2021

Machine Learning for Auditory Hierarchy

William Coleman
Technological University Dublin, d15126149@mytudublin.ie

Machine Learning for Auditory Hierarchy

William Coleman

Supervisors: Dr. Charlie Cullen

Prof. Sarah Jane Delany

Dr. Ming Yan

School of Electrical and Electronic Engineering

Technological University Dublin

This dissertation is submitted for the degree of

Doctor of Philosophy (Engineering)

November 2021

Abstract

Audio content is today consumed in a plethora of ways. This may be on stereo headphones, via a home cinema system, in the car or on a smart speaker. The format used to deliver the content may be an MP3 or WAV, FLAC, AIFF, OGG, or any number of various other video and audio streaming options. The content may be a game, music or drama & current affairs broadcasting.

Audio content is predominantly delivered in a stereo audio file of a static, pre-formed mix. The content creator makes volume, position and effects decisions, generally for presentation in stereo speakers, but has no control ultimately over how the content will be consumed. This leads to poor listener experience when, for example, a feature film is mixed such that the dialogue is at a low level relative to the sound effects. Consumers can complain that they must turn the volume up to hear the words, but back down again because the effects levels are too loud. Addressing this problem requires a television mix optimised for the stereo speakers used in the vast majority of homes, which is not always available.

The concept of object-based audio envisages content delivery not via a fixed mix, but as a series of auditory objects which can be flexibly controlled individually. This method would increase the flexibility available to creators such that they could design sound mixes for multiple consumption paradigms. A package of audio content could then come provided with a menu of mix configurations, giving consumers the option of choosing which to use. Object-based audio could also be used to automate content decisions in an informed manner for different scenarios. If a television mix is required for a film where none is available,

a model could be applied to automate an appropriate mix which balances dialogue and effects levels. If it became necessary to reduce the amount of data transmitted, variable compression could be applied to objects, selectively reducing data file sizes. In this way, the most important objects could be reproduced at highest quality with no file compression. Those less critical could be rendered at lower quality, having been heavily compressed. From these examples it follows that an ability to predict the importance of auditory objects would be useful as it would permit the selective treatment of assets for both creative and delivery strategies.

This thesis provides a research roadmap for a machine learning investigation of auditory hierarchy, and thus serves two communities. For those from a machine learning background, it introduces perceptual auditory theory and gives insight into how humans perceive sound. For those from an audio background, it provides insight into common machine learning methods and best practices. To begin, perceptual audio research is reviewed and a theory of auditory hierarchy is offered, which outlines factors relevant to hierarchical classification in the context of modern media consumption paradigms. A review of audio machine learning research is then presented, which frames hierarchical prediction as a problem complicated by the subjective nature of the labelling task, distinct from other prediction problems such as environmental sound classification where correct sound identification results in an objective label. The nature of auditory hierarchy is then explored via a number of experiments. The machine learning techniques employed are exploratory and provide insight into the performance of common methods. This is with the intention of illuminating a problem area which to date has not received widespread interest from the machine learning community. It is hoped that the experiments described in this work will thus inform further applications of machine learning methods to auditory hierarchy.

The first experiment described in this work is a perceptual labelling task, which investigates the inherent sound hierarchy between a small corpus of isolated sounds. A subsequent

machine learning analysis produces promising results, achieving a foreground recall score of 93.3%, but the size of dataset used is noted as an issue, highlighting the requirement for a larger dataset of hierarchically labelled sounds. For this reason, Active Learning methods for minimising the manual effort required to label large numbers of experimental stimuli are investigated. It is found that labels can be predicted to high degrees of accuracy (95.5% of the total possible) by selecting just a small percentage (1.7%) of the most informative instances. This method is then used in tandem with data augmentations to build a corpus of 100,000 instances with hierarchical labels. The performance of Support Vector Machine (SVM) and Convolutional Neural Network (CNN) algorithms on a sound hierarchy prediction task using different feature representations is then presented.

It is found in this case that performance of the CNN is superior (82.2% average class accuracy), but it is noted that this is not greatly superior to that of an SVM (77.5%) trained on a smaller dataset. This is an interesting result, as it suggests that the manual effort required to label datasets large enough for deep learning algorithms may not be justified for every application.

Declaration

I certify that this thesis which I now submit for examination for the award of Doctor of Philosophy, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work.

This thesis was prepared according to the regulations for graduate study by research of the Technological University Dublin and has not been submitted in whole or in part for another award in any other third level institution.

The work reported on in this thesis conforms to the principles and requirements of the TUD's guidelines for ethics in research.

TUD has permission to keep, lend or copy this thesis in whole or in part, on condition that any such use of the material of the thesis be duly acknowledged.

Candidate Signature: _____

Date: _____

William Coleman

November 2021

Acknowledgements

This thesis would not have been possible without the love, support and time graciously afforded by an extensive network of supervisors, colleagues, friends and family.

Firstly, to my supervisors, Dr. Charlie Cullen, Professor Sarah Jane Delany and Dr. Ming Yan. In various guises holders of my sanity for the duration of the project, I found your insight and input formative to both my work and to the way that I aspire to work. Thank you for your extensive feedback, good advice and endless patience, particularly in the face of numerous emails that turned into mini chapters!

I would also like to thank both my industrial sponsor for this project, Xperi, and the Irish Research Council, who funded my research for three years. In particular, I would like to thank Dr. Ton Kalker of Xperi for his willingness to go above and beyond the call of duty. It is greatly appreciated.

To all my TUD PhD office colleagues of Room 2006 in Aungier Street: Linda Adams, Cliodhna Pierce, Naoise Collins, Carolina De Pascuale, Eoghain Meakin and all the others who also passed through: salutations! You were a constant source of good cheer, emotional support, and PhD navigation advice! Thanks for all your help and wishing you all the very best now and in the future.

I am also fortunate to have the support of a large collection of family & friends, who have helped by being generally wonderful people with whom I am fortunate to be associated. Numerous chats have been had, perspectives shared and musical moments experienced. I thank you all for each and every one of them.

To my parents, John and Mary. Thank you for everything you've done for me. You've made me who I am.

And finally, to my wife, Orla, and daughter, Edie. For persisting, and joining us in this world, thereby enriching it and helping me be the best version of me. Thank you.

Table of contents

List of figures **xvii**

List of tables **xxi**

Nomenclature **xxv**

1 Introduction **1**

1.1 Motivation 1

1.2 Thesis Objectives, Research Questions and Contributions 4

1.3 Document Structure 7

1.4 Publications 8

2 Predicting Auditory Hierarchy: A Roadmap **11**

2.1 Introduction 11

2.2 Auditory Scene Analysis 12

2.2.1 Listening Modes 13

2.3 Object-based Audio and Modern Media Consumption Paradigms 17

2.4 Sound Datasets 20

2.4.1 Sound Taxonomies 20

2.4.2 Stimuli Selection 25

2.4.3 Existing Sound Stimuli Corpora 30

2.5 Perceptual Testing and Audio Standards 33
2.5.1 Listening Test Standards 34
2.5.2 Bias in Perceptual Testing 37
2.5.3 Listening Test Implementation 38
2.6 A Map of Auditory Hierarchy 43
2.7 Conclusion 51

3 Audio Machine Learning **55**
3.1 Introduction 55
3.2 Overview of Machine Learning 57
3.3 Machine Learning Methodology 59
3.3.1 Feature Representation 59
3.3.2 Building Supervised Machine Learning Models 61
3.3.3 Algorithm Choice 64
3.3.4 Feature Extraction 65
3.3.5 Feature Selection 68
3.3.6 Model Evaluation 70
3.4 Audio Features for Machine Learning 71
3.4.1 Time Domain Features 71
3.4.2 Frequency Domain Features 73
3.4.3 Cepstral Domain Features 74
3.4.4 Visual and Other Features 75
3.4.5 Summary of Feature Types 77
3.5 Algorithms for Audio Prediction 79
3.5.1 Information-based Learning 79
3.5.2 Similarity-based Learning 81
3.5.3 Probability-based Learning 82

3.5.4 Error-based Learning 84
3.5.5 Deep Learning Algorithms 86
3.6 Active Learning 94
3.6.1 Selection Methods 95
3.6.2 Active Learning in the Auditory Domain 97
3.6.3 Active Learning Summary 99
3.7 Data Augmentation 100
3.8 Conclusion 103

4 Perceiving and Predicting Auditory Hierarchy 107
4.1 Introduction 107
4.2 Perception of Auditory Hierarchy in Isolated Sounds 108
4.2.1 Methodology 108
4.2.2 Results 113
4.2.3 Discussion 121
4.2.4 Conclusions 122
4.3 Predicting Auditory Hierarchy 123
4.3.1 Methodology 124
4.3.2 Results 130
4.3.3 Discussion 131
4.4 Conclusions 133

5 Active Learning for Auditory Hierarchy 135
5.1 Introduction 135
5.2 Active Learning 135
5.3 Methodology 137
5.3.1 Dataset Creation 137

5.3.2 Feature Extraction 142
5.3.3 Algorithm Selection 143
5.3.4 Performance Measures 144
5.3.5 Cross Validation Experiment 144
5.3.6 Active Learning Process 146
5.4 Results 146
5.5 Discussion 151
5.6 Conclusion 153

6 Deep Learning for Auditory Hierarchy 155
6.1 Introduction 155
6.2 Active Learning Experiment 156
6.2.1 Methodology 157
6.2.2 Results 159
6.2.3 Discussion 161
6.3 Category Threshold Experiment 163
6.3.1 Methodology 163
6.3.2 Results and Discussion 165
6.4 Deep Learning Experiment 168
6.4.1 Methodology 169
6.4.2 Results and Discussion 177
6.5 Conclusions 181

7 Machine Learning Methods Applied to Auditory Hierarchy 185
7.1 Summary of Research Questions 189
7.1.1 RQ1: What factors are involved in the perception of Auditory Hierarchy? 189

7.1.2 RQ2: Does a hierarchy of importance exist between sounds isolated from context? . . . 191
7.1.3 RQ3: Is it possible to accurately predict AH using supervised ML methods? . . . 192
7.2 Future Work . . . 198

References **203**

Appendix A Computer Code **231**
A.1 Experiment 1 . . . 231
A.1.1 Experiment 1 - R Data Exploration Code . . . 231
A.2 Experiment 2 . . . 241
A.2.1 Experiment 2 - Random Forest Python Code . . . 241
A.3 Experiment 3 . . . 287
A.3.1 Experiment 3 - EGAL Python Code . . . 287
A.4 Experiment 4 . . . 299
A.4.1 Experiment 4 - SVM Python Code . . . 299
A.4.2 Experiment 4 - CNN Python Code . . . 307

Appendix B Model Evaluation **321**

Appendix C List of Publications **327**
C.1 Journal Papers . . . 327
C.2 Conference Papers . . . 327
C.3 Other Papers . . . 328

Appendix D List of Employability and Discipline Specific Skills **329**

List of figures

2.1 The Gestalt dog on a beach image where the outline of a dog is suggested by the alignment of Dalmation spots. Subjects typically fail to perceive the dog if the picture is first presented to them upside down, but quickly form the dog percept once the picture is presented as above. 15

2.2 A representation of the progression from background to foreground listening as a continuum, and how this relates to soundscape/semantic listening and auditory perception in general. 16

2.3 A reproduction of Gavers' taxonomy of everyday sounds, delineated by classes of materials and by interactions which may cause them to sound. . . 21

2.4 Categories of sounds used for the World Soundscape Project. 21

2.5 A taxonomy of the acoustic environment for soundscape studies as presented by Brown *et al.* [1]. 22

2.6 An urban soundscape taxonomy as developed by Raimbault and Dubois [2]. 23

2.7 The top-level categories of the ontology used by Gemmeke *et al.* [3] to organise over 2 million sound stimuli curated from YouTube. 24

2.8 A framework outlining factors which influence subjective hierarchical ranking of sound objects derived from the literature review outlined in Chapter 2 which shall be used to guide experimental design. These are components around which audio object hierarchy is hypothesised to vary. 47

2.9 A conceptualisation of temporal variance in auditory object hierarchy. . . . 50

3.1 An illustration of the data pipeline from unstructured data to prediction for supervised ML. 60

3.2 An illustration of the supervised ML process. 62

3.3 A representation of 5-fold Cross Validation, which provides 5 different training and test splits, each of which are used to build a model. 63

3.4 A cross validation split implemented to fit parameters. 63

3.5 A visual representation of a neuron showing input (a_{1-N}), weight (w_{1-N}), bias, and activation function (g) elements (reproduced from [4]). Here, z is the result of adding the bias term to the sum of the products of inputs, a, and weights, w. 88

3.6 A visual representation of a NN consisting of layers of neurons (adapted from [5]). 89

3.7 An example of CNN architecture depicting the VGG16 model proposed by [6]. The image was sourced from *https://neurohive.io/en/popular-networks/vgg16/*, Accessed: 5th December, 2019. 91

3.8 An outline of the Active Learning process. The purpose of the selection method is to select unlabelled instances that will be most informative of the dataset. Once labelled by a human annotator, a model can then be trained on the instances in the labelled pool to predict labels for remaining unlabelled instances. Selection methods are outlined in Section 3.6.1. 94

4.1 Methodology overview for Experiment 1. 108

4.2 The test environment. 111

4.3 Scatterplots of BG, N and FG counts. The categories in this plot are derived from the median ratings noted in Experiment 1. A strong linear relationship is noted between BG and FG ratings. 116

4.4 Relationship between mean sound score and standard deviation separated by class. Sounds ranked more FG are to the right. Those considered more BG are to the left. Sounds with a smaller standard deviation (closer to the bottom of the plot) indicate that there was more consensus between subjects as to category in these instances. There is no clear categorisation pattern by sound class. 117

4.5 The relationship between mean sound rating and gender. Once again, sounds considered FG are towards the right of the plot, BG sounds are to the left. . 119

4.6 A comparison of sound ranking standard deviation by gender. Sounds considered FG are towards the right of the plot, BG sounds are to the left. . . . 120

4.7 Methodology overview for Experiment 2. 125

5.1 Methodology overview for Experiment 3. 138

5.2 Boxplots outlining the variance in average sound ratings grouped in broad bands for the 3,002 sounds for which at least 3 ratings were gathered. Note that the minimum average score for BG sounds is 1, hence there is no quartile or minimum whisker below this value. Similarly, the maximum average score for FG sounds is 3, hence this band has no quartile or maximum whisker above this value. Also, the width of each boxplot is proportional to the number of instances summarised in each band. 141

5.3 Comparison of Active Learning selection methods displaying balanced accuracy (ACA) scores achieved from 10 - 2,501 labels. Each line denotes the overall average score for each method per batch. The shaded area denotes the variance observed from the random selection method. 148

5.4 Comparison of selection methods for early stage (between 0 and 500 labels) Active Learning runs. Each line denotes the overall average score for each method per batch. The shaded area denotes the variance observed from the random selection method. 149

6.1 Methodology overview for Experiment 4. 156

6.2 A comparison of scores noted for different thresholds. Note that 'FG P' denotes FG Precision, 'FG R' FG Recall etc. 166

6.3 The process and dataset details for comparing the performance of SVMs trained using all manual and a mixture of manual and predicted labels. . . . 172

B.1 An example of a confusion matrix for a binary classifier. 322

List of tables

4.1 Summary results ordered by mean sound rating from top to bottom. Sounds ranked *More Background* are towards the top, while those *More Foreground* are towards the bottom. 114

4.2 A description of features extracted as objective measures of the sound stimuli. 126

4.3 The parameter grid used to find optimal hyperparameters for baseline models for the RF algorithm. 128

4.4 The parameter grid used to find optimal hyperparameters for baseline models for the SVM algorithm. 128

4.5 Summary results for baseline (BL) and optimised (OP) models. CA is the FG class accuracy rate (or FG recall rate). ACA is the Average Class Accuracy for both FG and 'nonFG' classes. 130

5.1 A summary of instance count, average score and standard deviation (σ) per class for all 3,002 sounds for which at least 3 ratings were gathered. The highest occurrences are reproduced in **bold**, the lowest are underlined. . . . 139

5.2 A summary of feature representation data vectors and their dimensions. For each representation (MFCC, chroma and LPMS) zero-order (ZO) and 1st (1OD), 2nd (2OD) and 5th order (5OD) delta vectors are computed, resulting in a total of 12 initial representations. The Dimensions column denotes the number of instances x number of frequency bins x temporal feature extraction frames for each feature representation. These vectors were flattened prior to input to SVM models for AL. 143

5.3 Default parameters used per kernel in the initial classification exercise. The 'scale' value for the gamma parameter uses $1/(no.features * variance)$ as value of gamma. 144

5.4 Average Class Accuracy (ACA), and Class Accuracy scores for FG and nonFG classes per kernel and feature representation. As noted, the 'All' representation is an amalgamation of the other 3. 145

5.5 A summary of model accuracy and AULC scores for points in the labelling run per AL method. Using Diversity EGAL it is possible to achieve high classification accuracy (74%), using the 50 most informative instances selected using this method. 150

6.1 Class distribution of the splits used in both Active Learning process and subsequent validation. 159

6.2 An outline of grid search parameters used. 'C' is the only parameter varied for the linear kernel. The 'degree' parameter is varied for the polynomial kernel only. 159

6.3 A comparison of accuracy scores achieved on the Validation Split in the current instance compared to results noted in Experiment 3. 160

6.4 A classification report outlining the accuracy of predicted versus manual labels for the Validation Instances. 160

6.5 A confusion matrix achieved by tuning the margin information provided by the SVM model. A margin value of -0.937 was used to classify instances. . 161
6.6 A frequency table showing the count per average rating for all 3,599 manually labelled instances. 164
6.7 Class distribution per threshold value. An instance is classified as FG for a particular threshold if its average rating value is greater than or equal to the threshold value. 164
6.8 Parameter settings for the DRC augmentations. In the following, 'ms' are milliseconds, 'dB' are Decibels, 'CSK' refers to 'Classic Soft Knee', 'SL12' to 'Soft Limit -12dB' and 'SL24' to 'Soft Limit -24db'. 170
6.9 A summary of feature representation pools used when building SVM and CNN models. 171
6.10 Train and test split sizes used to train SVM and CNN models. 173
6.11 An outline of the smaller parameter grid used for large data representations. 173
6.12 An example of CNN architecture applied in this research, in this instance the final configuration for the 'CNN 10k Zero Order' model. An initial architecture based on that described by Chen *et al.* [7] was implemented and adapted for each CNN outlined in Table 6.9. The notation '5 x 5 Conv2D(pad=2, stride=2) x 12 - BN - ReLU - DO(0.3)' denotes a 2D convolutional layer with 12 filters of size 5 x 5 followed by batch normalisation, ReLU activation function and Dropout where p=0.3. 175
6.13 An example of CNN architecture applied in this research, in this instance the final configuration for the 'CNN 100k Delta' model. The notation '5 x 5 Conv2D(pad=2, stride=2) x 12 - BN - ReLU - DO(0.2)' denotes a 2D convolutional layer with 12 filters of size 5 x 5 followed by batch normalisation, ReLU activation function and Dropout where p=0.2. 176

6.14 Composition of a contingency table based on the results of two models, A and B. 176

6.15 Summary of average ACA, precision and recall scores noted across three randomly selected hold-out test sets. Note that 'P' indicates Precision and 'R' indicates Recall in the following. 177

7.1 Summary of average ACA, precision and recall scores noted across three randomly selected hold-out test sets. Note that 'P' indicates Precision and 'R' indicates Recall in the following. 197

Nomenclature

Acronyms / Abbreviations

ACA Average Class Accuracy, page 70

AES Audio Engineering Society, page 34

AH Auditory Hierarchy, page 2

AL Active Learning, page 6

AL Active Learning, page 188

ASA Auditory Scene Analysis, page 2

AUC Area Under the Curve, page 70

AUC Area Under the Curve, page 324

AULC Area Under the Learning Curve, page 144

BAQ Basic Audio Quality, page 28

BG Background Sounds: For the purposes of this study, Background sounds are those considered less likely to be a focus of attention, page 7

BiLSTM Bi-directional Long Short Term Memory, page 92

CNN Convolutional Neural Network, page 56

Nomenclature

DCASE Detection and Classification of Acoustic Scenes and Events, page 3

DL Deep Learning, page 3

DNN Deep Neural Network, page 69

DRC Dynamic Range Compression, page 100

EBU European Broadcasting Union, page 34

EEG Electroencephalogram features. Images representing brain activity., page 76

EER Expected Error Reduction, page 96

EGAL Exploration Guided Active Learning, page 96

FG Foreground Sounds: For the purposes of this study, Foreground sounds are those considered likely to be a focus of attention, page 7

FIAH Factors Influencing Auditory Hierarchy, page 46

FN False Negative, page 321

FP False Positive, page 321

GMM Gaussian Mixture Model, page 83

GTCC Gammatone Cepstral Coefficients, page 76

HMM Hidden Markov Model, page 66

ISO International Standards Organisation, page 34

ITU International Telecommunications Union, page 34

JAES Journal of the Audio Engineering Society, page 34

JASA Journal of the Acoustical Society of America, page 34

kNN k-Nearest Neighbours, page 57

LLD Low Level Descriptor, page 71

LPMS Log-Power Mel Spectrogram, page 75

LSTM Long Short Term Memory, page 92

MFCC Mel Frequency Cepstral Coefficients, page 75

ML Machine Learning, page 3

MuSHRA Multiple Stimuli with Hidden Reference and Anchor, page 35

N Neutral Sounds: For the purposes of this study, Neutral sounds are all those not judged to be definitively FG or BG, page 7

NN Neural Network, page 65

ObA Object-based Audio conceptualises audio content as a collection of individual audio assets controlled by accompanying metadata, as opposed to a fixed mix of a cohesive sound scene., page 1

OBJ Research Objectives, page 4

PCA Principal Component Analysis, page 69

PHI Potential Hierarchical Indicators, page 46

PLP Perceptual Linear Prediction, page 76

QBC Query-by-Committee, page 96

QoE Quality of Experience, page 46

RASTA-PLP Relative Spectral-Perceptual Linear Prediction, page 77

ReLU Rectified Linear Unit, page 89

RF Random Forest, page 68

RMS Root Mean Square, page 71

RNN Recurrent Neural Network, page 90

ROC Receiver Operating Characteristic, page 324

STFT Short Time Fourier Transform, page 73

SVM Support Vector Machines, page 56

TN True Negative, page 321

TP True Positive, page 321

UA Unweighted Accuracy, page 98

USAL Uncertainty Sampling Active Learning, page 95

WSPTL World Soundscape Project Tape Library, page 14

ZCR Zero Crossing Rate, page 71

Chapter 1

Introduction

1.1 Motivation

Recent technological advances have driven changes in how media is consumed in home, automotive and mobile contexts. Multi-channel audio home cinema systems are not ubiquitous, but have become more prevalent. The consumption of broadcast and gaming content on smartphone and tablet technology via telecommunications networks is also more common. Research in object-based broadcasting [8, 9] and auditory object categorisation [10] has underlined a growing interest in the area. Object-based Audio (ObA), introduced in Section 2.3, may lead to new modes of content creation and consumption by providing audio on an object level with metadata which controls how the media is delivered depending on the consumption paradigm and other considerations.

Delivering audio content as a collection of objects, as opposed to a fixed stream, suggests new possibilities and consequently poses new challenges for audio content delivery. A stereo audio file is adequate for consumption in a mobile context using headphones, for example, but it is limited to stereo presentation in the context of a surround-sound home entertainment

system. Delivering this content as a collection of objects controlled by metadata allows the possibility for many mix configurations accompanying the raw audio to accommodate numerous consumption paradigms. The variability of telecommunications network bandwidths is another factor which constricts data transmission capacity for consumers 'on-the-move'. In this context, an ability to adapt audio content based on the importance of each object to perception of the auditory scene as a whole would allow file size optimisation based on end user experience in addition to network capacity. This could be achieved by varying the degree of compression applied to elements of the auditory scene, rendering the most important objects at highest quality.

These examples motivate the requirement for a perceptual understanding of which auditory objects are deemed important and how the relative importance of sounds may change with time. Any real-world implementation of these concepts would require a method of accurately predicting Auditory Hierarchy (AH) without human intervention which motivates investigation of models trained for this purpose.

There is a considerable body of research in the area of Auditory Scene Analysis (ASA) , the study of human sound perception. ASA involves a constant activity of sound categorisation which Bregman [11] outlines as both a conscious (schematic or *top-down*) and unconscious (primitive or *bottom-up*) process of soundscape perception. Guastavino [12] has noted converging evidence from both behavioural and neurophysiological domains that provides support for the notion that amalgamation of these processes is integrated, rather than serial. Thus, ASA can be considered as a constant analysis of the surrounding sound scene, subject to varying levels of influence from a number of external factors, which involves continual innate identification of interesting sounds which may then be consciously analysed for semantic information or further meaning, or not, as deemed necessary based on the interaction of these functions.

Machine Learning (ML) is another extremely active area of research both generally [13] and in audio terms [14]. There is a rich recent history in this area deriving from events such as the Detection and Classification of Acoustic Scenes and Events (DCASE) challenges [15, 16], which provide background to a variety of sound classification tasks. Performance in some ML domains has begun to approach and even surpass human accuracy levels, which suggests that an ML implementation can be successfully applied to automate classification of auditory objects on a hierarchical scale.

Given the multifaceted nature of hierarchical classification, not to mention the individual, subjective nature of auditory perception, it can be presumed that predicting a phenomenon such as AH would be a non-trivial task. This presents a number of practical challenges. While much perceptual work uses small numbers of stimuli to investigate aspects of ASA, ML research typically requires much larger datasets. Indeed, the lack of large datasets is a problem common to many domains of ML research [17, 18], particularly given the tendency of Deep Learning (DL) models to outperform others once supplied with sufficient data [19, 20, 21]. This motivates the formulation of a large corpus of hierarchically labelled sounds to provide an assessment of ML algorithm performance when predicting AH.

The material presented in this work constitutes a roadmap of research into the domain of AH and outlines a series of ML experiments on the subject. It is therefore of practical use for both the ML and audio research communities. For audio practitioners, it provides a summary of ML methods and best practices, which can serve as an introduction to the domain. For ML researchers, a grounding in auditory theory is provided, and an introduction to audio features for ML work is presented. To those interested in the problem of AH and how to predict it using ML, a series of methods and experiments are employed, and the findings may be used to inform further work in the domain.

1.2 Thesis Objectives, Research Questions and Contributions

It is clear that in order to fully understand the nature of AH and appreciate its application to modern media consumption patterns a thorough understanding of ASA will be required. Furthermore, accurately predicting AH will require an in-depth appreciation of ML as it applies to the auditory domain. In cognisance of this and drawing from the motivation of this thesis a number of Research Objectives (OBJ) were formulated to structure the research described in subsequent chapters. These are summarised as follows:

OBJ 1: To develop an understanding of ASA with particular attention to the concepts of object-based audio, AH and modern media consumption paradigms.

OBJ 2: Informed by perceptual audio research, to propose a machine learning approach for the task of predicting AH.

OBJ 3: To assess the performance of supervised ML algorithms when predicting AH.

Setting these OBJs has helped to define a number of Research Questions (RQ) to address specific issues raised by consideration of the objectives. A review of ASA concerns was formative in defining factors which influence AH, which were critical to decisions made for the ML analysis. In order to provide a basis for further investigations of these factors it was decided to use stimuli isolated from context, and this necessitated an investigation into the nature of the hierarchical relationship between such sounds. These initial questions framed the assessment of ML methods described. The RQs formulated are as follows:

RQ 1: What factors are involved in the perception of AH?

RQ 2: Does a hierarchy of importance exist between sounds isolated from context?

RQ 3: Is it possible to accurately predict AH using supervised ML methods?

The contributions of this thesis outline how AH can be effectively predicted in a set of isolated sounds. This knowledge can then be applied to media content delivery strategies to improve the user experience and the efficiency of content delivery. They are organised in major and minor contributions as follows:

Major Contributions

Maj. Contrib. 1: ***A roadmap for research into ML methods for AH.*** AH has received relatively little attention in terms of ML research. This work explores perceptual audio theory and applies a number of common ML methods to the domain and the findings are offered in the shape of a roadmap which can inform future research in the area.

Maj. Contrib. 2: ***A published working theory of AH.*** AH is theorised to vary due to the influence of factors such as the physical properties of sounds and individual biases. Sounds are proposed to be characterised hierarchically in terms of a number of indicators such as whether they indicate the presence of humans or not, whether the sound contains semantic information or not, and others.

Maj. Contrib. 3: ***Evidence of a hierarchy of importance between sounds isolated from context is presented.*** The understanding of AH is enhanced by conducting a perceptual experiment where the hierarchical relationship between sounds isolated from context is investigated.

Maj. Contrib. 4: ***Validation of the use of ML methods to predict AH with competitive performance. Average Class Accuracy of 82.2% is noted using a Convolutional Neural Network (CNN).*** A series of experiments are described which address the problem of hierarchical prediction in an audio context.

Performance comparable with other audio ML applications is noted using Random Forest (RF), Support Vector Machine (SVM) and CNN algorithms.

Maj. Contrib. 5: ***Applied to AH, the Exploration Guided Active Learning (EGAL) algorithm can be used to select a minimal number of labels (in this case 1.7% of the total) to achieve 95.5% of possible model accuracy, outperforming other selection methods.*** In an assessment of Active Learning (AL) selection methods, EGAL is found to be most effective in selecting informative instances to reduce manual labelling effort, outperforming Uncertainty Sampling Active Learning (USAL). Use of EGAL is more computationally efficient and less time consuming than USAL as it does not require a model to be trained at each iteration of the algorithm.

Minor Contributions

Min. Contrib. 1: ***In the context of AH, the Log Power Mel Spectrogram (LPMS) zero order feature representation is found to be an effective compromise for predicting AH, providing comparable performance to larger representations which are considerably more expensive in terms of computation time. Delta representations are found to provide performance improvement in some, but not all cases.*** A number of feature representations have been utilised in the course of this research. While it is noted that in certain cases superior performance is possible from larger data representations it is debatable as to whether the increase in performance is justified by the computation cost entailed.

Min. Contrib. 2: ***The development of a hierarchically labelled corpus of 10,000 sounds consisting of both manual and predicted labels.*** Future investigations

of AH are facilitated via the corpus developed during the experiments conducted for this thesis. To our knowledge, this corpus represents the largest audio database of hierarchically labelled audio instances.

1.3 Document Structure

Chapter 2 provides an introduction to ASA (in Section 2.2), and introduces the concept of ObA and discusses how this may impact on consumption of audio content in Section 2.3. An overview of existing sound taxonomies and datasets is offered in Section 2.4 and Section 2.5 covers relevant industry standards for perceptual testing to include listening test design and implementation. These discussions inform a map of AH outlined in Section 2.6 conceived to encapsulate theory around the functioning of AH and how this can be predicted using ML methods. This commentary is based on that published by the authors previously, see [22].

Chapter 3 firstly provides an overview of ML research as it pertains to audio hierarchy in Section 3.2. In Section 3.3, methodological concerns are outlined in the areas of feature representation, algorithm choice, feature extraction and selection in addition to how models are built and evaluated. Section 3.4 provides a review of feature representations commonly used in the audio ML domain. An overview of algorithms applied to audio ML tasks is offered in Section 3.5. Methods used to minimise manual effort in ML labelling tasks, such as AL and data augmentation are outlined in Sections 3.6 and 3.7.

Chapter 4 describes the methodology and results of an experiment investigating the subjective evaluation of isolated environmental sounds on a Foreground (FG) — Neutral (N) — Background (BG) scale. This research, published previously by the authors [23], offers evidence that an AH exists even among sounds which have been removed from context to the extent this is possible in such a test. The application of ML analysis to this dataset is described in Section 4.3, and is also based on work recently published by the authors [24]. Encouraging results are noted that motivate further investigation on larger datasets.

Chapter 5 investigates application of AL to the problem of AH. Feature representations and algorithms are compared in a cross validation experiment outlined in Section 5.3.5 and three AL selection methods are contrasted in Section 5.4. Results, as published by the authors [25], suggest that minimal manual labelling can be used to label large corpora hierarchically.

This leads to an implementation of these concepts and data augmentation techniques to build a large labelled dataset, described in Chapter 6. The resultant analysis compares algorithms used in prior investigations with Deep Learning methods, held to be state-of-the-art [17, 26, 27] in audio domains.

Finally, in Chapter 7, the work undertaken is summarised and discussed in the context of the research objectives and questions outlined in earlier chapters. The contributions of the thesis are summarised, conclusions are offered and possible avenues for future work are considered.

1.4 Publications

The following publications directly exploit work presented in this document.

- Coleman, W., Delany, S. J., Yan, M., & Cullen, C. (2020). **A Machine Learning Approach to Hierarchical Categorisation of Auditory Objects.** *Journal of the Audio Engineering Society.* 68(1/2), 48–56.

- Coleman, W., Delany, S. J., Yan, M., & Cullen, C. **Active Learning for Auditory Hierarchy.** *Cross Domain Conference for Machine Learning and Knowledge Extraction (CD-MAKE)*, Dublin, Ireland; 25-28 August, 2020.

- Coleman, W., Cullen, C., & Yan, M. (2018). **Categorisation of Isolated Sounds on a Background - Neutral - Foreground Scale**. *Proceedings of the 144th Convention of the Audio Engineering Society*, Milan, Italy; May 23-26, 2018.

- Coleman, W., Adams, L., Cullen, C., & Yan, M. (2017). **Perception of Auditory Objects in Complex Scenes: Factors and Applications**. *Institute of Acoustics - 21st Century Developments in Musical Sound Production, Presentation and Reproduction (pp. 1–16)*, Nottingham, UK; November 21st, 2017.

The following publications constitute other work which has informed the context of this research.

- Coleman, W., O'Sullivan, L., Cullen, C., & Yan, M. (2017). **sonicPainter: Modifications to the Computer Music Sequencer Inspired by Legacy Composition Systems and Visual Art**. *International Festival and Conference on Sound in the Arts. Science and Technology (ISSTA 2017)*, Dundalk, Ireland; 8-9 September, 2017.

- Coleman, W., O'Sullivan, L., Cullen, C., & Yan, M. (2017). **iPhone FM Tilter: A Frequency Modulation Instrument for Improvisational Performance using iPhone and Arduino**. *International Festival and Conference on Sound in the Arts. Science and Technology (ISSTA 2017)*, Dundalk, Ireland; 8-9 September, 2017.

- Cullen, C., & Coleman, W. (2016). **Human Pattern Recognition in Data Sonification**. *6th International Workshop on Folk Music Analysis*, Dublin, Ireland; 15th-17th June, 2016.

Chapter 2

Predicting Auditory Hierarchy: A Roadmap

2.1 Introduction

This chapter introduces the area of ASA (Section 2.2), the perceptual study of sound, and also outlines the concept of ObA (Section 2.3) which conceives the auditory scene as made of a collection of audio 'objects'. These sections frame the further study of the concept of AH in the context of existing research. In furtherance of that end, Section 2.4 reviews a number of datasets available in the audio domain and nominates one as suitable for inclusion in this work given the requirements outlined in Sections 2.2 and 2.3. Section 2.5 then offers a review of approaches to perceptual testing and a rationale for those methods chosen for use in this case because of the specific requirements of this study. Finally, these concerns are synthesised in Section 2.6, where a theory of AH is offered together with a roadmap which frames the work presented in this thesis.

