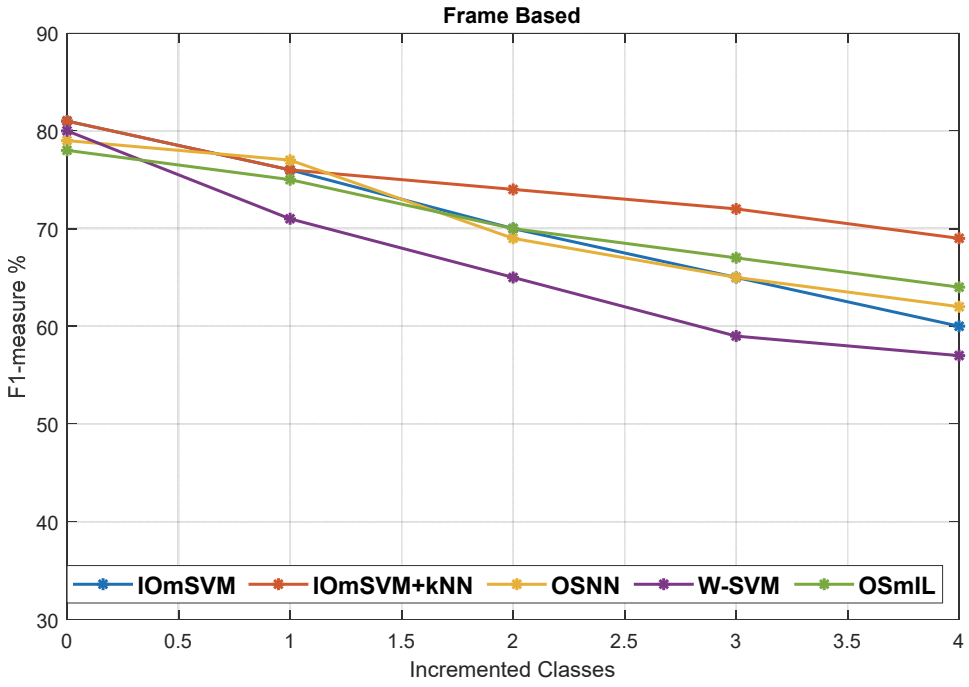


Figure 6.8 F1-measure for the incremental learning process in the closed set scenario, with frame-based metrics.

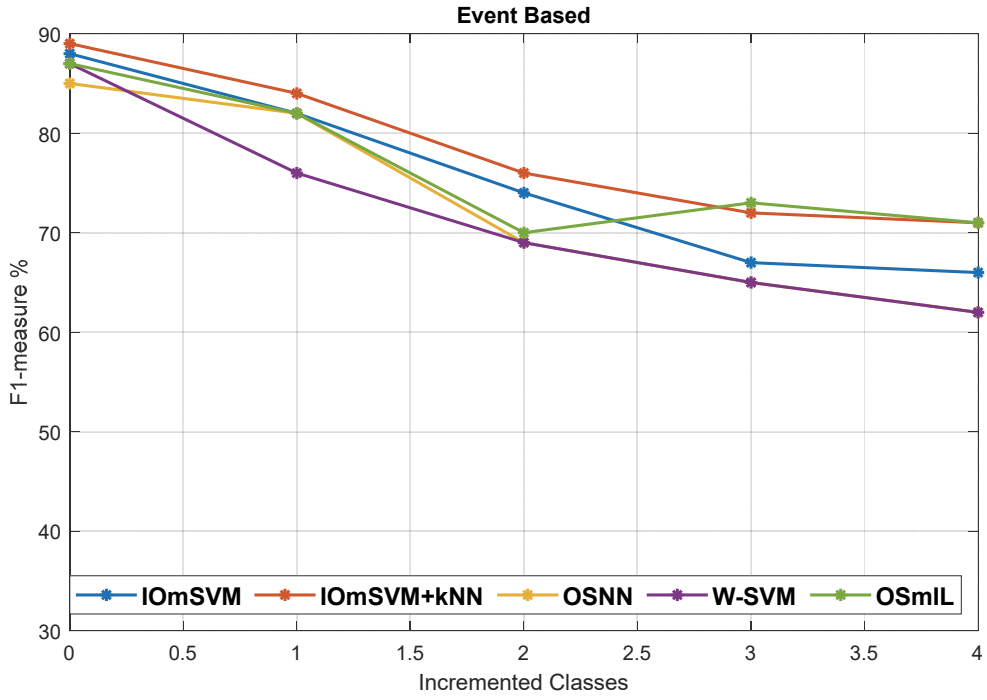


Figure 6.9 F1-measure for the incremental learning process in the closed set scenario, with event-based metrics.

6.4.4.2. Multiclass Open-set Incremental Learning Performance

This part of incremental learning consists of two processes: open set recognition and incremental learning. It is a more realistic incremental scenario. In Figure 6.10, the axis on the right side illustrates the incremented classes. Considering the F1-measure as the evaluation metric, the more incremented classes are added, the poorer the performance of the classifier is expected to become. The left side axis depicts the different degrees of openness considered in the simulations. When the openness equals zero, it becomes a closed set incrementing as in the previous sub-section. For zero increment, it becomes open set recognition without incrementing. It is clear from Figure 6.10 that our proposed methods (especially IOmSVM+ k-NN) can maintain a very competitive (typically better) performance in terms of recognizing previously trained classes and identifying new unknown classes, for different openness values and different numbers of incremented classes.

We can see that as we incrementally add more classes, the performance gradually drops as the system needs to model an SVM classifier to each new class, which means optimizing an extra hyperplane and its margin among the existing hyperplanes. The goal of the experiments is to run each iteration until the performance drops significantly.

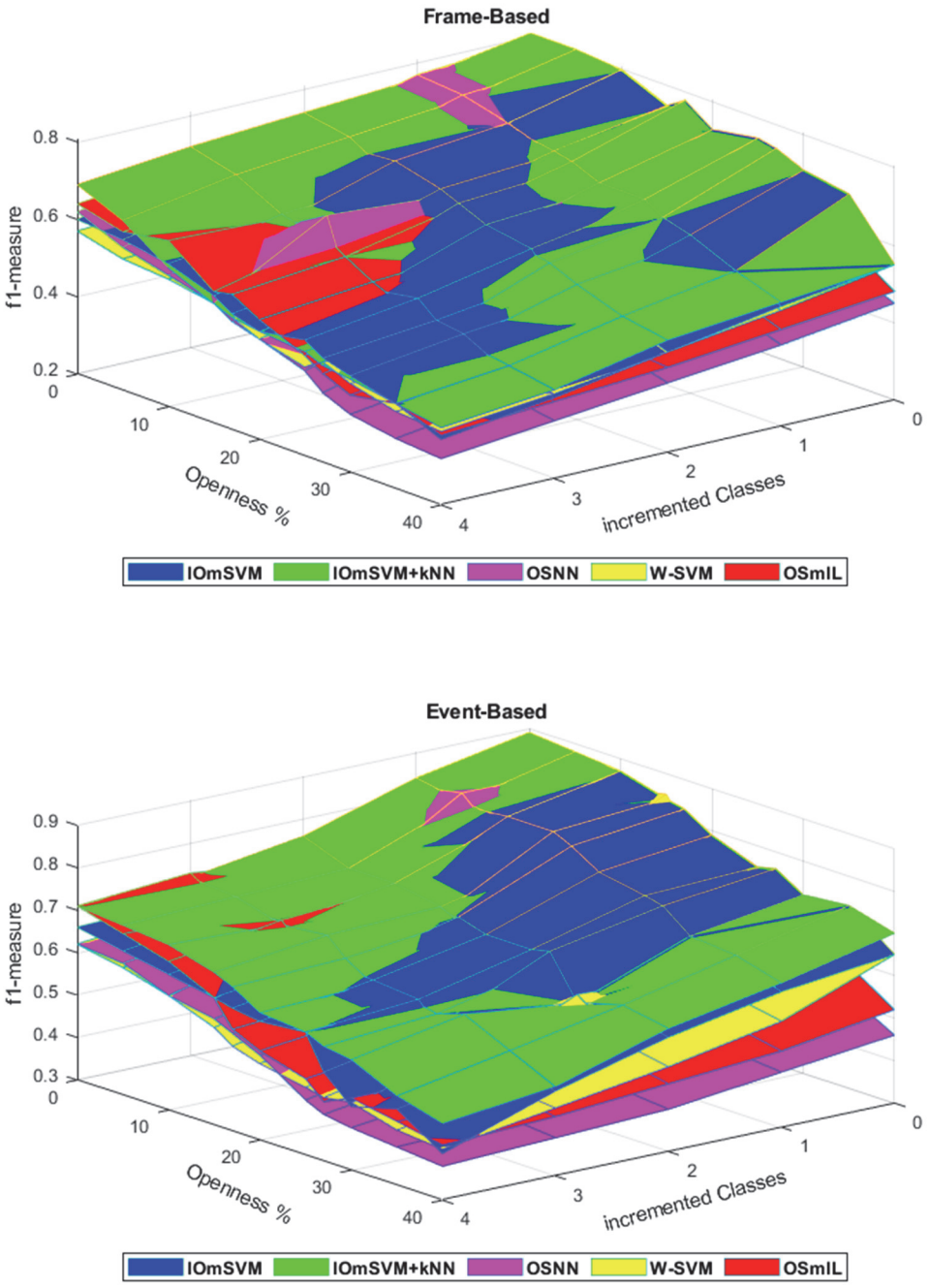


Figure 6.10 Multiclass open-set recognition with incremental learning.

6.5. Supplementary Experiments

6.5.1. Analysis of Adding Classes at One Time

This is an experience that shows the severe challenges for the algorithm. We set up a situation where classes were added for incrementing. At the same time, we put a few samples from other classes to be rejected. In the first step, we have chosen randomly 6 classes to be trained to model the system. In each iteration, samples of two new classes are added to test data to form open-set recognition. At the same time, samples from one class with their labels are incremented to the system to form incrementing learning. Figure 6.11 shows the result of this experiment. We can notice that when the number of added classes increases, the system performance gradually drops. It is noticeable again that the second algorithm (IOmSVM+k-NN) can handle more new classes than the first algorithm (IOmSVM). A likely explanation for this is that the classifiers are struggling to build a good new model for the newly added class. This performance computes the error of incremental learning loss in addition to misclassification error and the error that comes when the system does not detect/reject new classes.

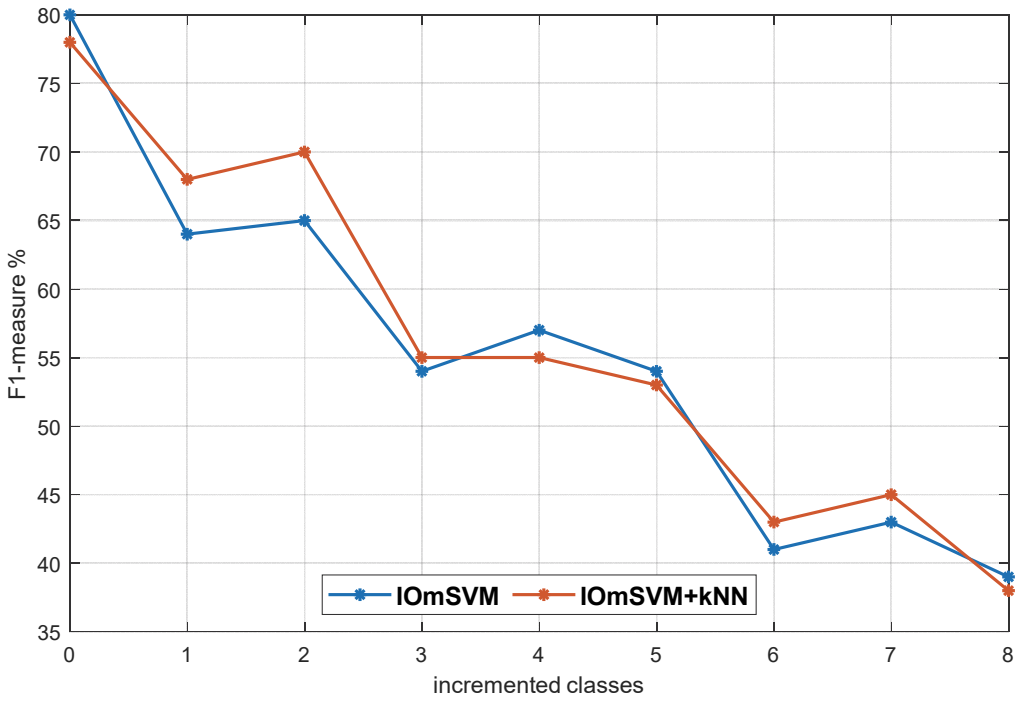


Figure 6.11 Open set recognition and incremental learning

6.5.2. Optimizing k-NN Parameters

We designed the k-NN classifier to evaluate the number k-value. We set k from 1 to 18 for comparison. In addition, five-fold cross-validation is used to divide each dataset into two sub-datasets: 80 % of the dataset is for the training and the rest 20 % is for the testing.

For distance functions, a variety of distance functions have been introduced; for example, Prasath et al. [113] summarized more than a hundred distance functions and organized them by their concepts. They studied the effect of distance functions on k-NN performance. The study concluded that there is no optimal distance metric that can be used for all types of datasets. However, the family of Euclidean distance formulas or Minkowski distances is more common in recognition algorithms, and it has given good efficiency and productivity [114], [115]. Minkowski distance is

also the default parameter of the *sci-kit-learn* package in Python [116]. Therefore, we use this distance function in our implementation, where the formula of Minkowski distance is computed as,

$$Dist(A,B)=\left(\sum_{i=1}^{n}\left|Pn_i-qu_i\right|^{\rho}\right)^{1/\rho} \tag{6.9}$$

Where n is the number of variables, and the value ρ determines the shape of the distance. When $\rho=2$, it becomes the Euclidean distance. When $\rho=1$, it becomes the Manhattan distance. Moreover, Chebyshev distance is a variant of Minkowski distance when $\rho=\infty$.

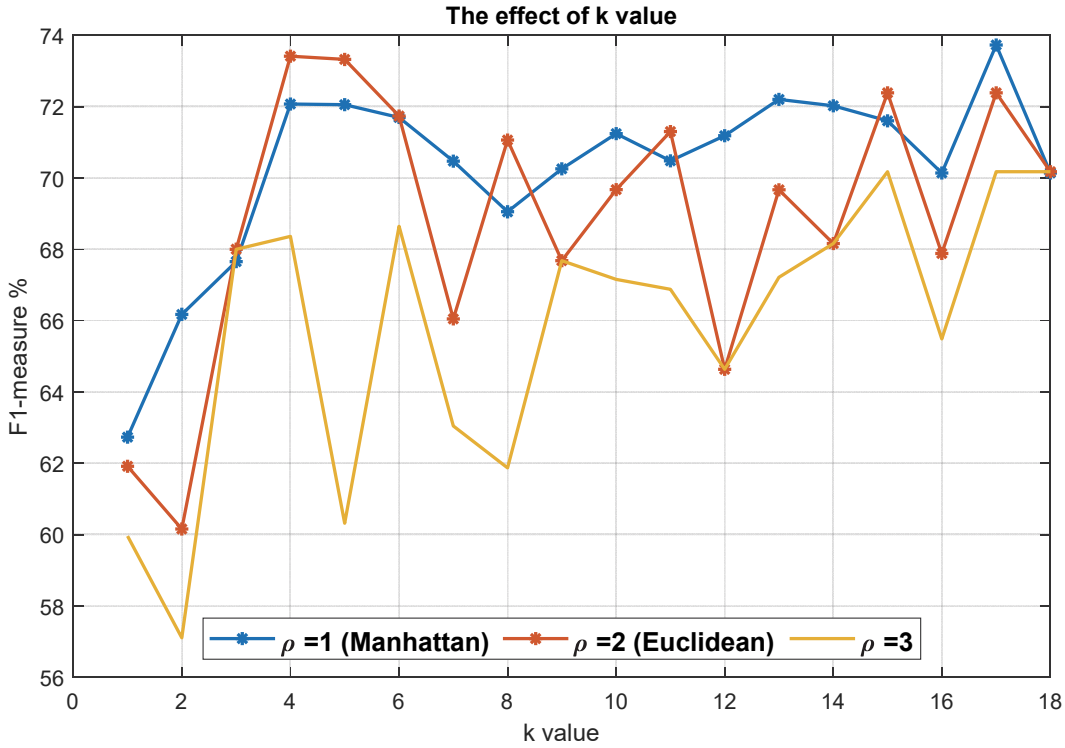


Figure 6.12 The average F1 score of k-NN with different k values for three distance measures.

As can be seen in Figure 6.12, the performance changes every time the k value changes, but without a clear relationship. Therefore, we have chosen the minimum k value that gives a good result, which was $k = 4$. This value has been used as a constant for the other experiments.

6.6. Summary

This chapter studied a novel multi-class incremental learning (MCIL) approach, where unknown audio classes are identified and then the unknown data are labeled so that the classifier incrementally expands on its knowledge. We proposed two algorithms for this process. Both algorithms applied the PSR-SVM classifier to detect the unknown classes. They differ in their incrementing methods. When the user labels the unknown class, it feeds back to the classifier. The first algorithm builds the new model by using all the known classes training samples to generate negative samples, in addition to positive samples obtained from the new class. The second algorithm selects only the closest classes and uses their samples as negative samples. It uses the k-nearest neighbors classifier (k-NN) to determine the distances between classes. An experimental setup was designed by building a complete cycle model training on provided data and then expanding the training classes based on new data discovered for future testing. Incremental learning audio classification experiments compared with representative previous algorithms have shown that the proposed incremental learning methods provide a better performance overall.

Chapter 7: Extreme Value Machine in Open-Set Recognition

7.1. Introduction

Machine learning algorithms in closed set recognition can perform empirical risk minimization very well, using their ability to handle large feature spaces and to identify outliers. However, in practical applications, out-of-set data may be encountered in the testing stage, yet this is classified as one of the known classes. Even if this sample lies far from any of the training samples, it may be classified with a high probability, that is, the algorithms will not only be wrong, but they may also be very confident in their results [109], [117]. As it was explained earlier in this thesis, the challenge of classifying open sets is far more difficult than just questioning the test samples whether from known (inlier) classes or the unknown (outlier) classes. Along with finding the unknown instances, the algorithm must appropriately classify the known class samples. The system in an open-set setting does not know the unknown classes and thus cannot estimate the probability of unknown classes. Instead, we compare the probabilities distribution of recognition output. If the highest probability value is far away from other values, this class is considered as known and is recognized. The probabilities distribution is the key to determining the novel classes. If the recognition scores are not calibrated very well, we cannot apply a threshold to separate the known classes from unknown classes. In chapter 5, we proposed a peak-side-ratio to calibrate the posterior probabilities. In this chapter, we apply the extreme value machine that uses Compact Abating Probability (CAP), which assumes that the probability of known classes decreases as points move from known data to open space. The first contribution of this chapter is to utilize the extreme value machine on open set recognition using a similarity measure to recognize audio sounds, where the decision boundary is made from extrema points identified within the training data. The second contribution is to directly apply the EVM score distribution to supervised

machine learning classifiers to calibrate their output. Therefore, we used the EVM with both "shadow learning" and "deep learning". For shadow learning, we used the SVM classifiers that were introduced earlier in this thesis. For deep learning, we used the Convolutional Neural Networks (CNN), which have been adopted from the image processing field. Experimental results reveal that these methods provide good performance over a wide range of openness values.

7.2. Extreme Value Machine Background

Extreme Value Machine (EVM) has been applied in [50] [118] to normalize the inclusion score for the novelty detection problem. The main assumption of this method is that the problem of distinguishing known and unknown classes is related to the problem of estimating the distribution of the known examples. The distribution will be estimated using Fisher-Tippett Theorem [119]. Additional details on EVT distributions can be found in Appendix B.

7.2.1. Density Function (PSI)

The EVM is based on the concept of margin distributions and uses techniques such as maximizing the mean or the median margin. The model is fit on the distribution of margins corresponding to negative and positive samples. In the same way, the extreme value theorem (EVT) estimates the distribution and normalizes it by the Weibull distribution. The cumulative distribution function (CDF) of the Weibull distribution is defined as:

$$Weibull_f(x) = \begin{cases} 1 - e^{-\left(\frac{x}{\lambda}\right)^{\kappa}} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{7.1}$$

where κ and λ are the Weibull scale and shape parameters, respectively. The probability of inclusion for a sample or point x' is given by,

$$\Phi(x_i, x', \kappa_i, \lambda_i) = \exp\left(-\frac{\|x_i - x'\|}{\lambda_i}\right)^{\kappa_i}, \tag{7.2}$$

where $\|x_i - x'\|$ is the distance between x' and class centroid x_i. The EVT rejection model, $\Phi(x_i, x', \kappa_i, \lambda_i)$, is the probability that a sample does not lie beyond a certain threshold. The EVM builds a margin distribution (MD) that requires the τ closest samples from any negative class. We used the libMR library [120] (provided by the authors of [121]), which returns the optimal τ and the parameters of the Weibull model κ, λ .

7.2.2. Multi-Class Open Set Probability

If Nk is the number of classes and x_i is the set of attributes for the j^{th} element, the element probability of class i is $P_{ij} = P(y = i \mid x_j),\ i \in \{1, 2, ..., Nk\}$. Several methods have been developed to estimate and calibrate the probability output [43],[96],[122]–[126], where the classification decision is based on thresholding the probability calibrations and distributions. Bendale et al [6] and Scheirir et al [30] used a Compact Abating Probability (CAP) Model. This model assumes that the probability of known classes data decreases when the probability values move from known data. Data beyond a thresholded radius are subsequently labeled as the unknown or open space. In [31], a posterior probability is estimated for each training class. A test instance is predicted into a known class only if the maximum probability exceeds a specific threshold. The probability that a sample x' belongs to class Y_i can be computed as:

$$P(Y_i \mid x') = \underset{i}{\operatorname{argmax}}\ \Phi(x_i, x', \kappa_i, \lambda_i) \tag{7.3}$$

Then, the output recognition will be:

$$\hat{y} = \begin{cases} \underset{i}{\operatorname{argmax}}\ P(Y_i \mid x') & \text{if } P(Y_i \mid x') \geq \delta \\ \text{"Unknown"} & \text{otherwise} \end{cases} \quad \forall i \in (1, \ldots, Nk) \tag{7.4}$$

where δ is a threshold that defines the boundary between known and open space. The EVM code was retrieved from [127]. The EVM outputs are obtained by running model reduction. This reduces the size of the model by discarding the minimal impact values.

7.3. Problem Notation and Evaluation Measure

The instances of the training set are given as $X \in \mathbb{R}^{d \times n}$ and their corresponding labels are described as $y \in \{1, \ldots, Nk\}^n$ where Nk is the number of known classes, n is the total number of instances and d is the dimension of each instance. In open set recognition, we learn a model $f : X \to \{1, \ldots, Nk+1\}^n$ such that the model accurately classifies an unseen instance. If Y is the set containing all the class labels in a given data, the training data are $Y_{train} = Y_{known}$ and the testing dataset is a combination of all classes $Y_{test} = Y_{known} \bigcup Y_{unknown}$.

We use different values of openness to evaluate the system performance against the F-measure, which was defined in chapter 3. The True Positive (TP) predictions are computed when the model predicts a positive class with a probability higher than threshold γ. A False Positive (FP) prediction is assigned when the model predicts the wrong class with a probability greater than γ, which includes the case where an unknown sample is classified as any of the known classes. A False

Negative (FN) prediction occurs when the probability of a certain known sample for its correct class is below threshold γ.

7.4. Discussion

In this section, we describe our method for audio open-set recognition. The methodology is composed of four parts, which are elaborated in the following subsections.

7.4.1. Protocol

We use the same database as in previous chapters, which contains audio events recorded in an office-like environment retrieved from the DCASE 2013, 2016 challenges [95] and the Freesound online database [96]. The input audio signals are preprocessed with normalization, pre-emphasis, segmentation. An audio classification task involves front-end processing such as audio analysis and features generation. The first step before feature generation is silence removal, where a VAD algorithm is used to remove silence, as previously described.

The first step is to test the classifier on closed set recognition. This experiment is conducted to evaluate the efficiency of the classifier. The system is evaluated using a 5-fold cross-validation procedure. Since the purpose of the closed set recognition experiment is to validate the ability of our proposed algorithm to discriminate among known classes, we did not conduct comparisons with other algorithms in this part. Comparisons with other algorithms are made for open-set recognition experiments. Like most supervised learning problems, our approach is based on two canonical stages: the training stage where the classifier is modulated using samples of certain classes, and the testing stage where a test sample can belong to one of the trained classes or none of these classes. Therefore, it must be classified as unknown if it belongs to a novel class. The second step

is to conduct experiments on open-set recognition. We measure the performance results of the classifiers by varying the openness from 0% (i.e., closed-set scenario) to a maximum of 54 % open space.

7.4.2. EVM Recognition

The algorithm uses an optimization solution. It can be defined as the minimization of the number of extreme training samples on either side of the boundary and minimizing a loss function related to the probability estimation. The EVM training function iterates through every class to fit a model vector Φ and then it iteratively optimizes the training models within a certain coverage threshold ς. The EVM optimization objective is to select the smallest number of extreme vectors using cover model reduction, as described in Appendix B. The algorithm code was retrieved from [44]. The training function is described in Algorithm 7.1.

Algorithm 7.1 Pseudocode for EVM training

Required: $D = \left\langle x^{(i)}, y^{(i)} \right\rangle, i = 1,..,N$ **samples from audio classes**

Required: (τ, ς) tail size, and coverage threshold

$C_{training} \leftarrow \{C_1, C_2, ...C_{Nk}\}$, **set of known classes**

For $C_i \in C_{Nk}$ **:**

$\Phi \leftarrow fit$ Weibull(D, τ, C_i) via LibMR toolbox **[120]**

$Model_i \leftarrow model\ reduction(D, \Phi, \varsigma)$ supplemental material of **[44]**

End loop

The algorithm trains the dataset of *Nk* audio classes to build its models. In the testing stage, the algorithm requires the test samples x_t, classes models, and the rejection threshold δ. The

following subsection discusses the threshold criteria. The output of the testing phase is one of the following classes $\{1, 2, ..., Nk, o\}$, where o represents the unknown class. The decision function for classifying the *Nk* known classes is set based on the predefined threshold. When the system receives a sample for recognition, it is compared to the stored models to determine the EVM function.

The cover probability is set as a default of $\varsigma = 0.5$ and a tail size $\tau = 300$. Details on the tuning of these parameters can be found in [6], [12]. We tested the EVM output score regarding its ability to distinguish known classes from unknown classes. For algorithm validation, Figure 7.1 shows the distribution of EVM probabilities corresponding to known and unknown classes.

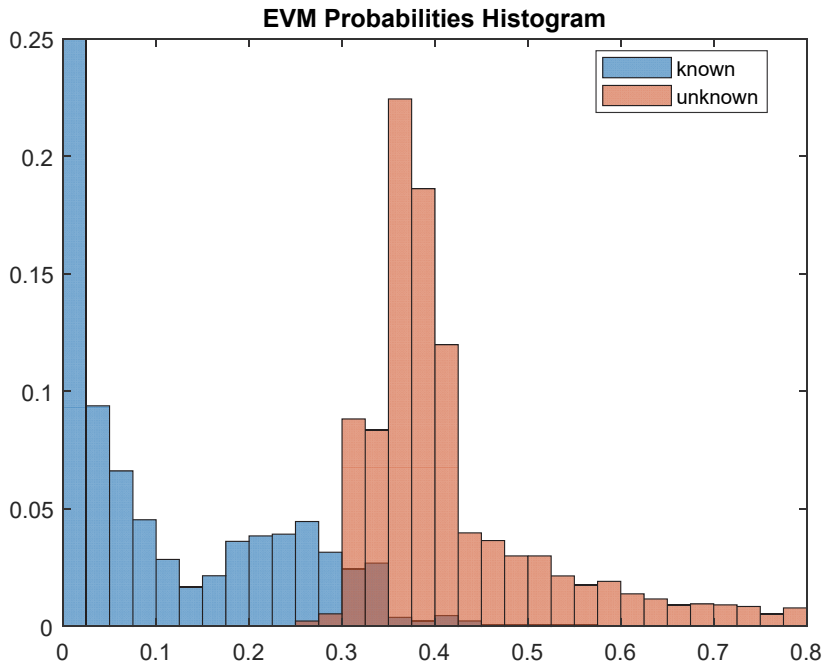


Figure 7.1 Histogram of EVM probabilities

7.4.2.1. Thresholding Criteria

To distinguish between the known classes and unknown classes, a threshold value is used. If a decision value is greater than a certain threshold, the tested samples will be considered to belong to one of the known classes, otherwise, the sample will be rejected. In order to get an optimum threshold, we selected 10% of the audio samples from all classes for validation, i.e., for threshold optimization. The remaining data are used later for the classification experiments. We want to tune the threshold so that it works well in an open environment. Therefore, we followed the threshold optimization method in [110], where the cross-validation technique was applied to the validation data with a different collection of openness to obtain the best possible threshold.

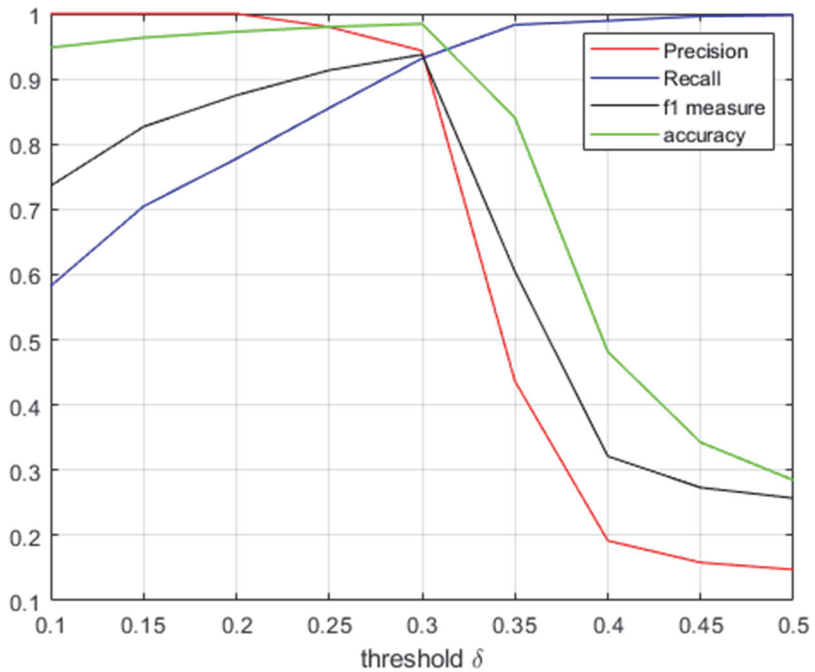


Figure 7.2 Search grid for threshold optimization

Figure 7.2 shows the grid search to find the best possible threshold. As can be seen, the threshold $\delta = 0.3$ leads to the best performance. That is, the threshold value optimizing the performance of the classifier on the validation set is fixed when applying the classifier on new samples in a test set.

7.4.2.2. Evaluation

This part presents the experimental study that was carried out for evaluating the efficiency of the classifiers in open set recognition. For features generation, we computed the same audio features is in previous chapters of the thesis, which are the Mel frequency cepstral coefficients and the Gammatone cepstral coefficients, with the spectral flux and spectral centroid features. These features carry important information of audio signals. For our experiments, we applied stratified k-fold cross-validation, where the dataset is split on k folds such that each fold contains approximately the same percentage of samples of each class. Algorithm 7.2. illustrates the pseudocode for this mechanism.