2.2 Auditory Scene Analysis

Auditory perception, like timbre itself [28], is a many splendoured thing, subject to influence from a series of factors external to the physiological functioning of the human auditory system. Bregman [11] has described ASA as the process by which auditory scenes are parsed into individual sounds, referred to in this work as auditory *objects*. This is a complex task because sounds are interleaved and overlap in both temporal and frequency domains, and the human auditory system only has access to an amalgam of all sounds that are presented to the ear at any one moment. Bregman describes how the human auditory system addresses this using processes of sequential and simultaneous grouping, where perception is governed by primitive low-level and schematic high-level structures that parse the sound scene presented to the ear for individual objects.

Sequential grouping occurs when similarities in sounds from one moment to the next result in them being grouped to form a *stream*. This is demonstrable via variations in tempo, frequency, timbre, spatial direction and duration of exposure (what Bregman describes as *cumulative effects* [29, pg. 5]). Other factors known to aid sequential grouping are onset/offset synchrony, origination from the same spatial location, similarities in pattern of fluctuation and frequency proximity [11]. Simultaneous grouping occurs when properties of the sound scene match patterns that tend to be true when components of sound come from the same source. If a subset of frequencies that are multiples of a common fundamental are detected, this suggests that the subset is from a common source. Sounds which have a different fundamental frequency tend to be segregated and considered separate. Periodic sounds, such as the human voice and many musical instruments, are an example of this phenomenon [11].

Both forms of grouping are functions of primitive and knowledge-based processes (see [30] and [31]). The term *bottom-up* is frequently used to refer to primitive, sometimes unconscious processes which are thought to be innate, have been found in non-human

animals [32] and in the perception of speech [33] and music [11]. Knowledge-based processes are frequently referred to as schematic, or *top-down* processes which involve conscious attention or past experience [34], for example.

2.2.1 Listening Modes

The listening modes of Truax [35] provide a useful framework for different levels of auditory perception. Truax outlines three modes in total, *BG listening*, *Listening-in-Readiness* and *Listening-in-Search*.

BG listening is outlined as a class of sounds that are not actively monitored, but for which awareness exists. While subliminal auditory perception is acknowledged as controversial, Kotzé and Möller [36] note significant galvanic skin response (changes in the electrical resistance of the skin) to subliminal auditory stimuli. Norman *et al.* [37] offer evidence of awareness of stimuli without conscious attention. Linzarini *et al.* [38] offer a review of consciousness and awareness studies and suggest that cognitive control can operate on conflicting subliminal information. Furthermore, the concept of *change deafness* [39] positions conscious attention as critical for auditory change detection even in very simple auditory scenes [40].

Dupoux *et al.* [41] suggest that conscious and unconscious processing are distinguished by "high-level perceptual streaming factors" rather than stimulus energy and duration distinct from Truax's mode of *BG listening*. Sounds that are actively attended to can be thought of as *figure* sounds while others form the *ground*, similar to the Gestalt example of figure/ground perception [42]. The *cocktail party effect* [43] highlights the ability of the auditory system to pull different auditory objects in and out of focus as required. This frames *BG listening* as a complex process of constant evaluation and re-evaluation of the auditory scene [44], where objects are continually evaluated for whether they are worthy of greater attention [45] or not.

The second of Truax's listening modes is *Listening-in-Readiness*, described as being an intermediate mode of listening where familiar sounds, such as the sound of our own name, are continually monitored while primary attention is focused elsewhere. Truax highlights the example of a parent capable of sleeping through traffic noise who wakes at the sound of their baby crying as an example of this mode in action.

The last of these modes, *Listening-in-Search* is when listening is most analytical, where the sound itself is searched for meaning. This is illustrated by the cocktail party effect, where a conversation within one group can be focused on to the exclusion of the conversations of others.

The foregoing supports the view that FG/BG categorisation of a sound can be established with a reasonable degree of confidence, with the caveat that this could not be considered a universal, unchanging categorisation. FG/BG categorisation, in other words, retains a somewhat subjective nature, dependent on other factors and can be considered to continually be in a state of flux. This has been illustrated by the dog on a beach Gestalt image, reproduced in Figure 2.1 and recently in the auditory realm via the *Yanny/Laurel* stimulus [46], which illustrates how perception can vary on an individual level due to small changes in timbre.

Sound categorisation is therefore seen to be a complex process of auditory perception and subsequent organisation and constant reorganisation, likely on both conscious and unconscious levels. This is supported by Guastavino [12], who posits sound categorisation as an aggregation of inputs from different classification schemas such as source identification, source action and context. Framing our investigation of the FG/BG categorisation task through the listening modes of Truax, this positions the categorisation of auditory objects as fluctuating due to perceived importance relative to activity in the observed scene, semantic meaning derived from the sound itself, and/or the action that it represents.

Thorogood *et al.* [47] examine the consistency of an arbitrary BG/FG categorisation of sounds drawn from the World Soundscape Project Tape Library [48] (WSPTL) . Subjects'

Fig. 2.1 The Gestalt dog on a beach image where the outline of a dog is suggested by the alignment of Dalmation spots. Subjects typically fail to perceive the dog if the picture is first presented to them upside down, but quickly form the dog percept once the picture is presented as above.

were asked whether they agreed or disagreed with the categorisation provided by the WSPTL. Strong levels of consensus were observed between study participants and the arbitrary tagging of the WSPTL on what constitutes an FG sample (80%), BG sample (92%) and BG with FG samples (75%).

The authors [47] make several further points about the nature of the FG/BG categorisation task which are worthy of mention. Firstly, that it is dependent on context (consistent with [1]) and focus of attention, which can be encapsulated with the idea of listening modes, as outlined by Truax [35], Chion [49] and Wolvin and Coakley [50]. Such listening modes treat a sound as BG or FG, depending on the amount of attention being paid to the sound. For example, Truax's *Listening-in-Search* can be characterised as focused, FG listening. His *BG Listening* can conversely be thought of as unconscious BG monitoring of a sound scene. Ubiquitous sound can be thought of as the BG quality of a soundscape. Finally, as

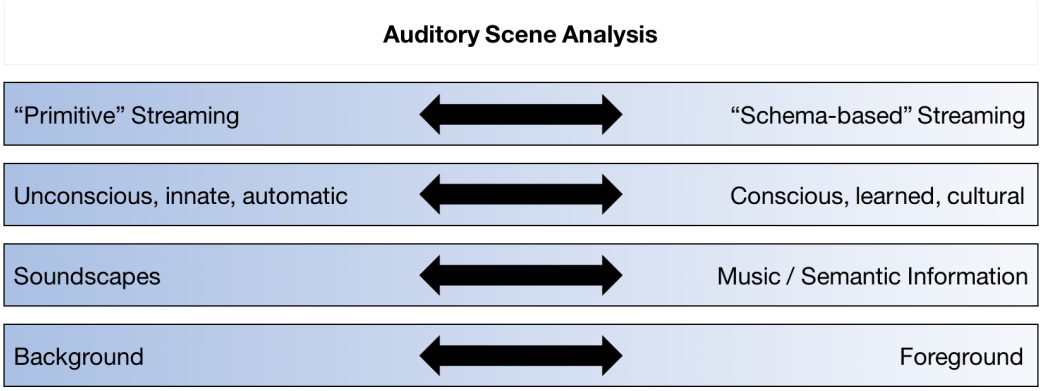

Fig. 2.2 A representation of the progression from background to foreground listening as a continuum, and how this relates to soundscape/semantic listening and auditory perception in general.

summarised by Augoyard and Torgue [51], sound can seem to come from everywhere and nowhere.

Developed from a literature review of the areas of ASA, related soundscape and sound categorisation research, Figure 2.2 outlines a series of axes proposed as those upon which AH acts. This theoretical mapping envisages a constant unconscious monitoring of the auditory scene while conscious attention is focused on FG sounds to derive semantic meaning from them [52]. This hypothesises AH operating as a process of constant identification of sounds deemed worthy of closer attention [53]. Sounds thus identified become the focus of FG attention while others, deemed less important for the time being, ‘fade’ to become part of the BG sound scene. This is not to suggest that all sounds containing semantic information or all sounds that suggest movement are constantly part of the FG sound scene. Rather, there is a constant interaction of different functions (context, attention, training and others), which alter the position of an auditory object on a hierarchical scale. This suggests two problems. Firstly, that of identifying which objects are of most importance and secondly, how and to what degree this relative importance changes due to the influence of context, attention, training and other factors.

This section has introduced the area of ASA and as part of this offered a high-level roadmap for the function of AH. The next section will introduce the concept of ObA and position it relative to ASA and sound hierarchy research.

2.3 Object-based Audio and Modern Media Consumption Paradigms

Modern audio content consumption has in many ways been informed by the legacy technology used to make and present audio recordings. The earliest methods of sound recording rarely involved more than one microphone, and output was via a single channel of audio [54]. As technology developed, content creators drove the desire for more flexibility in how broadcast content could be presented. Gradually, the concept of multi-channel recording became the norm, and this in turn created the need for more sophisticated mixing systems. The desire to individually treat mix elements with different *equalisers* (used to balance frequency levels of audio content), *compressors* (used to alter relative loudness levels within audio content) and other effects was thus facilitated. With the advent of digital recording systems content management became ever more complex, as with the capability to record hundreds of tracks came the proliferation of elements over which creators required control in order to produce compelling content.

In addition to content creation, advances in technology have also greatly changed how audio is consumed. The popular phrase, 'Put a sock in it!', may or may not have its origins as a rudimentary volume control for early gramophones. What can be said is that consumers now have greater control than ever before over audio content both in terms of mode of consumption (headphones, stereo speakers, home assistants, sound bars or multi-channel home cinema systems) and control (equalisation, volume, genre). However, with a few exceptions, such as cinema sound, bespoke art installations and home multi-channel audio

systems, the predominant method for mass media audio consumption is still the stereo audio file.

ObA is a concept which replaces the stereo mix with a bundle of audio 'objects' controlled by accompanying metadata. It envisages audio delivered not as a static mix of many individual elements combined into one stereo file, but as a collection of individual audio assets which are presented according to a provided metadata schema. For content creators this potentially allows the freedom to optimise the delivery of audio depending on content type (broadcast, game or music audio), end-user configurations (stereo, headphones or multi-channel) and other factors, even adapting automatically to local conditions (varying bandwidth capacities, individual preferences and differing environments) [9]. For consumers, the concept may materialise via the ability to control elements of the sound mix delivered to their televisions. The BBC has experimented with object-audio football broadcasting, for example, providing individual consumers control over crowd noise from different parts of the stadium and a commentary feed. Participants tended to balance their mix in the first minute of the broadcast and did not alter it subsequently. Preferred mixes were observed [55]. Another configuration could see control of film audio surfaced to consumers in the home, either offering a choice of mixes or allowing the audience control of individual audio objects themselves.

The interest in providing audio as a series of individual assets as opposed to a fixed mix carries with it a number of questions as to how content might best be managed by creators in order to leverage the full range of possibilities the concept facilitates. In many cases, content creators will have specific needs and a vision for the presentation of pieces. The proliferation of possibility that technology allows, however, suggests the usefulness of an ability to derive semantic meaning from individual assets to ease their integration into production workflows. This applies not only to the identification and classification of objects but also their categorisation in other semantically meaningful ways, such as which are perceived as being most important at the current moment, and monitoring how this changes

over time. The ability to predict such semantic properties could then be used to formulate codec(s) for use in the generation of audio content for different media forms and for differing consumption paradigms.

Environmental sound categorisation, of which hierarchical classification could be considered a sub set, is a multi-faceted problem which requires control of numerous effects bearing individual study. Previous sections have identified a series of possible influences on temporal variance in the AH which suggests a need to study each influence in isolation, so far as this is possible, in order to identify the degree to which each exerts influence on the hierarchical fluctuation. This suggests a necessary simplification of a complex system from a fluid continuum to one of discrete categories in order to facilitate understanding and assimilation of the concept into modern media production workflows. This work proposes that in order to do so in a structured manner, a corpus of sounds must be utilised that are isolated from context to the degree that this is possible, while still providing a broad palette of sound types to choose from. By proceeding in this manner, other influences on hierarchy can be introduced and studied in isolation, allowing a broader understanding of the phenomenon.

Considerable sensory research exists regarding soundscapes (e.g. [2, 56, 57]), sound categorisation (e.g. [30, 34, 58]) sound taxonomies (e.g. [1, 59, 60]) and how attentional, contextual and other processes affect our perception of the environment (e.g. [10, 61, 62]), which includes the recent multi-stable *Yanny/Laurel* percept [46]. However, there is little focused on hierarchies of importance between sound objects in complex auditory scenes and on the movement of sounds from BG sound scene to FG conscious attention. A more complete overview is offered in Section 2.6 but to summarise, the author is unaware of any studies which investigate this phenomenon and provide a broad palette of isolated sounds with hierarchical information on a BG — N — FG scale. Lewis *et al.* [63] provide a rating on an *object-like* versus *scene-like* axis for a selection of mechanical and environmental sounds. Thorogood *et al.* [47] use a selection of soundscape recordings derived from the

World Soundscape Project Tape Library database [64] and categorise them in BG, FG and 'FG with BG' categories. These sounds were selected with the intention of allowing the listener to identify sound context. Salamon *et al.* [59] perform subjective labelling of BG and FG urban sounds and validate their accuracy with experimental testing, but the sounds used are confined to urban contexts and are not isolated from context. This suggests that a database of hierarchically labelled sound objects, isolated from context in so far as this is possible, would be a useful contribution to research in this domain.

The next section will offer a selective summary of existing sound datasets and taxonomies, as this will be of assistance in elucidating stimuli selection choices for the experiments outlined in later chapters.

2.4 Sound Datasets

This section reviews sound taxonomies and existing sound datasets to investigate the methods applied to sound organisation. Implications are drawn for further study, given the envisaged need for a dataset to study AH.

2.4.1 Sound Taxonomies

Numerous taxonomies of sound exist, which serve to highlight the different ways in which it is possible to categorise sounds. In general, these tend to be organised in terms of the materials that produce sounds [65], or the activities that they indicate [1].

Gaver [65] outlines a taxonomy (Figure 2.3) of sounds delineated between classes of materials (vibrating objects, aerodynamic and liquid sounds) and by interactions which may cause them to sound (impacts, explosions, dripping etc.). Gaver further suggests the thesis, supported by Gygi *et al.* [62], that everyday listening, or "the experience of listening to events rather than sounds" (pg. 2), focuses on acoustic factors most useful for source identification,

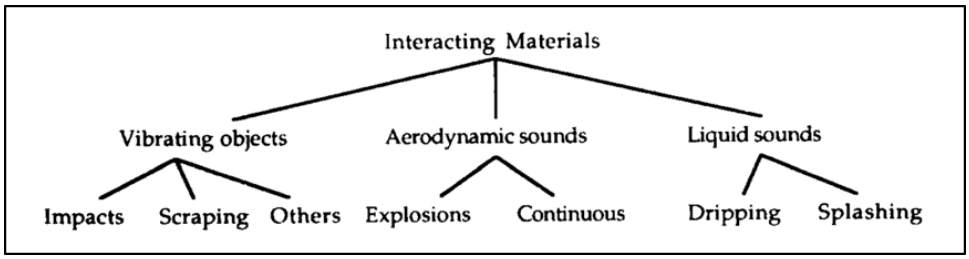

Fig. 2.3 A reproduction of Gavers' taxonomy of everyday sounds, delineated by classes of materials and by interactions which may cause them to sound.

Natural Sounds: Bird, chicken, rain, sea shore	Human Sounds: Laugh, whisper, shouts, talk, cough	Sounds & Society: Party, concert, grocery store	Mechanical Sounds: Engine, cars, air conditioner	Quiet & Silence: Wild space, silent forest	Sounds as Indicators: Clock, doorbell, siren

Fig. 2.4 Categories of sounds used for the World Soundscape Project.

as distinct from musical listening, where the "perceptual dimensions and attributes of concern have to do with the sound itself" (pg. 1). It is interesting to note that this taxonomy is outlined according to qualities of the sounds themselves rather than the objects which produce the sounds, a facet which is prevalent in more recent similar taxonomies.

R. M. Schafer outlines an extensive catalogue of sound types as used in the World Soundscape Project in [66]. The organisation used in the catalogue is arbitrary, but also comprehensive, having been built up over a period of years, and is empirically derived. Regarding the bias inherent in any such organisation of objects, Schafer makes the point that "the only framework inclusive enough to embrace all man's undertakings with equal objectivity is the garbage dump" [ibid., pg. 137]. An illustration of the broadest categories of sounds is offered in Figure 2.4.

Brown *et al.* [1] offer a comprehensive review of the perceptual assessment of human sound preference compiled by working group 54 of ISO/TC 43/SC 1. The authors suggest that for this area of study to be standardised there needs to be a parallel standardisation of language usage across disciplines that have an interest in the area. They emphasise the importance of context in the perception of sound, noting that sounds which are unacceptable in one context

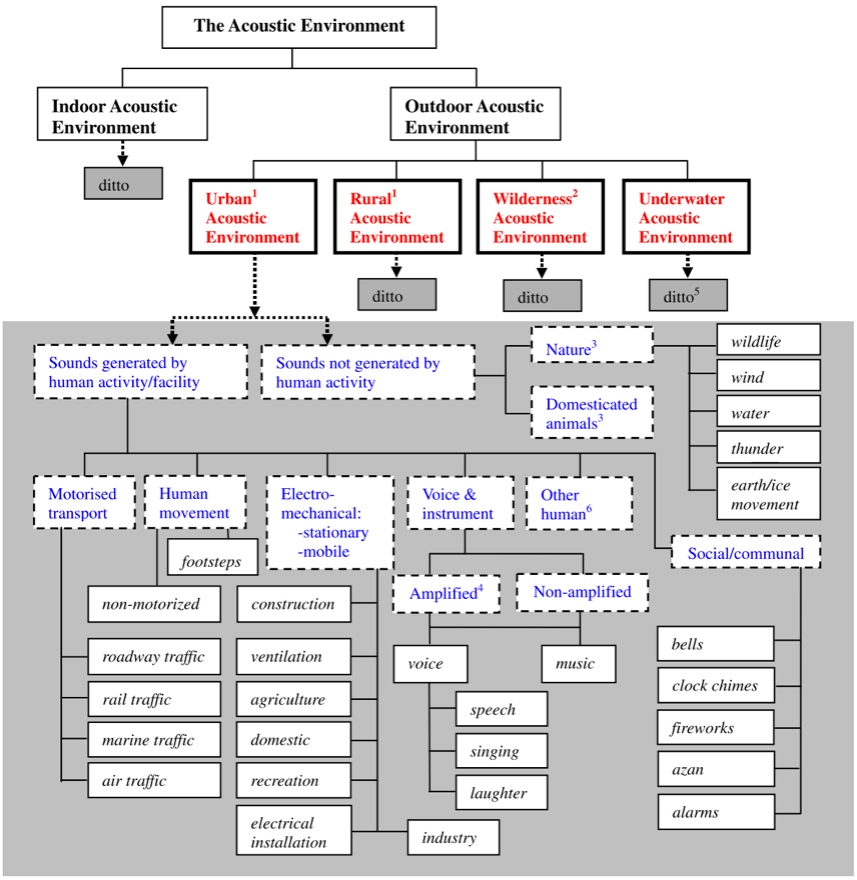

Fig. 2.5 A taxonomy of the acoustic environment for soundscape studies as presented by Brown *et al.* [1].

may be acceptable in others. Further to this they call for the analysis, identification and categorisation of such contexts that are germane to soundscape studies. The study offers a taxonomy of the acoustic environment for soundscape studies (reproduced in Figure 2.5), which has been used in other soundscape research [59], and shows promise for exploitation in sound categorisation tasks. It is interesting to note that a significant organising rule of this taxonomy is whether sounds do or do not indicate the presence of humans.

Raimbault and Dubois [2] suggest a taxonomy for urban sound scenes (shown in Figure 2.6) and summarise research on two particular topics of interest. It is again interesting to note that presence or absence of people in an auditory scene is a significant organisational structure. Furthermore, they reinforce the idea that certain noises are identified in terms

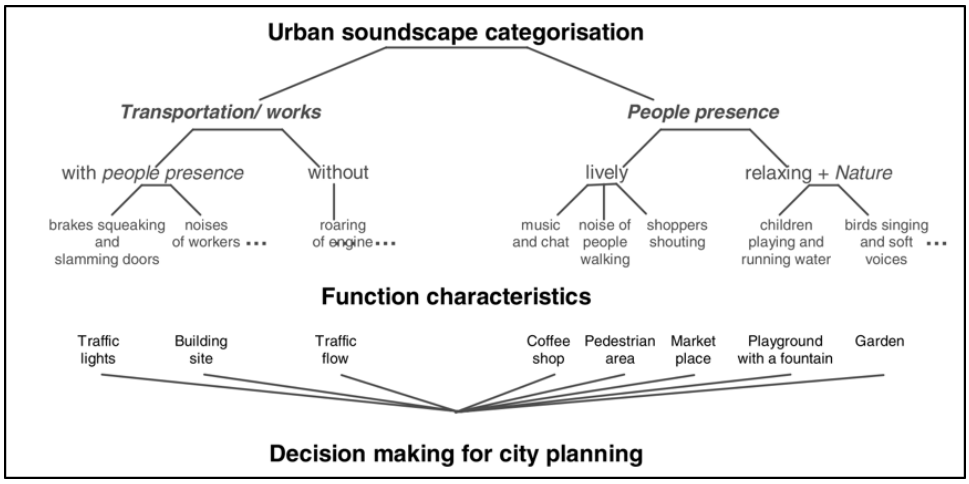

Fig. 2.6 An urban soundscape taxonomy as developed by Raimbault and Dubois [2].

of the semantic content the sound suggests in the goal of source determination. They also summarise research that suggests soundscape perception is affected by factors such as air pollution and physical appearance —- this recalls research reinforcing the importance of context in deriving meaning [1].

A more recent example of such organisations is offered by Gemmeke *et al.* [3] which consists of a dataset of sounds [67] manually curated from over 2 million YouTube [68] videos. These events are organised using a hierarchically structured ontology of 632 audio classes, which demonstrates the complexity to which any sound categorisation task is subject. A summary of the top level of organisation of this ontology is offered in Figure 2.7.

The taxonomies outlined in this section reflect the arbitrary nature of the sound categorisation task, but are indicative of the general principles used in the research and reflect much of the subsequent literature. They illustrate that multiple approaches are valid, and many similarities can be observed. For example, note the presence of *human sound* and *natural sound* categories in both the taxonomy of Schafer and the ontology from Gemmeke *et al.* [3]. Similar themes are recognised in the area of sound categorisation, with semantic elements noted in subject categorisation of sound by [30, 57], and presence or absence of

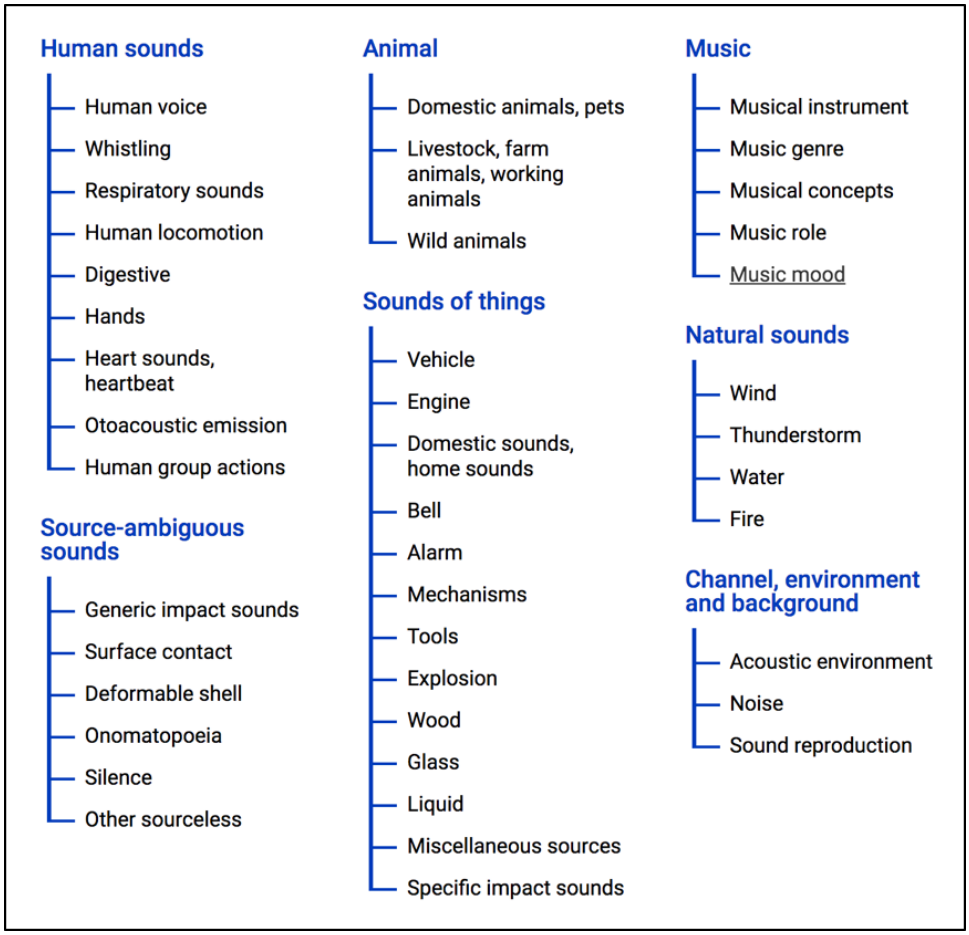

Fig. 2.7 The top-level categories of the ontology used by Gemmeke *et al.* [3] to organise over 2 million sound stimuli curated from YouTube.

humans noted as an organising factor by both [58] and [1], for example. Similar structures are envisaged as being useful in content organisation for subsequent experiments.

2.4.2 Stimuli Selection

To the author's knowledge, there exists no systematic strategy for the selection of listening test stimuli. This is supported by the literature (see Ekeroot *et al.* [69] for a review) who suggest that accounts of stimuli selection for published experiments should be collated to develop a systematic strategy in this regard.

Section 2.5.1 outlines relevant audio listening test standards which stipulate that stimuli must be critical in that they should stress the systems under test [70] and that a panel of suitable material should be parsed by a small group of expert listeners to finalise test items. Stimuli selection is also acknowledged as being time-consuming and resource-intensive, but remains a critical task as it has been shown to be the single biggest factor in intra-subject variance [71]. Evidence from both soundscape categorisation [72] and music information retrieval literature [73] would concur with the need to codify the stimuli selection process, as both domains have seen that use of a skewed dataset can lead to misleading results. Obviously, this underlines the criticality of the stimuli selection task.

An abundance of different sounds have been used in listening test experiments. Music and/or speech stimuli gathered from commercial music CDs, live music recordings or speech excerpts have been used extensively for BS. 1116 and MuSHRA tests by Bates *et al.* [74], Wüstenhagen *et al.* [75], Feiten *et al.* [76], Barbour [77], George *et al.* [78] and Schinkel-Bielefeld *et al.* [79], for example. Quackenbush and Gross [80], Stoll and Kozamernik [81], and De Man and Reiss [82] use speech only stimuli, whilst Sun *et al.* [83] utilise a singing voice stimulus only. Environmental and mechanical sounds are used in an experiment by Lewis *et al.* [63] where subjects used Likert scales to rank sounds as *object like* (low score) or *scene like* (high score). Environmental, musical and vocal sounds are used to

compare the perception of subjects with and without hearing aids by Collett *et al.* [84]. Gygi and Shafiro [85] use everyday environmental sounds in an experiment that demonstrates what they term the *incongruency advantage*, the fact that sounds deemed out of place in an auditory scene are more likely to be noticed. Gygi *et al.* [62] use environmental sounds in a categorisation experiment that identifies three broad categories (harmonic, discrete impact, and continuous sounds) using multidimensional scaling. Lewis *et al.* [86] use animal and tool sounds to investigate brain activity evoked by each using magnetic resonance imaging. Animal sounds are observed to activate middle portions of the left and right superior temporal gyri, whereas tool sounds activate a predominantly left hemisphere cortical *mirror network*, associating the sound heard to the motor action judged likely to have produced it. Wilson and Fazenda [87] use 20-second excerpts from 63 tracks of popular music to investigate the quality perception of recorded music. Hold *et al.* [88] use a surround mix of an up-tempo pop track recorded specifically for their experiment to investigate the impact on listening preference evoked by introducing variation in the spatial mix. They identify variations in virtual source positioning, loudness and dynamic range compression as being capable of positioning sounds in either FG or BG. Woodcock *et al.* [58] use excerpts from various broadcast scenarios in an object audio categorisation test. Stimuli used include:

- Radio drama (BBC productions of “The Wizard of Oz” and “The Hitchhiker’s Guide to the Galaxy: Tertiary Phase”)
- Nature documentary (BBC production of “Life: Challenges of Life”)
- Live events (BBC productions of the last night of the proms (live music), tennis at Wimbledon, and a soccer match)
- A feature film (Woman in Black)
- Naturalistic soundfield recordings of urban soundscapes around the city center of Manchester, UK

Rummukainen *et al.* [34] use audio-visual soundscape stimuli recorded in locations around the city of Helsinki in a scene categorisation task. Scenes were subjective categorised as 'calm', 'still', 'noisy', 'vivid' and 'open', with perceived movement, noisiness and eventfulness highlighted as factors used in the categorisation process. Guastavino [30] also uses soundscape recordings in an experiment that investigates sound categorisation on the basis of object identification, function or low-level features. Soundscape categorisation was found to diverge on two broad dimensions: the presence or absence of humans and the level of mechanical noises observed. These categories were further subdivided into the different activities undertaken according to the presence and/or absence of humans and by type of engine in scenes dominated by mechanical noises.

This research is concerned specifically with investigating the nature of any possible hierarchy of importance of auditory objects within auditory scenes. Papers summarised utilise accepted tests, such as the ITU MuSHRA [70] and BS.1116 [89] standards, or tests based on a modified version of these, for evaluating auditory aspects of perceptual codecs or investigating the perception of *everyday* auditory scenes.

Numerous studies in this section are noted as using varied stimuli in different contexts. Use of such stimuli in this manner can be held to pass the ecological validity test, as the stimuli used consistently reflect either end use cases or the specific environment under investigation. For example, contemporary and classical music recordings are good choices for tests evaluating audio codecs and loudspeakers, as they reflect a significant proportion of the eventual end use cases. Similarly, urban and rural soundscape recordings are a logical choice for stimuli in experiments investigating the categorisation of sounds from such scenes. Care has been taken in each test design to take account of the specific end use case. Indeed, in some instances stimuli from previous tests are re-used in slightly different contexts, with additional stimuli added based on broadening the test case or encapsulating a development in end use, with care taken to ensure the validity of new stimuli. For example, Gygi *et al.*

[62] conduct an acoustic similarity and categorisation task using a subset of the sounds used by the same authors in [90] to investigate spectro-temporal factors in the identification of environmental sounds.

Notable here is the fact that stimuli selection is tailored in each instance to match the task at hand. Consequently, in this work consideration was given to optimal stimuli to establish a baseline regarding hierarchical ranking of sounds, given the evaluation task in this instance differs somewhat from that of the tests mentioned. The experiments outlined previously evaluate either some notion of Basic Audio Quality (BAQ) or the factors behind the categorisation of auditory scenes and/or objects. BAQ in this research is a reference to a single, global attribute used to judge all perceptible differences between two auditory stimuli, to include aspects of timbre, loudness, spatial presentation, distortions, noise, pops, clicks and other artefacts as defined in the various ITU standards [70, 89]. When concerned with BAQ the purpose is generally to arrive at a grading of some function of the audio delivery paradigm such as a compression codec, a microphone or a sense of envelopment. In these cases, the differences between stimuli are relatively small, and the stimuli can be said to be very similar to each other, lending themselves naturally to comparison. This is a fundamentally different evaluation task to that of applying hierarchical labels to a series of environmental sounds.

This research is concerned with investigating the participant's perception of an auditory scene. Most specifically, this concerns the identification of which auditory objects a subject is likely to think more important. This suggests that selecting stimuli from auditory scenes that are agreed to be ecologically valid should be a priority. In this context, using stimuli which consist of sine tones and/or noise bursts would not appear to be ecologically valid, as these will seldom (if ever) comprise the audio in most end-use cases. A consideration of end use cases has already been outlined in Section 1.1 to include broadcast, game, music and

entertainment audio, with specific regard to mobile content delivery in limited bandwidth situations. This provides a considerable palette of possible options for test content selection.

As observed, a wide variety of audio stimuli are utilised in evaluation and categorisation listening tests, from commercial music [87, 91, 92], synthesised tones as with many of the demonstrations presented by Bregman [11], and in more recent studies [93, 94], to soundscape-based stimuli noted to contain FG and BG elements [95, 96] and soundscape field recordings [97, 98]. Bearing in mind the future relevance and possible applicability of any resultant research, it would then seem logical to base experimental stimuli on contexts and modes of consumption which are most relevant to future implementation. Such usage is likely to include multiple entertainment delivery scenarios both in public and home-based contexts, automotive and mobile consumption of audio-visual content and gaming. This would envisage use of a selection of broadcast, musical, gaming, film and environmental based stimuli.

The broad palette of sound types identified indicates that investigating sound objects in isolation to determine what hierarchy, if any, exists in this state will be a useful first step to in building a model to predict AH. For this work it is considered that scenes which use primarily non-music elements such as speech and effects will be analogous to visual streaming content such as drama or sports broadcasting and much computer game content which form the core of expected end use-cases. Thus, the stimuli selected should reflect this position excepting the use of speech stimuli, which could naturally be considered to form a core component of the FG of many sound scenes, as indicated by its primacy in many categorisation schemas [99]. The FG/BG nature of isolated sounds initially derived can then be used to inform design of further experiments to either improve the ability to predict or to deepen understanding of other parameters important in determining sound importance.

2.4.3 Existing Sound Stimuli Corpora

In approaching the design of an experiment to investigate categorisation of sounds, a review was undertaken of existing sound corpora to assess their suitability for use as test stimuli. The lack of large, labelled datasets for experimental purposes is an acknowledged problem in the field [17]. A non-exhaustive summary is offered below, which serves to outline the different options available and illuminate the stimuli selection process.

R. Murray Schafer outlines an extensive catalogue of sound types as used in the World Soundscape Project [48] in his book *The Soundscape: Our Sonic Environment and the Tuning of the World* [66]. This catalogue is connected to an extensive database of sound recordings collected since the inception of this project in the 1970s called the World Soundscape Project Tape Library database [64]. These recordings have been used as source stimuli in [47], and there are further examples of soundscape recordings used as stimuli in perceptual testing ([30, 34, 100]) and repositories of recordings compiled with the express purpose of providing a source of stimuli for such tests ([3, 59, 101]).