Algorithm 7.2: Pseudocode for open-set stratified k-fold cross-validation

---

function Stratified **k-fold** cross-validation

T_Metric $\leftarrow 0$

For iter =1 to 5 iteration do **:**

$(K\_classes, U\_classes) \leftarrow$ pick randomly known and unknown classes

For fold =1 to $(k = 5)$ do **:**

$training\_set, testing\_set \leftarrow$ PARTITION($examples, K\_classes, U\_classes, fold$)

For *Openness =0:0.5*$\leftarrow$ *combination of (2* :K_classes) & *(0* :U_classes)

Models$\leftarrow$Train_ML_algorithm with *training_set(* only K_classes*)*

Output$\leftarrow$*ML_recog(training_set(* K_classes*),testing_set* (K_classes$\cup$ *U_classes),conf)*

$T\_Metric(openness) \leftarrow T\_Metric(openness) + evaluate\_metric(output)$

End loop

End *fold* **loop**

End *iter* **loop**

$Metric\_Total \leftarrow Metric\_Total / (5 \times 5)$ averaging scores

Throughout the experiments, we randomly selected classes to be the known and unknown classes, creating a gradual transition from a closed set configuration to an open set configuration. The evaluation metrics consider the unknown classes as one basic class in a way it is appropriate for the multiclass closed-set situation. We approach the performance evaluation with metrics that compute similarities after aligning the recognition outputs with a reference ground truth. Two approaches have been used in the DCASE/ AASP challenges [66],[94] to evaluate how individual audio sounds are classified: segment-based evaluation and event-based evaluation.

The evaluation metrics consider the unknown classes as one basic class in a way it is appropriate for the multiclass closed-set situation. We approach the performance evaluation with metrics that compute similarities after aligning the recognition outputs with a reference ground truth. Two approaches have been used in the DCASE/ AASP challenges [66],[94] to evaluate how individual audio sounds are classified: segment-based evaluation and event-based evaluation.

Table 7.1 shows the results of the segment-based evaluation for open set recognition, where the algorithm outputs and ground truths are compared for each segment length. Table 7.2 shows the results of the event-based evaluation, where the algorithm outputs are considered to be the same within the whole duration of events. This means that event labels in the recognition output will be compared to the ground truth event. It can be seen that the algorithms can handle the open set recognition in most cases, but they became inaccurate when openness becomes too large. Results of audio recognition using EVM are compared with other methods in a later section of this chapter.

Table 7.1 Evaluation metric of segment-based open-set recognition using the extreme value machine

Openness	accuracy	Precision		Recall		F1-measure	
		Macro	Micro	Macro	Micro	Macro	Micro
0	0.836	0.806	0.842	0.831	0.846	0.818	0.844
0.087	0.78	0.750	0.781	0.786	0.776	0.768	0.778
0.106	0.79	0.760	0.762	0.809	0.801	0.784	0.781
0.134	0.775	0.745	0.735	0.843	0.851	0.791	0.789
0.155	0.798	0.768	0.781	0.818	0.765	0.792	0.773
0.184	0.786	0.756	0.748	0.799	0.881	0.777	0.809
0.184	0.802	0.772	0.701	0.702	0.703	0.735	0.702
0.209	0.806	0.776	0.752	0.755	0.742	0.765	0.747
0.225	0.702	0.702	0.703	0.702	0.707	0.702	0.705
0.244	0.78	0.725	0.721	0.725	0.715	0.725	0.718
0.255	0.677	0.647	0.645	0.643	0.66	0.645	0.652
0.261	0.74	0.671	0.682	0.688	0.695	0.679	0.688
0.293	0.652	0.622	0.627	0.719	0.718	0.667	0.669
0.321	0.662	0.632	0.651	0.658	0.648	0.645	0.649
0.326	0.735	0.685	0.694	0.693	0.665	0.689	0.679
0.333	0.677	0.547	0.579	0.575	0.586	0.561	0.582
0.345	0.763	0.663	0.671	0.689	0.631	0.676	0.650
0.355	0.729	0.649	0.651	0.652	0.651	0.650	0.651
0.368	0.708	0.658	0.657	0.643	0.676	0.650	0.666
0.38	0.684	0.604	0.611	0.603	0.614	0.603	0.612
0.388	0.629	0.599	0.588	0.624	0.651	0.611	0.618
0.397	0.634	0.604	0.616	0.614	0.606	0.609	0.611
0.402	0.644	0.614	0.607	0.706	0.737	0.657	0.666
0.423	0.6	0.570	0.569	0.606	0.601	0.587	0.585
0.445	0.682	0.612	0.617	0.617	0.673	0.614	0.644
0.452	0.645	0.615	0.601	0.596	0.580	0.605	0.59
0.465	0.684	0.554	0.573	0.579	0.582	0.566	0.577
0.478	0.664	0.634	0.644	0.643	0.649	0.638	0.646
0.5	0.666	0.636	0.614	0.611	0.621	0.623	0.617

Table 7.2 Evaluation metric of event-based open-set recognition using the extreme value machine

Openness	accuracy	Precision		Recall		F1-measure	
		Macro	Micro	Macro	Micro	Macro	Micro
0	0.868	0.838	0.880	0.876	0.879	0.857	0.879
0.087	0.836	0.806	0.808	0.865	0.851	0.834	0.829
0.106	0.841	0.811	0.816	0.858	0.858	0.834	0.836
0.134	0.828	0.798	0.790	0.863	0.866	0.829	0.826
0.155	0.833	0.803	0.808	0.819	0.818	0.811	0.813
0.184	0.821	0.791	0.792	0.843	0.841	0.816	0.816
0.184	0.836	0.760	0.768	0.768	0.768	0.764	0.768
0.209	0.821	0.791	0.780	0.822	0.820	0.806	0.800
0.225	0.751	0.721	0.720	0.769	0.775	0.744	0.746
0.244	0.822	0.762	0.763	0.761	0.765	0.761	0.764
0.255	0.722	0.712	0.719	0.757	0.742	0.734	0.730
0.261	0.816	0.746	0.741	0.758	0.755	0.752	0.748
0.293	0.709	0.659	0.662	0.759	0.758	0.705	0.707
0.321	0.716	0.676	0.666	0.757	0.751	0.714	0.706
0.326	0.812	0.782	0.760	0.756	0.776	0.769	0.768
0.333	0.739	0.649	0.656	0.642	0.642	0.645	0.649
0.345	0.812	0.782	0.783	0.695	0.689	0.736	0.733
0.355	0.705	0.675	0.681	0.686	0.692	0.680	0.686
0.368	0.813	0.683	0.695	0.707	0.709	0.695	0.702
0.38	0.705	0.675	0.684	0.681	0.696	0.678	0.69
0.388	0.708	0.678	0.659	0.647	0.642	0.662	0.65
0.397	0.696	0.656	0.643	0.644	0.646	0.650	0.644
0.402	0.782	0.732	0.735	0.736	0.727	0.734	0.731
0.423	0.654	0.624	0.632	0.677	0.674	0.649	0.652
0.445	0.735	0.675	0.686	0.687	0.683	0.681	0.684
0.452	0.715	0.655	0.654	0.644	0.655	0.649	0.654
0.465	0.731	0.561	0.654	0.651	0.653	0.603	0.653
0.478	0.695	0.665	0.683	0.687	0.674	0.676	0.678
0.5	0.668	0.638	0.649	0.656	0.638	0.647	0.643

7.4.3. Incorporating EVM with SVM

The extreme value machine (EVM) has a good performance in determining probabilities, regardless of the overall distribution of the data, while the support vector machine (SVM)

algorithms have typically good classification performance. Therefore, in this section we combine both methods to boost the performance of audio open set recognition. To achieve this, we modify the algorithm mechanism introduced in [31]. The decision function of each SVM classifier is given by:

$$f_{\omega,b}(x) = \mathrm{sgn}\left(\langle \omega, \psi(x) \rangle + b\right) \tag{7.5}$$

where b and ω are an offset parameter and a multi-dimensional weight vector, respectively, and $\psi : R^d \rightarrow H_{feat}$ is a kernel function. The Radial Basis Function (RBF) kernel $K(x_i, x_j) = \exp(-\gamma \left\| x_i - x_j \right\|^2)$ is used here, where $\gamma > 0$ controls the extent of influence for a single training sample. The input data are transferred into a high-dimensional feature space H_{feat}. Practically, the transformation ψ is indirectly defined by the (RBF) kernel function, so that,

$$K(x_i, x_j) = \langle \psi(x_i) \psi(x_j) \rangle \tag{7.6}.$$

The multi-class EVT-based probability estimator uses the Weibull distribution of the extrema of the positive decision scores, estimated from an RBF kernel trained with machine learning. The training process of this algorithm is described in Algorithm 7.3.

Algorithm 7.3: Pseudocode for incorporating EVM with SVM

Req: $D = \left\langle x^{(i)}, y^{(i)} \right\rangle, i = 1,..,N$	▷ **samples from audio classes**
$C_{training} \leftarrow \{C_1, C_2, ...C_{Nk}\},$	▷ **set of known classes**
For $y = 1 \rightarrow \lvert C_{training} \rvert$ **:**	
Compute the support vector ℓ_y	▷ Training SVM classifier
$[\tau_y, \kappa_y, \lambda_y] \leftarrow Weibulfit(\ell_y)$	▷ Fit a Weilbull distribution
$\theta_y \leftarrow [\tau_y, \kappa_y, \lambda_y]$	
End loop	
Return $W = [\theta_1, ..., \theta_{Nk}]$	▷ parameters set

For testing, the traditional SVM decision scores are uncalibrated values and are not posterior probabilities, and thus a rejection process is not possible. Therefore, utilizing the distribution of posterior probabilities produces a better SVM classifier [117]. The probability of inclusion Φ of the EVT can work as an assistant where the margin distance is optimized using a Weibull distribution. The Weibull distribution produces probabilities for all binary classifiers, and if none of the classifier's probabilities is less than a certain threshold, the test sample is considered unknown.

7.4.3.1. Evaluation

In this section we use the same feature extraction process as in previous sections and chapters. For the SVM classifiers, we conduct a cross-validation grid search with exponentially growing sequences of parameters c and γ to select the combination that provides the best cross-validation accuracy. To have robust evaluation metrics, the experiments were performed in a stratified five-fold cross-validation setup. The experiments were performed to recognize audio sounds where the

testing set may not include the same categories as in the training set. These experiments measure the capability to discriminate known classes from unknown classes, and to discriminate known classes from one another.

Therefore, the simulation of this scenario is done by selecting a portion of the available classes and considering them as known in the training and testing stages. During the testing, we consider rejected samples as either true (if from an unknown class) or false (if from a known class). Furthermore, the algorithm assigns accepted samples to one of the known classes. The class with the maximum score, probability, or votes is the predicted class. Any algorithm that does not produce a good rejection will have very poor precision as the simulation setup becomes more open because if they miss accepting or rejecting a sample at the threshold stage it will appear as misclassified in the metrics used. The F1-score, precision, and recall measures are computed, and the experiments are conducted with different numbers of known and unknown classes. Table 7.3 and Table 7.4 reveal the results of the macro-average and the micro-average metrics. As to be expected, when more classes are available during training (less openness), the classifiers become more accurate, and vice-versa. We note that the performance of event-based metrics is again better than the performance for frame-based metrics. Results of audio recognition using EVM with SVM are compared with other methods in a later section of this chapter.

Table 7.3 Evaluation metric of segment-based open-set recognition, incorporating EVM with SVM algorithm.

Openness	accuracy	Precision		Recall		F1-measure	
		Macro	Micro	Macro	Micro	Macro	Micro
0	0.856	0.826	0.822	0.851	0.856	0.838	0.839
0.087	0.852	0.822	0.821	0.866	0.876	0.843	0.848
0.106	0.830	0.800	0.802	0.829	0.821	0.814	0.811
0.134	0.875	0.805	0.795	0.793	0.791	0.799	0.793
0.155	0.848	0.818	0.814	0.835	0.838	0.826	0.826
0.184	0.838	0.778	0.772	0.801	0.809	0.789	0.79
0.184	0.842	0.812	0.822	0.822	0.812	0.817	0.817
0.209	0.844	0.764	0.762	0.793	0.78	0.778	0.771
0.225	0.828	0.798	0.798	0.792	0.742	0.795	0.769
0.244	0.822	0.792	0.791	0.786	0.782	0.789	0.786
0.255	0.803	0.773	0.754	0.755	0.758	0.764	0.756
0.261	0.791	0.661	0.656	0.656	0.636	0.658	0.646
0.293	0.760	0.673	0.673	0.715	0.714	0.693	0.693
0.321	0.752	0.722	0.732	0.736	0.74	0.729	0.736
0.326	0.732	0.702	0.728	0.732	0.734	0.717	0.731
0.333	0.702	0.702	0.706	0.696	0.684	0.699	0.695
0.345	0.698	0.718	0.734	0.738	0.726	0.728	0.730
0.355	0.762	0.732	0.699	0.703	0.701	0.717	0.700
0.368	0.774	0.704	0.704	0.704	0.706	0.704	0.705
0.38	0.762	0.632	0.628	0.694	0.661	0.662	0.644
0.388	0.756	0.726	0.706	0.692	0.656	0.709	0.68
0.397	0.728	0.698	0.682	0.672	0.660	0.685	0.671
0.402	0.708	0.618	0.618	0.688	0.677	0.651	0.646
0.423	0.688	0.658	0.682	0.586	0.565	0.620	0.618
0.445	0.688	0.658	0.684	0.604	0.591	0.630	0.634
0.452	0.691	0.601	0.608	0.677	0.749	0.637	0.671
0.465	0.701	0.671	0.674	0.678	0.672	0.674	0.673
0.478	0.685	0.625	0.628	0.672	0.786	0.648	0.698
0.5	0.642	0.588	0.580	0.629	0.866	0.608	0.695

Table 7.4 Evaluation metric of event-based open-set recognition, incorporating EVM with SVM algorithm.

Openness	accuracy	Precision		Recall		F1-measure	
		Macro	Micro	Macro	Micro	Macro	Micro
0	0.868	0.838	0.838	0.876	0.879	0.857	0.858
0.087	0.836	0.806	0.808	0.865	0.861	0.834	0.834
0.106	0.841	0.811	0.806	0.858	0.858	0.834	0.831
0.134	0.828	0.778	0.790	0.853	0.856	0.814	0.822
0.155	0.851	0.861	0.853	0.846	0.844	0.853	0.848
0.184	0.881	0.851	0.862	0.856	0.893	0.853	0.877
0.184	0.851	0.851	0.849	0.820	0.820	0.835	0.834
0.209	0.878	0.848	0.844	0.873	0.871	0.860	0.857
0.225	0.814	0.804	0.804	0.862	0.858	0.832	0.830
0.244	0.862	0.832	0.838	0.843	0.832	0.837	0.835
0.255	0.840	0.781	0.782	0.782	0.7804	0.781	0.781
0.261	0.836	0.781	0.781	0.781	0.7808	0.781	0.781
0.293	0.814	0.784	0.794	0.798	0.775	0.791	0.784
0.321	0.807	0.777	0.780	0.800	0.802	0.788	0.791
0.326	0.805	0.715	0.718	0.749	0.754	0.732	0.736
0.333	0.76	0.734	0.737	0.714	0.712	0.724	0.724
0.345	0.704	0.707	0.703	0.722	0.72	0.714	0.711
0.355	0.786	0.776	0.782	0.777	0.768	0.776	0.775
0.368	0.786	0.756	0.753	0.757	0.756	0.756	0.754
0.38	0.784	0.684	0.698	0.682	0.676	0.683	0.687
0.388	0.736	0.716	0.714	0.735	0.736	0.725	0.725
0.397	0.744	0.704	0.701	0.691	0.693	0.697	0.697
0.402	0.768	0.718	0.712	0.735	0.734	0.726	0.723
0.423	0.790	0.790	0.760	0.719	0.721	0.753	0.74
0.445	0.765	0.735	0.729	0.713	0.714	0.724	0.721
0.452	0.764	0.734	0.728	0.726	0.727	0.730	0.727
0.465	0.640	0.640	0.620	0.729	0.724	0.682	0.668
0.478	0.756	0.726	0.719	0.719	0.718	0.722	0.718
0.5	0.758	0.728	0.735	0.735	0.7466	0.731	0.741

7.4.4. Incorporating EVM with CNN

For a third approach, we use the extreme value machine with a convolutional neural network (CNN) for robust audio recognition. CNNs were introduced by Lecun et al. [128] for document recognition and have become widely used, in particular for image recognition. The architecture of a CNN consists of three layers: convolutional layer, pooling layer, and activation layer, e.g., SoftMax layer. The first two layers are responsible for features extraction, whereas the third layer performs the recognition decision. The extreme value machine is combined with the third layer. We provide some background theory for CNN in Appendix C. However, CNNs are typically used to perform closed set recognition, where classes are the same in the training and testing stages. For open-set recognition, Bendale et al [50] introduced a modified version of CNNs by replacing the Softmax activation function with a function called OpenMax. The OpenMax function uses meta-recognition [121] and activation vectors to provide the probability of a novel class. It measures the distance between the model vector for the known classes and an activation vector (AV) from the input sound.