At the original time of writing *The Soundscape*, in 1977, this catalogue numbered several thousand cards. An outline of the broadest categories of these sounds is offered in Figure 2.4. Schafer makes the point that a sound can appear in several places in this catalogue, as a sound can function in more than one context. In terms of the corpus as a whole, these sounds are predominantly of whole soundscapes, and so use of isolated samples from this dataset would require an extensive selection process.

Salamon *et al.* [59] present a taxonomy of urban sounds based on the urban subset outlined in [1], and a dataset of audio sounds, entitled *UrbanSound* [102], which contains 27 hours of audio. They define 4 top level groupings in their taxonomy: Human, Nature, Mechanical and Music. Interestingly, each sound in this dataset has been labelled with what they term a saliency characteristic. This description indicates whether the sound was subjectively perceived as being in the FG or BG of the recording. While this was manually

labelled by the authors, a subsequent automatic categorisation experiment found that, with only one exception (a siren noise) items labelled as BG sounds were significantly more difficult to identify than those labelled FG, suggesting this manual labelling process is robust. This is an extensive dataset based on urban sounds which has been organised to a specified taxonomy and has potential to be a useful building block for establishing a test dataset which is not confined to urban sounds.

Piczak [101] presents the ESC dataset of sounds for use in categorisation research. This dataset consists of 2,000 short clips which are annotated and span 50 different classes of audio events. Also included is an unlabelled compilation of 250,000 audio excerpts culled from the Freesound [103] project, a collaborative database of Creative Commons licensed sounds available to all. A categorisation comparison experiment was carried out with this dataset to compare human accuracy with that of automatic classifiers. In general, it was found that humans achieved greater accuracy in the categorisation task, with three broad delineations in categories highlighted by the author as follows:

- Easy categories (human and animal sounds, some distinct sound sources e.g. siren, water drops, breaking glass)
- Average categories (sounds ranging between easy and difficult categories)
- Difficult categories (soundscapes and some mechanical noises)

This is a collection of individual sounds in short clips which offers the flexibility of testing individual sounds in a categorisation task or the possibility of composing bespoke sound scenes with separate audio object stems should this be deemed appropriate. Furthermore, there is a considerable body of categorisation research which uses these sounds for various purposes, for examples, see [104, 105, 106].

The DCASE events are a series of sound classification competitions, the sixth of which is running in 2020. The challenge was created “to support the development of computational

scene and event analysis methods by comparing different approaches using common publicly available datasets" [107]. The challenge encompasses a number of different cases which include acoustic scene classification, general-purpose audio tagging of Freesound content (this element is hosted on the popular machine learning competition platform, Kaggle [108]) and semi-supervised sound event detection in domestic environments. Audio data from a variety of sources is provided for each task. In the case of acoustic scene classification, files recorded in large European cities are provided. Diverse sound events which feature musical instruments, human and domestic sounds and animals are culled from Freesound for the tagging task. YouTube [68] video excerpts focusing on domestic context are provided for event detection tasks. All audio data is available for download and could potentially be used as experiment stimuli, though much of it consists of sound scene stimuli rather than isolated sounds.

Gemmeke *et al.* [3] provide a large dataset [67] of manually labelled audio events curated from YouTube. These events are organised using a hierarchically structured ontology of 632 audio classes, which has been compiled via the literature and manual curation. The top-level structure of this ontology is outlined in Figure 2.7. At the time of writing, the dataset consists of more than 2 million YouTube videos using 527 labels. While not organised in terms of audio event importance, this is a large corpus of potential listening test stimuli, culled from a source which is highly relevant to modern audio delivery. A significant degree of mobile audio entertainment consumption is via video platforms like YouTube, arguably making it a relevant source for ecologically valid test stimuli. The stimuli would need extensive parsing to provide a corpus of isolated sounds to test, however.

This section has identified and summarised a number of existing sound corpora, but is not offered as an exhaustive list. Other sound corpora certainly exist, though not all are freely available to the scientific community. Indeed, a repository of such sound sets, both freely available and not, is curated by Toni Heittola [109], one of the DCASE organisers. At the

time of writing, more than 40 datasets were listed which furnish access to a wide variety of sounds for experimental purposes.

Few of these corpora feature an extensive set of isolated sounds, an exception being the ESC50 and associated datasets [101], although in order to select only sounds isolated from context an extensive auditioning process will be required. Given the desire to prioritise an investigation of isolated sounds in order to establish a dataset from which further parameters of variance could robustly be investigated, it was decided to use this dataset as a source for stimuli for initial experiments.

2.5 Perceptual Testing and Audio Standards

Hierarchical categorisation of audio objects is essentially an environmental sound classification problem for content such as game audio, much visual streaming content and drama, entertainment and current affairs broadcasting. This involves an investigation of individual subjective judgement of sound, specifically with regard to which sounds are most important when. As such, this should be seen as distinct from studies focussed on variations in BAQ between experimental stimuli, which will further be reviewed in Section 2.5.3. In other words, our focus in predicting AH will be on subjective perception of macro sound categorisation on a hierarchical level, rather than on micro differences between stimuli which may become important in the fine-tuning of any real world implementation of such a system.

The following sections outline the design decisions for an initial experiment, referred to henceforth as Experiment 1, whose aim is to establish the nature of AH that exists between sounds isolated from context to the extent that this is possible. This will entail an examination of current best practice in perceptual audio testing including a review of relevant standards, the dangers of bias in perceptual test design and experiment implementation concerns. This discussion frames the basis for design decisions taken and highlights issues encountered throughout the process.

2.5.1 Listening Test Standards

Numerous bodies have published standards regarding the correct procedures to be followed when conducting listening tests investigating the perceptual evaluation of audio. These include, but are not limited to, the Audio Engineering Society (AES) the International Telecommunications Union (ITU) , and the International Standards Organisation (ISO) . It was decided to focus primarily on the ITU standards as the weight of material from audio domain sources such as the Journal of the Acoustical Society of America and the Journal of the Audio Engineering Society cite these standards when conducting audio perceptual evaluation tests. Furthermore, as the delivery of broad bandwidth, multichannel audio in broadcasting situations is one of the potential foci of this research, it was decided that cognisance should be taken of the methods used by industry broadcasting bodies to evaluate audio in similar paradigms. The European Broadcasting Union (EBU) has issued a number of papers on the evaluation of audio in broadcast situations, in which they make extensive use of the ITU standards. Their members include the prominent national broadcasting companies of Europe [110], such as ARD (Germany), the BBC, ITV, Channel 4 (UK), and Canal Plus (France).

Within the ITU there are two categories of standards relating to the perceptual evaluation of audio: ITU-T and ITU-R. The scope of the ITU-T standards is confined to telecommunications applications, and they refer to either narrowband (300 – 3400 Hz) or wideband (150 – 7000 Hz) bandwidths. The ITU-R family of standards pertains to audio of the bandwidth 20 Hz – 20 kHz, that is generally accepted [11] as the range of human hearing. For this reason, the ITU-R standards were chosen as most applicable for the current research as it is intended for multiple areas of application, not simply telecommunications.

Within the ITU-R category, there are a number of methods detailed for the perceptual evaluation of audio that cover both subjective (Rec. ITU-R BS.1116, BS.1285, BS.1534 and

BS.1679) and objective (Rec. BS.1387-1) evaluation of audio. There are also guidelines on the evaluation of audio in audio-visual contexts (BT.500-11, BS.775-1 and BS.1286).

Rec. ITU-R BS.1116 [89] is intended for the subjective assessment of impairments so small that they cannot be detected without rigorous control over test conditions and stringent statistical analysis. Respondents are required to be experienced audio listeners, and the test entails a double-blind setup where neither participant nor moderator knows the order of stimuli to be presented. This test facilitates a very high level of detail when evaluating test stimuli. Such rigour comes at a cost in terms of the facilities, experimental precision required and time and resources needed to adequately administer the test. This test is therefore time-consuming to implement, and extreme care must be taken lest factors external to the test impact on results. The ITU-R BS.1116 test is designed to grade the impairment of an audio signal. The scale used is a value from 0.0 to -4.0, with the individual steps being categorised as 'imperceptible' (0.0), 'perceptible, but not annoying' (- 1.0), 'slightly annoying' (-2.0), 'annoying' (-3.0) and 'very annoying' (-4.0).

ITU BS.1534 [70] is intended for the subjective assessment of intermediate quality levels of audio coding systems. This test, known as MuSHRA (Multiple Stimuli with Hidden Reference and Anchor), uses a series of stimuli which the respondent can compare at will and, like ITU-R BS.1116, is a double-blind test. The stimuli presented include a high quality reference signal, the test signal(s) and anchor signal(s). MuSHRA tests have become widely used [83, 111, 112, 113], and software versions are available for online implementation, in addition to numerous other ABX test variants [114].

The goal of the MuSHRA test is to grade the absolute quality of an audio signal. Respondents are asked to grade stimuli between 0 and 100, giving their perspective on the quality of each sample. The scale is labelled as 'excellent' (100-80), 'good' (80-60), 'fair' (60-40), 'poor' (40-20) and 'bad' (20-0). Up to three 'anchor' stimuli can be used for comparison purposes. The first of these is a hidden reference, which is identical to the original. The

second is a 3.5 kHz low pass filtered version of the original. The third anchor is of an optional design but is required to be inhibited in the same modality as the audio artefacts being measured, thus giving respondents context for their grading of the material.

The EBU, in their own assessment of multichannel audio codecs [115] and [116], elected to use ITU-R BS.1534 for their top-line assessment of multichannel audio codec performance. While greater resolution is possible with the ITU-R BS.1116 method, the EBU in this instance decided to use the MuSHRA method as it "covers the whole quality range and is easier to run than ITU-R BS.1116" [115, p. 15]. MuSHRA is regarded as a method which is not as laborious and time-consuming to implement as ITU-R BS.1116, yet still provides results which are highly accurate, reliable and consistent [117]. The scoring system used for MuSHRA is more suitable in some instances than that used by ITU-R BS.1116 (an impairment scale). Although the resolution provided by ITU-R BS.1116 is greater, it is not always appropriate to use this method because of the time-consuming and resource heavy nature of the test.

In summary, over a period of years, methods for evaluating the subjective perception of auditory stimuli have been formulated and honed. The current consensus is that subjective human rating is the gold standard for perceptual testing [118], and that such tests should be conducted double-blind [119], where neither participant nor moderator is aware of the presentation order of test stimuli. However, the standards outlined are designed to detect small differences between multiple stimuli and are commonly used in tests comparing loudspeakers [120, 121, 122, 123], compression codecs [81, 111, 124, 125], multi-channel presentation of audio [126, 127, 128, 129] and more recent tests investigating surround sound envelopment [78, 130, 131]. Notable in the literature is the existence of many instances where tests are based on these standards, but do not adhere to them in every single detail (see [78, 132, 133] for example), meaning that the standards are often used as a baseline method to instil scientific rigour but adapted to fit specific use cases. It would seem logical

therefore to use the outlined standards similarly in the design process for Experiment 1, adapting them appropriately to suit the nature of the task defined as a categorisation of sounds on a macro level with a subjective hierarchical label as opposed to ranking a series of similar stimuli on a BAQ basis.

2.5.2 Bias in Perceptual Testing

Unless carefully designed and administered, there is potential for the results of any listening test to be compromised by various forms of bias. Numerous papers have been published on this topic [112, 120, 134, 135] and the following is a brief summary of the important points outlined in this literature. Poulton [136] categorises bias types into three areas: contraction bias, bias caused by lack of familiarity with units of measurement and bias caused by unfamiliarity with the mapping of the responses to the stimuli.

Contraction bias occurs when a respondent tends to be conservative in underestimating large differences and overestimating small differences between stimuli. This includes effects caused by the order of presentation of the stimuli. Poulton suggests counteracting this effect by counterbalancing the order of presentation of stimuli, or by using a Latin squares design, where every possible order of stimuli presentation is used.

Bias caused by a lack of familiarity with units of measurement is a particular problem for audio listening tests, as many subjects lack a frame of reference for the judgement they are asked to make. For example, while subjects may be confident in making a weight comparison judgement as they have everyday experience of judging the relative weight of objects, asking them to judge the audio quality of a presentation will not in most instances be a task they are familiar with. This can be tackled by inserting a training routine as part of the listening test.

Bias caused by unfamiliarity with the mapping of response to stimulus takes the form of logarithmic response, range equalising, centering, stimulus spacing, and stimulus frequency bias. These can be tackled using techniques outlined by Zielinski *et al.* [134], which include,

but are not limited to, ensuring some stimuli are not presented to subjects more than others, familiarising listeners with the range of sounds they will be presented, the use of a label-free scale to avoid bias caused by a perceptually nonlinear scale, modifying the scale used to grade stimuli, using anchoring techniques, and systextual design methods, where the range and distribution of stimuli are systematically changed and the influence of this activity on the results is analysed.

One further instance of bias is mentioned by Poulton, that of transfer bias, where an assessment of one attribute is affected by the impression of another. This is commonly an issue where the same group of subjects are used to assess different attributes or different conditions of the same attribute. Counteracting this effect must be balanced against the practicalities of using separate groups with a common rating scale and the advantages of efficiency that attend presenting all stimuli to all subjects.

Bech and Zacharov [112] have noted that no study of this effect relevant to listening tests currently exists. However, it is possible to examine the data gathered for evidence that notable transfer has taken place between significant stimuli, as in the case of systextual design. For instance, if Stimulus B can be shown to consistently drag the scoring for Stimulus A upwards when presented in the order B – A, then this can be adjusted for in the statistical treatment.

The consideration of bias effects is critical in any perceptual test design phase, as there are multiple opportunities for the introduction of confounding elements. Careful consideration of the bias problem is warranted due to the multi-faceted nature of the area and the ease with which unintended effects can influence experiment results.

2.5.3 Listening Test Implementation

Listening tests could generally be said to focus on one of two broad areas of research. The first of these, referred to here as perceptual experiments, is broad auditory scene analysis, which investigates how sound scenes are perceived, parsed and categorised. The second,

designated evaluation experiments for the purposes of this research, generally investigate the perceived BAQ of system components. Compression codecs, loudspeakers and microphones have all been the subject of such evaluation research and Section 2.5.1 has listed examples of a series of such tests.

Methods differ somewhat between these two purposes. The first is generally related to the process of our perception of sound and has given rise to a variation of experimental design approaches, a range of which have been outlined in Section 2.2.1. The most prominent proponent of this research is Albert Bregman [29], who investigated perception of auditory *streams* using a series of experiments that often made use of synthetic tones to establish the basic principles of auditory scene analysis. The success of using such stimuli to broaden understanding of the general workings of the auditory system suggests that they may be of use in understanding more complex sounds. However, evidence suggests that the response of neurons to complex sounds cannot be estimated from their response profile to pure tones [137]. More recent soundscape research using naturalistic stimuli [56, 138, 139, 140] investigates human perception of complex sound scenes to evaluate how they, and the audio objects that comprise them, are perceived and categorised by listeners.

Evaluative research is often based around ITU test methods and standards already mentioned, the ITU-R BS.1116-3 and ITU-R BS.1534-3 (MuSHRA) standards. Such tests are concerned with forensically parsing audio stimuli to detect fractional differences between the element(s) under investigation to determine which is superior. The stimuli used in such experiments generally reflect the intended end use of the factor under investigation, so a listening test comparing headphones, for example, will often use popular music for stimuli [141].

Experiment 1 could logically be thought of as a perceptual experiment rather than an evaluation of some audio system component, suggesting that the primary focus is less on minor differences between sounds and more the semantic information derived from them.

This informs experiment design significantly, as it suggests that the stringent laboratory setting required by the ITU standards is not necessary.

Participants

A common characteristic of evaluative testing is that subjects are skilled audio practitioners of some form [133, 142, 143], as stipulated in the standards. This is a sensible precaution when evaluating minor differences in performance between audio codecs, but not necessarily applicable to every test scenario. Furthermore, the stringent requirements of the standards mean that all experiments complying with them are conducted in laboratory conditions, generally in highly treated rooms designed to eliminate any possible confound caused by room reverberation or other such test environment factor. These requirements combine to keep the number of test participants low, frequently between 10—20 subjects (20 expert listeners is generally held to be enough to obtain a reliable evaluation [133]), and to keep the relative 'cost' of running each test high, given that constructing such facilities to the required standard is expensive.

Perceptual experiments, on the other hand, are more interested in subject perception or classification of the sound presented rather than the minutiae of marginal differences between different sound files. In the case of evaluative research, the same sound may be reproduced multiple times, while the bitrate at which the files are encoded is varied, or a series of different loudspeakers are used to audition the sound, and the subject is asked to pick which returns superior sound quality. The perceptual experiment task is not one of BAQ evaluation, in other words. Participants are generally asked to classify the sounds they hear, or provide some other semantic feedback evoked, but are not asked to compare sounds searching for fractional difference. Indeed, the EBU has stated that the ITU.R 1116 standard is excessively stringent for the assessment of internet audio codecs, proposed the MuSHRA standard as an appropriate alternative [81], and has found the resolution provided by the 1116

standard only necessary where the finest possible discrimination between auditory stimuli is required [115].

Perceptual experiments are often carried out in highly controlled environments when this is appropriate [10, 34]. These seldom directly adopt established standards, however, but frequently adapt to the particular use-case. Moreover, and just as frequently, subjective data is gathered via other methods, such as fieldwork surveys [144, 145] or even via in-situ smartphone applications [146]. Additionally, Lemaitre *et al.* [142] has noted that acousticians conceptualise sounds as abstract acoustic phenomena, whereas non-acousticians conceptualise sounds as indicating the presence of an object that is not abstracted from the sound source. Bech [147] notes that experts are more sensitive to artefacts and are generally more reliable in their ratings than non-experts, however, Schinkel-Bielefeld *et al.* [79] also note that while inexperienced listeners tend to give test audio higher scores than experts, they tend to rank them the same in the vast majority of cases. This suggests that, while there may be more noise in the data from inexperienced listeners, it will generally be in line with ratings from experienced listeners, even for tests investigating marginal impairments between audio signals.

This suggests that non-expert input may be just as desirable as expert input for the purposes of refining a model to predict AH for media consumption paradigms if non-experts are thought of as *users* who can give insight into audio object perception, and *experts* as having a role in evaluating early iterations of applied schemas. Additionally, any robust ML model of sound object hierarchy would require greater participant numbers than is common in evaluative testing, generating ratings on potentially thousands of sounds by hundreds of subjects in order to avoid analysis problems generated by having a great many more features than samples in a given dataset (the so-called *Curse of Dimensionality* or *large 'p', small 'n' problem* [148]).

Environments

The standards outlined in Section 2.5.1 contain detailed instructions regarding appropriate environments for listening tests. Guidelines for room size, shape, dimension proportions, reverberation time, operational room response curve, background noise level, height and orientation of loudspeakers, distance of loudspeakers from room walls and reference listening position for monophonic, stereo and multi-channel setups are covered in detail in the briefing document [89] for the ITU-R BS.1116 standards, which cover the [70] MuSHRA standard also.

These standards are formulated for a specific task, however: discriminating between audio representations on a very fine level of detail. This level of detail is deemed unnecessary for the purposes of this research, a perceptual sound categorisation task.

The desire to maximise the number of participants also mitigates directly against the use of strictly controlled laboratory conditions as a test environment. Furthermore, a number of studies [149, 150] suggest that online testing displays minimal differences to laboratory experiments for comparable tests that do not require forensic examination of minimal differences between stimuli. Most interestingly Cartwright *et al.* [118] implement a MuSHRA-like test via Mechanical Turk [151], an online environment suitable for collecting 'crowdsourced' annotations for perceptual experiments. In crowdsourcing, participants are not selected from a small potential pool but via an open invitation circulated via social media [152] or, in the case of this research, to large communities such as the 'Auditory' mailing list [153] or the R&D department of the industrial partner for this research, Xperi [154]. This suggests that an online listening test is a viable alternative to strict laboratory conditions for certain types of sound labelling tasks. Given these factors, it would seem viable to initially proceed with an online test environment to maximise the number of respondents. This approach can be revised, or indeed cross-validated with tests conducted in laboratory conditions, should this be deemed necessary.

2.6 A Map of Auditory Hierarchy

In this section, research reviewed thus far is summarised and a series of factors hypothesised to have an effect on AH is outlined. Findings on the perceptual function behind hierarchical ranking of sounds are then presented and encapsulated in a theory of AH.

The foregoing has outlined how the level of attention granted sounds by listeners [52, 155, 156], volume level [157, 158], proximity [138], sound event context [35], level of anticipation [159], prior training [147, 160] and experience [161], listening mode [35] and other senses (sight [162, 163], smell [164] and touch [165]) are all known to affect our perception of sounds to some degree. However, the extent to which these factors interact with each other, how they affect any inter-object hierarchy of importance and how this manifests in auditory scene perception is less well understood. The inherent BG or FG nature of a sound in isolation is also speculative, though it can reasonably be hypothesised that certain sounds (speech, alert noises, such as alarms) would likely be thought of as FG. Detailed knowledge in these respects would be central to any well-functioning model based on object audio theory, if each of these factors is thought as requiring a weighting proportional to the influence they exert on sound hierarchy fluidity, which maps how such hierarchies vary over time. Further investigation is required, but the likelihood exists that the some outlined factors are more important than others as regards such sound importance fluidity.

Critical functions of such a model will include a number of considerations. Firstly, the inherent nature of a sound, and its predisposition to be either FG or BG, if any such predisposition exists. Secondly is the identification of the relative importance of different influences on auditory perception and an establishment of appropriate weights in each instance. Thirdly is the development of an understanding of how influences on perception interact in order that changes in the inter-object hierarchy over time could be predicted. Fourthly is consideration of how an ML model to predict the hierarchy of auditory objects

can be built which can then be extended to an implementation which takes consideration of how the nature of how such hierarchies change over time.

Environmental sound classification holds a significant similarity to this task in that it involves the identification of individual sounds within auditory scenes. It could be said that in the act of identification and subsequent categorisation sounds can be thought of as having an importance level allocated which would view these processes as interlinked, the act of categorisation being a subsequent function of identification. Existing studies of sound categorisation have been reviewed to establish what consistencies may be observed in subject approach to such a task. Dimensions of such a categorisation-space will be useful in the formulation of any rule-set to predict sound object FG/BG ranking. To that end, this section will attempt to outline this categorisation-space with a view to formulating strategies for automated hierarchical classification.

As mentioned in Section 1.1, Lewis *et al.* [86] present a study where subjects were asked to rank stimuli as either *object-like* or *scene-like*. In general, sounds judged as objects were more likely to be mechanical and those thought of as belonging to a scene were predominantly natural. Additionally, scene-like sounds tend to have a more gradual change characteristic, differentiating continuous sounds from those with more abrupt change characteristics. In a study investigating the categorisation of broadcast audio objects, Woodcock *et al.* [58] identified three dimensions in sound object categorisation using multidimensional scaling. One of these dimensions ranged between continuous and discrete impact sounds. Another was proposed to be related to the presence of absence of humans. A third dimension progressed from continuous BG sounds to clear speech. The authors maintain that this dimension is related to whether the sound carries semantic meaning or not, which is mirrored in neurocognitive studies such as [86] and [166]. Interestingly, subjects' perceived importance of sound objects correlated with this dimension, suggesting that sound objects which carry semantic information are more important than those which do not.

Collett *et al.* [84] found that musical and vocal stimuli were easier to categorise than environmental sounds which, supported by [65], [62] and [63] suggests that sound categorisation is easier when more semantic information is discernible from the sound. Additionally, Guastavino [57] suggests that people organise sounds and soundscapes in terms of the meaning attached to a sound as a semantic clue to source identification as opposed to any abstract physical property of the sound. Dubois *et al.* [167] investigate meanings attributed to soundscapes both on an individual and collective level in an attempt to outline the similarities between the two. They present converging evidence that the subjective effects of complex acoustic scenes rely on semantic meanings attributed to sounds via cognitive processes. Furthermore, they outline two generic cognitive categories for sounds encountered in soundscapes. The first they term *event sequences*, from which the sources of sounds can be easily identified. The second is *amorphous sequences* where sources cannot be easily identified. These *event sequences* are further subcategorised by subjects according to either the source involved (vehicles, parts of vehicles, human sounds) or a qualitative evaluation of whether the sound was pleasant or not. *Amorphous sequences* are mostly described as background noise and are further subdivided by whether they are pleasant or by an evaluation of acoustic parameters (sound intensity, spectral content, temporal structure).

Gygi and Shafiro [85] demonstrate what they term an *incongruency advantage* by showing that sounds perceived as out of place in an auditory scene are more likely to be noticed. This is supported by Winkler and Schröger [53] and by Sussman-Fort and Sussman [61], who suggest that the auditory system maintains a representation of the environment that is only updated when new information indicates that re-analysing the scene is necessary. This is consistent with Rummukainen *et al.* [34] who find that humans are attentive to perceived movement, noisiness and eventfulness when analysing real-life urban environments. They note that arousal can affect selective attention, increasing focus on certain sounds to the detriment of attention paid to others.

Salamon *et al.* [59] present a taxonomy of urban sounds which they have labelled with a saliency characteristic, which indicates a subjective labelling of the sound on an FG/BG scale. A subsequent categorisation experiment found that BG sounds were significantly more difficult to identify than FG sounds, with only one exception —- a siren noise. This suggests that subjective labelling, if done with care, is a robust mechanism for sound categorisation.

Guastavino [30] suggests that sounds are either classified into taxonomic categories ('car', 'truck', 'street', 'acceleration') according to low-level features or into script categories ('doing the groceries','taking a walk','having a drink') according to high-level features concerned with the situation of use or the end-use purpose of the object. Raimbault and Dubois [2] support the idea that certain noises are identified in terms of the semantic content the sound suggests, and also outline research that suggests that psychological and sociological factors can affect sound scene perception. They suggest that street scenes which are visually appealing will generally be thought to sound more pleasant than those which do not.

A number of factors known to influence the perception of sound have been outlined at the beginning of this section. In the interests of clarity these will henceforth be referred to as Factors Influencing Auditory Hierarchy (FIAH) and are illustrated in Figure 2.8. Furthermore, an organisational distinction is made between the physical properties of sounds themselves [46, 168, 169] and variation on the level of the individual which manifests in terms of sound context, training level and so on. Characteristics of sounds which may be indicators of their hierarchical placement are also highlighted, such as the presence or absence of humans, and discreet and continuous sounds. These will be referred to as Potential Hierarchical Indicators (PHI) .

This model bears a number of similarities to the Quality of Experience (QoE) model [170] which identifies 'Influence Factors' relevant to experience of multimedia content. The definition of QoE offered in [170] notes that it is influenced by "content, network, device, application, user expectations and goals, and context of use." (cited after [171]). Influence

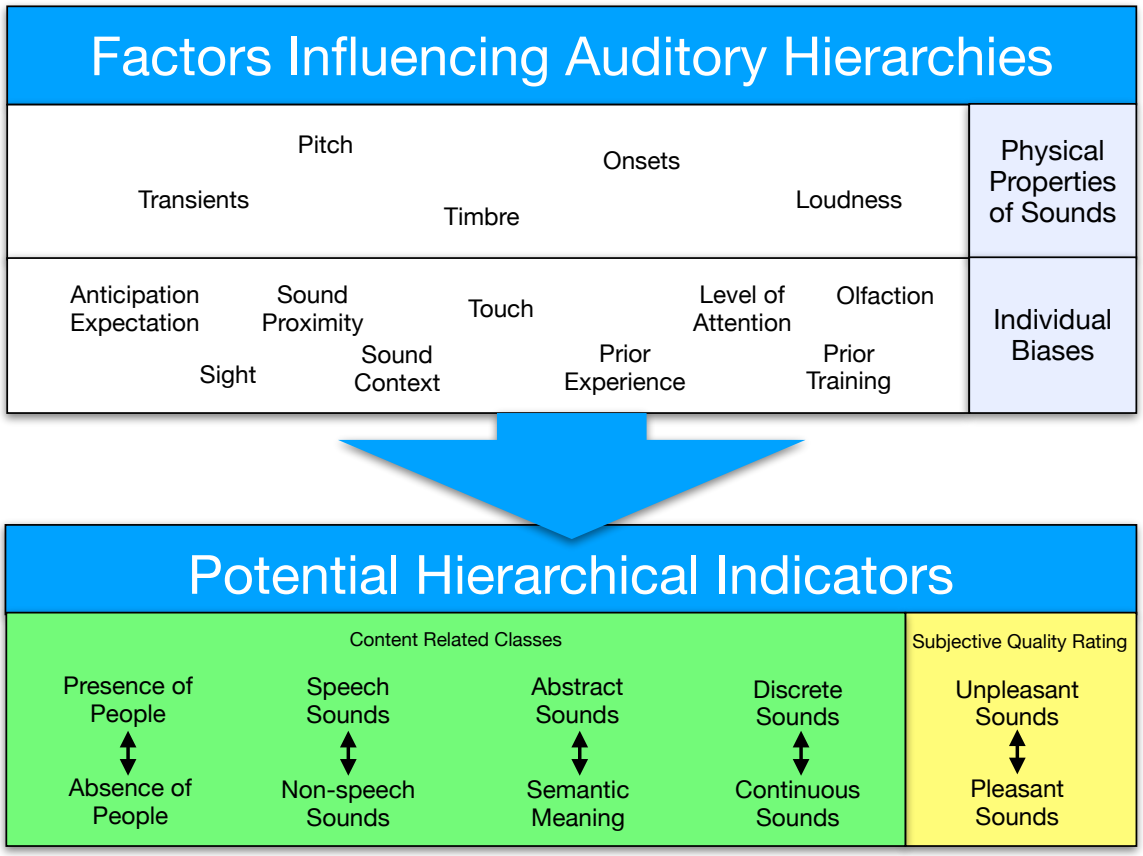

Fig. 2.8 A framework outlining factors which influence subjective hierarchical ranking of sound objects derived from the literature review outlined in Chapter 2 which shall be used to guide experimental design. These are components around which audio object hierarchy is hypothesised to vary.

Factors are noted as interrelated and grouped in three categories: human, system (notably including content related factors) and context. As the QoE model is concerned with experience as a whole it differs from the model of AH outlined in this thesis, but the similarities between each suggest that the hierarchical model has a sound theoretical base given the broad acceptance of the definition of QoE. The PHI outlined in Figure 2.8 can be considered as four content related classes and one subjective quality rating, as indicated.

Both this section and Section 2.2.1 have outlined evidence in support of the view that multiple factors, designated FIAH, affect human perception of auditory scenes and the focus of attention on individual sound objects in those scenes. This evidence suggests that

hierarchical perception is constantly in flux, that it is a continuum. It suggests that sound categorisation on a hierarchical scale is based on multiple FIAH, not confined to the relevance of a sound to current activities, the semantic information carried by the sound, whether the sound indicates human presence or perceived movement and sounds which are incongruous in the sound scene. This information elucidates the process of hierarchical categorisation which this thesis studies in a simplified manner, seeking to explore ML performance in the domain. In doing so, the intention is to offer a roadmap as to how this technology can be part of an intelligent system for content delivery optimisation. These thoughts will inform the design of subsequent investigations.

From similarities in groupings observed in these sources, the author has identified a series of characteristics, outlined in Figure 2.8 as PHI, which suggest relationships between sound types and define some dimensions of a possible categorisation space for hierarchical object classification. They may be of use in investigating the fluctuation of relative importance between sounds as a function of time. As conceptualised, PHIs suggest dimensions of sound hierarchies which range from those that indicate the presence of humans, to those that do not, between sounds which carry a high degree of semantic information about the object, action or event that caused their creation, and sounds that do not. This could also be referred to as the variation between *sounds often described by the event that caused them*, and thus easier to identify, versus *sounds often described using some abstract quality of the sound itself*, which are more difficult to identify. Further possible dimensions include continuous sounds (more likely to be BG and harder to identify) versus discrete sounds (connected to an object or event, easier to identify) and pleasant (people, nature, music, harmonic, lively ambience) versus unpleasant (traffic, alerts, inharmonic, alert) sounds.

Several studies examining the perception of acoustic scenes and the categorisation of sound have been reviewed in Section 2.2, Section 2.2.1 and Section 2.4.1. Subjective labelling of BG and FG sounds and testing of these labels in [47] and [59] suggest that this is a valid use

of such labelling systems. This provides a possible ground truth of hierarchical classification which may be of use when isolating test stimuli for new experiments once caution is used in their application, however, an entirely empirically derived hierarchical corpus would also undeniably be interesting for comparison if nothing else.

Many of the studies reviewed ([63, 65, 84], [58], [2, 57, 167]) reinforce the point made in Section 2.3 regarding semantic meaning in sounds. Additionally, Woodcock *et al.* [58] suggest one of the dimensions in which sounds are categorised pertains to the semantic information contained in a sound, or a lack thereof, and that this dimension was found to correlate with subjects' importance rating of individual sounds.

Evidence of an incongruency advantage in sound recognition [85] was also outlined. Further research suggests that the auditory system maintains a representation of the environment that is only updated when this is deemed necessary [61]. Additionally, Wustenhagen *et al.* [124], Mason *et al.* [117], and Marston *et al.* [111] observe that certain sound types, such as applause, are noted as being more difficult to encode than others. In a variable bandwidth situation, such sounds could be earmarked for high quality reproduction even if the model indicates they are of low importance. Sounds indicated as being FG can also be encoded at the highest quality possible, and minimum resources can be allocated to the less important stream. This points toward a working methodology which may be of use in applications of this research. A proposed outline of the temporal nature of change in hierarchical auditory perception is outlined in Figure 2.9.

This conceptualisation envisages a division between FG and BG sounds varying due to the FIAH outlined in Section 2.2 and manifesting in the PHI summarised in Figure 2.8. Sounds newly presented are evaluated and incorporated to perception and categorised as either FG or BG sounds as deemed appropriate. The process is continuous and fluid, meaning a constant re-evaluation of sound importance occurs, with sounds moving from BG to FG and vice versa. This outlines potential for a model predicting sound hierarchy which would

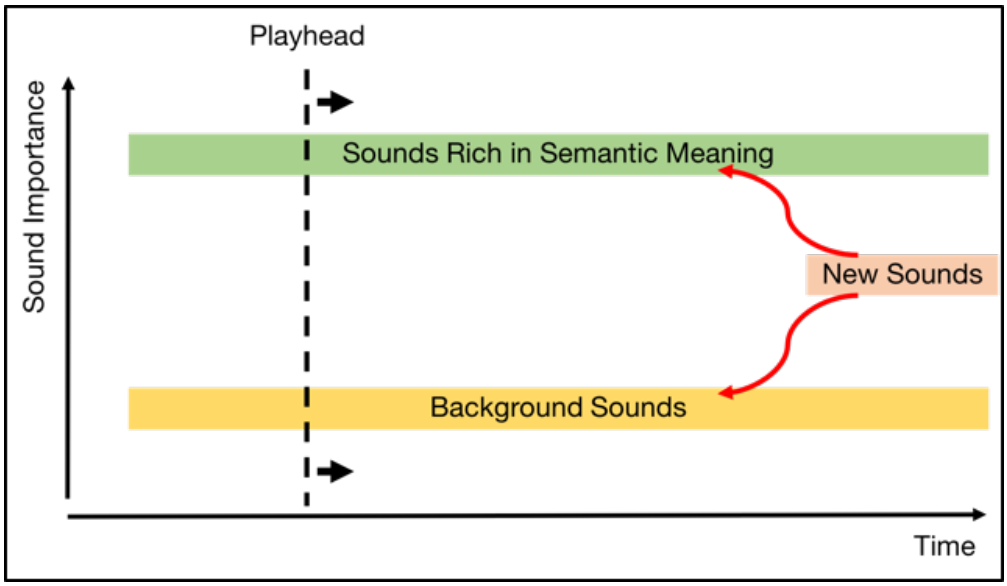

Fig. 2.9 A conceptualisation of temporal variance in auditory object hierarchy.

incorporate weightings for different FIAH which vary in the temporal domain due to new sounds being introduced. Such weightings would take the form of codec encoding parameters which would vary the weights accorded per PHI such as the presence or absence of humans, whether the sound contains semantic content or not, the degree of perceived movement in a sound and whether a sound is pleasant or unpleasant in addition to the FIAH mentioned such as focus of attention, sound context and so on.

Given the multi-faceted nature of AH it is logical to break down its functions to individual units of study, accepting that interaction is likely between them and each will require separate investigation. With this in mind, it was decided to first examine sounds in isolation from an auditory scene in order to establish what sounds are thought of as inherently either FG or BG. Additionally, establishing the feasibility of building a model using ML methods would constitute *proof of concept*, which could subsequently be augmented with other parameters as these are subjected to study. This positions the task of understanding an inherent hierarchy of importance between isolated sounds and examining how the phenomenon responds to ML analysis as a basic building block for understanding how a fully featured model could be

built. This could then be augmented by investigating the effect of context, attention, training and other FIAH.

Previous sections have derived an overview of AH as an initial step to building an ML model to predict the phenomenon. Section 2.2 has established ASA as a complex operation subject to multiple influences. Auditory object categorisation, of which hierarchical categorisation is a sub-task, is also subject to influence from FIAH outlined in Section 2.6, such as the inherent FG/BG nature of individual sounds, level of attention, sound event context and others.

This is a complex process with numerous interacting effects, which suggests that considerable detail and care must be undertaken in order to understand how hierarchical sorting works and how it might be robustly predicted for content delivery applications. Therefore, an investigation is proposed using sounds isolated from context, in so far as this is possible, as this will allow study of different FIAH as deemed necessary.

2.7 Conclusion

This chapter has provided an overview of ASA, introduced the concept of ObA and discussed these areas in the light of modern media consumption patterns. Auditory perception has been highlighted as a complex process subject to influence from a series of factors, identified as FIAH in Section 2.6, which require consideration in the formulation of a framework outlining AH and a method to predict audio object importance.

This has informed a discussion of sound taxonomies (Section 2.4.1) and available sound datasets (Section 2.4.3) which revealed no hierarchical classification data based on isolated sounds that were empirically derived. Some examples of arbitrarily-labelled non-isolated sounds that were subsequently validated by listeners have been previously mentioned ([47, 59, 63]), but these sounds are not isolated entirely from a sound scene and in some cases

are confined in scope ([59] for example is an entirely urban sound dataset). One of the contributions of this thesis is to derive this data.

Perceptual testing methods and listening test standards were reviewed in Section 2.5. The forensic level of detail afforded by the ITU-R BS.1116-3 and MuSHRA standards was deemed inappropriate for this study as the focus is not on BAQ differences between stimuli, but rather on respondent subjective judgement of the hierarchical placement of isolated sounds. This relaxes the necessity for laboratory listening conditions. Accordingly, it was decided to deploy subsequent experiments in an online environment as it has been shown, by Disley *et al.* [149] and McGraw *et al.* [150] for example, there is minimal difference between laboratory and online experiments for comparable tests. Furthermore, the potential response rate for an online experiment is far greater than that of one confined to a laboratory and the flexibility of the medium allows easy adaption at relatively little time cost should this be required.

Background research relevant to AH has been outlined, and a working concept for how such a hierarchy may function has been proposed. This is presented in Figure 2.9. The design of an experiment to investigate the existence of an AH between isolated sounds as an initial step towards designing an ML model to predict the phenomenon has also been outlined. Experiment 1 has been formulated to investigate isolated sounds only, as it is felt that this must first be established before other influences on hierarchy can be examined. The danger in not establishing such a baseline would be that future manipulations would potentially be open to multiple interpretations, thus invalidating any subsequent analysis. Once the nature of the relationship between isolated sounds is understood, then the empirical data derived can be used in further experiments.

The literature review offered in this chapter has addressed OBJ 1, namely:

OBJ 1: To develop an understanding of ASA with particular attention to the concepts of object-based audio, AH and modern media consumption paradigms.

The material reviewed in this chapter has identified FIAH and the value of directly investigating the hierarchical relationship between isolated sounds for media consumption applications. This has been addressed via the formulation of RQ1 and 2, as follows:

RQ 1: What factors are involved in the perception of AH?

RQ 2: Does a hierarchy of importance exist between sounds isolated from context?

The review of sound stimuli corpora offered in this chapter has established that existing datasets of sound stimuli do not provide scope for the study of FIAH in a suitable manner. This identifies a need to form such a database using sounds isolated from FIAH to the greatest extent possible, in order that a study can be made of each factor individually. Chapters 4, 5 and 6 will describe research relevant to this aim. In the first element of this, Chapter 4 will detail the implementation and results of Experiment 1, which also addresses RQ1 directly. First however, Chapter 3 will outline audio ML research in the context of sound object hierarchies in complex auditory scenes.

Chapter 3

Audio Machine Learning

3.1 Introduction

There is a considerable extant audio ML literature [14] and a rich recent history in the application of such knowledge to speech recognition [172, 173, 174], music information retrieval [175, 176, 177] and various automated personal assistant technologies such as Google Home [178] and Amazon Echo [179]. In this context, an ML analysis of the objective features of the stimuli used in the experiment described in Chapter 4 will be illustrative of the potential of such an approach for building a model to accurately predict AH as outlined in Section 2.6. This chapter will give a brief overview of ML relevant to concerns around building a model to predict AH. The choice of audio features will be discussed, as will the choice of ML algorithms for the specific task of hierarchical modelling. This will include a discussion of those referred to as Deep Learning (DL) algorithms, and an outline of the advantages and disadvantages of each approach generally. ML concerns specific to the audio domain will be covered in addition to methods for evaluating model performance.

This discussion will underline how important large volumes of labelled data are to the performance of supervised ML. Section 2.5 has introduced the area of perceptual testing in the audio domain which establishes the labelling task as an expensive undertaking in terms of the resources required to accumulate large volumes of labels. Section 2.4 has outlined the lack of an audio dataset consisting of isolated sounds labelled hierarchically, and selected the ESC datasets [101] as suitable for forming the basis of such a corpus. This has established the requirement for two tasks in order to form a dataset of auditory objects labelled hierarchically. Firstly, the ESC sounds require a manual review to select a subset which evinces a single auditory object only. This process is described in more detail in Chapters 4 and 5. Secondly, a hierarchical label must be sourced for each object. To address this second problem in Section 3.6 the technique of AL will be described as a method to reduce the manual effort required to label audio. In Section 3.7 methods for data augmentation relevant to audio applications will be discussed.

Two points on nomenclature are relevant at this point. In Section 2.6 we have introduced the concept of a 'model' for AH, which we use as a term in reference to the conceptualisation of AH, influenced by a number of different factors (FIAH), which we hypothesise to have varying levels of influence on the perception of AH. This is intended to be distinct from the ML concept of a 'model'. The terms 'algorithm' and 'model' are used frequently in the following. In ML parlance, 'algorithms' are held to be a framework of assumptions used to structure the prediction process. Logistic regression, Support Vector Machine (SVM) , Naive Bayes and Convolutional Neural Networks (CNN) are all examples of different algorithms. A 'model' is held to be a deployment of an algorithm with suitably fitted parameters, trained using appropriate data and methodology, which can be used to make an actual prediction, and this is the sense in which these terms are used in the following.

The remainder of this chapter will offer an overview of ML concepts and methodologies with particular relevance to the auditory domain. Section 3.2 outlines ML generally and will structure a further discussion around methods employed in this research.

3.2 Overview of Machine Learning

ML has been extensively and successfully applied to numerous audio problems. A full review of the area is beyond the scope of this work, however, a brief introduction is in order to outline the state of the art.

ML can be understood as the process of deriving insight into real-world processes by analysing patterns in data they produce. There are numerous flavours of ML, which can broadly be categorised into two types: supervised and unsupervised learning [180].

Supervised learning involves predicting some outcome from the analysis of data which must be labelled in some manner. The goal is to build a model capable of accurately predicting on unseen data instances. The label can be categorical, as with a *classification* task where the label is a discrete category, such as whether an image is a cat or a dog. The label may alternatively be a continuous number, as with a *regression* task where the result is a quantity which is continuous [181], such as trying to predict the price of a house. Examples of supervised algorithms are SVMs [182] and Linear and Logistic Regression [183]. Unsupervised learning, on the other hand, seeks to learn structure in data without reference to labels. For example, clustering algorithms, such as the k-Nearest Neighbours (kNN) algorithm seek to define the proximity of instances to each other identifying clusters of similar examples as demonstrated by Noda *et al.* [184], who classify fish species by clustering vocalisation data.

In the case of this research, perceptual hierarchical labels were required as an indication of audio object importance for modern media consumption. This informed the choice of supervised ML for subsequent work.

There is a distinction made between some supervised ML techniques, outlined in Section 3.5, which generally consist of minimal layers of abstraction between source data and model prediction, and the 'Deep' Learning class of algorithms [185], outlined in Section 3.5.5, that learn a function between the representations of data and a target output with multiple layers of abstraction. Furthermore, algorithms can be considered in terms of whether they are more suited to discriminative training, in that they learn the boundary between classes, or lend themselves more to generative training, in that they learn the characteristics of the distributions of different classes [180]. Additionally, some algorithms are known as *lazy learners* because they typically delay making a prediction until they are queried, as is the case with the Nearest Neighbour algorithms reviewed in Section 3.5.2 where query instances are compared to training instances in order to make a prediction. Other algorithms are known as *eager learners* because they construct some function during training which is then used to make predictions, as is the case with the SVM algorithms reviewed in Section 3.5.4.

The fundamental requirement for training accurate ML models is the availability of representative data. In the case of supervised learning, labelled data is required, and lack of such data is a noted problem in audio domains such as acoustic scene classification [17] and speech emotion recognition [18]. While large volumes of data are not necessarily required for all algorithms, it is a noted characteristic of deep learning models that they are capable of superior performance to other algorithm types once sufficient, large volumes of representative data are available [19, 20, 21].

This work focusses on using supervised ML methods in a classification task to predict AH. Section 3.3 first offers an overview of ML methodology and then introduces the areas of feature extraction and selection in addition to methods of model evaluation. Section 3.4 contains a review of audio features used in ML research and Section 3.5 outlines algorithm choice concerns. Finally, Section 3.6 reviews application of Active Learning methods to the auditory domain, while Section 3.7 considers data augmentations methods.

3.3 Machine Learning Methodology

A series of decisions are required during an ML project regarding a number of factors not limited to data treatment, feature representation, algorithm choice and evaluation method. In general, the ML process can be outlined as a series of iterative steps:

1. Domain knowledge accumulation.
2. Data gathering, analysis.
3. Feature representation and algorithm choice. Feature extraction and selection.
4. Training models.
5. Evaluation.
6. Predicting.

The iterative nature of the process is important, as learnings from prior stages can be recycled with further data analysis in an attempt to improve model performance. In many instances specific domain detail, such as the form of available data, will mandate choice of methods.

3.3.1 Feature Representation

Chapter 2 presents a summary of relevant perceptual audio research for the problem of hierarchically labelling auditory objects. This has outlined a broad range of possibilities in terms of *feature representation*. That is, how auditory objects will be represented in data form for input to ML models for this research.

In the case of some ML problems, the source data is already *structured*, presented in an organised table of *instances*, the historical events or items under analysis (in the case of this research, auditory objects), and *features* which describe the instance in some manner.

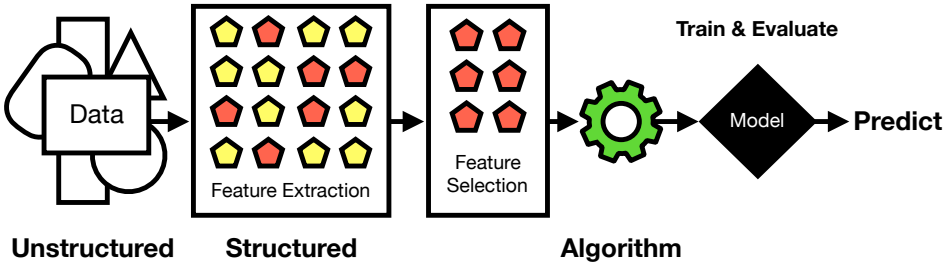

Fig. 3.1 An illustration of the data pipeline from unstructured data to prediction for supervised ML.

Features can consist of numerical, continuous, textual, categorical (cannot be ranked, such as countries, animals), binary (consisting of two categories, e.g. gender) or ordinal (categorical, which can be ordered in some fashion, such as small, medium, large) descriptions of instances. Often, however, the data is said to be *unstructured* and must be gathered and organised in order to be useful as input to an ML algorithm, as illustrated in Figure 3.1.

The broad breadth of features introduced in Section 3.4 outlines the scale of the task confronting practitioners in choosing which feature representation to use when building ML models. In the audio domain alone, numerous works exist demonstrating the superiority of one representation over another for certain tasks [172, 186, 187, 188]. Additionally, recent evidence [189] suggests that learning features directly from the data, as is the case with NN algorithms, is a more robust method of approaching sound classification tasks. Choice of feature representation is therefore a difficult task with multiple options, but central to the success or otherwise of final model performance.

The data pipeline from source to prediction can be summarised as presented in Figure 3.1. Given the wide range of potential representations and raw data forms, the data for this research is unstructured, requiring a number of decisions to finalise feature representation. Many of these are relevant to the process of *feature extraction* (surveyed in Section 3.3.4): The process of transforming raw data into features suitable for ML analysis. In the audio domain, feature types can be extracted for analysis at varying resolutions using a variety of temporal and spectral options. *Feature selection*, surveyed in Section 3.3.5, is the process

of removing features which do not help the ML task, as reducing dataset dimensionality by removing features has been found to improve learning performance in addition to lowering computational complexity and decreasing storage requirements [190]. Once a final feature representation has been identified, this can be used as input to train a model, which can then be used to predict.

3.3.2 Building Supervised Machine Learning Models

Once domain knowledge has been assimilated, data gathered and analysed, decisions can be made as regards which algorithm to use and how to approach feature extraction. The feature representations used in ML vary widely, not least because there are a number of possible approaches to feature extraction and selection.

An outline for the process of building supervised ML models is offered in Figure 3.2. In the training phase, features are extracted from the available training data and a model is built to predict an outcome, which in the case of this research is a categorical label pertaining to AH. In the classification phase, the model trained in the training phase is given unseen data for which it predicts labels.

When building models, it is desirable to do so in a manner that ensures maximum possible performance on unseen data. To do so, the available data is split into *training* and *test* data subsets, where the model is built using only the training data and the test split is used only to evaluate the model. Effectively, test data is treated as unseen data to give a robust assessment of performance. One method of implementing such a split is by using a *hold-out test set*, where a portion of the available data is selected at random to form a test set. The drawback of this method is that it runs the risk of returning a misleading result should the test split contain numerous instances that are easy to predict. Implementing repeated hold-out test sets is one way of controlling for this possibility, averaging the performance of models across each split to provide a final result [180].

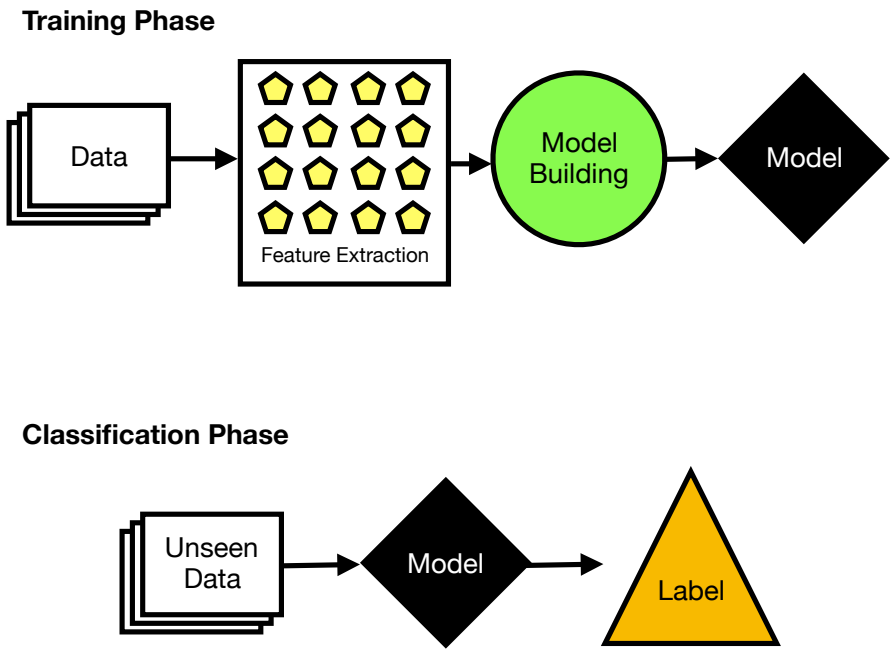

Fig. 3.2 An illustration of the supervised ML process.

A more comprehensive method for data splitting is *k-Fold Cross Validation*, where the available data is divided into k equal sized folds and each fold is used to build a model. Illustrated in Figure 3.3, each fold is divided into training and test splits, and a model is trained for each fold using only the training data for that fold and evaluated using only the test data for that fold. In this way, all the available data is used in both training and evaluating models. Treating the data in this way gives a more robust estimate of how a final model, trained on all the available data, will perform on unseen instances [180].

Once the method of splitting data is decided, algorithm *hyperparameters* can be fitted to each fold of training data by further splitting the training data into train and *validation* splits and subjecting the subsequent train split to a hyperparameter *grid search*. To do this a parameter grid is formed using a range of hyperparameter values and a model is trained for each combination using the train split and evaluated using the validation split. Figure 3.4 illustrates this split implemented using a 4-fold cross validation split with the 'Fold 1' training

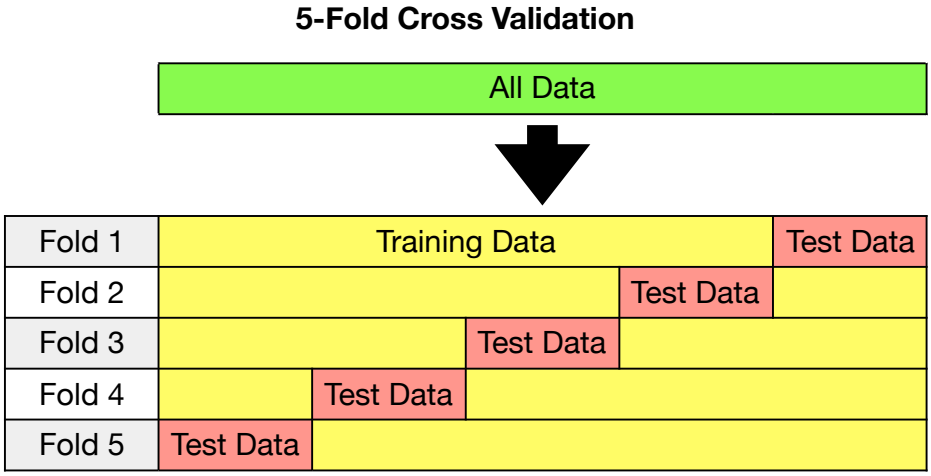

Fig. 3.3 A representation of 5-fold Cross Validation, which provides 5 different training and test splits, each of which are used to build a model.

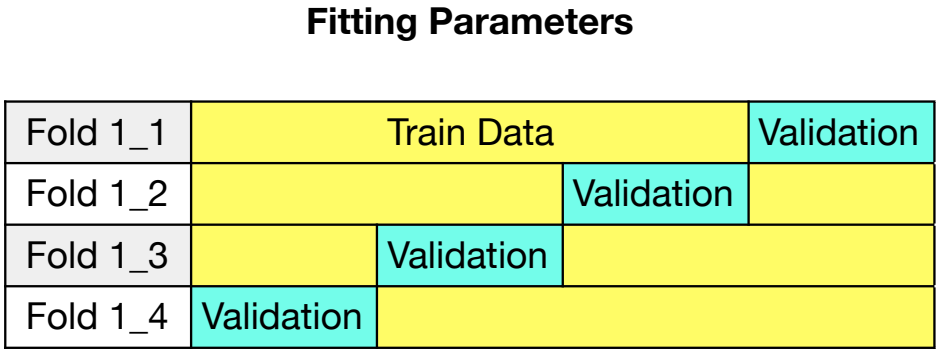

Fig. 3.4 A cross validation split implemented to fit parameters.

split outlined in Figure 3.3 divided into 4 further folds, designated 'Fold 1_1' through 'Fold 1_4', each with a validation split used to evaluate the performance of different hyperparameter combinations. In this example, each combination of hyperparameters is evaluated 4 times using different train and validation data. The performance of each combination is averaged over the four folds to identify the best hyperparameters. A model is then trained using the best set of hyperparameters and all train and validation data (the 'Fold 1' training data outlined in Figure 3.3 in this example) and evaluated using the test data for 'Fold 1'.

The size of *training* and *test* sets is generally set to 70-80% *train* versus 30-20% *test*, though this can vary according to dataset size. In scenarios with particularly small datasets,

allocating a 20% test set may lead to a non-representative sample giving a misleading impression of model performance, in which case the size of the test set should be increased. Equally, in situations where there the dataset consists of millions of instances, a much smaller test set can be implemented once there is confidence it will be representative of the dataset as a whole.

3.3.3 Algorithm Choice

The discussion around the general ML process outlines the numerous decisions that must be taken. These may be impacted by algorithm choice, which in itself is a major decision in the data modelling process. Each of the algorithms outlined in Section 3.5 have different inductive biases and sets of assumptions and there is widespread acknowledgement that there is no *best* approach that consistently outperforms all others. This idea even has its own term — the *No Free Lunch Theorem* [191]. Algorithm assumptions manifest in distinctive characteristics of decision boundaries they draw for classification tasks. For example, kNN models, will manifest notably jagged decision boundaries because of the influence of nearest neighbours. Decision tree boundaries will have a step characteristic because of the way instances are split in the tree, and so on. Which algorithm works best will depend on particular dataset characteristics, and for this reason it is advisable to choose a number of different models initially in order to evaluate the strengths and weaknesses of each as they pertain to a particular project [180].

A number of different factors feed into the algorithm selection process. Firstly, that of prediction speed. Some algorithms predict more quickly than others. Logistic regression tends to be quite quick to make predictions because the calculations involved are relatively simple in comparison with other ML algorithms. In contrast, kNNs can be very slow to make predictions as they must compare each and every instance in a training set, which can run to thousands of individual instances, thus adding to computation time. Secondly, the capacity

for retraining of the model can vary between algorithms due to how they accommodate new data instances. If new, previously unseen, instances are presented which are different from those on which the model was trained this can lead to *concept drift*, where the model no longer accurately reflects the data and ceases to make good predictions. In such a case, the model must be adapted to the changing data, generally by retraining using new data.

Finally, the degree to which an algorithm is interpretable or not may be of critical importance for the end model to be accepted. It will sometimes be necessary to derive insights into why the final model is making particular predictions. A rationale for how particular categorisations are arrived at can drive development of further research and deepen understanding of the processes being studied. This is not always possible for every ML algorithm with some, such as the NN family of models (see Section 3.5.5), still referred to as 'black box' models because of this interpretability issue. To this point however, Zeiler and Fergus [192] present a technique for visualising the inputs that are important to final outputs in some forms of Neural Network (NN) models.

3.3.4 Feature Extraction

The concept of feature extraction has been introduced within the context of preparing data for introduction to ML algorithms. Auditory stimuli can be represented in many ways, and for the purposes of a broad overview in the context of ML research this can be summarised as a process of extracting continuous, numerical data that describes an auditory event or 'object', as in the case of this research. A review of feature types is offered in Section 3.4 and all represent audio in numerical terms. Even the visual features outlined in Section 3.4.4 are fundamentally a pictorial presentation of audio data represented numerically.

There are a number of decisions to be made in the feature extraction process when dealing with audio data. Generally, short term features are calculated on a *frame* level. The temporal length of this analysis frame is a parameter that varies across the studies surveyed, and thus

requires consideration. Typical values observed in the literature reviewed range between 25 and 100 milliseconds. Furthermore, the step size, and whether there is to be any overlap between analysis frames, also requires consideration with typical values for this parameter ranging between 25 and 50% overlap.

The aggregation of analysis frame information over time is another possible point of variance. A number of possibilities exist in this respect, as outlined by Ruvolo *et al.* [193]. One approach, utilised by Barrington *et al.* [194], involves modelling the short-term features on a frame level and then combining them in a *bag of features* manner, with the overall categorisation being a product of the individual frame probabilities. Also, temporal data aggregation by summary statistics is commonly used, as applied by Grimm *et al.* [195], where mean, standard deviation and other summary statistics are computed for the frame-level features and these are used for modelling purposes. However, Lagrange *et al.* [72] question the sufficiency of these approaches for some applications, such as soundscape classification. While they acknowledge that soundscape categorisation is a difficult computational problem they suggest that such temporal aggregation should be allied with methods which align individual sources with what they term sound 'textures' [72, p. EL491] such as urban or rural sound settings, in order to provide more scientific rigour. With regard to building a model to predict AH, such additional information could be derived from subjective ratings provided in the experimental process.

A number of the algorithms reviewed in Section 3.5 do not directly address the variation of audio data over time, though Hidden Markov Models (HMM), capable of describing a sequence of events where the probability of each event depends only on the state attained in the previous event, may be used to do so [196]. If SVM, Linear Regression, Decision Tree or NN algorithms are to be deployed, some method of introducing the temporal variance of sound data should be attempted. In this case, the use of *delta values* between frame analysis windows may be deployed as in [197]. A *delta* value, also referred to as a *first order*

delta value, is obtained by calculating the difference between two adjacent analysis frames of the same feature, for illustrative purposes these can be labelled frames 'A' and 'B'. A *double-delta*, or *second order* delta value is obtained by calculating the difference between frames 'A' and 'C', 'C' being the direct neighbour of 'B'. Such delta values can be calculated for increasing orders if deemed necessary by the experimenter.

Finally, consideration must be given to the treatment of the data thus derived. Outlier values can cause problems in any statistical analysis process, and ML is no different. Both normalisation (adjusting feature values to a common scale) and standardisation (re-mapping a feature to fall within a number of standard deviations of the mean value of the sample) [180] techniques are available to the experimenter and there is no rule of thumb as to which should be applied in particular situations. Again, the decision is generally subject to a degree of experimentation before a final model is derived.

Numerous software environments exist in which it is possible to extract features from audio data. Matlab [198] is a well established and widely used programming environment designed specifically for scientific applications. A number of bespoke libraries [199, 200, 201] have been written in Matlab specifically for the purpose of audio feature extraction. Similar libraries [202, 203, 204] have also been written for the R [205] and Python [206] environments, both of which are well established and extensively used for scientific experimentation and data modelling purposes. There also exist a number of bespoke audio analysis programs such as MARSYAS [207], Praat [208], which was specifically designed for phonetics analysis but has been applied to other domains, and openSMILE [209]. All of these programs offer access to an ample selection of features to choose from, and in the case of Matlab, Python and R, a framework within which to calculate bespoke features should these be deemed necessary.

The existence of numerous libraries for feature extraction means that it is relatively straightforward for the experimenter to simply calculate a series of standard features and evaluate each in terms of their suitability of the task under consideration. Having said

this, in practice automatically extracting a broad range of features may ultimately make interpretation of the results more difficult as well as causing unnecessary logistical difficulty (e.g. increased training times). It is noted that feature extraction should ideally be guided by the intended final application, as some features may be more appropriate than others in specific cases. Additionally, evaluating feature performance on an isolated set of sounds does not guarantee that similar performance can then be assumed for all sounds. Some features may not be as discriminatory, could potentially be outperformed by features previously deemed unsuitable, or the relative importance level of auditory objects may change. In practice, different configurations would need to be evaluated before finalising choices for real-world applications.

3.3.5 Feature Selection

Feature selection, the elimination of redundant features from training and test sets, is important as this alleviates the so-called *curse of dimensionality*, where few instances n are spread over many features p, meaning that the target feature to be investigated is distal in p-dimensional space to its nearest neighbouring instances [210]. This section offers a compact overview of the area, a more complete review is offered by Özseven [211].

There are a number of ways to identify the features with high predictive power. *Filter-based* approaches involve ranking each individual feature according to some predefined metric. *Information gain* (a measure of the amount of information a feature brings to a training set) and *information ratio* (information gain divided by the amount of information used to determine the value of the feature) are common metrics to use in such an instance [180]. Some algorithms, for example Random Forest (RF) [212] , offer a grade for each feature included and this can be used as a selection metric also. Thus ranked, those features below a determined cut off point are discarded and modelling proceeds with the most informative features only. This makes the model more efficient. However, as the predictiveness of the

features is evaluated in isolation, this excludes any possible interacting features which could potentially be more informative than the features evaluated in isolation.

So-called *wrapper* approaches attempt to incorporate interacting effects among features by searching the feature set for subsets that perform best. This involves generating subsets of features, commonly using either *forward sequential selection*, where the search starts with no features and gradually adds single features, or *backward sequential selection*, where the search starts with the full feature set and iteratively eliminates single features from each subsequent trial. Individual feature subsets are evaluated according to the potential performance of models based only on that subset.

Feature selection is also possible via a process of Principal Component Analysis (PCA) , a commonly used statistical technique for finding a linear transformation for dimension reduction [213]. If each feature is considered a potential axis along which data points can be plotted, PCA sees these axes being transformed such that feature vectors are orthogonal to each other. This resolves the feature set into a lower dimensional representation, thus filtering the potential input features for modelling [197].

It is important that feature selection is performed on the data in the training set only. This point is noted as often neglected in the literature Hastie *et al.* [214, p. 245], who point out that feature selection on all instances in the dataset gives the model an unfair advantage in that the most important features will have been selected having *seen* both training and test data. Performing feature selection — on the training set only — ultimately leads to more robust models that will generalise better to unseen data. Deep Neural Network (DNN) models differ at this point as they essentially learn features directly from the data itself without outside intervention.

3.3.6 Model Evaluation

The final step in the modelling process is measuring the performance of the models trained. There are numerous metrics used for this purpose, the applicability of which will vary for different use cases and scenarios. In other words, there is no acknowledged consensus on which metric is useful for every particular application [215]. The following section will offer a brief summary of those utilised in the course of this work. An extended version of this section is offered in Appendix B.

In the context of supervised ML categorisation task, the predictions made by a model can be compared to the actual categories to generate a number of metrics. Those used in this work include:

- **Accuracy** — In total, what percentage of predictions made by the model are correct?
- **Average Class Accuracy (ACA)** — Sometimes referred to as 'balanced' accuracy, where individual class accuracies are averaged.
- **Precision** — What percentage of instances predicted as YES are correct?
- **Recall** — What percentage of YES instances are correctly predicted as YES?

Choice of metric can be highly dependent on the intended use for the model. For example, should identification of a maximal number of instances in a particular category be deemed important, then thought should be given to utilising the recall measure. If correct identification of all instances in a category is deemed important, this would suggest the precision metric should be used.

There are numerous other metrics available to the machine learning practitioner. One other was utilised extensively over the course of this work. A variant of the Area Under the Curve (AUC) metric was used during the experiment outlined in Chapter 5. An AUC value calculates the area underneath a curve which outlines model performance at varying

categorisation thresholds. In this way, different models can be compared using a single digit for each.

This section has offered an overview of ML methodology. The next section will review commonly used feature representations for audio ML applications, as this will inform our subsequent review of algorithms and techniques for the domain.

3.4 Audio Features for Machine Learning

Feature extraction and selection is a significant element of any audio based ML process given that there exist many possible ways to represent sound events in data form. Numerous spectral and temporal features of sound files have been used in a series of audio experiments with varying degrees of utility. These include Low Level Descriptors (LLDs) , such as the time and frequency domain features summarised in Sections 3.4.1 and 3.4.2, and 'global' features such as spectrograms, outlined in Sections 3.4.3 and 3.4.4, so called as they summarise an entire audio event from beginning to end in one representation.

These sections offer a brief summary of features used in audio applications, together with a commentary on the extraction methods employed. Much of this overview is based on the in-depth reviews offered by Mitrovic *et al.* [216] and Alías *et al.* [186], who detail a taxonomy of audio features which delineates them by temporal, frequency, cepstral and other features.

3.4.1 Time Domain Features

Features in the time domain represent how a signal changes over time and include measures of amplitude change, Zero Crossing Rate (ZCR) and power change of a signal. Such measures can include the Root Mean Square (RMS) of a signal, which mainly describes the power envelope of an audio signal. RMS is usually thought of as an approximation of the volume

of a signal [216], and is derived by calculating the mean root of individual values from each analysis frame. Comparable to these are the maximum absolute values of each frame, known as amplitude envelope values. ZCR measurements, in their simplest form simply a count of the number of times an audio signal crosses zero level, can be used as an indicator of the perceptual attribute of brightness of an audio signal. Autocorrelation, a measure of the self-similarity of a time series derived by multiplying the signal with a delayed version of itself, can be used to extract periodicity information about the signal [217].

Also used are statistical and cumulative distribution, mean and statistical noise levels: L_1 L_{10} L_{50} $L_{99.}$ These are indicators of dynamic properties as outlined in [218]. A statistical noise level of L_{99} for example, indicates the noise level that has been exceeded in a particular piece of audio data for 99% of the audio excerpt length. Other measures such as Sharpness, Roughness, Fluctuation strength, Tonality and measures of temporal variability as outlined in [219] are also candidates for use.

ZCR features have been used in soundscape context classification [197], sound recognition feature comparison [220], environmental sound classification [221] and animal sound classification [188]. RMS measures, or measures derived using them, such as Crest Factor, have been used in speech/music categorisation studies [222], environmental sound identification [90], urban soundscape differentiation [223] and speech segmentation [224] tasks. Autocorrelation features have been used in various sound categorisation [84], environmental sound categorisation [62] and soundscape evaluation [225] tasks.

The variety of applications outlined suggests that time domain features have proven useful in a number of different audio ML tasks. However, from this research, there is no single gold standard feature set or extraction method suggested. To summarise, the level of usage suggests the efficacy of this feature type, however there is no consensus as to the superiority of ZCR over RMS measures or vice versa, for instance. The numerous frameworks available make the task of extracting common features relatively straightforward, and a frequently used

approach is to extract a broad battery of features as opposed to minimising those selected via guesswork. Some frameworks, such as openSMILE [209], even provide pre-formulated feature sets for common audio ML applications. It is therefore easy to include these features and then implement a feature selection exercise, if required, to isolate the most informative features for the task at hand.

3.4.2 Frequency Domain Features

Frequency domain features reveal information about the spectral content of a signal and can be used to analyse the harmonic structure, bandwidth and tonality of a signal. They include features such as brightness, pitch, harmonicity and short-time Fourier transforms of a signal.