7.4.4.1. Audio Processing

A CNN is a type of deep learning algorithm that is typically used for image recognition. We will use them for audio recognition by extracting features that look like images and then shaping them to be fed into a CNN. We use a Mel-spectrogram as the input feature to the CNN mechanism since it has proven to be suitable for the human auditory system [129]. The Mel spectrogram is a spectrogram where the frequencies are converted to the Mel scale. We use log-Mel spectrogram coefficients, which is a representation of the power spectrogram in the Mel scale domain. We compute the log-Mel spectrogram coefficients by applying a bank of overlapping triangular filters that determine the energy of the spectrum in each band. The relationship between the Mel spectrum

and the frequency and their computations were discussed in chapter 4.

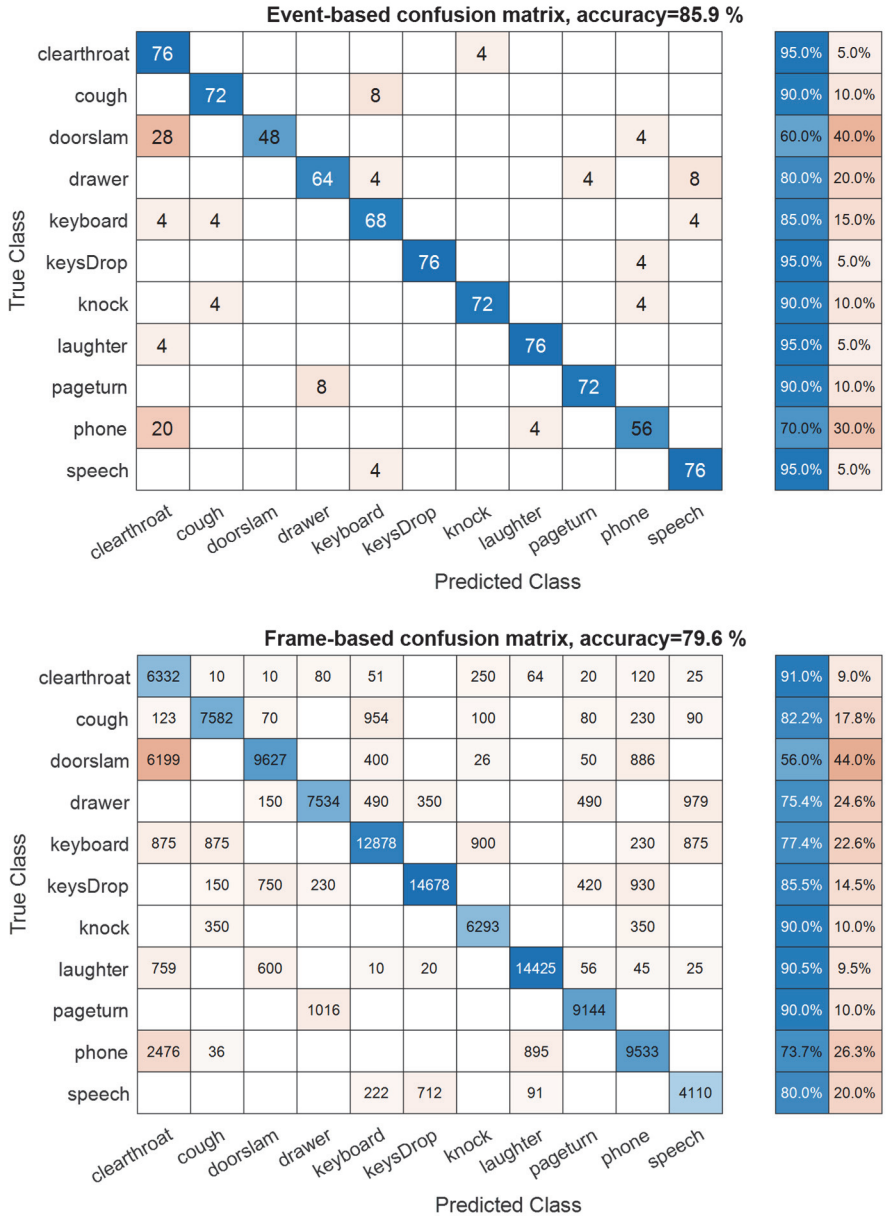


Figure 7.3 Confusion matrix of closed-set recognition of CNN algorithm.

7.4.4.2. Evaluation

The evaluation of the performance obtained when combining EVM with CNN is presented in this section. First, we carry out closed set recognition experiments where the audio dataset is split into training and testing datasets. We use 5-folding cross-validations where 20% of the data are used for testing, and 80% of the data are used as the training dataset to model the classifiers. The experiment is conducted on the DCASE2016 dataset. The confusion matrix in Figure 7.3 shows that most of the signal samples are correctly classified. The event-based confusion matrix reveals that most of the classes have been recognized very well except door-slam and phone-ring classes, whose accuracies were 60% and 70 %, respectively.

The second experiment is for open-set recognition, where we randomly selected the known and unknown classes, creating a gradual transition from a closed set configuration to an open set configuration. Table 7.5 and Table 7.6 show the segment-based and event-based evaluation metrics for this experiment, where we consider the unknown classes as one basic class. Results of audio recognition using EVM with CNN are compared with other methods in the next section of this chapter.

Table 7.5 Evaluation metric of segment-based open-set recognition, incorporating EVM with CNN algorithm.

Openness	accuracy	Precision		Recall		F1-measure	
		Macro	Micro	Macro	Micro	Macro	Micro
0	0.846	0.816	0.812	0.861	0.876	0.838	0.843
0.087	0.816	0.786	0.791	0.866	0.876	0.824	0.831
0.106	0.810	0.780	0.782	0.819	0.811	0.799	0.796
0.134	0.825	0.795	0.793	0.795	0.799	0.795	0.796
0.155	0.823	0.793	0.783	0.836	0.831	0.814	0.806
0.184	0.812	0.742	0.743	0.801	0.806	0.770	0.773
0.184	0.828	0.798	0.892	0.818	0.816	0.808	0.852
0.209	0.831	0.801	0.814	0.813	0.812	0.807	0.813
0.225	0.819	0.789	0.791	0.799	0.791	0.794	0.791
0.244	0.815	0.705	0.708	0.735	0.740	0.720	0.724
0.255	0.801	0.701	0.702	0.762	0.755	0.73	0.728
0.261	0.789	0.759	0.753	0.754	0.733	0.756	0.743
0.293	0.766	0.736	0.741	0.741	0.731	0.738	0.736
0.321	0.754	0.724	0.729	0.731	0.731	0.727	0.73
0.326	0.740	0.710	0.720	0.723	0.725	0.716	0.722
0.333	0.734	0.704	0.711	0.713	0.872	0.708	0.783
0.345	0.732	0.702	0.707	0.711	0.713	0.706	0.71
0.355	0.740	0.710	0.697	0.700	0.708	0.705	0.702
0.368	0.747	0.707	0.694	0.696	0.685	0.701	0.689
0.38	0.745	0.715	0.704	0.783	0.766	0.747	0.734
0.388	0.742	0.682	0.687	0.684	0.655	0.683	0.671
0.397	0.670	0.697	0.676	0.670	0.656	0.683	0.666
0.402	0.703	0.683	0.680	0.674	0.654	0.678	0.667
0.423	0.693	0.663	0.673	0.669	0.749	0.666	0.709
0.445	0.626	0.656	0.671	0.669	0.654	0.662	0.662
0.452	0.610	0.611	0.615	0.665	0.662	0.637	0.638
0.465	0.630	0.653	0.660	0.662	0.657	0.657	0.658
0.478	0.585	0.605	0.604	0.654	0.65	0.629	0.626
0.5	0.604	0.604	0.609	0.649	0.663	0.626	0.635

Table 7.6 Evaluation metric of event-based open-set recognition, incorporating EVM with CNN algorithm.

Openness	accuracy	Precision		Recall		F1-measure	
		Macro	Micro	Macro	Micro	Macro	Micro
0	0.868	0.838	0.838	0.876	0.876	0.857	0.857
0.087	0.836	0.806	0.808	0.865	0.871	0.834	0.838
0.106	0.841	0.811	0.806	0.858	0.858	0.834	0.831
0.134	0.828	0.798	0.809	0.843	0.846	0.820	0.827
0.155	0.862	0.832	0.835	0.819	0.814	0.825	0.824
0.184	0.851	0.821	0.821	0.874	0.876	0.847	0.848
0.184	0.828	0.808	0.802	0.865	0.864	0.836	0.832
0.209	0.786	0.783	0.783	0.813	0.812	0.798	0.797
0.225	0.856	0.826	0.828	0.853	0.858	0.839	0.843
0.244	0.852	0.822	0.802	0.837	0.838	0.829	0.82
0.255	0.838	0.808	0.824	0.82	0.818	0.814	0.821
0.261	0.831	0.801	0.814	0.814	0.817	0.807	0.815
0.293	0.814	0.784	0.799	0.792	0.783	0.788	0.791
0.321	0.705	0.724	0.733	0.792	0.786	0.756	0.759
0.326	0.718	0.718	0.718	0.778	0.779	0.747	0.747
0.333	0.796	0.766	0.778	0.777	0.769	0.771	0.773
0.345	0.793	0.763	0.785	0.778	0.766	0.77	0.775
0.355	0.791	0.761	0.776	0.769	0.775	0.765	0.775
0.368	0.703	0.723	0.726	0.741	0.743	0.732	0.734
0.38	0.792	0.762	0.754	0.748	0.742	0.755	0.748
0.388	0.788	0.758	0.751	0.787	0.777	0.772	0.764
0.397	0.773	0.743	0.725	0.725	0.722	0.734	0.723
0.402	0.780	0.735	0.736	0.766	0.754	0.750	0.745
0.423	0.768	0.718	0.719	0.719	0.714	0.718	0.716
0.445	0.667	0.707	0.706	0.718	0.721	0.712	0.713
0.452	0.764	0.734	0.723	0.713	0.727	0.723	0.725
0.465	0.668	0.680	0.650	0.714	0.741	0.697	0.693
0.478	0.655	0.625	0.618	0.708	0.701	0.664	0.657
0.5	0.646	0.616	0.617	0.716	0.735	0.662	0.671

7.5. Comparison

In this section, we compare the performance of audio recognition using the methods of the previous sections (EVM, EVM with SVM, and EVM with CNN), when varying the openness from 0% (i.e., closed-set scenario) to very open-set classification scenarios.

7.5.1. Comparison Among the EVM-based Methods.

Fig. 7.4 shows the comparison of the three methods with segment-based and event-based evaluations, using the F1-measure computed after comparing the recognition outputs with the reference ground truths. The results reveal that all the methods produced good results, with the EVM-SVM and EVM-CNN methods producing similar overall results and outperforming the EMV method.

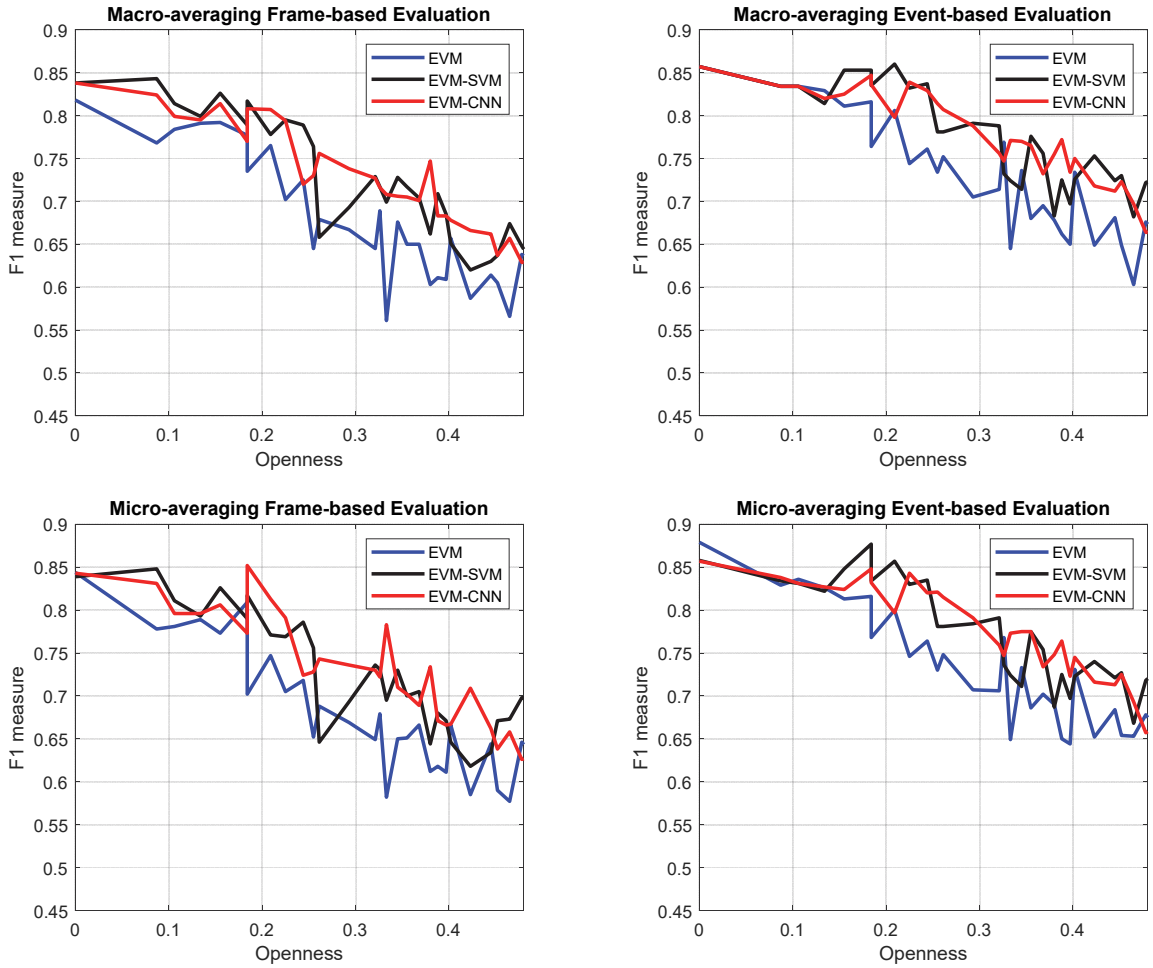


Figure 7.4. F1-measure for open-set recognition as a function of openness for the three EVM-based algorithms.

7.5.2. Comparison with Other Methods

In this section, we compare the performance of EVM-based methods from this chapter with other representative previously published algorithms: W-SVM [30], OSNN [110], OSmIL [111], and our IOmSVM proposed in chapter 6. The parameters of all previous algorithms were set according to the corresponding paper. Fig 7.5 and Fig 7.6 show that the combination of EVM and SVM and the combination of EVM and CNN proposed in this chapter both outperform the other methods, over a wide range of openness values.

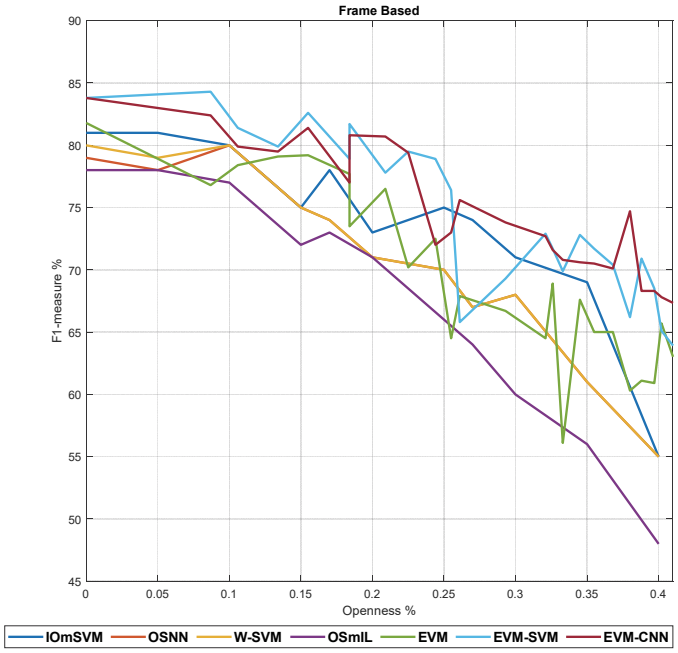


Figure 7.5 F1-measure for open-set recognition as a function of openness. Results computed for frame-based metrics

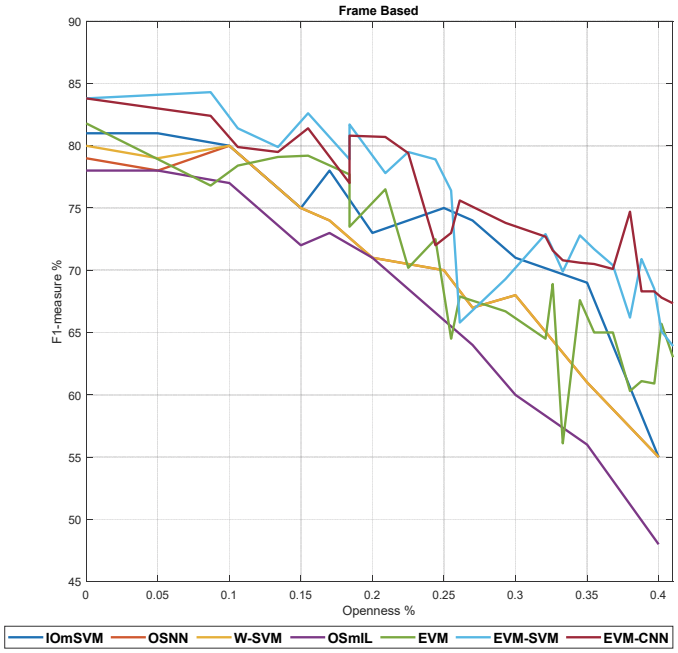


Figure 7.6 F1-measure for open-set recognition as a function of openness. Results computed for event-based metrics

7.6. Summary

In this chapter, we addressed the open set audio recognition problem using the extreme value machine (EVM) method, as an attempt to make the performance of open set recognition more robust. The EVM method was also combined with SVM and CNN classifiers. An experimental performance evaluation was conducted for these methods and other representative methods. The results show that the methods proposed in this chapter are suitable for open-set multiclass classification and provide better overall performance for different levels of openness.

Chapter 8: Conclusions and Future Work

8.1. Summary

Real-world implementation of machine learning-based approaches is still a work in progress. The assumptions of traditional statistical learning models that are based on Bayesian models often do not hold. The goal of this research was to study the feasibility of recognition systems in open-set environments, where the model was trained with a few classes from a larger space that has more classes. Our contributions are to detect novel sound categories, recognize sounds belonging to familiar ones, and increment additional categories for update learning.

The first contribution has been to propose a multi-class open set recognition with rejection using SVM classifiers. In addition to audio recognition, our method was able to determine if a sample sound does not belong to one of the known classes and reject it. The most challenging part of rejection approaches is the difficulty to find an appropriate threshold to distinguish between known and unknown classes. Therefore, we proposed the use of a similarity ratio: the Peak-Side-Ratio (PSR). The PSR ratio computes the distribution of posterior probabilities for all classifier outputs, to determine whether a particular sound or event belongs to a certain group of known classes or not. We conducted extensive experiments using different sound features proposed in the literature for closed-set audio identification. We performed experiments using SVM classifiers on database sounds mostly retrieved from the DCASE challenge. We evaluated performance results by varying the openness from 0% (i.e., closed-set scenario) to a maximum of 46 % openness. The same setting and database were used to compare the method with previous work. Our results delivered the best or nearly the best performance over a wide range of openness values.

The second contribution has been in incremental learning. We introduced two methods that incrementally add new classes to a classifier. One method utilizes the open set SVM classifier to create a new model from the new class sounds, using old data as negative samples when it trains the new model. The other method used a hyper machine learning algorithm, with an SVM combined with a k-NN classifier to implement the incremental update and train the model for the new data. We have compared the results of our proposed methods with representative prior work and shown that our proposed methods delivered better performance overall, for a wide range of openness values.

We extended our work for audio open set recognition by using an Extreme Value Machine (EVM). This is the third contribution of this thesis. Our work utilizes the implementation of EVM in [44] and applies it for open set recognition. Comparisons with previous representative work have shown that: 1) an EVM model can perform effectively for multi-class audio recognition, and 2) the combination of EVM with either an SVM or CNN classifier can produce the best open set recognition performance overall, for a wide range of openness values.

To sum up, this thesis addressed problems associated with open set audio recognition. Using open set machine learning algorithms as classifiers allowed to detect novel sounds as unknown classes and fit them to new classes, in addition to traditional closed set recognition.

8.2. Future Work

Several of the items explored in the thesis could be expanded upon. More specifically, possible future work could include the following:

1. Audio features and representation are still an issue for audio recognition. Possible future work could be to explore and analyze the performance of proposed learning algorithms with different feature representations for extensive datasets.
2. The decisions for novel classes detection are made based on a fixed threshold as illustrated in chapter 5. Possible future work could be to explore the idea of adaptive thresholds - perhaps via more reject thresholds - and analyze the capabilities of the systems.
3. In chapter 6, the incremental algorithms had a satisfactory performance when dealing with few new classes, but as the number of classes increases, their performance degraded. Therefore, updates to the algorithms may be required to improve their scalability.

Appendix A: Support Vector Machine

In this Appendix, we present the principle of the support vector machine (SVM). SVM is a discriminative classifier, which is known as an effective approach for pattern recognition. The main aim of SVM is to determine a decision boundary or hyperplane. The hyperplane separates two classes of input data points.

A.1 SVM Separability

A.1.1 Separable Classes

The hyperplane is found based on the principle of maximum margin. Let us consider classifying d-dimensional vectors $\mathbf{x} = \{x_1, \ldots, x_L\}$ which belong to class labels $y_i \in \{+1, -1\}$ $i = 1, \ldots, L$

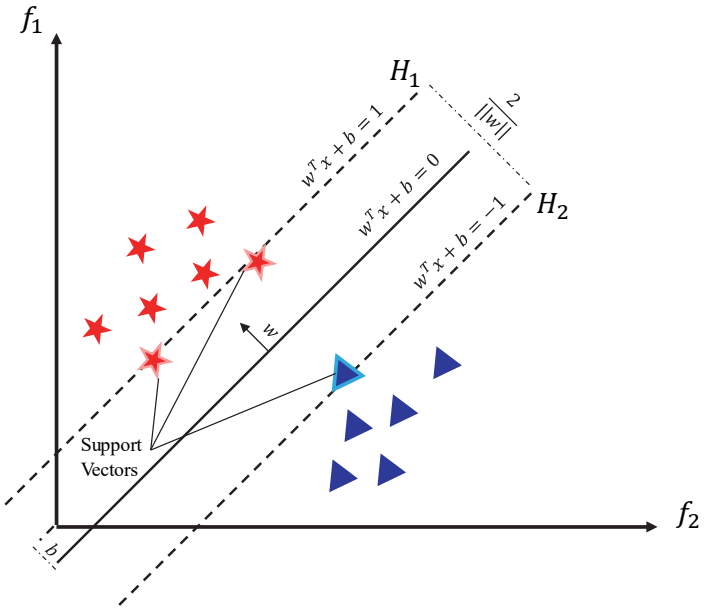


Figure A.1 SVM structure of separable two-classes.

where L is the number of feature vector-label pairs, and there exists a hyperplane defined by $w^T x = 0$. We can define the decision function:

$$g(x) = w^T x + b \tag{A.1}$$

where w is a d-dimensional weight vector and b is a bias term. For $i = 1, \ldots, L$, we consider the following inequalities [107]:

$$w^T x_i + b \begin{cases} \geq 1 & \text{for } y_i = 1 \\ \leq -1 & \text{for } y_i = -1 \end{cases} \tag{A.2}.$$

These can be combined into one set of inequalities:

$$\forall x_i : \quad y_i [w^T x_i + b] - 1 \geq 0 \tag{A.3}$$

and the SVM problem can be formulated as:

$$\max_{w,b} \min \{ \| x - x_i \| : w^T x + b = 0, i = 1, ..., L \}, \tag{A.4}$$

where w is the weight vector and b is the bias for the model. Having defined this measurement to find the optimal decision boundary, the next step is to actually compute it from a given set of data points. In linear binary SVM classification, w and b can be rescaled in such a way that the point closest to the hyperplane $w^T x + b = 0$ lies on hyperplanes $H_1 : w^T x + b = 1$ and $H_2 : w^T x + b = -1$ for either class, as illustrated in Figure A.1. It can be seen that the width of the margin is simply equal to $2/\| w \|$. To find the pair of hyperplanes H_1 and H_2 which gives the maximum margin by minimizing $\| w \|$, the optimization problem can be restated as:

$$\tau(w) : \min_{w,b} \tau(w) = \frac{1}{2} \| w \|^2, \tag{A.5}$$

subject to constraints. Such problems can be solved by considering the so-called Lagrangian duality [81]. The problem can be stated equivalently by its Wolfe dual representation form, that is:

$$L(w,b,\alpha) = \frac{1}{2} \| w \|^2 - \sum_{i=1}^{L} \alpha_i (y_i [x_i^T w + b] - 1) \tag{A.6}$$

where $\alpha_i > 0$ are Lagrange multipliers. The minimization leads to $\sum_{i=1}^{L} \alpha_i y_i = 0, w = \sum_{i=1}^{L} \alpha_i y_i x_i$. Since the problem is quadratic, there is a unique optimum. When we find that optimal solution, the Karush–Kuhn–Tucker (KKT) conditions will be satisfied: $\alpha_i (y_i [x_i^T w + b] - 1) = 0$ for $i = 1, ..., L$. The non-zero α_i have a crucial role in the solution of the optimization problem, known as support vectors. The training vectors and labels enter the classifier in pairs, in the form of inner products. The cost function does not depend explicitly on the dimensionality of the input space. This property allows for efficient generalizations in the case of nonlinearly separable classes.

A.1.2 Non-separable Classes

In theses cases, any attempt to draw a hyperplane will never end up with a class separation band with no data points inside it. The problem is approached as an optimization task [81], which can be solved by the SVM:

$$\begin{aligned} \min_{\omega, b} \quad & \frac{1}{2} \omega^T \omega + c \sum_{i=1}^{L} \zeta(\omega, b, x_i, y_i) \\ \text{subject to} : \quad & y_i \left(w^T x_i + b \right) \geq 1 - \zeta_i \end{aligned} \tag{A.7}$$

$\zeta(\omega, b, x_i, y_i)$ is a loss-function, and $c \geq 0$ is a regularization parameter or a penalty on the training error. The maximum-margin hyperplane can be defined by computing both ω and b in high-dimensional space [1]. Given a separating hyperplane, the decision function of the SVM is given by:

$$f_{\omega,b}(x) = \mathrm{sgn}\left(\langle \omega, \psi(x) \rangle + b\right) \tag{A.8}$$

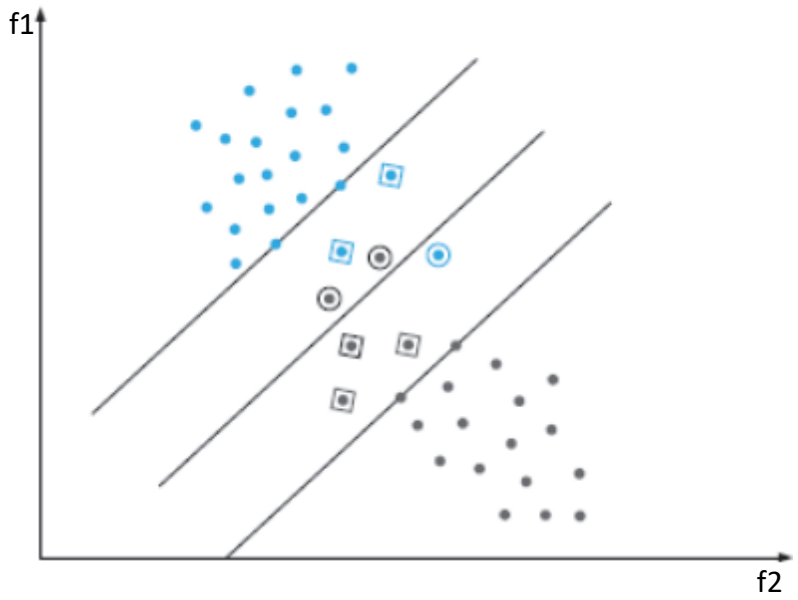


Figure A.2 Non-separable classes case.

where b is a parameter that shows the offset of ω, and $\psi : R^d \rightarrow H_{feat}$ is called a kernel function, which is used to transform the input data into a high-dimensional feature space H_{feat}. Practically, the transformation ψ is indirectly defined by a kernel function, which computes the inner product of vectors $K(x_i, x_j) = \langle \psi(x_i) \psi(x_j) \rangle$. The selection of kernels depends on specific applications. The most popular kernels are:

1. Polynomial kernel. It is popular in image processing $K(x_i, x_j) = (x_i . x_j + 1)^{deg}$, where deg is the degree of the polynomial.
2. Gaussian radial basis function (RBF) Kernel. It is used in general purpose, especially when there is no prior knowledge about the data $K(x_i, x_j) = \exp(-\gamma \left\| x_i - x_j \right\|^2)$ where $\gamma > 0$.

3. Laplace RBF kernel. $K(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|}{\sigma})$.

4. Sigmoid kernel. $K(x_i, x_j) = \tanh(\alpha x_i^T x_j + c)$

A.1.3 The Multi-class Case

In all our discussions so far, we have been involved with the two-class classification task. Consider an Nk -class problem, where we have *L* training samples: $\{x_i, y_i\}, ..., \{x_L, y_L\}$. Here $x_i \in R^d$ is a *d*-dimensional feature vector and $y_i \in \{1, 2, ..., Nk\}$ are the corresponding class label. A straightforward approach is to decompose the Nk -class problem into a set of *Nk* two-class problems (*one-against-all*), which constructs Nk SVM classifiers, where the i^{th} SVM is trained with all the training examples of the i^{th} class with positive labels, and all the other training examples with negative labels. We seek to design an optimal discriminant function, $g_i(x),\ i = 1, 2, ..., Nk$, so that $g_i(x) > g_j(x)\ \ \forall j \neq i$. The i^{th} SVM solves the following problem that yields the i^{th} decision function $g_i(x) = w_i^T \psi(x) + b_i$. At the classification phase, a sample vector *x* is classified as belonging to the class i^* for which g_i produces the largest value:

$$i^* = \arg\max_{i=1,...,Nk} \{g_k(x)\} = \arg\max_{i=1,...,Nk} \{w_i^T \psi(x) + b_i\}. \qquad \text{(A.9)}$$

A.2 Tuning Parameters

Some parameters must be chosen carefully to obtain a good performance. It is common to tune several parameters using hyper-parameters. The process is done by minimizing the estimated generalization error like the k-fold cross-validation error or the leave-one-out error. The SVM parameters are:

- The regularization parameter c. It produces a trade-off between minimizing model complexity and minimizing the training error.
- Kernel function parameter γ. It defines the nonlinear mapping from the input space to the higher dimensional feature space.