These features can be approached in a number of different ways. Linear Predictive Coding (LPC) is used to estimate parameters of a signal by predicting the value of a sample based on the values of previous samples [216]. As it accommodates the source-filter model of speech reproduction it has been used extensively in automatic speech recognition [172], and has also been used in audio context recognition [197], animal sound categorisation [226] and environmental sound classification [221].

The Short Time Fourier Transform (STFT) is extensively used in audio file analysis for time-frequency decomposition for feature extraction purposes, being necessary for the derivation of measures such as spectral centroid and sharpness, which relate to the brightness and sharpness of sounds, and spectral flux, rolloff, entropy and crest, which relate to the tonality of sounds. Such measures have been used in a variety of audio studies, such as sound identification, by Yang and Kang [227] who use spectral flux and Ogg *et al.* [228] who use spectral centroid and flatness as features, environmental sound classification by Bountourakis *et al.* [221] via use of spectral centroid, spread, rolloff, skewness, sharpness and smoothness features and by Eronen *et al.* [197] who use spectral flux, rolloff and centroid and in bird species identification by Fagerlund [229] who also use spectral centroid detail.

Other tonal measures include pitch, chroma and harmonicity features. Pitch information has been used in musical genre classification [230] via the use of pitch histograms, sound categorisation [231] and soundscape categorisation [225]. Chroma features present audio information in the form of spectral energy divided by pitch bands, essentially a time-frequency distribution, that are designed to represent the cyclic attribute of pitch perception [232]. They have been used in audio thumbnailing [233], music information retrieval applications [234] and can be used to model listener response times in melody and harmony tasks [235]. A feature called chromatic entropy, a measure of the change in energy between frequency bands, has also been used in speech/music discrimination tasks, such as [236].

Similar to the other LLD features outlined in Section 3.4.1, frequency domain features have been utilised in a broad variety of audio ML tasks and are relatively easy to extract using the various software frameworks available, outlined in Section 3.3.4. Once more, the literature reveals no superior subset of frequency features which provide universally high accuracy levels in a variety of tasks. Given the level of usage, it then seems wise to include them in a feature extraction exercise in addition to time domain features. It is interesting to note, however, the general predominance of global summary features, such as spectrograms, in the more recent literature, as evinced on the various DCASE challenge leaderboards [237, 238, 239], for instance.

3.4.3 Cepstral Domain Features

Cepstral domain features are based on the inverse Fourier transform of the logarithm of the estimated spectrum of a signal and were initially extensively used in speech analysis applications [240]. The cepstrum in general can be thought of as giving information about the rate of change of a signal in different frequency bands. Pitch information is known to be particularly strong in the cepstral domain for vocal signals because vocal formants and pitch

excitations are additive in the logarithm of the power spectrum, which helps to delineate them [241].

Cepstral features, primarily Mel Frequency Cepstral Coefficients (MFCC) are generally thought to capture timbral information well and have been widely used in various audio specific ML applications [216]. Specifically, they have been used in music information retrieval [230], automatic speech recognition [172], acoustic scene [242] and animal sound categorisation [184, 188, 243] in addition to general sound classification [197, 221, 244, 245] tasks.

MFCCs were initially designed to model the human vocal tract and were implemented for speech analysis and recognition applications [246]. If the vocal tract is thought of as a source-filter production model, MFCCs are designed to mainly discard the source element, with the result that they are somewhat pitch independent [247]. MFCCs as they are generally implemented are also phase-blind, meaning they lack finely grained temporal information [248, 75]. Yet, MFCCs have been extensively and successfully used in music and environmental sound analysis tasks where pitch would be a critical factor. Furthermore, they provide a global summary of an audio event in a single representation, which in theory bundles a number of the LLDs outlined previously together. The widespread utilisation of MFCCs across many audio applications speaks to their efficacy in multiple scenarios.

3.4.4 Visual and Other Features

Numerous other features have been extracted from audio files that are designed for a specific purpose [186]. MFCCs are often treated as a visual input, given they are a global representation of temporal changes in energy at different frequencies. There are numerous variants on the standard MFCC feature, chroma features being one such example, which organise the frequency representation in terms of musical semitone spacing. Another variant is Log-Power Mel Spectrogram (LPMS) images, which scale the power representation in

decibels, generally providing a more informative image feature. LPMS features are noted as being very popular in state-of-the-art audio deep learning research [20] and have been used in a number of audio ML tasks [249, 250]. Image-based features have also been used in sound event recognition [251] and a robust sparse spike coding of a 40-dimension Mel-filtered spectrogram is used in [252] for a sound event classification task. Gammatone Cepstral Coefficients (GTCC) are computed using the same method as MFCCs but by using a Gammatone instead of a Mel filter bank. They have been found to give a good approximation of the human auditory systems' impulse response, magnitude response and bandwidth [186], and have been used in computational auditory scene analysis [253] and road traffic noise mapping [254].

Electroencephalogram (EEG) features are generated from EEG signals of a subject who is listening to an auditory stimulus, their favourite music in the case of [255], who generate features from the EEG signals as well as RMS, ZCR and others, which they then correlate with the EEG signals enabling the generation of EEG data directly from the audio features. Purwins *et al.* [20] also note that raw audio waveforms are popular as input for deep learning approaches. Other features used in ML audio tasks include those that capture activity in the frequency modulation domain designed to represent the hearing percepts of fluctuation strength and roughness, and have been used in music and sound categorisation applications [256]. Phase space reconstruction features can approximate the non-linear behaviour of a system, which other features represent poorly, and estimate its entropy. They are usually used in combination with other features and have been successfully used in music genre classification [257]. Perceptual Linear Prediction (PLP) attempts to represent spectral contour more accurately by predicting future values based on prior occurrences and including some human auditory system inspired elements such as the use of a Bark frequency scale and asymmetrical critical-band masking curves [258]. They were specifically designed for speech analysis purposes and have also been used in infant crying sound event

recognition [259]. Relative Spectral-Perceptual Linear Prediction (RASTA-PLP) [173] is a version of the PLP method which bandpass filters each frequency channel in order to derive a more noise-robust feature [186]. It has been used in speech recognition [172] and animal sound recognition [188], but may not be optimal for an environmental sound classification task as the effect is to suppress components of spectral variations that are not speech related [248, 84].

Finally, there are also a number of examples of *end-to-end* systems in the literature. In these cases a neural network architecture front end is used to learn features of the audio input which are then fed to a back end classifier. For example, Mao *et al.* [260] attempt automatic feature learning on an emotion in speech task and achieve performance superior to that based on LLD-type features alone. Abdoli *et al.* [261] use a similar approach on an environmental sound classification problem and outperform approaches that utilise handcrafted features or spectrograms. This suggests systems which learn a feature representation directly from the audio signal hold much promise in audio ML research and deserve consideration for applications where sufficient data is available to implement a deep learning approach.

3.4.5 Summary of Feature Types

This section has reviewed the rich variety of feature representations available for use in audio ML research. It is difficult to isolate a single approach that outperforms all others consistently, though image-based features are utilised extensively in more recent research. This is perhaps due to the popularity of deep learning approaches, such as CNNs, which are designed to take a visual input. End-to-end architectures have shown promising results, but these are also built using deep learning methods which depend on having access to large amounts of data which may not be available for every application. LLD inputs, while outperformed in some cases, cannot therefore be entirely ruled out of consideration as they have been shown to perform reliably in a variety of applications. This suggests that, for any exploratory work,

a mix of both LLD and global feature representations should be first experimented with in order to establish their efficacy for the ML task at hand.

Section 2.5 has noted the similarity between hierarchical and environmental sound classification. Linkage has also been drawn in Section 2.6 between the theoretical AH model proposed and the QoE model. Emotion detection [175] is also a related research area. While similar, these categorisation tasks entail differing evaluation in terms of the level of focus on the sound itself and the listener's perception of the sound. Sound categorisation requires the object to be identified and is inherently an objective task. Hierarchical categorisation, quality assessment and emotion detection are more subjective and subject to differing opinions and thus may merit selection of different features for a prediction task. Furthermore, AH is predicted in this work using solely audio features, independently of sound labels or categories. While the ML work described in this thesis focusses on predicting AH to a context free binary level it should be emphasised that this is a simplification of the theoretical model presented in Section 2.5 which allows for a wider understanding of hierarchy.

Section 3.3.4 has noted that feature choice should be driven by the intended application. For example, Section 2.6 has noted that certain sounds, such as applause, are more difficult to encode transparently. This indicates that quality could be a consideration which should influence feature choice if a multi-faceted approach is identified as being necessary. This acknowledges that object identification, hierarchical classification and content clarity are distinct considerations where LLD or LPMS may prove more or less appropriate.

The discussion presented in Section 3.4 demonstrates the rich variety of visual feature representations used in audio ML tasks. It is difficult to assess which of those outlined is optimal for a specific application. It would seem logical, therefore, to begin by investigating those in most common usage, such as MFCCs and LPMS representations, and to focus on variants of these subsequently should this be deemed necessary.

Algorithm choice is one factor that bears relevance to representation selection. For example, use of a CNN algorithm suggests use of a visual feature representation as input, given the algorithm's popularity in the visual ML domain. The next section will review audio ML literature for algorithms in common usage and assess which is most appropriate for an investigation of AH.

3.5 Algorithms for Audio Prediction

There are numerous examples in the literature of various ML algorithms applied to different problems in the audio domain, such as soundscape classification [47, 197, 262, 263], animal classification [184, 264, 265, 266] and environmental sound classification [267]. Those mentioned following are not intended to be exhaustive, merely to give context to those chosen for predicting AH in the following.

The following sections consider algorithms within the context of supervised learning, as that is the focus of this research. Information-based (Section 3.5.1), similarity-based (Section 3.5.2), probability-based (Section 3.5.3) and error-based (Section 3.5.4) algorithms will be reviewed, in addition to DL algorithms, which will be outlined in Section 3.5.5.

3.5.1 Information-based Learning

Information-based machine learning is based around calculating the reduction in entropy provided by splitting dataset instances using the most informative features. The most informative feature is used to split the rest of the data, and this process is repeated until all instances in the dataset are categorised according to the target feature. Both categorical and continuous data can be treated in this manner by discretising continuous data [180].

RF algorithms are an example of decision tree-based ML, the most common information-based ML approach. RF models incorporate elements of bootstrap aggregating and subspace

sampling. Bootstrap Aggregating (or *'bagging'*) is a process whereby a collection (known as an *ensemble*) of models is used, each trained on a random sample of the dataset using sampling with replacement. This results in an ensemble of models based on different instances of the dataset, each of which will vary. Subspace sampling is the process by which a subset of dataset features are used to build different models, thus introducing more diversity into the component models.

Yang and Su [262] have utilised RF in a 21-class soundscape classification problem and achieved 79.7% average class accuracy across the 10 classes reported using a composite feature representation on a dataset of 5,250 instances. Unfortunately, this study does not report accuracy for all classes. However, the results reported suggest RF are capable of high accuracy levels on some sound categories. Noda *et al.* [184] compared the performance of RF, kNN and SVM algorithms on a fish species classification task. They noted RF are slightly outperformed by both kNN and SVM models, but are still capable of 93.56% median accuracy on 102 different species. RF was also noted as being more time-consuming to train than the other two algorithms in this study. Malfante *et al.* [264] also approached a fish species problem using RF for both feature selection and classification, and compared the results with those achieved by an SVM model. They found little difference in terms of algorithm performance in this case. While RF are noted as outperformed by other algorithms on occasions, the degree to which this is the case does not suggest they should be neglected as an option for audio classification tasks.

Decision tree models are generally easily interpretable, which is very important in contexts where an understanding of how the model arrives at its decision is important. They can become unwieldy, and therefore difficult to interpret, when dealing with large amounts of continuous data, however. Decision trees also struggle with datasets containing many features, and most especially when there are few samples in these datasets. They are also known as eager learners and are not most suited to modelling change over time [180].

3.5.2 Similarity-based Learning

Similarity-based learning techniques consider the proximity of instances to each other as a means of clustering and thereby identifying similar instances for grouping or classification purposes. The distance between feature vectors (the feature representation for each instance) can be computed using a number of different measures based on instance location in a *feature space*, an n-dimensional space where all dataset instances can be plotted, where n is equal to the number of features in the dataset. *Euclidean distance*, is one such measure, which computes the length of a straight line between two points. *Manhattan distance* is another distance metric, so called because it involves calculating the distance between two points in a block layout system, as would be the case in Manhattan, New York [180]. Implementing a nearest neighbour algorithm thus involves plotting all training instances in the defined feature space and identifying the nearest neighbour(s) to each instance. This is known as the kNN algorithm, where k can be any value ≥ 1. In a case where $k = 5$ for example, this would mean the majority target level of the 5 nearest neighbours to the query instance would be used.

kNN models have been used in environmental sound classification by Wang *et al.* [267], who use a hybrid kNN/SVM method to achieve an average accuracy rate of 85.1% across 12 sound classes in a dataset with 527 sounds. Esfahanian *et al.* [265] use kNNs when classifying dolphin whistles, and find their performance slightly inferior (94%) to that of a SVM (98%). Han *et al.* [266] find kNNs effective in acoustic classification of Australian Anurans, achieving an average accuracy of 98% across 9 frog species on a small database of 54 total instances. Eronen *et al.* [197] refer to kNNs as the most straightforward classification method, and this view is perhaps reflected in audio domain studies which use kNNs as a comparator or baseline to other options which demonstrate superior performance [184, 197].

When used in a prediction task, similarity-based models attempt to rank or categorise dataset instances by comparing them to adjacent instances. This makes them easy to interpret, though consideration must be given to the choice of distance metric used, as these are

appropriate to different kinds of data. As these models make use of all the features in a dataset they are sensitive to missing values (not usually a grave concern when deriving objective measurements of audio files) and also outlying values as these can unduly skew results. Consideration should be given to normalising or standardising the data if appropriate.

Nearest neighbour algorithms are also known as lazy learners, because they do not abstract from the data until asked to make a prediction. Consequently, when the number of instances in a dataset becomes large, the nearest neighbour algorithm slows down due to the increase in instances it must check. In other words, where speed of prediction is a critical factor, nearest neighbour may not be the optimal choice. However, they can handle different types of training features, and they are robust to concept drift, as is the case in spam email prediction tasks for example. Each time a correct classification is made in such a task, the instance can be added to the training set and thus constantly update the algorithm as input evolves [180].

3.5.3 Probability-based Learning

Probability-based learning involves using estimates of likelihoods to determine the most likely predictions for a given dataset. This involves the constant revision of predictions based on the accumulation of more data and other evidence. Each feature in a dataset is regarded as a random variable, and the set of all possible combinations of all possible values for each feature [180] is the sample space for the domain.

This approach is heavily based on Bayes' Theorem, which states that the probability an event has occurred is equal to the probability of the evidence being caused by the event multiplied by the probability of the event itself. Essentially, if beliefs about the causes of an event are modified proportionally with how measurements relate to the potential causes of that event, beliefs regarding the events that occurred which resulted in our measurements can be altered and iteratively improved. This idea has seen application in many scenarios,

primarily through the Naive Bayes model, so-called as there is an assumption of conditional independence between dataset features. Cai *et al.* [263] use a Bayesian network-based approach to integrate prior knowledge and statistical learning to investigate high-level semantics of an auditory context. When predicting on a 10 class dataset of 12 hours of audio data taken from film and television effects tracks, they achieve recall of 87.6% and precision of 78.1%.

Bayesian prediction is problematic in that the number of probabilities grows exponentially with the number of features. This is generally tackled by reducing the interactions between features and the number of model parameters, known as factorisation [180, pg. 313], meaning that these models can be trained using a small dataset. Although this factorisation is based upon the (naive, and often incorrect) assumption that each feature is conditionally independent of all the other features, Naive Bayes models often perform well as long as the error in the calculated probabilities does not affect the rankings between target levels. As a consequence, Naive Bayes models are not generally suitable for predicting continuous targets. They are often easy to interpret because of the factorised feature space, as it is possible to analyse the probabilities for each feature to see how each affects the model categorisation, which can be a useful tool to use when building more complex models.

Hidden Markov Models (HMM) are used to represent probability distributions over sequential data [268]. As referenced in Section 3.3.4 they can be used to address the temporal variation in audio data by providing a probability for events dependant only on the state attained in the previous event. HMMs have been used extensively in speech recognition systems for this purpose with Gaussian Mixture Models (GMM), another probabilistic algorithm, used to assess the match between states of each HMM and the acoustic representation used as input [269]. While successful, recent research suggests that HMM/GMM systems can be outperformed by those using deep learning architectures in both speech and acoustic event recognition domains [269, 270, 271, 272]. It should be noted however that this is not universal, with Schroder *et al.* [273] reporting DNN systems as less accurate than a

HMM/GMM model in a sound event detection task based in a multi-source environment (analogous to everyday life).

3.5.4 Error-based Learning

Error-based learning is an approach that attempts to minimise the error in the predictions made by a model by varying model parameters until the total error, or cost, is minimised. This envisages the error from a set of parameters as a surface via which the point of minimum error can be calculated. This approach can be applied to linear, logistic and multinomial models as required [180].

A simple linear regression model can be built by plotting two continuous variables which convolve to a straight line on the graph. This line can be expressed in the form:

$$y = mx + b \tag{3.1}$$

Here, m is the slope of the line, and b is the *intercept*, where the line cuts the y-axis when x is equal to zero. Logistic models, which use the logistic function in binary classification, and multinomial models, used to classify instances on multiple levels, can be easily expressed using more complex developments of this equation using squared, cubed or higher order expressions to define models that involve one or more curves. The values of y and x in Equation 3.1 then represent model parameters and when one is known the other can be predicted. The distance between each data point and the model line represents the model error for that instance. By summing the squares of all the instances an overall measure of the accuracy of the model can be identified, and by minimising this number the most accurate model can be built. Iterating this process, which is known as *gradient descent*, involves sequentially modifying the model parameters to trial a series of different models. Over a sequence of iterations, this process finds the global minimum error point at which the model is most accurate.

Multiple linear regression is noted as extensively used to model subjective audio preferences to objective measures in a review of the area by Pietila and Lim [274], but the authors note the approach is limited in terms of the number of sounds which can be evaluated and the linearity of the dataset. Neural network-based approaches are noted as having the potential to address these issues, but the lack of interpretability and scarcity of sufficient data are highlighted as drawbacks for these model types. Härmä *et al.* [275] compared performance of linear regression and neural network models on a spatial sound experience task and found the linear regression model slightly superior, which they note as interesting. In addition, logistic regression has been used to model subjective perception of urban soundscapes from objective measurements [276] and perceived changes in perceptual properties of object-based audio treatments [9], but these studies do not compare performance with other algorithms.

Linear regression models are accepted as easily interpretable and engender a deeper understanding of the interactions between the features used in the model, although a high degree of domain knowledge is required in order to develop an accurate model. Furthermore, as the name suggests, a linear relationship is assumed between the features studied, and this may not be the case [274]. Conversely, logistic and multinomial models can model more complex relationships, but may not be as easy to interpret. Additionally, this capability comes with increasing complexity in terms of model implementation. Regression modelling is well established through extensive use in multiple areas of research, and thus its application to different areas of investigation is uncontroversial.

SVMs are another type of ML modelling that is based on error-based learning. They differ from the approaches described above in that they find a decision boundary that defines the greatest separation between data instances, and so result in more robust models. They have been used extensively in audio machine learning experiments, including audio scene classification by Jiang *et al.* [224], who find them capable of consistently classifying to greater than 90% accuracy on instances featuring pure speech, non-pure speech, music and

environmental sound content. It should be noted that the categories used in this instance mean the classification task is straightforward in comparison to more recent work which focuses on more difficult tasks. For example, McLoughlin *et al.* [277] compare SVM and DNN models in a 50 class sound scene categorisation task, examining the effect of varying levels of background noise on classification accuracy. Here, the SVM is observed to outperform the DNN model in clean audio conditions, but the DNN proves more robust to the challenging condition of classifying correctly in increasing levels of background noise. As noted in Section 3.5.2, Esfahanian *et al.* [265] use SVMs on an animal call species classification task, in this instance on dolphin whistles, and find their performance slightly superior (98%) to that of a kNN (94%).

Although the popularity and potential of deep learning approaches is notable in the literature surveyed, the lack of large, suitably labelled datasets is an acknowledged problem [17]. SVMs are prominent in the foregoing review for competitive performance levels on datasets that are small by the standards of deep learning. For example, Wang *et al.* [278] use an SVM in a 15 class environmental sound classification task and achieve 91.7% accuracy on a dataset of 677 instances. Furthermore, SVMs are robust to overfitting and perform well for problems that use a multi-dimensional feature space [180].

3.5.5 Deep Learning Algorithms

DL is a subset of machine learning based on the family of NN algorithms. NNs seek to learn hierarchical relationships directly from datasets and are inspired by observations of the way the human cerebral cortex deals with natural signals. They have gained a reputation as being good predictors for a wide variety of tasks and are considered state-of-the-art [17, 26, 27] in audio domains such as speech recognition [269, 270] and in computer vision [279, 280]. They have been shown to perform comparably to humans in sound categorisation tasks [197], particularly when considering higher-level contexts such

as ‘Outdoor’, ‘Vehicles’ or ‘Public/Social’ opposed to lower-level sub-divided contexts of ‘Outdoor’ such as: ‘street’, ‘road’, ‘nature’ or ‘construction site’. Furthermore, applications such as Google’s translator, street view, image search and Android’s voice recognition [281], Apple’s automated assistant Siri [282] and IBM’s brain-like computer [283] are based on DL algorithms.

Neural Networks

NNs are built on layers of *neurons*, mathematical representations of biological structures based on the concept of a *perceptron* introduced by Rosenblatt [284]. The inputs for the first layer of neurons is the initial dataset under analysis. At its simplest (see Figure 3.5) a neuron takes a number of inputs and introduces weights for each input which signify the importance of that input to the output of the neuron. Each input is multiplied by its weight, these values are summed, and a bias term is added. The result is applied to a non-linear function called an *activation function*, and this determines the output of the neuron. Different activation functions can be applied to control the output of the neuron. NN weights are usually randomly initialised to values close to zero [214] and the process of training then involves updating the weights so that the error of the network is minimised [285]. The bias value (sometimes referred to as an offset) is used to adjust the position of the activation function to better fit the product of inputs and weights.

In a NN, neurons are arranged in layers, as illustrated in Figure 3.6. The first layer on the left in this illustration is referred to as the *input layer*, and in this illustration consists of 6 neurons. On the right-hand side of this illustration is the *output layer*, which here consists of a single neuron. In between can be a number of *hidden layers*, so called because they are neither input not output layers [5]. NNs consisting of multiple hidden layers were possible but relatively unpopular until recent developments in computer processing power. Greater

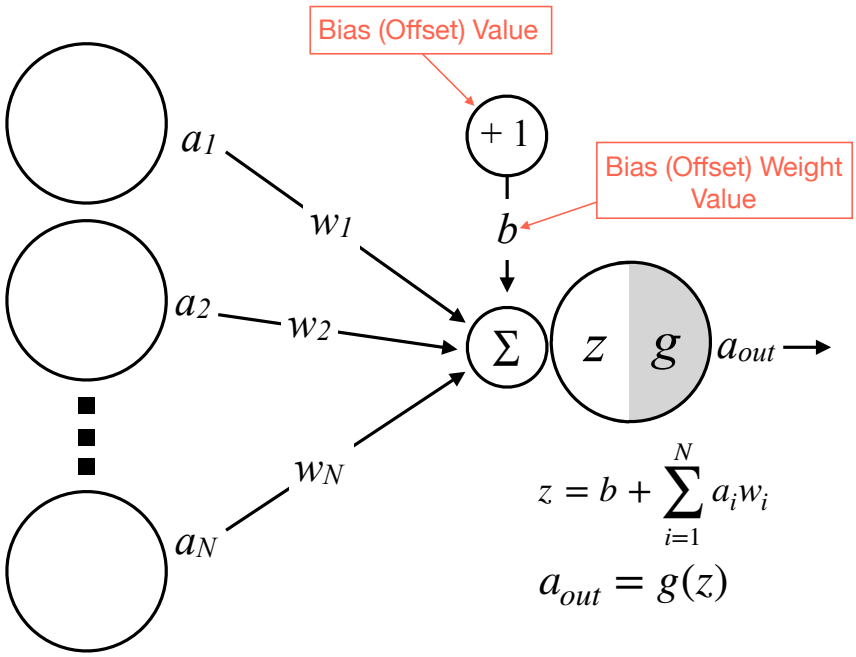

Fig. 3.5 A visual representation of a neuron showing input (a_{1-N}), weight (w_{1-N}), bias, and activation function (g) elements (reproduced from [4]). Here, z is the result of adding the bias term to the sum of the products of inputs, a, and weights, w.

computational capacity has led to NNs with significantly more hidden layers than before, leading to the term 'deep' learning.

A process akin to gradient descent, described in Section 3.5.4, known as *backpropagation* [196], then calculates the gradient of the error function with respect to the NNs weights and back-propagates this through the network of neuron layers. The gradients at each node are then used to update the weights before the next instance is presented. This process gradually refines to either a single output node, if the prediction takes the form of a number, or small number of output nodes, as in the case of a multi-class categorisation problem. A *learning rate* parameter is used to control the size of the update applied to model weights and can be adjusted if the network is found to learn inefficiently. *Optimisers*, such as AdaGrad (Adaptive Gradient [286]) and Adam (Adaptive Moment Estimation [287]) employ methods which automate learning rate adjustment and help to avoid error reduction getting stuck in local minima, thereby decreasing the time required to train models.

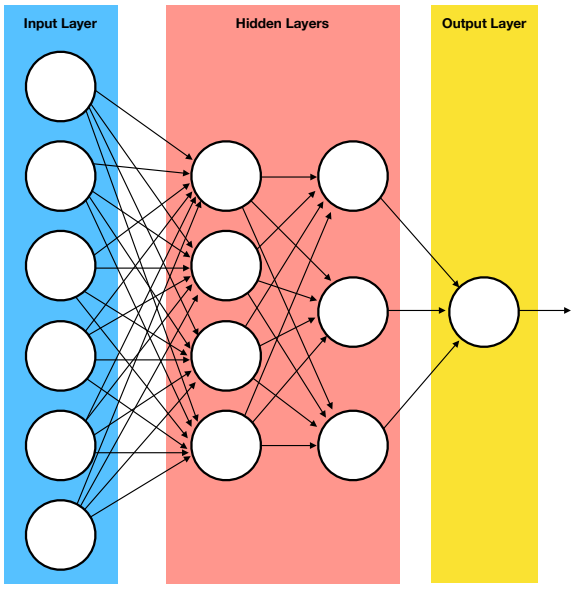

Fig. 3.6 A visual representation of a NN consisting of layers of neurons (adapted from [5]).

Different kinds of activation functions can be used when determining neuron output, though some of these (for example, *tanh*, *logistic* or *soft-sign*) are said to saturate when approaching maximum positive or negative values. In the case of a saturated node the gradient will be zero which means the weights will not update, and the node stops learning. This is referred to as the *vanishing gradient* problem [288], where updates to neural net weights decay exponentially, meaning over time the network ceases to learn [289]. Other activation functions, such as the Rectified Linear Unit (ReLU) or *soft-plus*, saturate only on negative inputs, or in the case of 'leaky' ReLU, not at all, which generally make them a better choice, though the choice of activation function is left to the discretion of the researcher.

For audio applications the scale of the data used is important, and it has been noted that coordinate-wise standardisation post a log-amplitude scaling of spectral magnitudes generally works well for a number of different audio applications [196]. An example of this is the LPMS representation outlined in Section 3.4.4. Neurons as described do not take full advantage of the temporal and spectral structure of audio data. This means that outputs can

vary widely, even if they are separated by only a few analysis frames. Variations of NNs exist which are designed to circumvent this issue will be outlined in the following sections.

The NN just described is known as a *feedforward* network — no feedback loops exist in the network, and information is always fed forward. The next sections will outline networks which use different architectures, such as CNNs, and sometimes contain feedback loops, as is the case with Recurrent Neural Networks (RNN) .

Convolutional Neural Networks

CNNs vary from standard neural networks in that they incorporate a form of subsampling with repetition termed *local receptive fields*, using *shared weights* for each hidden layer of the network and a process of *pooling* to simplify layer results. Local receptive fields essentially take a subset of the initial dataset and feed them into the next layer of the network, learning an associated weight and bias. This process is repeated with a degree of overlap such that the hidden layer consists of a different number of units, subject to the degree of overlap and number of units in the subsample. As the same weight and bias is applied to subsamples, each can be thought of as learning a single feature of the dataset from a slightly different set of data. Each set of units with the same weights and biases is referred to as a *feature map*, illustrating as it does a single feature. In order to learn another feature therefore, different weights and biases are applied similarly, building up a layer of feature maps which is referred to as a *convolutional layer*. Subsequently, the network will commonly consist of pooling layers after each convolutional layer. Each pooling layer simplifies the output from a convolutional layer, resulting in a representation of each feature map included in the convolutional layer that is smaller and thus easier to perform calculations on [290].

CNNs generally perform well in situations where the desired output is a series of predictions based on local interactions [196] and are able to extract features that are not influenced by local spectral and temporal variations [27]. They have been applied to a number of dif-

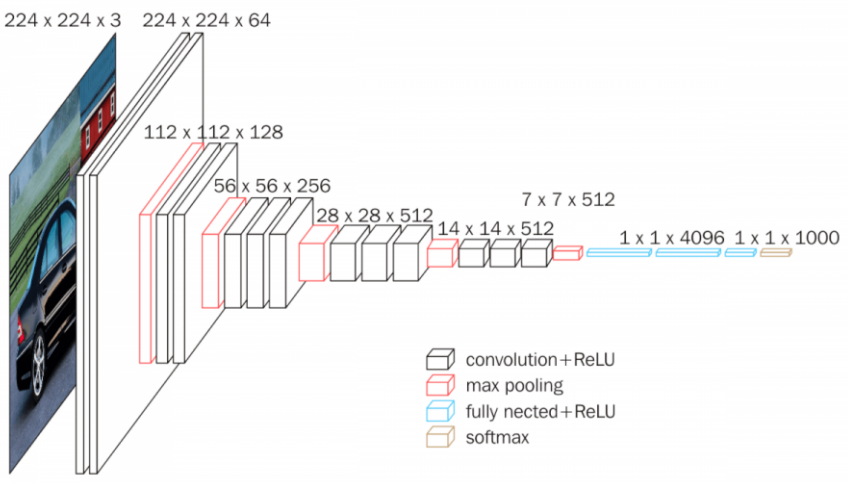

Fig. 3.7 An example of CNN architecture depicting the VGG16 model proposed by [6]. The image was sourced from *https://neurohive.io/en/popular-networks/vgg16/*, Accessed: 5th December, 2019.

ferent audio classification problems including environmental sound [104], soundtrack [291] and fish species [292] classification. Even a cursory review of CNN usage in the domain reveals a plethora of architectures, features and approaches. For example, Sharan and Moir [293] investigate feature representation types, comparing standard spectrogram, a frequency domain moving average representation they call a smoothed spectrogram, a mel-scale spectrogram and a cochleagram image which is based on the characteristics of the human cochlea. They find that the cochlear representation gives the best performance in an acoustic event recognition task. In an environmental sound classification experiment, Dai *et al.* [294] use raw waveforms as input to a very deep CNN (up to 32 weight layers) and report results that are competitive with CNNs using log-mel spectrogram inputs [106]. Kumar *et al.* [295] propose a transfer learning approach, where weakly labelled audio data is used to learn a representation which can then be tuned to state-of-the-art results on both ESC-50 and Audioset datasets. Hershey *et al.* [291] experiment with a number of CNN architectures drawn from the computer vision domain and report excellent sound event recognition results using AlexNet [296], VGG [6], Inception [297] and ResNet [298] configurations.

Given the rich variety of methods utilised, it is difficult to identify a reliable best practice implementation, however, the popularity of CNN approaches in successful solutions entered to significant domain competitions such as the DCASE environmental sound classification challenges [15, 21, 299, 300] speaks to their effectiveness in audio ML applications.

Recurrent Neural Networks

CNNs are feedforward networks where the route from input to output is fixed and unchanging. RNNs incorporate dynamic change over time to the network architecture and are noted as being strong in modelling long term temporal context in audio signals [27]. They have neurons which fire for a predefined, limited duration of time only. This activity can cause other neurons to fire, also for a limited time, which gives rise to the possibility not only of units from early layers activating those in later layers, but also vice versa. This behaviour can cascade through a network, resulting in multiple instances of such feedback loops which refine the models results. RNNs can directly model sequential information available from audio stimuli, and because of their feedback and feedforward nature can be said to 'remember' past states. This property can bypass the need for tailored postprocessing in some instances [289].

RNNs have been successfully applied to a number of problems in the auditory realm. Parascandolo *et al.* [289] have used them in a polyphonic sound event detection task using a Bi-directional Long Short Term Memory architecture , which was originally proposed to address the problem of vanishing gradient [288]. Long Short Term Memory (LSTM) architectures address this by implementing feedback structures which make it possible to propagate information from previous layers through the network. BiLSTM structures facilitate propagation of information from both previous and subsequent layers [301]. Such structures implemented as part of an RNN were found to surpass previous state-of-the-art performance in a sound event detection task [289]. Li *et al.* [19] also addressed a sound detection problem

using an RNN and compared performance with other DL algorithms (DNN and CNN) in addition to a GMM baseline. Using a number of different feature representations including MFCC and LPMS features, the authors found in this instance that performance varied for each algorithm and feature representation pairing, with both DNN (84.2%) and CNN (82.2%) slightly outperforming the RNN (80.2%) using an ACA measure. The best performing model was an ensemble which fused input from all three neural nets and achieved a score of 88.1%, indicating that diversity in algorithms and feature representations can augment overall performance.

These examples demonstrate successful applications of RNN to audio classification problems. It is interesting to note that in public ML competitions similar to that of predicting AH, such as the various DCASE challenges [237, 238, 239] it could be said that more of the superior performing solutions feature CNN, rather than RNN, architectures. This is not to suggest that CNNs are a superior solution to every audio ML application. However, in light of this, it is reasonable to suggest CNNs as the first choice for a new audio prediction problem.

Summary of Deep Learning

Deep ML algorithms such as those outlined above have in recent years become the most popular classification method for audio tasks such as environmental sound categorisation [299] as they have been observed to outperform other learning architectures consistently when trained on unstructured data. They are also capable of solving complex problems where other algorithms as yet cannot, such as recognising speech and objects [285]. Set against their considerable advantages are a number of disadvantages, however. They are more computationally expensive than other algorithms [302], arguably provide less insight into the data under analysis [192] and require far greater volumes of data in order to reach superior accuracy levels [299]. This suggests that they are a good choice for audio ML tasks in

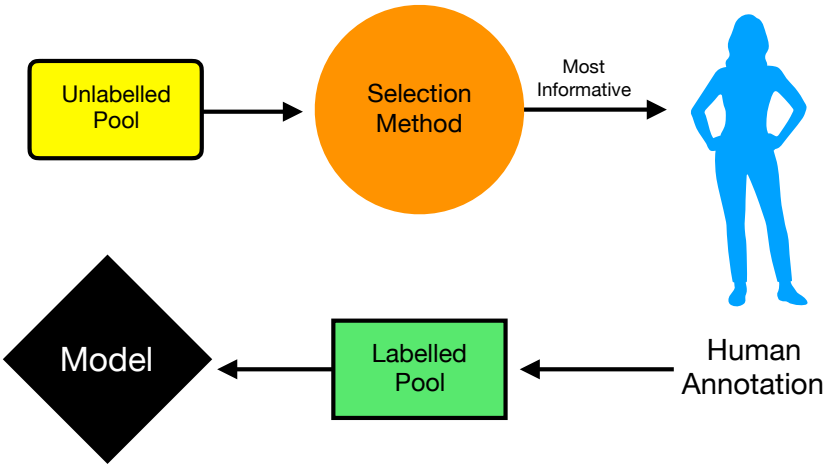

Fig. 3.8 An outline of the Active Learning process. The purpose of the selection method is to select unlabelled instances that will be most informative of the dataset. Once labelled by a human annotator, a model can then be trained on the instances in the labelled pool to predict labels for remaining unlabelled instances. Selection methods are outlined in Section 3.6.1.

situations where there is an abundance of processing power and appropriately labelled data, where they can be expected to provide higher accuracy levels than other algorithm options.

3.6 Active Learning

This work proposes addressing the labelling problem outlined in Section 3.2 using a combination of crowdsourced labels, AL and data augmentation. Methods for crowdsourcing labels have been outlined in Section 2.5.3. This section will outline AL as a ML method which can be used to label large numbers of instances with minimal manual effort. Data augmentation for the auditory domain will be examined in Section 3.7.

AL is a supervised ML technique, originally designed to build classifiers with minimal manual labelling effort. It can be used to label large datasets [303]. As outlined in Figure 3.8, an unlabelled pool of instances can be assessed for informativeness using a selection method, reviewed in Section 3.6.1. The instances deemed most informative can then be removed from the unlabelled pool and presented to a human oracle for labelling. The AL process

is applied iteratively, and more instances are presented for labelling, reducing the size of the unlabelled pool, until either the performance of a model trained on the labelled pool reaches a predetermined level of performance, a label 'budget' is reached or there are no more instances to label. The objective is to use minimal manual effort to label the entire dataset to the maximum level of accuracy possible.

3.6.1 Selection Methods

There are a number of methods for selecting the most informative instances from an unlabelled pool. The following is a non-exhaustive summary.

Uncertainty Sampling Active Learning

The most common selection approach [303] is Uncertainty Sampling Active Learning (USAL) which uses uncertainty in model prediction as a metric to select instances for labelling. The hypothesis behind USAL holds that the instances about which the classifier is most confident will provide the least useful information and that the instances most difficult to categorize will be more informative, allowing greater accuracy from fewer manually applied labels. It therefore selects these instances for labelling first.

Uncertainty can be identified in different ways. Settles [303] identifies three main methods. The least confident method ranks classification confidence based on the best prediction for a single instance. The predictions are ranked and the lowest ranking instance, that which the model is most uncertain, is presented for labelling. The margin method ranks instances by their proximity to a classifier decision boundary, presenting those closest to the boundary for labelling first, as they are the instances most difficult to categorise. These methods are limited in the case of a multi-class categorisation problem as they use only information on the most confident, or two most confident predictions to rank instances. An entropy measure can also be used to assess the average information content of an instance.

Settles [303, Pg. 16] notes that the entropy approach is most appropriate where the objective is to reduce log-loss and the margin and least confident methods are more appropriate if the desire is to reduce classification error, with the margin method being slightly more powerful as it uses more information to arrive at a decision.

Query-by-Committee

Query-by-Committee (QBC) uses an ensemble of models to select instances on which the models in the committee disagree the most, viewing these as most informative for labelling purposes. It relies on the theory that each classifier used will have a slightly different interpretation of the data, resulting in differences to class predictions.

Some general points about model ensembles are relevant to QBC. Committee members can be differentiated either by varying model parameters [304] or by using a 'bag-of-classifiers' [305]. Seung *et al.* [306] suggest that using a small number of classifiers is adequate and Melville and Mooney [307] note that using diverse classifiers is advisable.

Expected Error Reduction

Expected Error Reduction (EER) is an approach that uses the generalisation error of a model as a selection measure. Every unlabelled instance is ranked using an estimate of the degree to which the model's error will be reduced should it be labelled. Those which reduce the model's error the most are selected for labelling. This method has been successfully used for text classification [308], but is computationally expensive [303, 309].

Exploration Guided Active Learning

Exploration Guided Active Learning (EGAL) identifies useful instances for classification purposes in relation to their location in the feature space relative to neighbouring instances and proximity to already labelled instances. It has been used in text classification applica-

tions [310] but not to our knowledge on an audio domain problem. It differs from the other selection methods outlined above in that it is not dependant upon a model to select instances. EGAL seeks to identify instances in clusters that are furthest from labelled instances on the assumption that dense clusters that are diverse from labelled instances will be most informative for classification purposes. This is implemented by first calculating a *density* value per instance, defined as the sum of similarities between the instance and all other instances within a certain radius. Secondly, a *diversity* value is calculated by measuring instance distance to the nearest labelled instance of the dataset.

Variants of EGAL can be implemented by varying the balance between density and diversity measures. A density only approach selects instances from dense areas of the feature space only. Using the diversity metric in isolation will select instances that are most diverse from already labelled instances. These measures can be combined to select instances from the most dense areas of the feature space that are most diverse from already labelled instances. EGAL is arguably a computationally inexpensive method as once a similarity measure is calculated for all instances in the unlabelled pool, only the diversity calculation is required for each iteration of the algorithm. The other methods surveyed require a model to be trained at each iteration of the algorithm, which can considerably increase the time required to label instances.

3.6.2 Active Learning in the Auditory Domain

There are a number of example applications of AL in the audio domain. Mandel *et al.* [311] use AL in a popular music mood, style and artist classification task. They use MFCC features and an angle diversity selection method, which balances decision boundary proximity with coverage of the feature space, based on the findings of Chang *et al.* [312], who recommend it in an image retrieval task. Mandel *et al.* [311] find that AL proceeds quicker using smaller batch sizes (the number of instances selected for labelling at each iteration) as this gives the

algorithm more opportunities to select helpful instances. They also note that use of a small batch size can initially hurt classification performance, and postulate that beginning training using a larger number of instances may be a way to counteract this.

Gulluni *et al.* [313] utilise AL in an electro-acoustic music sound object classification task and compare three selection strategies based on the prediction confidence of an SVM model. They find that a selection strategy based on the instances the model is most uncertain in a binary classification case works best for this application, outperforming two other strategies: selecting instances the model is most confident classifying as class A, and those the model is most certain are class B. In an emotion in speech study, Zhang and Schuller [314] investigate a sparse-instance-based AL method and an uncertainty sampling technique, once more using an SVM. The sparse instance method involves randomly selecting instances that the SVM predicts are members of a sparse class in an imbalanced binary dataset. When compared to a passive selection approach, where instances are selected completely at random, they find that the sparse instance approach achieves 5% greater absolute Unweighted Accuracy (UA) and reduces the amount of data required by 64.2% versus passive selection. In this case, uncertainty sampling achieves 61.5% UA versus 65.5% from the sparse sampling, on the same number of instances. Using a balanced dataset, the authors find that an uncertainty selection approach outperforms passive selection by 1.3% absolute UA when trained using the same number of instances.

Han *et al.* [309] scrutinised both supervised and semi-supervised approaches to AL in an environmental sound classification task. They used distance from the decision boundary of an SVM to derive pseudo-probabilistic values which they used as an indicator of model confidence, scores close to the decision boundary indicating which instances the model was least confident about. In the supervised approach, instances the model was least confident about were manually labelled, and this process was repeated until there were no more unlabelled instances or the model stopped improving. In the semi-supervised approach,

an additional step was added after the least confident instances were identified. Once the newly labelled instances were added to the labelled pool, the model was re-trained and the instances about which the model was most confident were automatically labelled. Once more, this process continued until there were no more unlabelled instances or the model stopped improving. It is interesting to note that the semi-supervised approach was deemed feasible in this instance only after observing that a high proportion of the dataset was classified with high confidence, which may not be the case with every dataset. The authors find that in this case the semi-supervised approach outperformed both supervised AL and passive selection, making it possible to reduce by 52.2% the number of manual labels required to achieve the best performance.

3.6.3 Active Learning Summary

USAL is popular in AL applications both generally [303] and in audio scenarios as demonstrated above. Aggarwal *et al.* [315] note that QBC, used in sound event classification contexts [316], is similar to USAL in that the selection measure is based on the uncertainty of a model or committee of models. EER is highlighted as being computationally expensive in the foregoing, so is not investigated in this work. EGAL is a selection method that takes account of the position of all dataset instances relative to each other, which suggests it provides a contrast in method to USAL. This work examines the EGAL and USAL selection methods and compares them with passive (or random) selection to assess performance. To our knowledge, this is the first application of AL to a hierarchical audio task and the first application of the EGAL selection method in the audio domain.

3.7 Data Augmentation

Data augmentation involves applying a series of deformations to labelled training data to produce new training instances. Recent trends in ML have seen the growing popularity of deep learning methods, which tend to outperform other models when supplied with large volumes of data [19, 20, 21]. However, a general difficulty is noted in the auditory domain in terms of access to large datasets of suitably labelled data [17]. Data augmentation is one method of addressing this problem, as it provides an opportunity to scale labelled datasets to much larger sizes, making it possible to improve the accuracy of models [317]. Extensively applied in the visual domain [318], distortions such as image rotation, mirroring and scaling result in recognisable images of the original source, essentially providing a cheap source of labelled data. Equivalents of these manipulations exist for the auditory domain, with the caveat that care must be taken with their application such that the original semantic meaning of the sound so augmented is not changed in the process.

Auditory augmentations can be summarised as follows:

- **Pitch Shifting:** This augmentation either lowers or raises the pitch of the audio while retaining file duration unchanged. Negative and positive pitch shifts of 1 to 3.5 semitones were implemented by Salamon and Bello [250] where they were found to be responsible for greater improvements in accuracy than other augmentation types.

- **Time Stretch:** Manipulating the audio by altering the temporal duration (slowing it down or speeding it up), while retaining file pitch.

- **Dynamic Range Compression (DRC) :** Commonly applied in audio production workflows, DRC involves compressing the range between the loudest and quietest parts of an audio file. Many common parameterisations of this augmentation are available in audio production software.

- **Background Noise:** Entails mixing the audio with other sound, chosen to simulate different kinds of background auditory scenes. Given this artificially introduces a contextual element to stimuli it is not utilised for this research.
- **Random Cropping:** Involves randomly selecting short sections of audio content and cropping them entirely. The length of time cut can be randomly varied.
- **Gain:** Increasing or decreasing the loudness of the audio.
- **Equalisation:** Altering the relative loudness of frequency components to alter the timbre of the audio. High frequencies could be cut to make a sound seem more 'muffled', for example.
- **Band Limiting:** Somewhat similar to equalisation, band limiting implies selecting a bandwidth of frequencies and blanking them entirely.

McFee *et al.* [317] compare the performance of a CNN trained with and without data augmentations in a music classification task and find a small, but consistent improvement from the no augmentation condition to a number of pitch, time stretch and background noise augmentations. The pitch-shift augmentation improves average precision from 65.5% to 67.7%. Other augmentation schemas are less successful, but perform similarly to the pitch augmentation, there being no significant difference observed between the augmentation schemas in a Bonferroni-corrected Wilcoxon signed-rank test. All augmentation methods consistently outperform the no-augmentation condition, however. Salamon and Bello [250] investigate pitch, time stretch, DRC and background noise augmentations in an environmental sound classification task, also using a CNN. They find that their proposed model performs comparably to a clustering baseline method using a non-augmented dataset, but significantly outperforms the baseline according to a paired two-sided t-test when both are trained using augmentations. They also compare the per-class classification accuracy as a function of

each augmentation and find that the pitch augmentations generally have the greatest positive impact on classification performance.

The primary concern when selecting suitable data augmentations is to ensure the effect on semantic meaning is minimised. This is subject to debate in some cases more than others, though it could be considered that any alteration to an audio stimulus has the potential to alter semantic content. It is therefore advisable to minimise the augmentations applied and to be selective about which are used. Increasing volume level (gain) is likely to affect the hierarchical perception of a sound, based on the literature reviewed in Chapter 2. Equalisation and band limiting could be considered a similar case given they involve altering the volume of a selection of frequencies, though selective, light usage may be suitable. Addition of background noise can be considered unsuitable as it introduces a hierarchical element in an instance where care has already been taken to minimise such instances. DRC also involves a manipulation of perceived volume levels, but may be permissible if applied appropriately, and has been used in this manner in similar studies [250]. Random cropping and time stretch involves manipulating the temporal length of the stimulus, and may not be applicable for applications where file length could be considered a variable when labelling stimuli semantically. Additionally, random cropping in this instance could potentially crop important parts of an audio file which dictate its hierarchical position. Finally, pitch shifting could be considered less intrusive to semantic meaning than many of the options highlighted hitherto. Furthermore, it has been successfully applied in a similar prediction task [250] and for this reason it has been employed in this research in addition to DRC compression applied restrictively.

In theory, the application of data augmentation techniques to the auditory domain is logical, given its success in other domains. The area is ripe for investigation via perceptual testing in an audio context, as the effect on semantic meaning of different manipulations would be of interest from the point of view of confining future use to those manipulations

judged to have no effect and also defining the degree to which manipulations can be applied before semantic meaning is judged affected. At the time of writing, the authors are unaware of any such study.

3.8 Conclusion

ML analysis is a complex process requiring numerous decisions governed by specific domain details, available data and prediction task. Frequently an iterative approach is required to deepen analysis using learnings from simpler initial approaches to inform the building of more complex models. There is an extensive extant literature on the application of ML analysis to many audio problems using a range of algorithms, feature extraction and selection methods. The diversity of applications and approaches in the literature suggests that successful modelling can be arrived at via numerous methods, and an experimental approach is required in this regard. In other words, there is indeed no *free lunch* [191].

In Chapter 2 we have outlined AH from a perceptual point of view and as regards how it pertains to modern media consumption paradigms. This has inspired the desire to predict AH as perceived by consumers of such media, which motivates the need to establish a baseline dataset of sounds tagged with appropriate labels. This in turn has motivated the use of supervised ML, using these labels, to predict AH. Sections 3.2 and 3.3 have in turn highlighted the requirement for large volumes of data for supervised ML tasks and together with the summary of available audio datasets in Section 2.4 this mandates that considerable work is required to compile a dataset suitable for analysis in this respect. To that end AL approaches used to minimise the manual effort required to label large datasets have been introduced in Section 3.6 and data augmentation techniques relevant to the auditory domain have been outlined in Section 3.7.

A series of supervised ML algorithms have been surveyed in Section 3.5 with particular attention to their application in audio prediction tasks. Recent work on the popular ML

competition platform, Kaggle [319] and on audio specific ML challenges such as the DCASE challenges [15] is notable for the success of ensemble solutions, where the predictions of several algorithms are combined to make a more robust prediction. This is an interesting development, but not one that suggests other ML methods should be abandoned entirely. It should be noted that the importance of interpretability is a topic of hot debate in the wider ML community [320] with some citing the continued preference in commercial contexts to favour a more interpretable model over a more accurate model. Prominent practitioners, such as Rahimi and Recht [321], have recently called for a more systematic analysis of the DL modelling process in order to greater understand its workings.

It is logical in this context to proceed with an ML investigation of AH using algorithms which can successfully be applied to small datasets in order to establish the feasibility of predicting AH. In doing so, the value of implementing a large logistical and manual labelling task can be evaluated before significant effort is expended doing so.

The material covered in this chapter addresses OBJ 2 by providing an overview of ML and a description of how a supervised ML model that predicts AH can be built. To recap, this objective was formulated as follows:

OBJ 2: Informed by perceptual audio research, to propose a machine learning approach for the task of predicting AH.

This in turn has inspired the formulation of another RQ to address the requirement for a model to predict AH as follows:

RQ 3: Is it possible to accurately predict AH using supervised ML methods?

To answer this question, Chapter 4 will outline a semantic labelling exercise, referred to as Experiment 1. An initial ML investigation of the findings, designated Experiment 2, is also described in this chapter. Subsequent chapters will focus on further assessment of algorithms and feature representations applied to AH. Methods of building larger datasets

(Chapter 5) are then also investigated and further explored using ML and DL algorithms (Chapter 6).

Chapter 4

Perceiving and Predicting Auditory Hierarchy

4.1 Introduction

This chapter examines the perception of AH for isolated sounds using ML techniques. This examination comprises two parts. The first of these, Experiment 1, is an exploratory perceptual experiment where subjects rank sounds on a BG — N — FG scale to establish the nature of audio object hierarchy as it pertains to stimuli analogous to broadcast media content. The second part of this analysis, referred to henceforth as Experiment 2, applies ML to the perceptual labels gathered in Experiment 1 to evaluate ML performance on an audio hierarchy problem.

4.2 Perception of Auditory Hierarchy in Isolated Sounds

With a move towards object-based sound delivery in visual streaming scenarios, a deeper understanding of how auditory objects are parsed and hierarchically categorised will be useful in the development of strategies for sound file delivery. To that end, Experiment 1 explores inherent inter-object hierarchies of importance in the context of a BG/FG evaluation task.

4.2.1 Methodology

In Chapter 2 this work outlined theory behind the existence of a hierarchy of importance between sounds isolated from an auditory scene. This section will describe the methodology of an experiment investigating this question. Figure 4.1 presents an overview of the methodology for the experiment. The dataset used and online environment where the experiment was conducted will be described. Participant recruitment and profile will also be discussed.

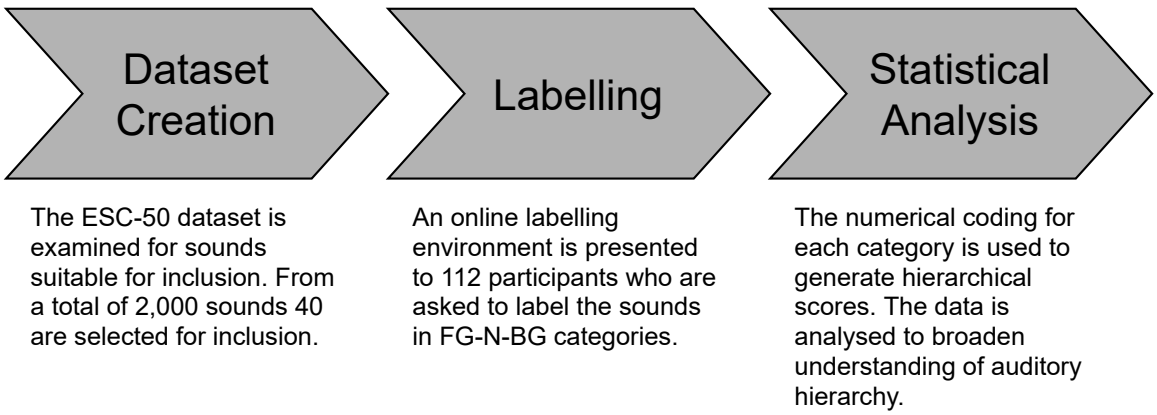

Fig. 4.1 Methodology overview for Experiment 1.

Dataset

It was decided to use stimuli analogous to visual streaming content as this is the envisaged end-use of object-based audio in media consumption scenarios. The stimuli for this initial

experiment were sourced from the ESC-50 [101] sound set. This dataset has been compiled for use in computational audio scene analysis contexts for training and testing automatic classification of sounds. A total of 40 sounds were deemed suitable for inclusion in Experiment 1. A list of these sounds is presented in Table 4.1. Dataset recordings are of approximately 5 seconds duration and are organised into 5 broad classes:

- Animals
- Natural soundscapes and water sounds
- Human, non-speech sounds
- Interior/domestic sounds
- Exterior/urban sounds

These classes are further subdivided into 10 sub-classes consisting of 40 sounds per sub-class, resulting in a dataset of 2,000 sounds in total. Sounds from every class were auditioned, and each test sound was selected with care so that each instance was that of an isolated sound, minimising the possibility of perception of a mini sound 'scene' due to the existence of other sounds at lower levels in the same file. This process resulted in some sub-classes not being represented as no individual recording was deemed suitably isolated, and others being rejected for reasons of similarity. For example, the sub-class 'Mouse click' was deemed to have a similar modality to 'Keyboard tapping' and thus one was excluded. It should be noted that a varying degree of scale is perceptible from some sounds, though this variance was minimised by auditioning multiple instances from each class and selecting sounds which were deemed acceptable. While the difficulty in removing context using any methodology is acknowledged, this approach was adopted as a practical solution to provide scope for future investigation.

Test Environment

The experiment was deployed in an online environment as it has been found ([149] and [150]) that there is minimal difference between laboratory and online experiments for comparable tests and the potential response rate for an online experiment is far greater than that of one confined to a laboratory.

Disseminating the experiment in this way is feasible due to the ease of distributing a website link to the experiment environment. There are inherent challenges because of the wide array of devices and programs in usage for browsing the web, however. Design of the environment must account for different browser versions, operating systems and devices, for smartphones, tablets, desktop computers and so on. What may work for one participant may not work for the majority of configurations. Consideration should be given therefore to options that will work for the majority of participants. Sufficient time must be allowed in advance of the experiment going live, not alone for the basic design and coding of the experiment, but also to allow for extensive testing of functionality in multiple different browsers and devices. Once this process is completed, the advantage is that the same design can be reused multiple times, thus repaying the initial resource investment.

The experiment asks subjects to rate sounds on a BG — N — FG scale. For the purposes of this study, FG and BG were defined as follows:

A Foreground sound: One you are likely to think prominent and give greater attention.
A Background sound: One you are likely to think less important and give less attention.

If unsure whether a stimulus was BG or FG, subjects were advised to mark the sound as neutral. Informed consent was obtained for all participants following guidelines approved by the Technological University Dublin Research Ethics Committee. Figure 4.2 shows the stimulus presentation and scale rating portion of the test environment. It should be noted that any definition of FG and BG sounds is problematic, given these categories are inextricably

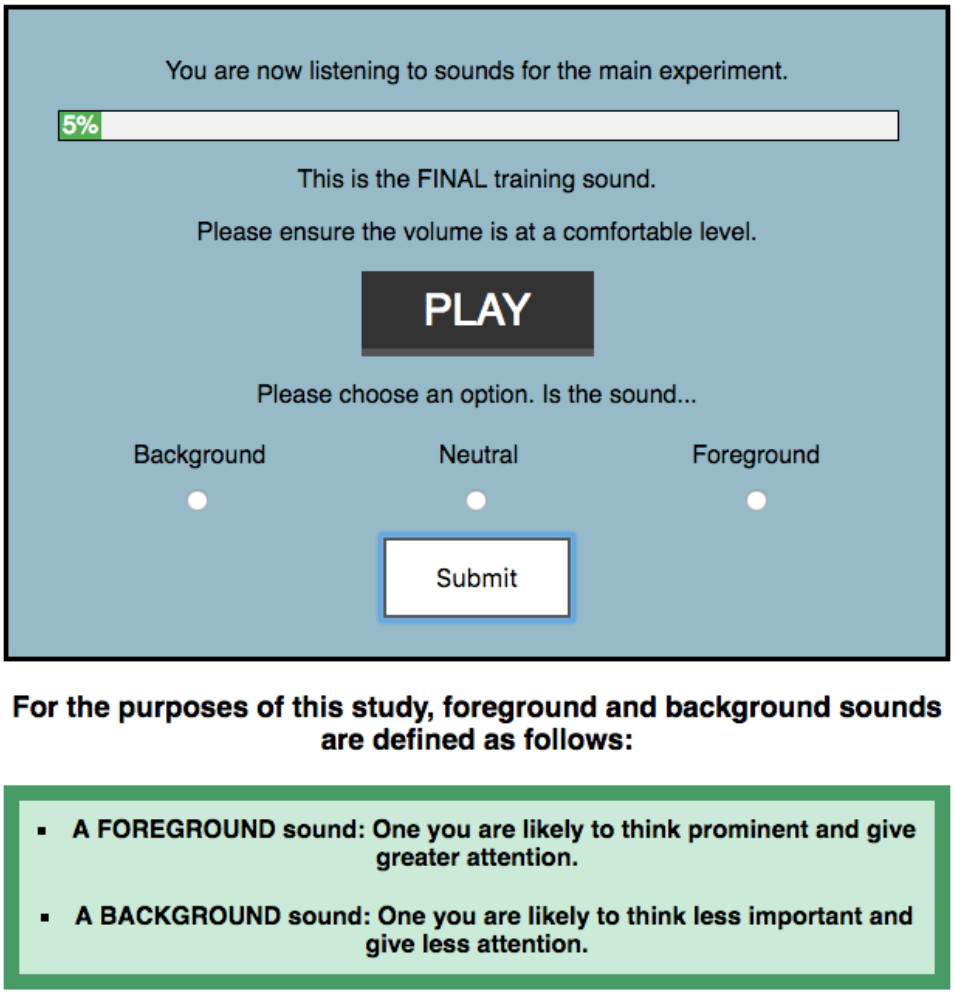

Fig. 4.2 The test environment.

linked to context. The screening process used to select stimuli which evinced only a single audio object addresses this concern.

The test website was designed using pre-formulated stimulus presentation orders which were seeded using randomised output as it was desirous to control directly for presentation order occurrences. Random orders were sourced from *www.random.org*, a source for true random sequences cited in a number of peer-reviewed publications [322]. These series were then analysed for repeated occurrences of order, and controlled so that the mean occurrence of a particular order was 5, with all combinations occurring at least once. This process was repeated so that 200 unique presentation orders were compiled for the experiment.

In the experiment environment, detailed instructions and a training phase were implemented, during which participants were asked to set the volume at a comfortable level and not to adjust it. Participants were asked to complete the test using headphones in a quiet environment, were required to submit basic demographic information and then rate 40 sounds.

Participants

The desire to maximise the number of participants was one of the considerations behind using an online environment for the experiment. This was judged an acceptable compromise as the study focus was not on BAQ differences between stimuli but rather on participant subjective judgement of the hierarchical placement of isolated sounds, which relaxes the necessity for laboratory listening conditions.

Consideration was given to the manner in which participants were recruited. Once the experiment environment was completed, the primary challenge to completion was engaging with a sufficient number of respondents. While social media can be an effective way of reaching a large pool of participants, this requires a broad reach from the accounts used to disseminate the experiment request. As such an account was not available, this challenge was addressed by utilising the following resources. Firstly, a request for participation was placed on the Auditory list [153], an online mailing list set up by Albert Bregman where listening tests of various forms are routinely circulated. Secondly, participation was requested of the staff and students of the TU Dublin School of Media, where participation requests for various studies are a common event. Thirdly, a curated list of the authors' contacts in the research and creative arts domains were asked to participate. Additionally, not being constrained by a hard completion deadline was useful, as it meant that the labelling exercise could be extended until sufficient tests were completed. In this way, a balance could be struck between moving to the analysis phase in a timely manner and securing enough participants for the

experiment. This process resulted in 112 complete tests collected from 36 women and 76 men. The majority (65%) of respondents were 25 — 44 years of age.

4.2.2 Results

Subject responses were collated in a tabular format in Microsoft Excel and a frequency table (summarised in Table 4.1) was compiled showing counts of BG, N and FG selections for each sound. The R statistical environment was used to generate additional summary statistics and plots of the results.

It was decided to use the median as the centre measure of this data, as it is generally accepted as the appropriate measure of centre for ordinal data. The median is the centre value in a series that is arranged sequentially. Ordinal data is categorical and has an order, though the distance between different levels on the scale may not be equal. The numerical coding used for sound categories was as follows: BG — 1, N — 2, FG — 3. A median value of 1 means that at least 50% of subjects categorised the sound as BG, while a median value of 3 signifies that at least 50% categorised the sound as FG. Scores for each sound were arranged in a series and the median value for each sound isolated and used as a basic categorisation rule for each sound as outlined in Table 4.1. It should be pointed out that in marginal cases this would mean, in the case of BG sounds for example, that nearly as many subjects rated the sound as either N/FG as rated it BG, thus weakening the strength of any such category membership.

The frequency counts were analysed using a scatter plot matrix to visualise the correlations between subject categorisations and see if autonomous clusters were apparent from which robust BG — N — FG categorisations could be made. This plot is reproduced in Figure 4.3, which colour codes results based on the median categorisation rule previously mentioned. Unsurprisingly, a strong linear correlation is observed between FG and BG scores

Table 4.1 Summary results ordered by mean sound rating from top to bottom. Sounds ranked *More Background* are towards the top, while those *More Foreground* are towards the bottom.

Sound	BG	N	FG	Category
Birds	95	12	5	BG
Keyboard_Tapping	81	25	6	BG
Clock_Tick	79	25	8	BG
Fire	76	31	5	BG
Crickets	81	16	15	BG
Water_Drops	73	23	16	BG
Wind	69	28	15	BG
Engine	69	23	20	BG
Helicopter	68	22	22	BG
Train	62	19	31	BG
Washing_Machine	61	20	31	BG
Rain	55	28	29	N
Drink_Sipping	51	31	30	N
Hen	50	32	30	N
Can_Open	53	25	34	N
Pouring_Water	50	26	36	N
Coughing	43	38	31	N
Snoring	46	28	38	N
Crow	45	27	40	N
Brushing_Teeth	42	33	37	N
Handsaw	36	40	36	N
Fireworks	37	30	45	N
Clapping	35	31	46	N
Pig	31	35	46	N
Church_Bells	34	26	52	N
Dog	28	36	48	N
Cow	35	20	57	FG
Door_Wood_Creak	31	25	56	FG
Insects	28	27	57	FG
Thunderstorm	30	22	60	FG
Rooster	24	25	63	FG
Cat	24	18	70	FG
Laughing	17	30	65	FG
Breathing	19	22	71	FG
Chainsaw	12	16	84	FG
Siren	11	12	89	FG
Baby_Crying	6	7	99	FG
Door_Knock	3	10	99	FG
Glass Breaking	2	11	99	FG
Clock_Alarm	1	7	104	FG

(as one increases, the other decreases, and vice versa). However, the plots do not suggest obvious autonomous clusters for groupings of BG, N and FG categories.

The numerical coding used for each category was used to calculate a mean of the rating values for each sound. This was used to draw an indicative spectrum to rank sounds from 'More Background' to 'More Foreground' in order to gain insight as to how sounds relate to each other on this spectrum. Similarly, the standard deviation was calculated to investigate the level of consensus between subjects for each categorisation. These values are compared in Figure 4.4. Sounds ranked as more BG are to the left and those more FG are to the right. Sounds with a smaller standard deviation are plotted towards the bottom of the chart, while those with a larger value are at the top. This plot demonstrates that there are relatively few sounds which most subjects agree are either BG ('Birds', 'Keyboard Tapping', 'Fire' and 'Clock Tick') or FG ('Baby Crying', 'Door Knock', 'Glass Breaking' and 'Clock Alarm'). There is greater disagreement between subjects regarding the category of the remaining sounds. Conversely, there is greater consensus regarding strongly FG or BG sounds at either end of the spectrum, with slightly more agreement regarding which sounds are FG than BG.

The exact point at which a sound can be said to have definitively changed from being BG to N, or N to FG is arbitrary and the efficacy of applying such a margin remains to be seen in further research and in real-world applications. The median rating value would appear to be insufficient, at least in the context of categorising all sounds with high inter-subject agreement, as the decision boundary encompasses sounds which evince considerable disagreement between subjects as to appropriate category based on the plot offered in Figure 4.4. Formulating a decision boundary threshold based on a function of overall rating and sample consensus would allow for more sophistication in the model, but is still subject to an arbitrary decision on where this boundary would best lie. For example, the following equations use rating and standard deviation (σ) values to isolate 'Birds', 'Keyboard Tapping', 'Fire' and 'Clock Tick' as BG sounds (Equation 4.1) and 'Baby Crying', 'Door Knock',

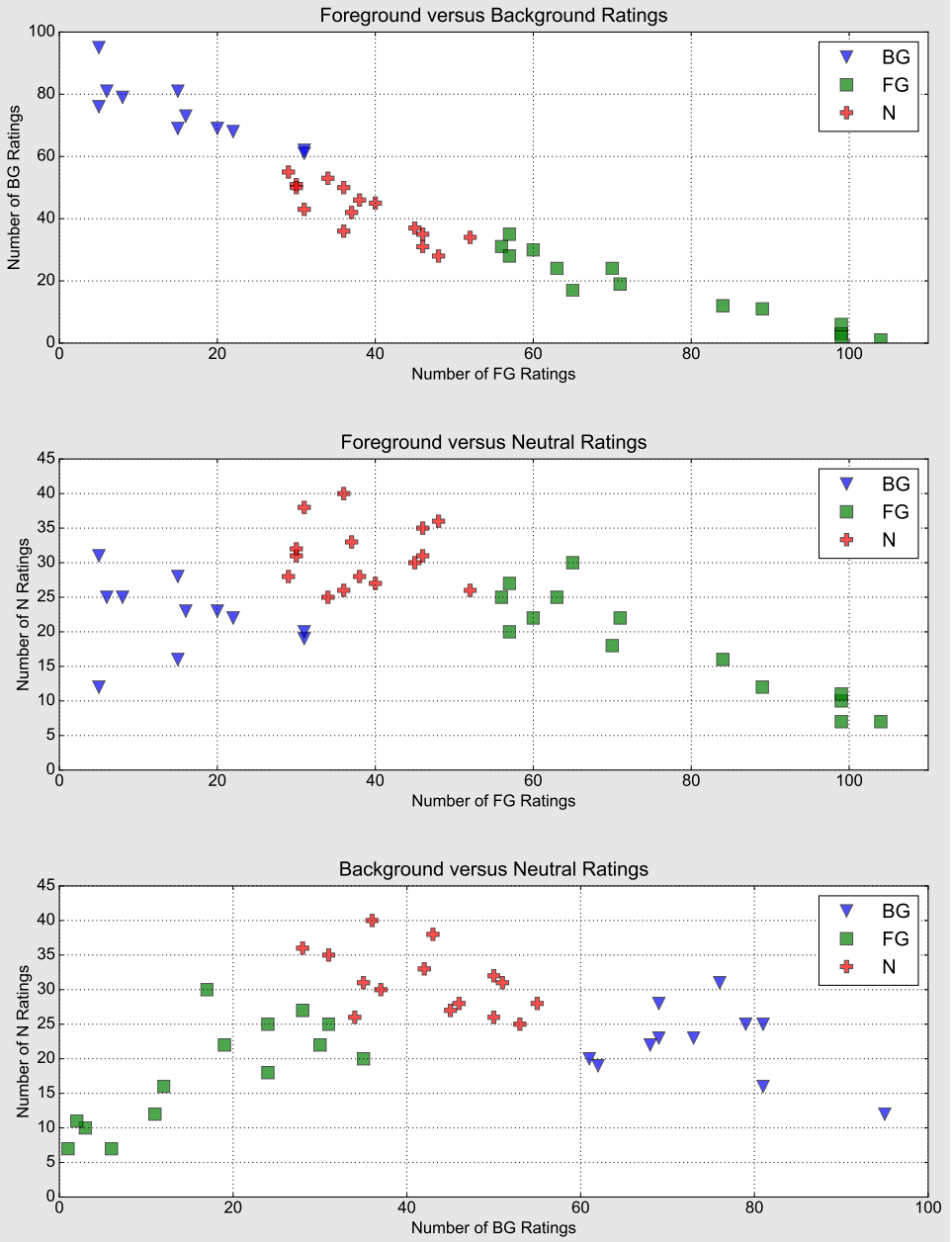

Fig. 4.3 Scatterplots of BG, N and FG counts. The categories in this plot are derived from the median ratings noted in Experiment 1. A strong linear relationship is noted between BG and FG ratings.