For the above two parameters, the tuning process fixes the regularization parameter c at some value and varies the width of the kernel γ. Then it fixes the value of γ and varies the value of c to build multiple pair of parameters.

Appendix B: Statistical Extreme Value Theory

B.1 Definition

Extreme Value Theory (EVT) is a branch of statistics that deals with extreme events stochastic actions contained in the tails of probability distributions. A stochastic model represents a condition where there is ambiguity. Extreme value distributions are defined as a limiting distribution for the minimum or the maximum of stochastic observations from arbitrary distributions.

Suppose that $x_i : i = 1,...,n$ are n ordered independent random variables, with $M_n = \max(x_1,...,x_n)$. If these variables are independent and identically distributed (IID) and a sequence of pairs of real number (a_n, b_n) exists, the finite-sample distribution probability satisfies [43]:

$$\lim_{x\to\infty} P\left\{\frac{M_n - b_n}{a_n} \le x\right\} \rightarrow F(x) \tag{B.1}$$

where $F(x)$ is a non-degenerate extreme value distribution. The most powerful feature of EVT is that its distribution does not depend on the exact common density, i.e., it depends only on the tail [130]. The EVT has many types of distributions, but the primary extreme distributions are the Gumbel, Frechet and Weibull [131]. They are a family of probability distributions developed within the generalized extreme value distributions.

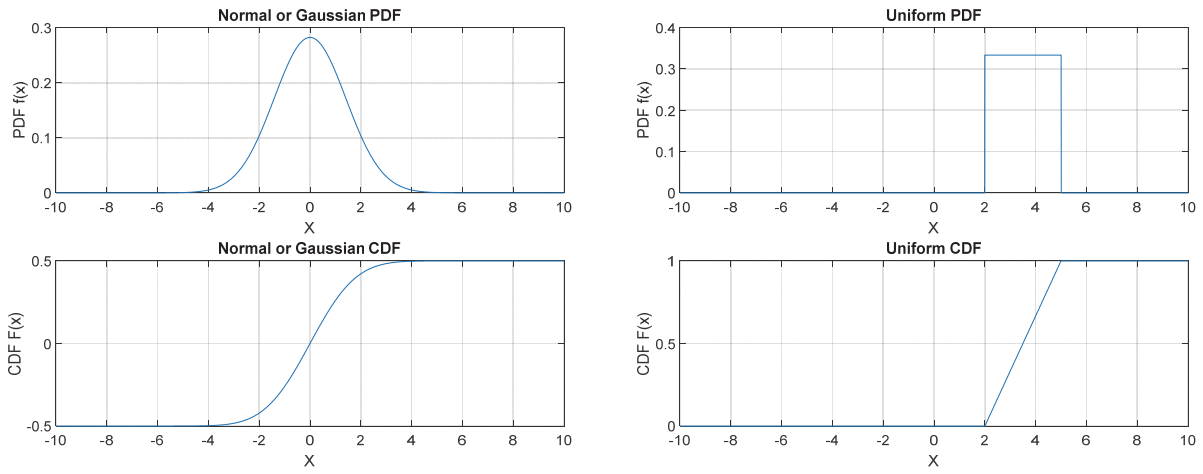


Figure B.1 Normal and uniform distributions

B.2 Distributions in the EVT Family

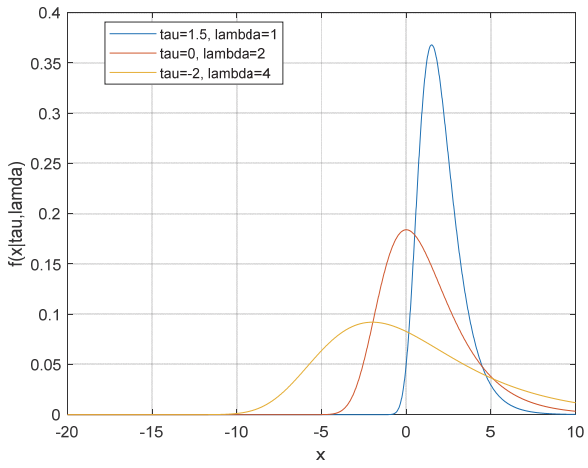
Let x be a sequence of i.i.d. samples, CDF be the cumulative distribution function and PDF be the probability density function. For the following distributions, we use the notation as in [43] where K, λ, τ are the shape, scale, and location parameters, respectively. They differ from the normal or Gaussian distribution, shown in Figure B.1s.

B.2.1 Gumbel Distribution

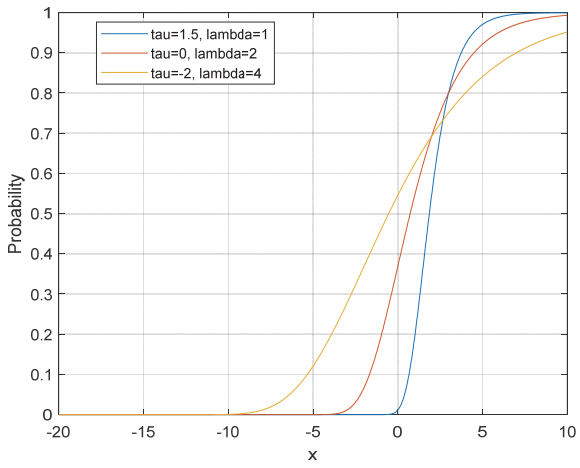
The shape of the Gumbel distribution is not determined by the parameters, only the location and scale are defined. The CDF of this distribution has an infinite right endpoint.

$$\text{PDF:} \qquad Gumbel_f(x) = \frac{1}{\lambda}\exp\left\{-\left(\frac{x-\tau}{\lambda} + \exp\left(-\left(\frac{x-\tau}{\lambda}\right)\right)\right)\right\} \tag{B.2}$$

$$\text{CDF:} \qquad Gumbel_f(x) = \exp\left\{-\exp\left(\frac{-(x-\tau)}{\lambda}\right)\right\}, \quad -\infty < x < \infty; \tag{B.3}$$



(a) Gumbel PDF



(b) Gumbel CDF

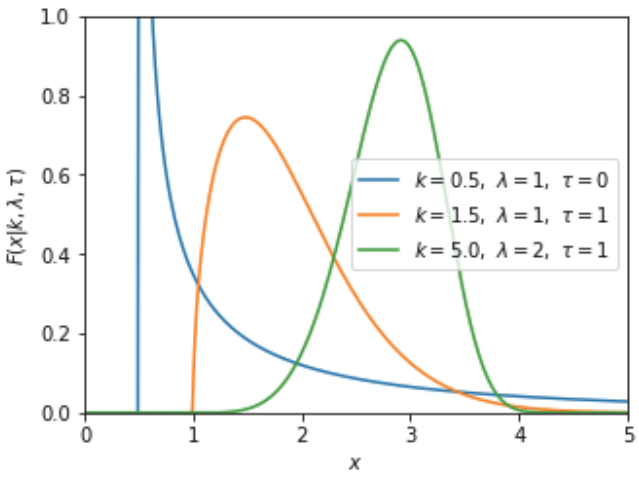
Figure B.2 Gumbel PDFs and CDFs

B.2.2 Fréchet Distribution

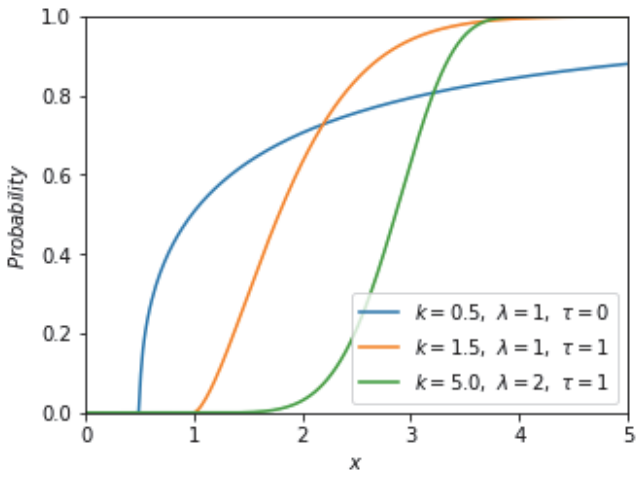
The Fréchet distribution is used when the data are dense in the upper tail.

$$PDF: \quad Fréchet_f(\mathrm{x}) = \frac{\mathrm{K}}{\lambda}\left(\frac{x-\tau}{\mathrm{K}}\right)^{-1-\mathrm{K}} \times \exp\left\{-\left(\frac{x-\tau}{\lambda}\right)^{\mathrm{K}}\right\} \tag{B.4}$$

$$CDF: \quad Fréchet_f(\mathrm{x}) = \exp\left\{-\left(\frac{x-\tau}{\lambda}\right)^{\mathrm{K}}\right\} \tag{B.5}$$



(a) Fréchet PDF



(b) Fréchet CDF

Figure B.3 Fréchet PDFs and CDFs

B.2.3 Weibull Distribution

The shape and scale parameters of the Weibull distribution and inverse Weibull distribution must be positive.

- Weibull distribution

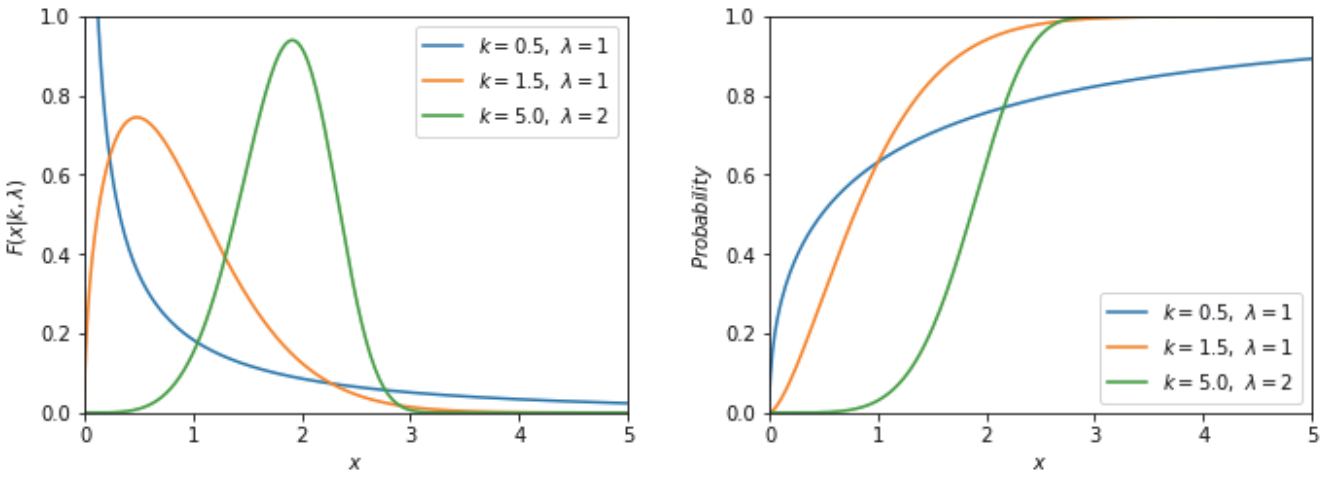
$$PDF: \quad Weibull_f(\mathrm{x}) = \begin{cases} \frac{\mathrm{K}}{\lambda}\left(\frac{x}{\lambda}\right)^{\mathrm{K}-1} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{B.6}$$

$$CDF: \quad Weibull_f(\mathrm{x}) = \begin{cases} 1 - e^{-\left(\frac{x}{\lambda}\right)^{\mathrm{K}}} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{B.7}$$

- Reverse Weibull distribution

  This is simply the opposite of the Weibull distribution function, and is thus used to model maxima.

$$RWeibull_f(\mathrm{x}) = \begin{cases} e^{-\left(\frac{x}{\lambda}\right)^{\mathrm{K}}} & x < 0 \\ 0 & x \geq 0 \end{cases} \tag{B.8}$$



(a) Weibull PDF (b) Weibull CDF

Figure B.4 Weibull PDFs and CDFs

B.3 Fisher-Tippett Theorem

This theorem is a general formulation of extreme value theory regarding the three types of distributions presented earlier. It states that as the number of observations $n \to \infty$, the $\max(x_1, ..., x_n)$ converges in certain distributions $F(x)$. It corrupts the normal law assigning a probability of one to the right endpoint of $F, x_F = sub\{x : F(x) < 1\}$ [132].

B.4 Set-Cover Model Reduction

The set-cover method is an algorithm for model reduction. This algorithm uses only a fixed number of reduced data points and distribution pairs per class to summarize each selected class to be modeled. The primary task is to find an optimal set-covering that has fewer sets. Let x_i be a point

in a certain class C_i and let probability inclusion $\psi\left(x_i, x'; k_i, \lambda_i\right)$ be its corresponding model. If we have another point x_j in the same class, its model will be $\psi\left(x_j, x'; k_j, \lambda_j\right)$. There is redundancy in coverage if $\psi_i\left(x_i, x_j; k_i, \lambda_i\right) \geq \varsigma$, where ς is the coverage probability threshold. Similarly, the function will loop iteratively through all points of the class. It can further be defined as a coverage set of indices where $s_i \equiv \left\{ j \in \{1,...,N\} \,|\, \psi\left(x_i, x_j; k_i, \lambda_i\right) \geq \varsigma \right\}$ for each of the $x_i, \psi\left(x_i, x'; k_i, \lambda_i\right)$ pair with a universe $\bigcup$. The goal of the process is to select the minimum number of sets that contain all the elements of the universe $\bigcup$ [44].

B.5 Recognition Score Normalization

Score normalization has attracted much attention from many researchers. [131], [133]. A good score normalization can be robust to model assumptions, modeling errors, and algorithm failure. Let us define the similarity score obtained as $\{s_k\}$, $k = 1, 2, ..., n$, where n is the number of observations. These scores should be normalized to allow a better interpretation. The following are some normalization scores used for recognition.

The min-max score: The most basic approach is min-max normalization. The normalized min-max score is represented by s'_k .

$$\text{(min-max):} \qquad s'_k = \frac{s_k - s_{\max}}{s_{\max} - s_{\min}} \tag{B.9}$$

where $s_{\max}$ and $s_{\min}$ are the maximum and minimum of the scores respectively. The normalized score is maintained in the range of [0,1]. This normalization approach preserves the original score distribution, but it is sensitive to outlier data samples.

The z-score normalization: Another approach is based on central tendency, called z-score normalization [134]. It uses the mean μ and standard deviation σ statistics to normalize the scores. A normalized z-score s'_k is calculated via:

$$\text{(z-score):} \qquad s'_k = \frac{s_k - \mu}{\sigma} \tag{B.10}$$

Outliers also affect the mean and standard deviation in this approach.

The MAD score: The third approach is similar to the z-score, it is called Median Absolute Deviation (MAD) [135]. The MAD approach is insensitive to the outliers, and it is computed by:

$$s'_k = \frac{s_k - median}{median\left(\left|s_k - median\right|\right)} \tag{B.11}$$

The w-score normalization: This approach was introduced in [133]. The procedure for the normalization is first to collect $\{s_k\}$ scores from a single recognition algorithm. These scores are then sorted by applying a Weibull distribution to fit the scores. Once the fitting has taken place, the CDF of the Weibull distribution is applied to normalize a raw distance score into a probability value.

Appendix C: Convolutional Neural Networks

Convolutional Neural Networks (CNN) are a type of neural network with a convolutional layer that performs a convolution between a filter and the input of the layer. CNNs exhibit very good performance on a variety of machine learning tasks.