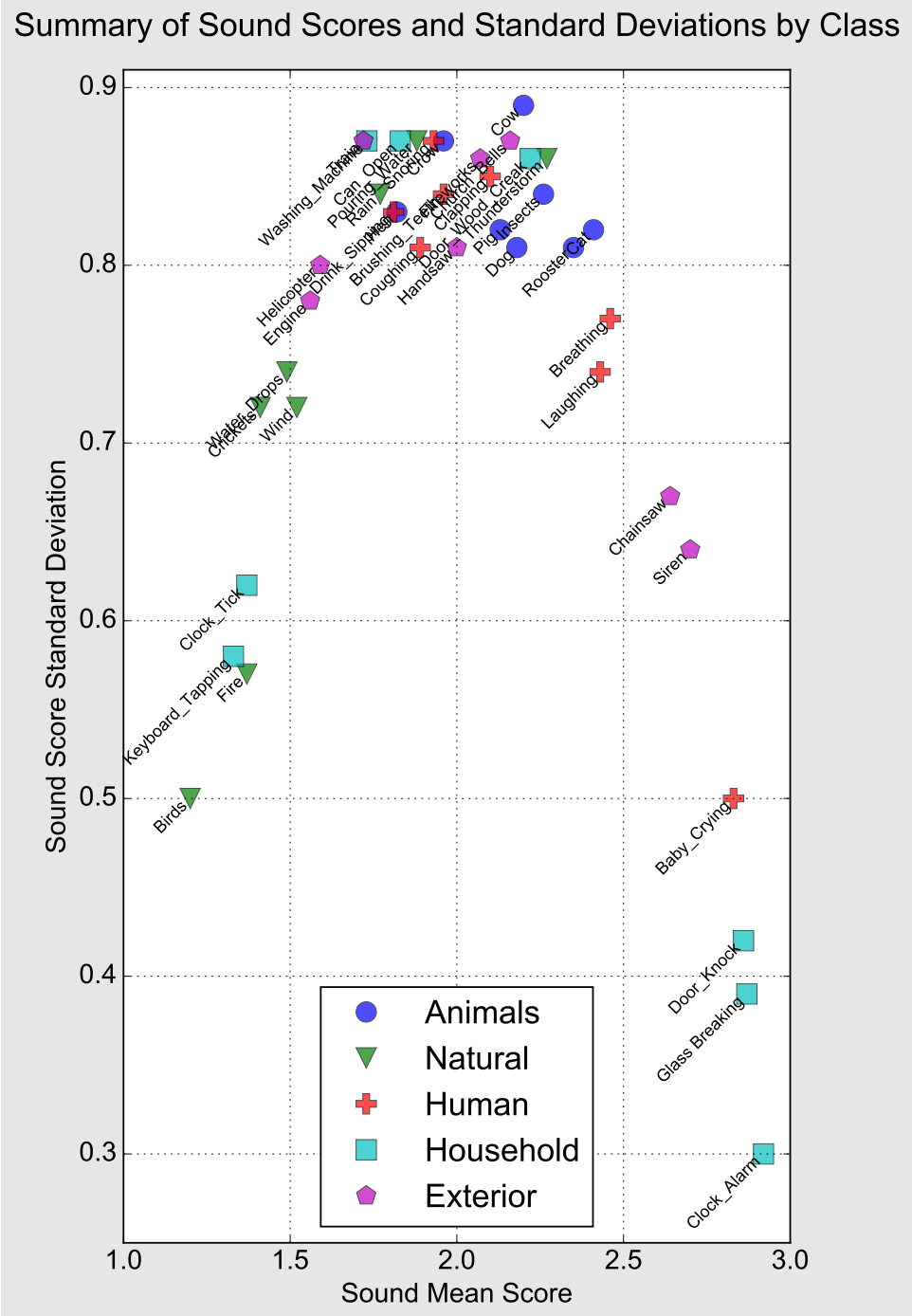

Fig. 4.4 Relationship between mean sound score and standard deviation separated by class. Sounds ranked more FG are to the right. Those considered more BG are to the left. Sounds with a smaller standard deviation (closer to the bottom of the plot) indicate that there was more consensus between subjects as to category in these instances. There is no clear categorisation pattern by sound class.

'Glass Breaking' and 'Clock Alarm' as FG sounds (Equation 4.2), but could be altered to include or exclude other sounds. These are but two possibilities suggested by the plot.

$$BG_{\text{RATING}} \geqslant 76 \wedge \sigma \leqslant 0.62 \tag{4.1}$$

$$FG_{\text{RATING}} \geqslant 99 \wedge \sigma \leqslant 0.5 \tag{4.2}$$

The data was also analysed for differences in sound ratings between genders. Figure 4.5 plots mean sound scores for the whole sample and both female and male subsets. Figure 4.6 is a similar plot showing the variance in standard deviation. While there are some variations in the rankings of sounds, no general trend emerges along gender lines. For example, female mean rating for 'Dog' (2.5) and 'Insects' (2.44) sounds are more FG than male 'Dog' (2.03) and 'Insects' (2.17) ratings, remembering that BG = 1, N = 2 and FG = 3. However, this trend does not extend to other animal noises, with female mean ratings for 'Hen' (1.61) and 'Pig' (1.92) being more BG than equivalent male mean ratings (1.92 and 2.24 respectively). Female data appears more spread out than the male equivalents, though this could easily be explained by the disparity in sample sizes (68% male). Given a larger female sample size, it could reasonably be expected that the scores would regress towards the mean.

Finally, the data were examined for any evidence of correlation between sound class and subjective categorisation. Figure 4.4 presents the sounds colour-coded by class. While there are some weak trends visible, there is no clear categorisation trend by sound class. 'Natural' sounds tend more to BG and N than other classes. 'Animal' sounds caused significant disagreement among respondents compared to other classes, having higher standard deviation values and no representatives from this class being considered highly BG or FG. 'Household' sounds contained a considerable spread across the BG—N—FG spectrum, with many of the most BG and FG sounds coming from this class.

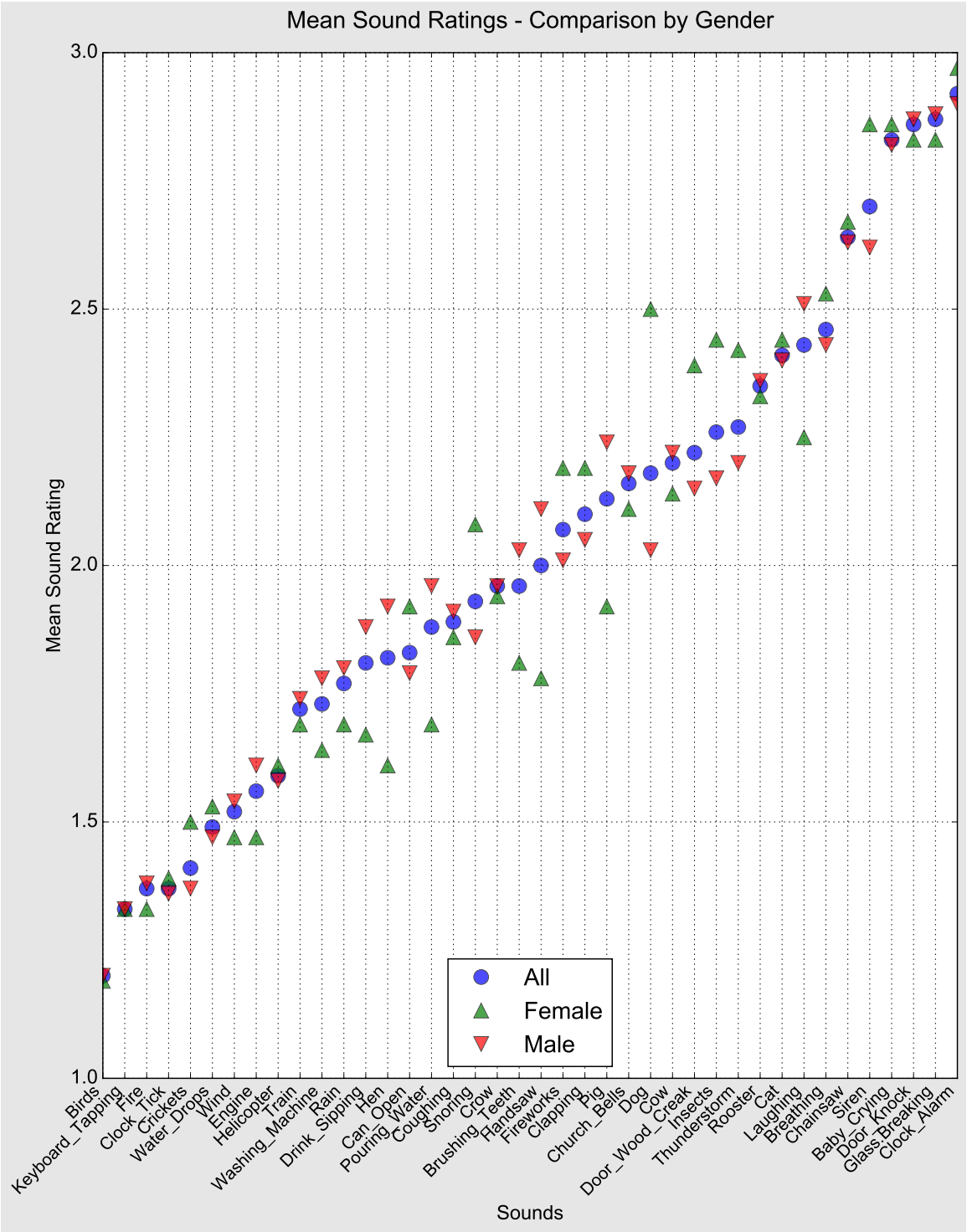

Fig. 4.5 The relationship between mean sound rating and gender. Once again, sounds considered FG are towards the right of the plot, BG sounds are to the left.

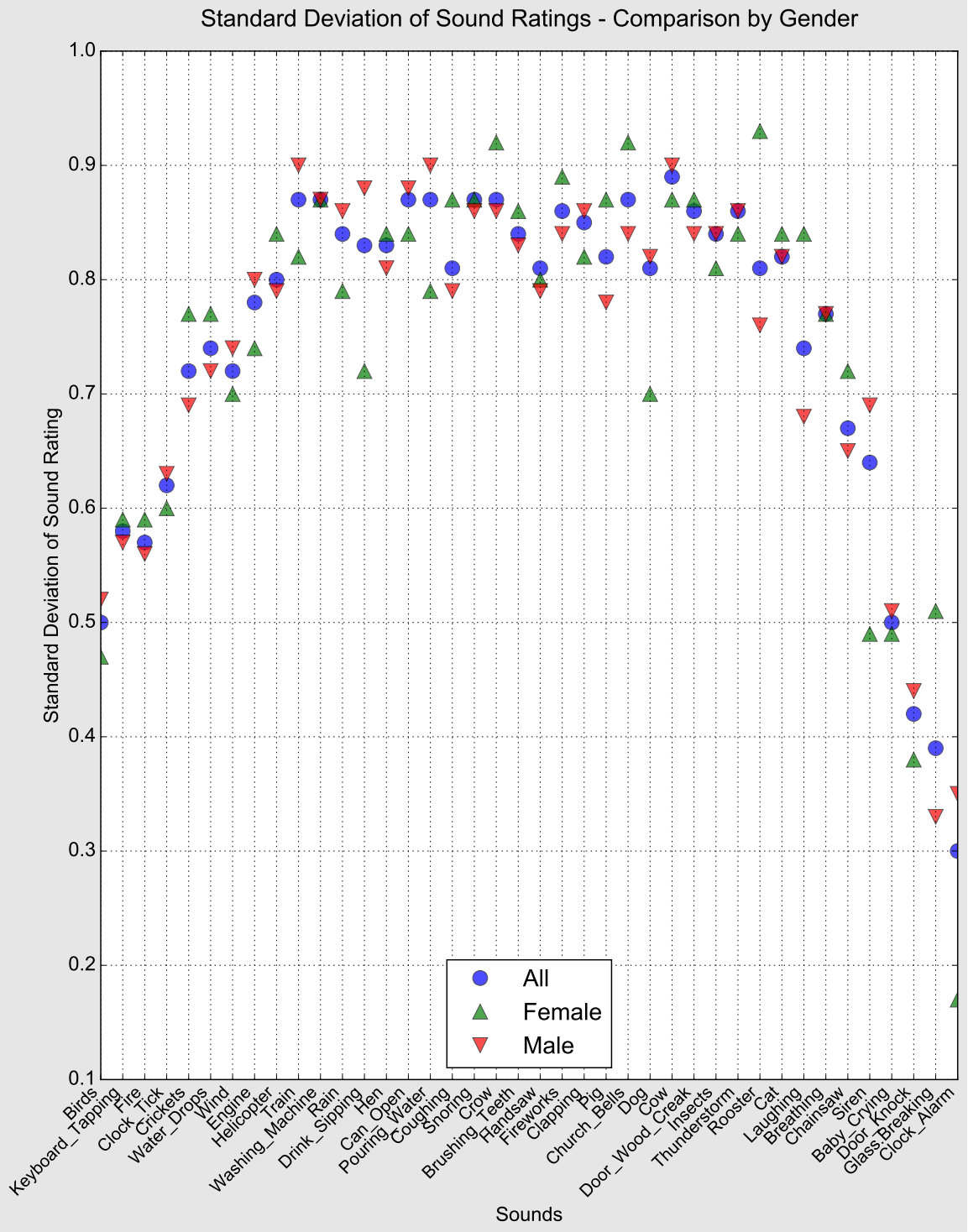

Fig. 4.6 A comparison of sound ranking standard deviation by gender. Sounds considered FG are towards the right of the plot, BG sounds are to the left.

4.2.3 Discussion

The results outlined suggest that a spectrum of sound hierarchy exists for isolated sounds and that this can potentially be predicted provided suitable objective measurements that correlate with subjective evaluation of sounds can be isolated. Subject ranking of sounds, as outlined in the scatter plot in Figure 4.5, suggests that BG — N — FG is a continuum. Chapter 2 has offered an overview of perceptual theory and a schema for AH hypothesised as being affected by factors outside the scope of this experiment such as, but not confined to:

- Sound context
- Prior experience and training of subject
- Attention/listening mode
- Sound loudness
- Physical characteristics of the sound
- Spatial location of the sound etc.

It is important to note that testing using isolated sounds is somewhat artificial as they are seldom, if ever, experienced entirely in isolation – a point made by some subjects in correspondence who reinforced the importance of context and sound meaning in making a decision in the categorisation task. Thus, further investigation of the effect of such factors is necessary in order to derive a useful categorisation schema for real-world implementation.

While a clear consensus among subjects was observed for certain stimuli, there were no unanimous decisions, an indication of the subjective nature of the experiment task. Sounds such as 'Clock Alarm', which received 104 selections as FG (104 — 92.86%), the most emphatic FG score, still received selections for either N (7 — 6.25%) or BG (1 — 0.89%). The converse holds for sounds considered overwhelmingly BG, such as 'Birds', which

received 95 selections as BG (95 — 84.82%), 12 for N (12 — 10.71%) and 5 for FG (5 — 4.46%).

4.2.4 Conclusions

Section 4.2.3 has noted the wide range in rating scores and the lack of unanimous categorisations. For the purposes of this discussion, sounds with an average rating below 1.5 will be referred to as being definitively BG (6 sounds: 'Birds', 'Keyboard Tapping', 'Clock Tick', 'Fire', 'Crickets' and 'Water Drop'). Conversely, those with an average rating over 2.5 are designated definitively FG (also 6 sounds: 'Chainsaw', 'Siren', 'Baby Crying', 'Door Knock', 'Glass Breaking', 'Clock Alarm'). If all other sounds are considered a Neutral rating, then this category dominates the dataset in terms of size: 28 sounds are rated between these two values. This indicates the level of disagreement between participants as to the appropriate hierarchical category in many cases, which effectively results in a noisy dataset for ML purposes.

It should be noted that it may not be possible to obtain a similar set of results from a different cohort of participants. While many elements would remain the same, such as the test environment and sounds, other factors are potentially beyond the control of the experimenter when using an online test. This experiment demonstrates inter-rater reliability, and it is argued that this is useful for the development, training and testing of an objective auditory hierarchy classification model. However, the experiment cannot fully validate a conceptual model of auditory hierarchy because it does not establish intra-rater reliability (participants only rate each sound once). For this reason, though the experiment provides some evidence of a hierarchy of importance between sounds isolated from context, it is not considered to be a conclusive demonstration of audio hierarchy in non-contextual sounds.

As noted, the decision boundary between what constitutes a BG, N or FG sound is open to debate. Indeed, the location of such boundary lines may form part of any final solution

in this regard, becoming a parameter used to tune a model for specific applications. In this context, compare the differences in categorisations suggested by median sound rating score and any variant on this, such as those outlined in Equations 4.1 and 4.2. What may prove more illuminating in this regard is a testing of the subject nominated categorisation schema for different applications. An object audio codec which encodes BG assets at lower bit rates than FG may prove to be perceptually transparent at specific thresholds yet to be determined, for example.

Equally, subjective categorisation of audio objects may not be a comprehensive indicator of asset importance with regard to a perceptual coding application. For instance, certain sounds may not necessarily rate as FG, but are known to be more challenging for compression codecs to deal with transparently. Applause is an example in this regard [111]. Simply put, what subjects perceive as a relatively unimportant sound in isolation may have a disproportionate effect on the perception of a sound scene if that element is delivered at an inappropriately low bit rate.

These points noted, the following section describes Experiment 2, an investigation of ML algorithms to the perceptual data collected in Experiment 1 to assess whether ML can be used to accurately predict AH.

4.3 Predicting Auditory Hierarchy

Section 4.2 summarises research which suggests the existence of a hierarchy of importance between isolated auditory objects by quantifying human subjective hierarchical ratings of sounds. The next step is to derive labels from these data for use in an ML classification exercise, which establishes the feasibility of predicting the hierarchy of a sound set using purely objective measurements. The arbitrary nature of deciding on a classification boundary location in advance of perceptual testing for the implications of such a decision on the end use case has been noted in Section 4.2. To facilitate investigation of ML methods applied to

AH, it was decided to use the median rating score from Experiment 1 as the categorisation schema used in Experiment 2.

4.3.1 Methodology

Section 4.2.2 has presented evidence of the perception of hierarchy as a continuum and noted that the positioning of borders between hierarchical categories is somewhat arbitrary. Respondents were asked to categorise sounds rather than score them on a continuous scale and Section 2.6 has stated the intention of providing a roadmap for how intelligent content optimisation systems can be built via a simplified study of AH. With these thoughts in mind it was decided to approach the modelling task as a categorical rather than a regression task using the median score as the category boundary. It was also decided to prioritise identifying FG sounds. Any real-world implementation of a variable compression codec, for example, would in theory require as many important sounds be correctly identified as possible and be forgiving of having some misclassified non-important objects. For this reason, it was decided to frame the task as a binary classification problem using the target labels of 'FG' and 'nonFG', the latter of which is simply the set of all sounds identified as N and BG according to the median rating derived in Section 4.2.

The following sections describe the process of building the dataset for this experiment by extracting audio features from the 40 sounds used in Experiment 1. Feature selection methods are described and a rationale for algorithm selection is offered. Figure 4.7 offers an overview of the methodology used for Experiment 2.

Dataset Creation

As the purpose of the experiment was an exploratory assessment of ML applied to AH it was decided to extract a series of LLD features which are summarised in Table 4.2. Subsequent experiments will contain an investigation of feature representations. Feature extraction was

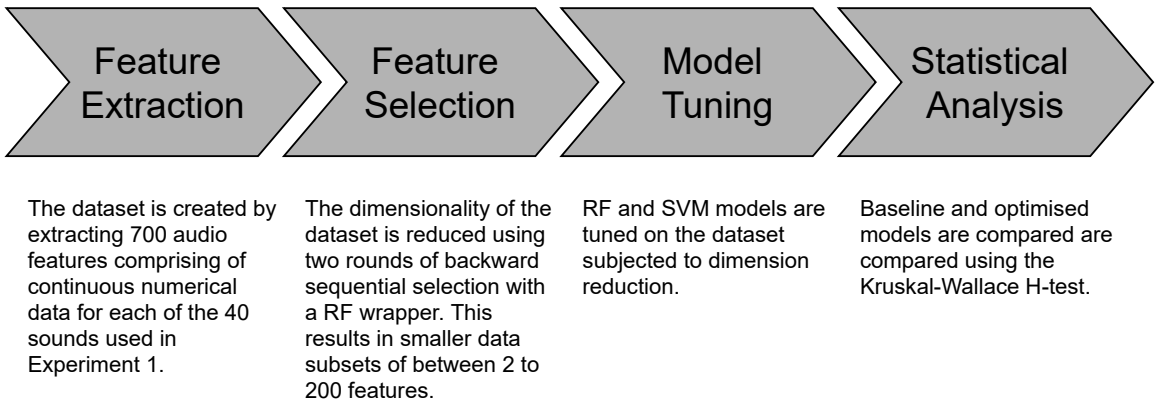

Fig. 4.7 Methodology overview for Experiment 2.

completed using Matlab [198] via the 'Matlab Audio Analysis Library' [323] as detailed in [200]. A Hamming window of the form outlined in Equation 4.3 (where n = sample number, N = the number of samples in the window, window length $L = N + 1$) was implemented with a size and step length of 0.05 and 0.025 secs (50% overlap) respectively. This resulted in an initial dataset of 35 features.

$$w(n) = 0.54 - 0.46\, cos(2\pi \frac{n}{N}),\ 0 \leq n \leq N \tag{4.3}$$

Standard statistical summaries (mean, median, standard deviation, standard deviation by mean ratio, maximum, minimum, mean non-zero, and median non-zero) were applied to each feature resulting in an initial vector of 280 features per sound. In addition to these global summaries, delta and double delta measures for the original 35 features were calculated to capture detail of local variation in the stimuli. These were derived from the frame level data and summarised using mean, median, standard deviation, standard deviation by mean ratio, maximum and minimum values resulting in another 420 features. This resulted in a final dataset of dimensions 40 sounds detailed by 700 features comprised of continuous numerical data.

Table 4.2 A description of features extracted as objective measures of the sound stimuli.

Feature	Description
Zero Crossing Rate	The number of times the signal changes value, negative to positive and vice versa, divided by frame length.
Energy	Sometimes referred to as the *power* of a signal, calculated as the sum of the squares of signal values normalised by the respective frame length.
Entropy of Energy	A measure of the abrupt changes in the energy of an audio signal, which can be thought of as an indication of signal predictability.
Spectral Centroid	An indicator of timbre. Higher values equate to brighter sounds.
Spectral Spread	A measure of how the sound spectrum is distributed about the spectral centroid. Higher values result from spectra not tightly grouped about the centroid, exhibiting more variety.
Spectral Entropy	Similar to energy entropy, but in the frequency domain. A measure of abrupt changes.
Spectral Flux	The degree of change in the frequency domain between two analysis frames.
Spectral Rolloff	Generally used to indicate the frequency below which 90% of the magnitude distribution of the spectrum is focussed.
MFCCs	Mel Frequency Cepstral Coefficients capture timbre detail of a signal efficiently. The frequency bands used to split the signal are not linear but distributed according to the mel-scale which is modelled on the human auditory system. In this instance, 13 bands are extracted.
Harmonic Ratio	The maximum value of the normalised autocorrelation function (the correlation of an analysis frame with itself at a defined time lag, in this instance, one analysis frame).
Fundamental Frequency	An estimate of the frequency equivalent of the length of the fundamental period of the signal.
Chroma Vector	A 12-element representation of spectral energy, where the bins are organised as per the 12 equal-tempered pitch classes of western music (semitone spacing).

Algorithm Choice

Numerous ML algorithms have been utilised in audio research as outlined in Section 3.5. The relatively small size of the available dataset was a factor in algorithm choice, as there are noted strengths and weaknesses for the different ML methods. As pointed out by Krstulovic [302], SVMs tend to outperform other algorithms on small datasets. Also, Deep Neural Networks require large amounts of data to outperform SVMs, which are noted to perform well using up to 10,000 instances, but deteriorate in performance thereafter [324]. This suggests that better results will be achieved with the current dataset using algorithms known to perform well with relatively small datasets, such as SVMs, which find the optimal hyperplane which separates instances by maximising the margin of distance from hyperplane to data point [325]. Data was normalised before input to SVM models as required [324].

It was decided to compare the performance of SVMs with RF models for this experiment. RFs are an ensemble of decision trees used extensively in ML classification problems [324]. Where a single decision tree can overfit the training data, an ensemble of trees is less prone to this problem, as the tendency to overfit in single trees can be averaged out throughout the ensemble. RF are often used to provide insight into relative feature importance to assist in the process of dimension reduction. They have been introduced in Section 3.5.1 where they are noted to be slightly outperformed by SVMs on some audio classification problems. Their selection for this task is motivated by their interpretability and the insight they may give as to feature importance. In this sense they add balance to the use of SVMs which are not easily interpreted. As RFs are known to perform poorly in situations where few instances are represented with many features [180] they will also be used in this instance to reduce the dimensionality of the dataset in order to improve performance.

Table 4.3 The parameter grid used to find optimal hyperparameters for baseline models for the RF algorithm.

Parameter	Values
No. of Estimators	50, 200, 500
Maximum Features	2, 5, 10, 20, 50
Maximum Depth	2, 3, 5, None
Minimum Samples per Split	2, 3, 5
Minimum Samples per Leaf	1, 2
Bootstrap	True, False

Table 4.4 The parameter grid used to find optimal hyperparameters for baseline models for the SVM algorithm.

Parameter	Values
kernel	radial basis function, polynomial, linear, sigmoid
C	0.001, 0.10, 0.1, 1, 10, 25, 50, 100, 1000, 10000
gamma	10, 1, 1e-1, 1e-2, 1e-3, 1e-4, 1e-5

Model Training and Validation

5-fold random, stratified, cross-validation was implemented to split the dataset into train and test sets and a further 4-fold cross-validation was used on the training sets to select features and to fix parameters. Before the dimension reduction process described in Section 4.3.1 was applied a parameter grid search was run to identify optimal hyperparameters using all features. The parameter grids for this search are reproduced in Tables 4.3 and 4.4. These parameters were used to train baseline models for comparison with optimised models trained after the dimension reduction was complete. Once the optimal feature set was identified, another grid search was conducted, as it was found in experimentation that the initial hyperparameters were not necessarily optimal on the reduced feature set. Once hyperparameters were finalised, models were trained on the training set and evaluated on the test set for comparison with baseline models.

Dimension Reduction

There are a number of feature selection procedures for ML features which include filter-based, wrapper and PCA approaches as introduced in Section 3.3.5. In the following, dimension reduction is applied using the training portion of the dataset only, as to apply it across the whole dataset in advance of any dimension reduction exercise would give an unrealistic picture of how models would perform on unseen data [214].

A *wrapper* approach was applied in this instance because the relatively small dataset size meant that the computational load entailed, prohibitive with large datasets [326], was feasible. To recap on the information provided in Section 3.3.5, the *wrapper* technique uses a prediction algorithm (the *wrapper*) to reduce the dimensionality of a dataset while incorporating interacting effects among features by searching the feature set for subsets that perform best [327]. This is achieved either via a process of *forward sequential selection*, where the search starts with a single feature and iteratively adds more, or *backward sequential selection*, where the search starts with the full feature set and iteratively eliminates single features from each subsequent trial.

Two rounds of backward sequential selection were applied to reduce the dimensionality of the initial dataset. Firstly, 5 subsets were generated using an RF *wrapper* trained using the best hyperparameters found in a grid search across the values outlined in Table 4.3, as it was noted that each repetition resulted in variations and numbers of features chosen. Each of the initial subsets were large, ranging from 200 - 600 features, so it was decided to conduct a further round of dimension reduction using a *wrapper* based on the final prediction algorithm, either RF or SVM. This produced smaller data subsets of sizes ranging from 2 - 200 features.

Model Evaluation

The final step in the modelling process is measuring the performance of the methods chosen, for which there are a number of popular metrics. The applicability of these varies for

Table 4.5 Summary results for baseline (BL) and optimised (OP) models. CA is the FG class accuracy rate (or FG recall rate). ACA is the Average Class Accuracy for both FG and 'nonFG' classes.

Metric	RF-BL	RF-OP	SVM-BL	SVM-OP
CA	30 %	73.3 %	50 %	93.3 %
ACA	60.8 %	80.3 %	67.7 %	88.1 %

different use cases. Given the priority of correctly classifying FG sounds outlined in earlier in Section 4.3.1, it was decided to use FG class accuracy (also referred to as recall) and Average Class Accuracy (ACA) as measures of model success. FG class accuracy indicates correct predictions of FG sounds only. ACA, on the other hand, indicates how many 'FG' and 'nonFG' predictions are on average correct.

Scores from baseline and optimised models from each of the 5 cross-validation folds implemented in the experiment were compared using the Kruskal-Wallace H-test, a non-parametric statistical test for comparing two or more independent examples which can be applied to data samples of 5 or more observations. A significance level of $p < 0.05$ was adopted in this instance to indicate a statistically significant difference between model scores [328].

4.3.2 Results

Table 4.5 summarises the results of ML modelling providing FG class accuracy and ACA scores for baseline and optimised models. The baseline class accuracy scores are poor, 30% of FG sounds captured by RF and 50% by SVM. However, ACA scores are more promising with RF successfully categorising 60.8% of sounds and SVM scoring 67.7%. Taken together, these results suggest that AH may plausibly be modelled using machine learning techniques, though improvement in categorisation success rates will likely be required for any real-world implementation.

The parameter tuning and dimension reduction process described in the foregoing were implemented in an attempt to improve these baseline scores to levels comparable with other

studies. If successful, this would strengthen the case for utilisation of ML in the domain. Regarding RF, FG class accuracy improves from 30% to 73.3%, and ACA from 60.8% to 80.3%. When comparing the fold scores using the Kruskal-Wallace test, the difference between class accuracy baseline and optimised models is statistically significant at the 95% level. The ACA scores are not statistically significant, but only marginally so ($p = 0.057$). SVM FG class accuracy improves from 50% to 93.3%, and ACA from 67.7% to 88.1%. Both of these results are statistically significant. While it is yet to be determined if these success rates would be effective in the implementation of a variable compression codec, the SVM FG class accuracy score of 93.3% is encouraging, given the stated priority of classifying FG sounds. Furthermore, the optimised model scores are comparable to similar studies [47, 264] which indicate that experimentation with feature extraction approaches may lead to further improvements. Finally, when comparing optimised RF with SVM scores, while we report better performance for SVM models in Table 4.5, the difference between optimised learning models was not statistically significant in this case.

In terms of the features selected for final optimised models, no pattern was observed in preponderance of the feature types utilised, those being temporal, frequency and cepstral features. It was interesting to note however that a disproportionate number of double delta features were selected as being most informative. An analysis was made of the features used in the optimised models for each fold and this shows that 23% of the features used are zero order, 20% are delta and 57% are double delta features.

4.3.3 Discussion

The study aim was to establish if predicting AH from objective measures of the sounds is feasible, and it can be regarded as successful in this respect. The FG recall rate achieved (93.3%) is an encouraging starting point, as it suggests that almost all FG instances can be successfully predicted and therefore prioritised for optimal delivery using a variable

compression approach. However, the wide variance in ratings for the majority of sounds reflects the subjective nature of the rating task, which possibly impacts classification scores. This should be contrasted with other audio ML tasks, such as environmental sound classification, where the equivalent labelling task could be said to be objective. In other words, deciding on a hierarchical category for the sound of a dog barking is a far more subjective task than identifying and classifying sounds, i.e. correctly categorising a dog barking versus deciding the sound is actually a cat coughing. Attendant to this discussion is the inherent difficulty of definitively isolating sound stimuli from context, given the variance in auditory perception on the level of the individual referenced in Section 2.6. In effect, the variance in subject responses has resulted in a noisy set of ratings that is not ideal for ML prediction. This demonstrates the difficulty inherent with any study of sound hierarchy concerned with establishing a universal FG/BG categorisation. Such a distinction may not be feasible except in more restrictive terms. The popularity of double delta features used suggests that temporal context is important in a hierarchical classification task.

The process has revealed two further issues requiring attention when attempting to predict AH for any real-world application. Firstly, the amount of labelled data available is a significant issue to address before further ML analysis. The dataset of 40 sounds derived previously as described in Section 4.2 is useful for initial modelling attempts to assess the application of ML techniques to the domain, but a larger dataset would enable a more robust analysis. It should also be noted that providing more sounds of the same kind may provide a more nuanced set of data with which to perform ML analysis. Given the selection of 40 different sounds for the experiments just described, it would be of great interest to examine the ratings returned for multiple examples of a sound. This applies to the example offered previously of a dog barking. Logically, given the perceived proximity of the stimulus would be likely to vary, even under conditions where every effort was made to isolate the sounds from context, different examples of dogs barking would be applied with

varying hierarchical labels, thus informing a more challenging prediction task, but ultimately facilitating the training of a more useful model. Given the performance of DL algorithms in the environmental sound classification literature noted in Section 3.5.5 it should be regarded as likely that a superior performing model can be derived once a suitable dataset is compiled. It should also be noted that while Mesaros *et al.* [215] recommends quantity over quality of data for sound classification applications, every effort should be made to improve the quality of data used whenever possible. Application of DL techniques to this domain will require a labelled dataset of significantly greater size than used in the foregoing. This could potentially be compiled by combining subjective ratings with Active Learning [329] techniques.

Secondly, further investigation is required on the hypothesised impact of how attentional, contextual and other processes, as outlined in Section 2.6, affect our perception of auditory hierarchies. Sound context, for example, may prove a more important indicator of importance than visual accompaniment, suggesting that a weighted schema could be derived experimentally which would model how different factors affect hierarchical categorisation and auditory scene perception. Once complete, such a schema would inform the functioning of a model, meaning that auditory objects could be compressed in terms of their importance to sound scene perception. Thus, audio content could be flexibly delivered to consumers, taking cognisance of the mode of consumption and the capacity of the delivery mechanism involved.

4.4 Conclusions

The thesis statement presented in Section 1.2 outlined the OBJs which have inspired this work. Experiments 1 and 2 are relevant to the following OBJs:

OBJ 1: To develop an understanding of ASA with particular attention to the concepts of object-based audio, AH and modern media consumption paradigms.

OBJ 3: To assess the performance of supervised ML algorithms when predicting AH.