Let $\{(x_i, y_i)\}_{i=1,\ldots,n}$ be samples drawn from a joint X and Y distribution, where $X \in \mathrm{X} \subset \mathbb{R}^d$ and $Y \in \Upsilon \subset \mathbb{R}$ are two random variables. The goal of machine learning is to learn a mapping function $\hat{f} : \mathrm{X} \to \Upsilon$ by minimizing the expected loss function, where the loss function is $L : \Upsilon \times \Upsilon \to \mathbb{R}$. The learning function can be written as [136]:

$$\hat{f} = \arg\min_{f \in F} E\left[L(Y, f(x))\right] \tag{C.1}.$$

The network is trained to learn the parameters Θ of a composite nonlinear function $f(x \mid \Theta)$. The model learns these parameters during the training process, and it updates them with each new training example. CNNs are comprised of multiple hidden layers, but the weights and bias values are the same for all hidden neurons in a given layer. The architecture of a convolutional network typically consists of four types of layers as shown in Fig. C.1: convolution, pooling, activation, and fully connected layers. Each layer generates several activation functions whose outputs are passed on to the next layer. Convolution is a crucial step for feature extraction in CNNs. In this layer, the input samples are put through a set of convolutional filters, each of which activates certain features from the input samples. The outputs of the convolutions can be called feature maps.

A pooling layer operates on blocks of the input feature map and combines or concatenates or downsamples the features [137].

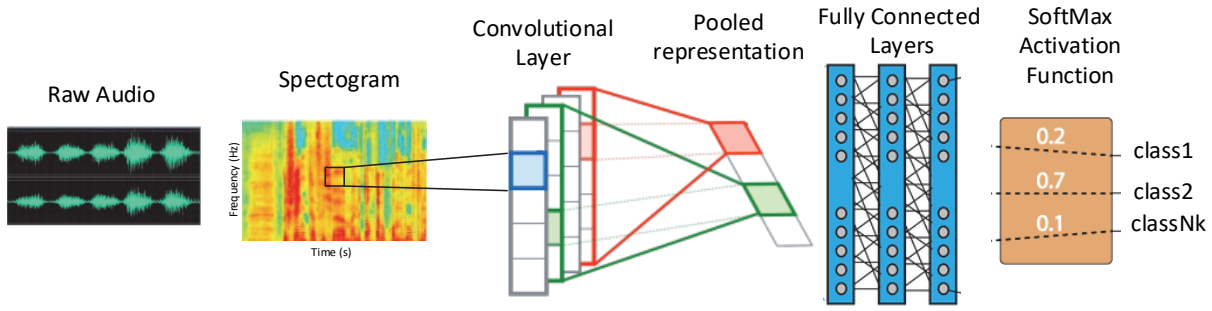


Figure C.1 Convolutional neural network with audio input features.

References

[1] W. J. Scheirer, A. D. R. Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 7, pp. 1757–1772, Jul. 2013, doi: 10.1109/TPAMI.2012.256.

[2] C.-X. Ren and D.-Q. Dai, "Incremental learning of bidirectional principal components for face recognition," *Pattern Recognit.*, vol. 43, no. 1, pp. 318–330, 2010.

[3] T. I. Dhamecha, R. Singh, and M. Vatsa, "On incremental semi-supervised discriminant analysis," *Pattern Recognit.*, vol. 52, pp. 135–147, 2016.

[4] M. Ristin, M. Guillaumin, J. Gall, and L. Van Gool, "Incremental learning of ncm forests for large-scale image classification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, 2014, pp. 3654–3661. doi: 10.1109/CVPR.2014.467.

[5] J. Henrydoss, S. Cruz, E. M. Rudd, M. Gunther, and T. E. Boult, "Incremental open set intrusion recognition using extreme value machine," in *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, Dec. 2017, pp. 1089–1093. doi: 10.1109/ICMLA.2017.000-3.

[6] A. Bendale and T. Boult, "Towards open world recognition," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 1893–1902. doi: 10.1109/CVPR.2015.7298799.

[7] Z. Zhong, L. Zheng, Z. Zheng, S. Li, and Y. Yang, "CamStyle: A novel data augmentation method for person re-identification," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1176–1190, Mar. 2019, doi: 10.1109/TIP.2018.2874313.

[8] L. M. Mazaira-Fernandez, A. Álvarez-Marquina, and P. Gómez-Vilda, "Improving speaker recognition by biometric voice deconstruction," *Front. Bioeng. Biotechnol.*, vol. 3, 2015, doi: 10.3389/fbioe.2015.00126.

[9] D. Pye, "Content-based methods for the management of digital music," in *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*, Jun. 2000, vol. 4, pp. 2437–2440 vol.4. doi: 10.1109/ICASSP.2000.859334.

[10] R. V. Sharan and T. J. Moir, "An overview of applications and advancements in automatic sound recognition," *Neurocomputing*, vol. 200, pp. 22–34, Aug. 2016, doi: 10.1016/j.neucom.2016.03.020.

[11] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Audio surveillance of roads: a system for detecting anomalous sounds," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 1, pp. 279–288, Jan. 2016, doi: 10.1109/TITS.2015.2470216.

[12] T. Heittola, A. Mesaros, A. Eronen, and T. Virtanen, "Context-dependent sound event detection," *EURASIP J. Audio Speech Music Process.*, vol. 2013, no. 1, pp. 1–13, 2013.

[13] S. Chu, S. Narayanan, and C. C. J. Kuo, "Environmental sound recognition with time-frequency audio features," *IEEE Trans. Audio Speech Lang. Process.*, vol. 17, no. 6, pp. 1142–1158, Aug. 2009, doi: 10.1109/TASL.2009.2017438.

[14] D. Mitrović, M. Zeppelzauer, and C. Breiteneder, "Features for content-based audio retrieval," in *Advances in Computers*, vol. 78, Elsevier, 2010, pp. 71–150. doi: 10.1016/S0065-2458(10)78003-7.

[15] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the CUIDADO project," *Cuid. IST Proj. Rep.*, vol. 54, no. 0, pp. 1–25, 2004.

[16] H. Wang, A. Divakaran, A. Vetro, S.-F. Chang, and H. Sun, "Survey of compressed-domain features used in audio-visual indexing and analysis," *J. Vis. Commun. Image Represent.*, vol. 14, no. 2, pp. 150–183, Jun. 2003, doi: 10.1016/S1047-3203(03)00019-1.

[17] Y. Yang, F. Nie, D. Xu, J. Luo, Y. Zhuang, and Y. Pan, "A multimedia retrieval framework based on semi-supervised ranking and relevance feedback," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 723–742, Apr. 2012, doi: 10.1109/TPAMI.2011.170.

[18] G. Muhammad, Y. A. Alotaibi, M. Alsulaiman, and M. N. Huda, "Environment recognition using selected MPEG-7 audio features and mel-frequency cepstral coefficients," in *2010 Fifth International Conference on Digital Telecommunications (ICDT)*, Jun. 2010, pp. 11–16. doi: 10.1109/ICDT.2010.10.

[19] V. Carletti, P. Foggia, G. Percannella, A. Saggese, N. Strisciuglio, and M. Vento, "Audio surveillance using a bag of aural words classifier," in *2013 10th IEEE International Conference on Advanced Video and Signal Based Surveillance*, Krakow, Poland, Aug. 2013, pp. 81–86. doi: 10.1109/AVSS.2013.6636620.

[20] H. Jleed and M. Bouchard, "Acoustic environment classification using Discrete Hartley Transform features," in *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, Apr. 2017, pp. 1–4. doi: 10.1109/CCECE.2017.7946646.

[21] T.-M. Huang, V. Kecman, and I. Kopriva, *Kernel based algorithms for mining huge data sets: supervised, semi-supervised, and unsupervised learning*. Berlin ; New York: Springer, 2006.

[22] V. N. Vapnik, "An overview of statistical learning theory," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 988–999, Sep. 1999, doi: 10.1109/72.788640.

[23] K. Łopatka, J. Kotus, and A. Czyżewski, "Evaluation of sound event detection, classification and localization in the presence of background noise for acoustic surveillance of hazardous situations," in *Multimedia Communications, Services and Security*, A. Dziech and A.

Czyżewski, Eds. Springer International Publishing, 2014, pp. 96–110. doi: 10.1007/978-3-319-07569-3_8.

[24] A. R. Hilal, A. Sayedelahl, A. Tabibiazar, M. S. Kamel, and O. A. Basir, "A distributed sensor management for large-scale IoT indoor acoustic surveillance," *Future Gener. Comput. Syst.*, vol. 86, no. Complete, pp. 1170–1184, 2018, doi: 10.1016/j.future.2018.01.020.

[25] W. Huang, S. Lau, T. Tan, L. Li, and L. Wyse, "Audio events classification using hierarchical structure," in *Proceedings of the 2003 Joint Conference of the Fourth International Conference on Information, Communications and Signal Processing, 2003 and Fourth Pacific Rim Conference on Multimedia*, Dec. 2003, vol. 3, pp. 1299–1303 vol.3. doi: 10.1109/ICICS.2003.1292674.

[26] P. Mahana and G. Singh, "Comparative analysis of machine learning algorithms for audio signals classification," *Int. J. Comput. Sci. Netw. Secur. IJCSNS*, vol. 15, no. 6, p. 49, 2015.

[27] P. R. M. Júnior, T. E. Boult, J. Wainer, and A. Rocha, "Specialized support vector machines for open-set recognition," *ArXiv160603802 Cs Stat*, Jun. 2016, Accessed: Dec. 04, 2018. [Online]. Available: http://arxiv.org/abs/1606.03802

[28] D. Battaglino, L. Lepauloux, and N. Evans, "The open-set problem in acoustic scene classification," in *2016 IEEE International Workshop on Acoustic Signal Enhancement (IWAENC)*, Sep. 2016, pp. 1–5. doi: 10.1109/IWAENC.2016.7602939.

[29] J. Manikandan and B. Venkataramani, "Design of a modified one-against-all SVM classifier," in *2009 IEEE international conference on systems, man and cybernetics*, 2009, pp. 1869–1874.

[30] W. J. Scheirer, L. P. Jain, and T. E. Boult, "Probability models for open set recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 11, pp. 2317–2324, Nov. 2014, doi: 10.1109/TPAMI.2014.2321392.

[31] L. P. Jain, W. J. Scheirer, and T. E. Boult, "Multi-class open set recognition using probability of inclusion," in *Computer Vision – ECCV 2014*, vol. 8691, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 393–409. doi: 10.1007/978-3-319-10578-9_26.

[32] M. D. Scherreik and B. D. Rigling, "Open set recognition for automatic target classification with rejection," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 52, no. 2, pp. 632–642, Apr. 2016, doi: 10.1109/TAES.2015.150027.

[33] J. D. Roos and A. K. Shaw, "Probabilistic SVM for open set automatic target recognition on high range resolution radar data," Anaheim, California, United States, May 2017, p. 102020B. doi: 10.1117/12.2262840.

[34] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," *J. Mach. Learn. Res.*, vol. 7, no. Mar, pp. 551–585, 2006.

[35] J. Xu, C. Xu, B. Zou, Y. Y. Tang, J. Peng, and X. You, "New incremental learning algorithm with support vector machines," *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 49, no. 11, pp. 2230–2241, Nov. 2019, doi: 10.1109/TSMC.2018.2791511.

[36] Y. Liu, Y. Su, A.-A. Liu, B. Schiele, and Q. Sun, "Mnemonics training: multi-class incremental learning without forgetting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12245–12254.

[37] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "Icarl: incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[38] L. Shu, H. Xu, and B. Liu, "Unseen class discovery in open-world classification," *ArXiv E-Prints*, vol. 1801, p. arXiv:1801.05609, Jan. 2018.

[39] J. Leo and J. Kalita, "Moving towards open set incremental learning: readily discovering new authors," *ArXiv191012944 Cs Stat*, Oct. 2019, Accessed: Aug. 23, 2020. [Online]. Available: http://arxiv.org/abs/1910.12944

[40] A. Garg and D. Roth, "Margin distribution and learning," in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, 2003, pp. 210–217.

[41] F. Aiolli, G. Da San Martino, and A. Sperduti, "A kernel method for the optimization of the margin distribution," in *International Conference on Artificial Neural Networks*, 2008, pp. 305–314.

[42] E. Vignotto and S. Engelke, "Extreme value theory for open set classification - GPD and GEV Classifiers," *ArXiv180809902 Cs Stat*, Aug. 2018, Accessed: Dec. 16, 2018. [Online]. Available: http://arxiv.org/abs/1808.09902

[43] W. J. Scheirer, "Extreme value theory-based methods for visual recognition," in *Synthesis Lectures on Computer Vision*, vol. 7, Morgan & Claypool Publishers, 2017, pp. 1–131.

[44] E. M. Rudd, L. P. Jain, W. J. Scheirer, and T. E. Boult, "The Extreme value machine," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 762–768, Mar. 2018, doi: 10.1109/TPAMI.2017.2707495.

[45] J. B. Broadwater and R. Chellappa, "Adaptive Threshold Estimation via Extreme Value Theory," *IEEE Trans. Signal Process.*, vol. 58, no. 2, pp. 490–500, Feb. 2010, doi: 10.1109/TSP.2009.2031285.

[46] H. Lee and S. J. Roberts, "On-line novelty detection using the kalman filter and extreme value theory," in *2008 19th International Conference on Pattern Recognition*, 2008, pp. 1–4.

[47] S. Luca *et al.*, “Detecting rare events using extreme value statistics applied to epileptic convulsions in children,” *Artif. Intell. Med.*, vol. 60, no. 2, pp. 89–96, Feb. 2014, doi: 10.1016/j.artmed.2013.11.007.

[48] A. Bendale and T. Boult, “Reliable posterior probability estimation for streaming face recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 56–63.

[49] Y. Shu, Y. Shi, Y. Wang, Y. Zou, Q. Yuan, and Y. Tian, “ODN: Opening the Deep Network for Open-Set Action Recognition,” in *2018 IEEE International Conference on Multimedia and Expo (ICME)*, Jul. 2018, pp. 1–6. doi: 10.1109/ICME.2018.8486601.

[50] A. Bendale and T. E. Boult, “Towards open set deep networks,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 1563–1572. doi: 10.1109/CVPR.2016.173.

[51] Z. Ge, S. Demyanov, Z. Chen, and R. Garnavi, “Generative OpenMax for Multi-Class Open Set Classification,” *ArXiv170707418 Cs*, Jul. 2017, Accessed: Nov. 16, 2021. [Online]. Available: http://arxiv.org/abs/1707.07418

[52] S. Marsland, *Machine Learning : An Algorithmic Perspective, Second Edition*. Chapman and Hall/CRC, 2014. doi: 10.1201/b17476.

[53] F. Camastra and A. Vinciarelli, *Machine Learning for Audio, Image and Video Analysis*. London: Springer London, 2015. doi: 10.1007/978-1-4471-6735-8.

[54] F.-R. Stöter, A. Liutkus, and N. Ito, “The 2018 signal separation evaluation campaign,” *ArXiv180406267 Cs Eess*, Apr. 2018, Accessed: Sep. 19, 2019. [Online]. Available: http://arxiv.org/abs/1804.06267

[55] J. S. Downie, “The music information retrieval evaluation exchange (MIREX),” *-Lib Mag.*, vol. 12, no. 12, Dec. 2006, doi: 10.1045/december2006-downie.

[56] J. Barker, E. Vincent, N. Ma, H. Christensen, and P. Green, “The PASCAL CHiME speech separation and recognition challenge,” *Comput. Speech Lang.*, vol. 27, no. 3, pp. 621–633, 2013.

[57] A. Mesaros, T. Heittola, and T. Virtanen, “TUT database for acoustic scene classification and sound event detection,” in *Signal Processing Conference (EUSIPCO), 2016 24th European*, 2016, pp. 1128–1132. Accessed: Jan. 17, 2017. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/7760424/

[58] X.-Y. Zhang, C.-L. Liu, and C. Y. Suen, “Towards robust pattern recognition: a review,” *Proc. IEEE*, vol. 108, no. 6, pp. 894–922, Jun. 2020, doi: 10.1109/JPROC.2020.2989782.

[59] S. Krstulović, “Audio event recognition in the smart home,” in *Computational Analysis of Sound Scenes and Events*, T. Virtanen, M. D. Plumbley, and D. Ellis, Eds. Cham: Springer International Publishing, 2018, pp. 335–371. doi: 10.1007/978-3-319-63450-0_12.

[60] D. R. F. Irvine, “Auditory perceptual learning and changes in the conceptualization of auditory cortex,” *Hear. Res.*, vol. 366, pp. 3–16, Sep. 2018, doi: 10.1016/j.heares.2018.03.011.

[61] T. Nowotny, “Two Challenges of Correct Validation in Pattern Recognition,” *Front. Robot. AI*, vol. 1, p. 5, 2014, doi: 10.3389/frobt.2014.00005.

[62] M. N. Murty and V. S. Devi, *Introduction to pattern recognition and machine learning*. New Jersey: IISc Press/World Scientific, 2015.

[63] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning*, 2nd ed., vol. 1. Springer series in statistics New York, 2009.

[64] G. Forman and M. Scholz, “Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement,” *ACM SIGKDD Explor. Newsl.*, vol. 12, no. 1, pp. 49–57, 2010.

[65] H. Jleed and S. Agaian, “Prediction of coding region in the DNA sequences,” in *2011 IEEE International Conference on Systems, Man, and Cybernetics*, Oct. 2011, pp. 1128–1133. doi: 10.1109/ICSMC.2011.6083826.

[66] A. Mesaros, T. Heittola, and T. Virtanen, “Metrics for Polyphonic Sound Event Detection,” *Appl. Sci.*, vol. 6, no. 6, Art. no. 6, Jun. 2016, doi: 10.3390/app6060162.

[67] N. Japkowicz, *Evaluating Learning Algorithms: a classification perspective*. Cambridge: University Press, 2011.

[68] D. Chicco and G. Jurman, “The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation,” *BMC Genomics*, vol. 21, no. 1, p. 6, Jan. 2020, doi: 10.1186/s12864-019-6413-7.

[69] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, Jul. 2009, doi: 10.1016/j.ipm.2009.03.002.

[70] H. Wechsler, *Reliable Face Recognition Methods*. Boston, MA: Springer US, 2007. doi: 10.1007/978-0-387-38464-1.

[71] M. Scherreik and B. Rigling, “An evaluation of open set recognition for FLIR images,” Baltimore, Maryland, United States, May 2015, p. 94760N. doi: 10.1117/12.2176585.

[72] H. Kadri, Z. Lachiri, and N. Ellouze, “Speaker change detection method evaluated on arabic speech corpus,” 2006.

[73] M. Grandini, E. Bagli, and G. Visani, “Metrics for Multi-Class Classification: an Overview,” *ArXiv200805756 Cs Stat*, Aug. 2020, Accessed: Dec. 03, 2021. [Online]. Available: http://arxiv.org/abs/2008.05756

[74] T. Giannakopoulos and A. Pikrakis, *Introduction to audio analysis: a MATLAB approach*, First edition. Kidlington, Oxford: Academic Press is an imprint of Elsevier, 2014.

[75] “Micro and Macro Averages for imbalance multiclass classification,” *knowledge Transfer*, Jul. 10, 2021. https://androidkt.com/micro-macro-averages-for-imbalance-multiclass-classification/ (accessed Dec. 03, 2021).

[76] D. Hand and P. Christen, “A note on using the F-measure for evaluating record linkage algorithms,” *Stat. Comput.*, vol. 28, no. 3, pp. 539–547, 2018.

[77] A. Temko, C. Nadeu, D. Macho, R. Malkin, C. Zieger, and M. Omologo, “Acoustic event detection and classification,” in *Computers in the human interaction loop*, London, 2009, pp. 61–73. doi: 10.1007/978-1-84882-054-8_7.

[78] L. Fan, “Audio Example Recognition and Retrieval Based on Geometric Incremental Learning Support Vector Machine System,” *IEEE Access*, vol. 8, pp. 78630–78638, 2020, doi: 10.1109/ACCESS.2020.2988686.

[79] L. Ptacek, L. Machlica, P. Linhart, P. Jaska, and L. Muller, “Automatic recognition of bird individuals on an open set using as-is recordings,” *Bioacoustics*, vol. 25, no. 1, pp. 55–73, Jan. 2016, doi: 10.1080/09524622.2015.1089524.

[80] T. Virtanen, *Techniques for noise robustness in automatic speech recognition*. Chichester, West Sussex, U.K. ; Wiley, 2012.

[81] S. Theodoridis and K. Koutroumbas, *Pattern recognition*, 4. ed. Amsterdam: Elsevier Acad. Press, 2009.

[82] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational Analysis of Sound Scenes and Events*. Springer, 2018.

[83] I. V. McLoughlin, *Speech and audio processing: a MATLAB-based approach*. Cambridge University Press, 2016.

[84] A. V. Oppenheim and R. W. Schafer, *Discrete-time signal processing*, 3rd ed. Prentice Hall, 2010.

[85] J. Xin and Y. Qi, *Mathematical Modeling and Signal Processing in Speech and Hearing Sciences*, vol. 10. Cham: Springer Science & Business Media, 2014.

[86] D. Mitrović, M. Zeppelzauer, and C. Breiteneder, "Features for Content-Based Audio Retrieval," in *Advances in Computers*, vol. 78, Elsevier, 2010, pp. 71–150. doi: 10.1016/S0065-2458(10)78003-7.

[87] Y. Shao, Z. Jin, D. Wang, and S. Srinivasan, "An auditory-based feature for robust speech recognition," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, Apr. 2009, pp. 4625–4628. doi: 10.1109/ICASSP.2009.4960661.

[88] J.-M. Liu *et al.*, "Cough signal recognition with Gammatone Cepstral Coefficients," Jul. 2013, pp. 160–164. doi: 10.1109/ChinaSIP.2013.6625319.

[89] X. Valero and F. Alias, "Gammatone Cepstral Coefficients: Biologically Inspired Features for Non-Speech Audio Classification," *IEEE Trans. Multimed.*, vol. 14, no. 6, pp. 1684–1689, Dec. 2012, doi: 10.1109/TMM.2012.2199972.

[90] R. Zhang and D. N. Metaxas, "RO-SVM: support vector machine with reject option for image categorization.," in *BMVC*, 2006, pp. 1209–1218.

[91] C. Chow, "On optimum recognition error and reject tradeoff," *Inf. Theory IEEE Trans. On*, vol. 16, no. 1, pp. 41–46, 1970, doi: 10.1109/TIT.1970.1054406.

[92] A. R. Webb, *Statistical pattern recognition*, 2nd ed. West Sussex, England ; New Jersey: Wiley, 2002.

[93] V. Tiwari, "MFCC and its applications in speaker recognition," *Int. J. Emerg. Technol.*, vol. 1, no. 1, pp. 19–20, 2010.

[94] A. Diment, T. Heittola, and T. Virtanen, "Sound event detection for office live and office synthetic AASP challenge," *Proc IEEE AASP Chall. Detect. Classif Acoust Scenes Events WASPAA*, 2013, Accessed: Nov. 11, 2016. [Online]. Available: http://c4dm.eecs.qmul.ac.uk/sceneseventschallenge/abstracts/OL/DHV.pdf

[95] F. Font, G. Roma, and X. Serra, "Freesound technical demo," in *Proceedings of the 21st ACM international conference on Multimedia*, 2013, pp. 411–412.

[96] J. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Adv. Large Margin Classif.*, vol. 10, no. 3, pp. 61–74, 1999.

[97] F. Li and H. Wechsler, "Open set face recognition using transduction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 11, pp. 1686–1697, 2005.

[98] K. Proedrou, I. Nouretdinov, V. Vovk, and A. Gammerman, "Transductive confidence machines for pattern recognition," in *European Conference on Machine Learning*, 2002, pp. 381–390.

[99] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, Apr. 2011, doi: 10.1145/1961189.1961199.

[100] W. J. Scheirer, "Open set recognition." https://www.wjscheirer.com/projects/openset-recognition/ (accessed Dec. 23, 2019).

[101] R. Polikar, L. Upda, S. S. Upda, and V. Honavar, "Learn++: an incremental learning algorithm for supervised neural networks," *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 31, no. 4, pp. 497–508, Nov. 2001, doi: 10.1109/5326.983933.

[102] D. M. J. Tax and P. Laskov, "Online SVM learning: from classification to data description and back," in *2003 IEEE XIII Workshop on Neural Networks for Signal Processing (IEEE Cat. No.03TH8718)*, Sep. 2003, pp. 499–508. doi: 10.1109/NNSP.2003.1318049.

[103] S. Huang, R. Jin, and Z.-H. Zhou, "Active learning by querying informative and representative examples," in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 892–900. Accessed: Sep. 13, 2020. [Online]. Available: http://papers.nips.cc/paper/4176-active-learning-by-querying-informative-and-representative-examples.pdf

[104] Y. Yan, F. Nie, W. Li, C. Gao, Y. Yang, and D. Xu, "Image classification by cross-media active learning with privileged information," *IEEE Trans. Multimed.*, vol. 18, no. 12, pp. 2494–2502, Dec. 2016, doi: 10.1109/TMM.2016.2602938.

[105] T. Diethe and M. Girolami, "Online Learning with (Multiple) Kernels: A Review," *Neural Comput.*, vol. 25, no. 3, pp. 567–625, Mar. 2013, doi: 10.1162/NECO_a_00406.

[106] R. Fletcher, *Practical methods of optimization*. John Wiley & Sons, 2013.

[107] S. Abe, *Support vector machines for pattern classification*, 2nd ed. London ; New York: Springer, 2010.

[108] H. Jleed, "Incremental-Multiclass-Open-set-Audio-Recognition," *GitHub*, 2021. https://github.com/hjleed/Incremental-Multiclass-Open-set-Audio-Recognition (accessed Oct. 27, 2021).

[109] H. Jleed and M. Bouchard, "Open set audio recognition for multi-class classification with rejection," *IEEE Access*, vol. 8, pp. 146523–146534, 2020, doi: 10.1109/ACCESS.2020.3015227.

[110] P. R. Mendes Júnior *et al.*, "Nearest neighbors distance ratio open-set classifier," *Mach. Learn.*, vol. 106, no. 3, pp. 359–386, Mar. 2017, doi: 10.1007/s10994-016-5610-8.

[111] S. Dang, Z. Cao, Z. Cui, Y. Pi, and N. Liu, "Open set incremental learning for automatic target recognition," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 4445–4456, Jul. 2019, doi: 10.1109/TGRS.2019.2891266.

[112] D. F. Williamson, R. A. Parker, and J. S. Kendrick, "The box plot: a simple visual method to interpret data," *Ann. Intern. Med.*, vol. 110, no. 11, pp. 916–921, 1989.

[113] V. B. S. Prasath *et al.*, "Distance and Similarity Measures Effect on the Performance of K-Nearest Neighbor Classifier -- A Review," *Big Data*, vol. 7, no. 4, pp. 221–248, Dec. 2019, doi: 10.1089/big.2018.0175.

[114] P. Mulak and N. Talhar, "Analysis of Distance Measures Using K-Nearest Neighbor Algorithm on KDD Dataset," vol. 4, no. 7, p. 5, 2013.

[115] K. Yu, L. Ji, and X. Zhang, "Kernel Nearest-Neighbor Algorithm," *Neural Process. Lett.*, vol. 15, no. 2, pp. 147–156, Apr. 2002, doi: 10.1023/A:1015244902967.

[116] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[117] C. Geng, S. Huang, and S. Chen, "Recent Advances in Open Set Recognition: A Survey," *ArXiv181108581 Cs Stat*, Jul. 2019, Accessed: Nov. 20, 2019. [Online]. Available: http://arxiv.org/abs/1811.08581

[118] M. Gunther, S. Cruz, E. M. Rudd, and T. E. Boult, "Toward open-set face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 71–80.

[119] R. A. Fisher and L. H. C. Tippett, "Limiting forms of the frequency distribution of the largest or smallest member of a sample," *Math. Proc. Camb. Philos. Soc.*, vol. 24, no. 2, pp. 180–190, Apr. 1928, doi: 10.1017/S0305004100015681.

[120] Vastlab, *Vastlab/libMR*. Vision and Security Technology Lab, 2021. Accessed: Jan. 20, 2022. [Online]. Available: https://github.com/Vastlab/libMR

[121] W. J. Scheirer, A. Rocha, R. J. Micheals, and T. E. Boult, "Meta-Recognition: The Theory and Practice of Recognition Score Analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1689–1695, Aug. 2011, doi: 10.1109/TPAMI.2011.54.

[122] C. Bravo, J. L. Lobato, R. Weber, and G. L'Huillier, "A Hybrid System for Probability Estimation in Multiclass Problems Combining SVMs and Neural Networks," in *2008 Eighth International Conference on Hybrid Intelligent Systems*, Sep. 2008, pp. 649–654. doi: 10.1109/HIS.2008.112.

[123] T.-K. Huang, R. C. Weng, and C.-J. Lin, "Generalized Bradley-Terry Models and Multi-Class Probability Estimates," p. 31.

[124] B. Zadrozny and C. Elkan, “Transforming Classifier Scores into Accurate Multiclass Probability Estimates,” p. 6.

[125] H. Jleed, “Open-Set-Audio-Recognition-for-Multi-class-Classification-with-Rejection/FeatureExtraction.m at master · hjleed/Open-Set-Audio-Recognition-for-Multi-class-Classification-with-Rejection,” *GitHub*, 2020. https://github.com/hjleed/Open-Set-Audio-Recognition-for-Multi-class-Classification-with-Rejection (accessed Nov. 15, 2021).

[126] E. Vignotto and S. Engelke, “Extreme value theory for anomaly detection – the GPD classifier,” *Extremes*, vol. 23, no. 4, pp. 501–520, Dec. 2020, doi: 10.1007/s10687-020-00393-0.

[127] EMRResearch, *ExtremeValueMachine*. 2020. Accessed: Oct. 16, 2020. [Online]. Available: https://github.com/EMRResearch/ExtremeValueMachine

[128] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[129] H. Meng, T. Yan, F. Yuan, and H. Wei, “Speech Emotion Recognition From 3D Log-Mel Spectrograms With Deep Learning Network,” *IEEE Access*, vol. 7, pp. 125868–125881, 2019, doi: 10.1109/ACCESS.2019.2938007.

[130] J. Beirlant, Ed., *Statistics of extremes: theory and applications*, Reprinted. Chichester: Wiley, 2005.

[131] W. Shao, W. Yang, and G.-S. Xia, “Extreme value theory-based calibration for the fusion of multiple features in high-resolution satellite scene classification,” *Int. J. Remote Sens.*, vol. 34, no. 23, pp. 8588–8602, 2013.

[132] I. F. Alves and C. Neves, “Extreme Value Distributions,” in *International Encyclopedia of Statistical Science*, M. Lovric, Ed. Berlin, Heidelberg: Springer, 2011, pp. 493–496. doi: 10.1007/978-3-642-04898-2_246.

[133] W. Scheirer, A. Rocha, R. Micheals, and T. Boult, “Robust fusion: extreme value theory for recognition score normalization,” in *European Conference on Computer Vision*, 2010, pp. 481–495.

[134] H. J. Jung and M. Lease, “Improving consensus accuracy via Z-score and weighted voting,” in *In Proceedings of the 3rd Human Computation Workshop (HCOMP) at AAAI*, 2011, pp. 88–90.

[135] S. Kundu and S. Ari, “Score normalization of ensemble SVMs for brain-computer interface P300 speller,” in *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Delhi, Jul. 2017, pp. 1–5. doi: 10.1109/ICCCNT.2017.8204119.

[136] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. USA: Prentice Hall PTR, 1998.

[137] U. Michelucci, *Advanced Applied Deep Learning: Convolutional Neural Networks and Object Detection*. Berkeley, CA: Apress, 2019. doi: 10.1007/978-1-4842-4976-5.