Experiment 1 informs the understanding of ASA as it pertains to media consumption by analysing a corpus of sounds analogous to modern media content and suggests that, even when detached from context to the extent possible in a listening test paradigm, a hierarchy of importance may exist between sounds. Experiment 2 subjects the labels thus gathered to ML analysis and shows promising results that arguably constitute a proof-of-concept working model that predicts AH. Work presented in this chapter is therefore formative in addressing the following RQs:

RQ 2: Does a hierarchy of importance exist between sounds isolated from context?

RQ 3: Is it possible to accurately predict AH using supervised ML methods?

This chapter has outlined perceptual research in Experiment 1 which confirms the existence of a hierarchy of importance between sounds isolated from context. In Experiment 2 these results were subjected to ML analysis and high performance levels (93.3% FG class accuracy) were observed predicting FG instances using an optimised SVM model. However, it is important to note that the dataset used comprised of only 40 sounds and this is noted as being small when compared to most ML datasets. Therefore, Experiment 3, outlined in Chapter 5, will outline methods of maximising the quantity of labelled data while employing minimal manual effort.

Chapter 5

Active Learning for Auditory Hierarchy

5.1 Introduction

One of the largest problems encountered when subjecting datasets to machine learning analysis is the lack of labelled data [20]. Chapter 4 has outlined initial steps to predict AH using a small dataset. In this chapter, methods to build large datasets using minimal manual labelling are outlined. An investigation of AL applied to an AH problem is described in addition to the assessment of selection methods and data representations.

5.2 Active Learning

AL has previously been introduced in Section 3.6, which outlined a number of selection methods for identifying informative instances for labelling by an oracle. The experiment described following, referred to henceforth as Experiment 3, compares three selection methods in an AH task: Uncertainty Sampling AL (USAL), Exploration Guided AL (EGAL) and random selection.

USAL is the most commonly used selection method, using uncertainty in model prediction as a metric to select instances for labelling. As outlined in Section 3.6.1, it has been used in a variety of audio applications including environmental sound classification [309], bird sound categorization [330] and speech emotion recognition [314]. USAL is a model-based selection method, which holds that the instances which will be most informative for labelling purposes are those the classifier categorises with the least confidence. It therefore selects these instances first.

A number of methods to identify uncertainty have been introduced in Section 3.6.1. The margin method is implemented in Experiment 3. This ranks instances by their proximity to a classifier decision boundary, presenting those closest for labelling, as they are the instances most difficult to categorise. USAL is computationally expensive and potentially very time-consuming for large datasets, as it requires a model to be trained every time labelled instances are added.

The EGAL selection strategy is a model-free method that addresses this shortcoming, which has been found to outperform USAL in other domains [331]. EGAL identifies useful instances for classification purposes in relation to their location in the feature space relative to neighbouring instances and proximity to already labelled instances. It has been used in text classification applications [310] but to our knowledge this is the first application of this technique to an audio problem. EGAL seeks to identify instances in clusters that are furthest from labelled instances on the assumption that dense clusters more diverse from labelled instances will be most informative for classification purposes. This is implemented by first calculating a *density* value per instance, defined as the sum of similarities between the instance and all other instances within a certain radius. Here the inverse of Euclidean distance for this measure is implemented, which indicates similarity to neighbouring instances. Secondly, a *diversity* value is calculated by measuring instance distance to the nearest labelled instance of the dataset.

In all, 3 forms of EGAL are investigated here: the first uses the density measure only, selecting instances from dense areas of the feature space. The second uses diversity in isolation, which selects instances that are most diverse from already labelled instances. The third EGAL form is a hybrid approach, which combines the first two to select instances from the most dense areas of the feature space that are most diverse from already labelled instances. To provide a baseline comparison for USAL and EGAL methods, a random selection strategy is also implemented.

5.3 Methodology

The following sections outline the methodology applied to our investigation of AL for AH. Audio stimuli are described, as are label collection methods based on those used in Experiment 1 (outlined in Section 5.3.1). Feature extraction and data preparation are covered in Section 5.3.2 and classifier choice is outlined in Section 5.3.3. Having extracted a number of feature representations, it was decided to compare these in an initial cross validation experiment which identifies the optimal feature representation to submit to AL. This is outlined in Section 5.3.5. Finally, AL using USAL, EGAL and random selection is applied to the chosen representation as described in Section 5.3.6. Figure 5.1 presents a methodology overview for Experiment 3. The Python language was used for implementation using the associated Scikit-learn [332], SciPy [333] and Pandas [334] libraries.

5.3.1 Dataset Creation

As introduced in Section 2.4.3, the ESC datasets [101] have been compiled from the Freesound website (*freesound.org*) for use in computational audio scene analysis contexts for training and testing automatic classification of sounds. They have been selected for use in

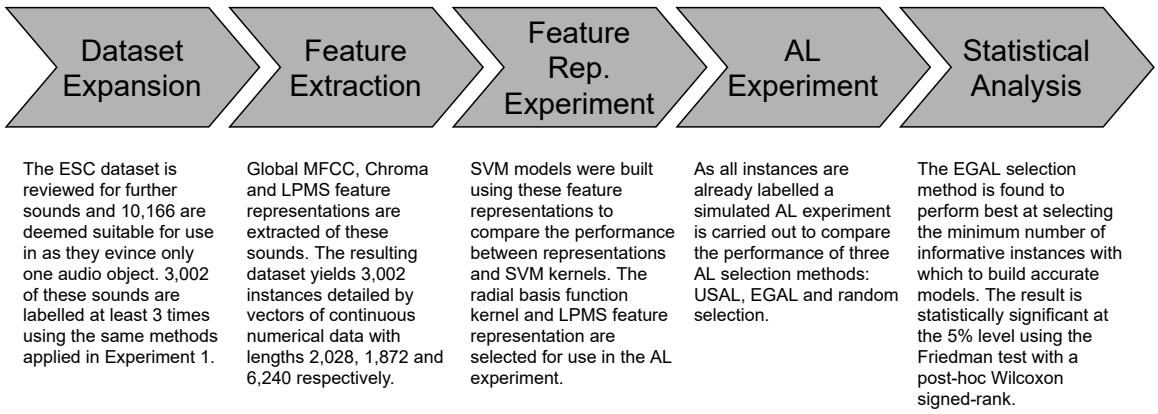

Fig. 5.1 Methodology overview for Experiment 3.

Experiment 3 because of their use in Experiments 1 and 2 and because they provide a large bank (>250,000) of potential stimuli with associated sound class metadata.

In excess of 20,000 sounds were reviewed by the authors for suitability of use in Experiment 3 with care taken to exclude sounds which evinced more than one sound event in order to provide a corpus of stimuli isolated from context in so far as this is possible. The vetting process for Experiment 3 sounds differed from that applied for Experiments 1 and 2 in that it was decided to include sounds judged to be amorphous combinations, such as urban and nature soundscapes, as long as no one sound of the combination was judged to predominate. An example of this would be the concept of "urban hum" [49, pg. 68] where the combination of numerous cars and other vehicles becomes a percept of 'traffic' rather than a series of individual objects. This process resulted in the selection of 10,166 sounds as suitable for inclusion as they did not evince more than one audio 'object'. Note that while not all of these sounds were labelled in Experiment 3 they will be utilised in subsequent experiments when applied with predicted hierarchical labels. Table 5.1 outlines the sounds selected for inclusion in Experiment 3 organised into 12 broad classes based on the metadata provided from the ESC dataset.

Table 5.1 A summary of instance count, average score and standard deviation (σ) per class for all 3,002 sounds for which at least 3 ratings were gathered. The highest occurrences are reproduced in **bold**, the lowest are underlined.

Class	No.	Average Score	σ
Nature	**523**	1.655	0.578
Ambience	507	1.477	0.504
Animal	408	2.121	0.569
Urban	370	1.382	0.437
Machine	285	1.941	0.585
Human	266	2.131	0.461
Other	226	2.325	0.564
Domestic	145	2.307	0.527
Travel	115	1.285	0.356
Actions	67	2.269	0.573
Alarms	55	**2.535**	0.41
Bells	35	2.41	**0.715**
Total/Average	**3,002**	**1.986**	**0.523**

These sounds were labelled using the same methodology as Experiment 1, described in Section 4.2.1. Participants for Experiment 3 came from employees of Xperi/DTS Inc. and researchers in the TU Dublin School of Media.

In all, 3,002 sounds were labelled a minimum of 3 times on a FG — N — BG scale by 149 participants (73% male, 7% 18-24, 49% 25-44). An average of 83.42 sounds were rated per participant, with each given the opportunity to rate 100 stimuli. The average time taken to complete the rating process excluding outliers greater than 1 hour in duration was 19 minutes 54 seconds. The numerical coding for each category (BG - 1, N - 2, FG - 3) was used to generate mean and standard deviation scores for each sound. The mean was used in Experiment 3 as it was felt this would give a more realistic view of the subjective nature of the labelling task given most sounds were rated 3 times. For example, in cases where a sound received 2 FG and 1 BG rating this would result in a mean score of $2.\dot{3}$, indicating a non-unanimous rating, versus a median score of 3 which in data analysis would indicate a unanimous FG score. The average rating score and standard deviation per class are provided in Table 5.1.

This table shows that sounds such as 'Alarms' are likely to be labelled as FG. Sounds categorised as 'Travel' are most likely to be labelled BG reflecting the interior public transport hum ambience present in many of these sounds. Standard deviation per sound class varies between 0.41 and 0.715. The variance in average rating is outlined using the boxplots reproduced in Figure 5.2 giving an indication of the variance in the data which in this instance indicates the degree of subject consensus on BG/N/FG sounds.

For illustrative purposes, the sounds are organised into three average rating score bands. There are 1,156 instances with an average rating of 1.5 or under which are designated BG sounds. There are 608 sounds with an average rating of greater than 2.5 that are designated FG sounds. The remaining 1,238 sounds have average ratings greater than 1.5 and less than 2.5. These are referred to as neutral sounds. The width of each box plot is proportionate to the number of instances summarized in each rating band.

Similar to the findings noted in Chapter 4, a greater consensus is noted among subjects as regards sounds considered most FG or most BG, there being less variance in the ratings for these bands than those sounds considered N. Interquartile range for both FG and BG bands is approximately 0.33 of a rating score. N sounds on the other hand exhibit greater variance in rating scores compared to BG and FG sounds, interquartile range here being twice that of BG and FG sounds, 0.67 of a rating score. The high degree of variance for some sounds, indicating a lack of consensus between subjects as to the correct sound class, is to be expected with a subjective labelling task and the dataset evinces disagreement between annotators as to the correct hierarchical category for many instances. The proposed application of a variable compression codec suggests a priority of identifying FG sounds, so for the purposes of subsequent investigations it was decided once again to address the data as a binary classification problem. Accordingly, all sounds achieving an average score $\geq$ 2.5 (608 instances, 20.25%) are categorized as 'FG'. All others (2,394 instances, 79.7%) are categorized as 'nonFG' sounds.

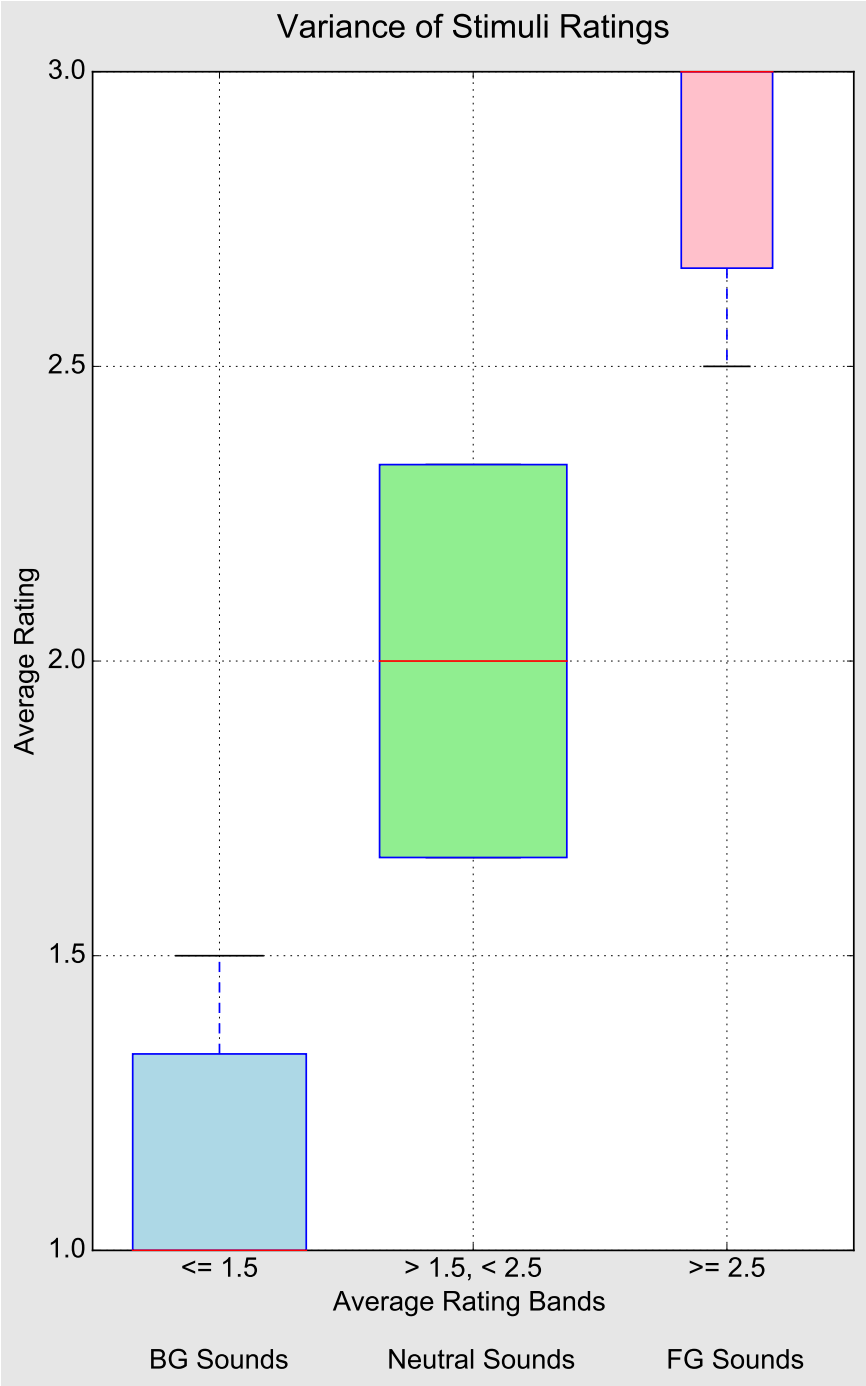

Fig. 5.2 Boxplots outlining the variance in average sound ratings grouped in broad bands for the 3,002 sounds for which at least 3 ratings were gathered. Note that the minimum average score for BG sounds is 1, hence there is no quartile or minimum whisker below this value. Similarly, the maximum average score for FG sounds is 3, hence this band has no quartile or maximum whisker above this value. Also, the width of each boxplot is proportional to the number of instances summarised in each band.

5.3.2 Feature Extraction

Experiment 2 has utilised statistical summaries of LLD features and applied feature selection to identify those of most use in the prediction task. For Experiment 3 it was decided to compare a number of global representations given the popularity of this approach in the audio ML domain.

The Python LibROSA [335] package was used to extract three different feature representations for each audio stimulus. Mel Frequency Cepstral Coeficients (MFCC) and Log Power Mel Spectrogram (LPMS) representations were extracted based on their popularity in audio machine learning applications [20] and a chroma representation was also extracted based on its usefulness in Experiment 2. All files were first downsampled to 16kHz to account for the variable recording quality of sounds sourced from Freesound, such as the ESC datasets. A Hann window of the form outlined in Equation 5.1, (where n = sample number, M = the number of points in the output window) is used to extract audio data. A number of different window types are available for audio extract, such as the Hamming window used in Experiment 2. The Hann window was used in this instance as it is the default window applied in the LibROSA framework [335] and has been used in multiple audio ML tasks [221, 336, 337].

$$w(n) = 0.5 - 0.5cos\left(\frac{2\pi n}{M-1}\right),\ 0 \leq n \leq M-1 \tag{5.1}$$

In line with similar experiments, an environmental sound classification task [250] and an acoustic event detection experiment [338], a window size of 128 ms (2048 samples at 16 kHz) and stride of 32 ms was used to extract 12 frequency bands of chroma, 13 bands of zero-order MFCC feature vectors and 40 bands of LPMS features. This results in the feature representation dimensions outlined in Table 5.2, where 3,002 instances with at least 3 hierarchical ratings are represented by the frequency bins noted for MFCC, chroma and LPMS features and 156 temporal analysis frames. From these zero-order, delta, double delta

Table 5.2 A summary of feature representation data vectors and their dimensions. For each representation (MFCC, chroma and LPMS) zero-order (ZO) and 1st (1OD), 2nd (2OD) and 5th order (5OD) delta vectors are computed, resulting in a total of 12 initial representations. The Dimensions column denotes the number of instances x number of frequency bins x temporal feature extraction frames for each feature representation. These vectors were flattened prior to input to SVM models for AL.

Type	Dimensions	Flattened Dimensions
MFCC ZO	3002 x 13 x 156	3002 x 2,028
MFCC 1OD	3002 x 13 x 156	3002 x 2,028
MFCC 2OD	3002 x 13 x 156	3002 x 2,028
MFCC 5OD	3002 x 13 x 156	3002 x 2,028
Chroma ZO	3002 x 12 x 156	3002 x 1,872
Chroma 1OD	3002 x 12 x 156	3002 x 1,872
Chroma 2OD	3002 x 12 x 156	3002 x 1,872
Chroma 5OD	3002 x 12 x 156	3002 x 1,872
LPMS ZO	3002 x 40 x 156	3002 x 6,240
LPMS 1OD	3002 x 40 x 156	3002 x 6,240
LPMS 2OD	3002 x 40 x 156	3002 x 6,240
LPMS 5OD	3002 x 40 x 156	3002 x 6,240

and fifth-order delta representations were extracted as delta features were prominent in the features selected as being most useful for categorisation purposes in Experiment 2. All data is scaled and bands from each data matrix are flattened and organized into 12 data subsets, 4 each for the MFCC, chroma and LPMS data, a summary of which is presented in Table 5.2.

5.3.3 Algorithm Selection

A Support Vector Machine (SVM) algorithm is used for classification purposes as it was most successful in Experiment 2 (as outlined in Section 4.3) and has been used extensively on audio ML applications [47, 221, 339]. A number of different kernels can be used with a SVM, three are investigated here: the Radial Basis Function (RBF), Polynomial and Linear kernels.

Table 5.3 Default parameters used per kernel in the initial classification exercise. The 'scale' value for the gamma parameter uses $1/(no.features * variance)$ as value of gamma.

Kernel	Parameters
Radial Basis Function	C=1, gamma='scale'
Linear	C=1
Polynomial	C=1, degree=3, gamma='scale'

5.3.4 Performance Measures

Average Class Accuracy (ACA), precision and recall scores were used to evaluate model performance. These measures have been introduced in Section 3.3.6. When building models the data was first split into 5 stratified folds and to fit parameters the training set was further divided into 4 stratified folds to form train and validation portions as outlined in Section 3.3.2. The results reported are therefore averages across 5 folds.

To assess performance during AL runs an Area Under the Learning Curve (AULC) metric was used to compare learning curves for different selection methods using a single number. This metric utilises the same concept as the AUC metric introduced in Section 3.3.6, applying it to the learning curve accuracy of models trained during the progression of AL as opposed to a ROC curve (see Appendix B) which plots TP against FP rates for a number of thresholds. AULC therefore calculates a single value to represent the area underneath the learning curve, so trials using different selection methods can be compared.

5.3.5 Cross Validation Experiment

In a preliminary experiment optimal feature representation was investigated, firstly for distinguishing between FG and nonFG sounds and secondly to examine which SVM kernel works best on these data. An SVM with three different kernels (RBF, polynomial and linear) was applied using default parameters outlined in Table 5.3.

Class weights were adjusted to penalise mistakes inversely proportional to the number of instances in each class to adjust for the class imbalance in the dataset. Results showed that

Table 5.4 Average Class Accuracy (ACA), and Class Accuracy scores for FG and nonFG classes per kernel and feature representation. As noted, the 'All' representation is an amalgamation of the other 3.

Kernel	Measure	MFCC	Chroma	LPMS	All
RBF	ACA	72.2%	65.7%	**73.9%**	74.3%
	FG	67.3%	53.6%	67.1%	69.7%
	nonFG	77.2%	77.8%	80.7%	78.9%
Linear	ACA	63.9%	53.1%	63.1%	60.3%
	FG	45.9%	31.9%	38.5%	35.9%
	nonFG	81.9%	74.3%	87.8%	84.8%
Polynomial	ACA	72.4%	63.0%	73.4%	**74.4%**
	FG	66.6%	61.0%	69.1%	70.1%
	nonFG	78.3%	65.0%	77.7%	78.7%

extracted delta representations gave no improvement on the zero-order versions in this case, and so these were discarded.

In addition to MFCC, chroma and LPMS zero-order representations one further representation is investigated: a concatenation of these three, labelled the 'All' representation in Table 5.4. This table also provides ACA and class accuracy scores per kernel and representation, where the best ACA performances are highlighted in bold typeface.

The best ACA score (74.4%) is achieved using the polynomial kernel on the 'All' representation. The ACA score for the RBF (73.9%) kernel on the LPMS representation is only slightly behind this, and training is considerably quicker using LPMS compared to 'All' representation. The MFCC and LPMS representations perform similarly to the 'All' representation, while the chroma is notably poorer. It was decided to proceed with the LPMS representation as it performs slightly better than the MFCC and takes significantly less time to train than the 'All' representation, while achieving scores only slightly lower. With regard to kernel choice, RBF and polynomial kernels are observed to perform more strongly than linear. The overall difference between RBF and polynomial is marginal, so the RBF kernel was selected as it is more commonly used [340].

5.3.6 Active Learning Process

As 3,002 instances were pre-labelled as described in Section 5.3.1, a simulated labelling exercise was next conducted to assess AL for AH. A stratified, randomly selected hold-out test set of 501 instances was extracted to measure performance. The remaining 2,501 instances form the pool of 'unlabelled' examples. Due to the random nature of the hold-out test set and 'unlabelled' pool, three random splits are formed to counteract the chance of a single iteration providing a misleading result. The results reported are therefore averages over 3 iterations.

Agglomerative clustering was applied on the 'unlabelled' pool to select the first set of instances forming 5 distinct clusters. A batch size of 10 instances were selected from the cluster centroids, 2 from each cluster, as this has been shown to be an effective way to initiate AL [310, 341]. During labelling runs, ACA was used to measure performance due to the imbalanced class distribution. The initial instances were labelled, a model trained on them and an ACA score calculated on the hold-out test set. The selection method was then used to pick the next batch of 10 instances from the 'unlabelled' pool, these were labelled, added to the other labelled instances and a new ACA score calculated on the hold-out test set. This process was iterated until no instances remained to be 'labelled'. The ACA values were used to plot a learning curve used to compare methods both visually and with an AULC value. The baseline comparison method used was a random selection strategy, which does not seek to intelligently select instances for labelling.

5.4 Results

In total five selection methods are investigated:

- USAL, which uses a SVM to identify the instances closest to the classification decision boundary.

- Diversity EGAL, which uses the diversity measure from EGAL to select instances that are most diverse from already labelled instances.
- Density EGAL, which uses the density measure from EGAL to select cluster centroids from the most densely populated areas of the feature space.
- Hybrid EGAL, which combines density and diversity EGAL measures to select cluster centroids that are most diverse from already labelled instances.
- Random selection, selects instances randomly. Three random selection runs are implemented to account for randomness.

Figure 5.3 shows results of labelling runs from 10 to eventually 2,501 'labelled' instances. It includes a shaded area that denotes the maximum and minimum values achieved by random selection for each batch, which demonstrates large variance.

The EGAL runs are noticeably strongest early in the training runs, all quickly achieving scores in excess of 70% accuracy. USAL does not match this performance and indeed, given the popularity of this method in other domains [303], is surprisingly less effective than random selection method apart from the earliest section of the run under 70 labels. There is considerable variance between the maximum and minimum scores from the random selection method, showing it is not reliable in this application. Figure 5.4 focusses on the early portion of the labelling run which tracks scores achieved between 0 - 500 labels.

This highlights the success of diversity EGAL, which achieves 74% ACA using only 50 labels. The other EGAL variants are fractionally behind this early result, but perform similarly up to approximately 120 labels, with the performance of density EGAL being notably strong beyond this point. The random selection strategy does not improve on the accuracy level of diversity EGAL at 50 labels until it is provided 350 labels. USAL requires 1,410 labels to achieve the same. Table 5.5 offers a summary of ACA and AULC scores at different points from each labelling run.

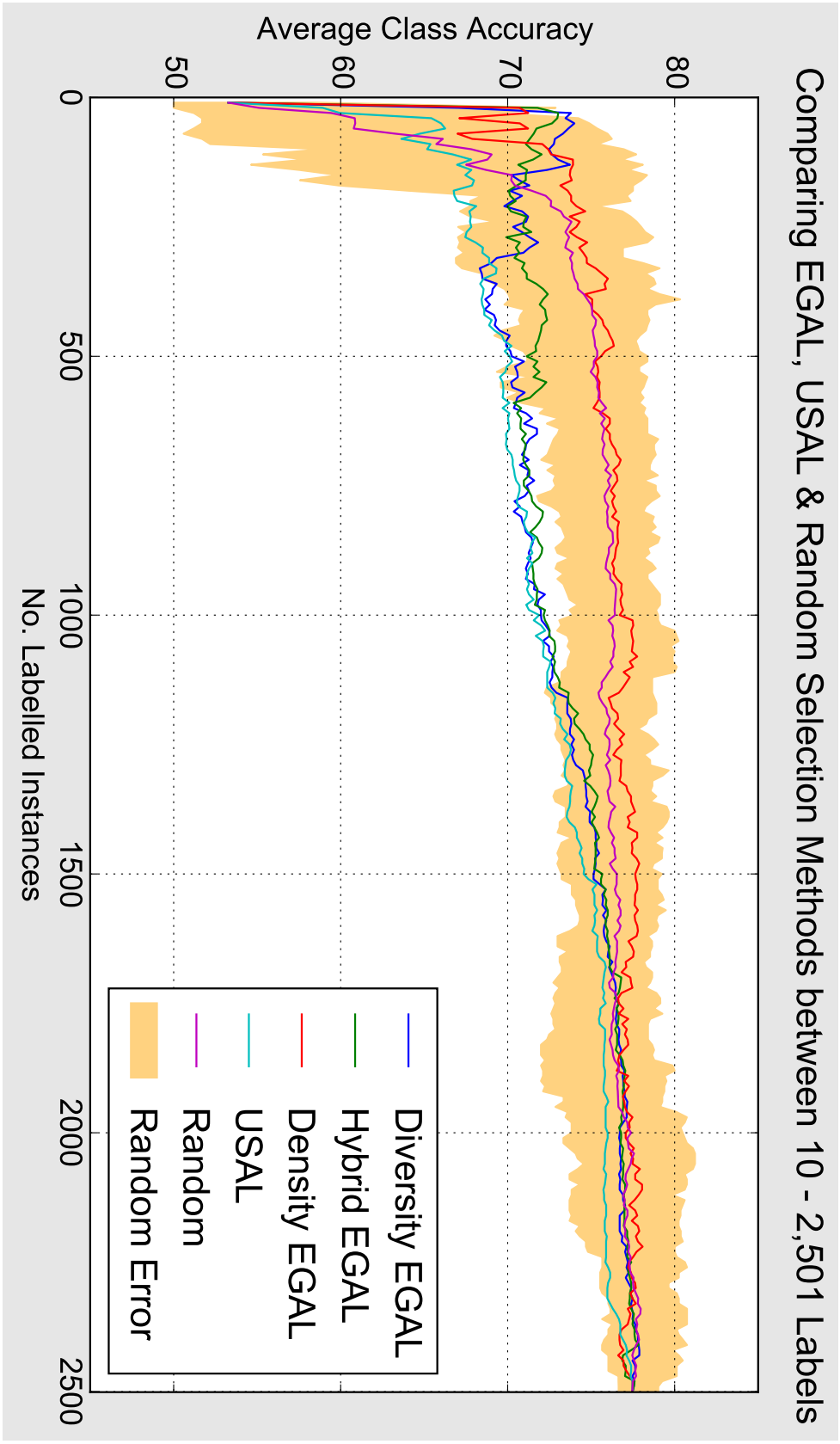

Fig. 5.3 Comparison of Active Learning selection methods displaying balanced accuracy (ACA) scores achieved from 10 - 2,501 labels. Each line denotes the overall average score for each method per batch. The shaded area denotes the variance observed from the random selection method.

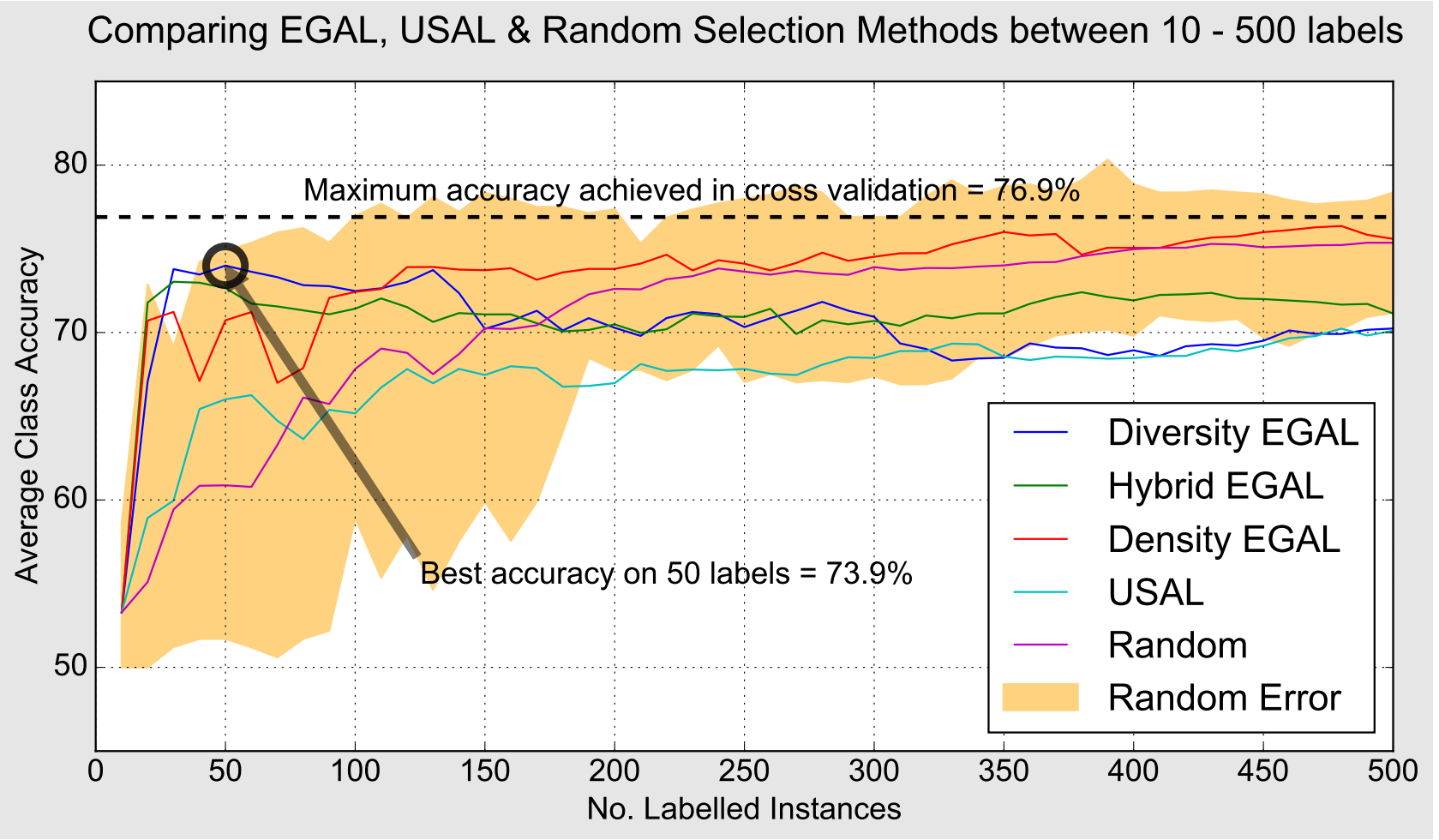

Fig. 5.4 Comparison of selection methods for early stage (between 0 and 500 labels) Active Learning runs. Each line denotes the overall average score for each method per batch. The shaded area denotes the variance observed from the random selection method.

Table 5.5 A summary of model accuracy and AULC scores for points in the labelling run per AL method. Using Diversity EGAL it is possible to achieve high classification accuracy (74%), using the 50 most informative instances selected using this method.

No. Labels	50	100	200	500	2501
Method			**ACA Scores**		
Diversity EGAL	**74.0%**	72.5%	70.3%	70.2%	77.5%
Hybrid EGAL	72.7%	71.4%	70.5%	71.1%	77.5%
Density EGAL	70.7%	72.4%	73.8%	75.6%	77.5%
USAL	66.0%	65.2%	67.0%	70.1%	77.5%
Random	60.9%	67.8%	72.6%	75.4%	77.5%
			AULC Scores		
Diversity EGAL	20.4	57.1	128.9	338.4	1838.2
Hybrid EGAL	20.8	56.7	127.7	341.6	1845.2
Density EGAL	20.2	54.9	128.2	353.3	1900.8
USAL	17.8	50.4	117.5	323.1	1808.6
Random	17.2	48.6	117.9	340.4	1877.5

Here we see that the performance achieved from 2,501 labelled instances across the three random splits used to compare selection methods was 77.5% ACA. In light of this, the score of 74% from 50 labels achieved by diversity EGAL is a strong result, meaning that AL in this instance can achieve 95.5% of total possible model accuracy using only 1.7% of labels. As noted, using random selection, 350 labels, or 11.7% of the total, are required to improve on this accuracy level.

The Friedman test to compare more than two samples and the Wilcoxon signed-rank as a post-hoc test between pairs of samples are used for statistical analysis. In the case of the Wilcoxon test a Bonferroni correction is applied for the significance level in order to reduce the Type I error rate (identifying a significant effect where there is none) [342]. This results in a revised significance level of 0.005 for the post-hoc Wilcoxon tests as 10 comparisons are made. Additionally, for the Wilcoxon test, runs of 20 measurements are compared as comparisons below this point are not recommended due to sample size [333]. The Friedman and Wilcoxon are non-parametric tests that look for differences between related samples

and are noted to be a safer option than using parametric tests as they do not assume normal distributions or homogeneity of variance [343].

A Friedman test on the AL balanced accuracy values up to 200 labels provided is significant at the 95% level (p = 8.03E-10). The Wilcoxon tests reveal that the differences between EGAL variants are not significant to the revised significance level. However, the differences between EGAL and USAL, and between EGAL and random selection methods are significant to the revised significance level. This indicates that EGAL is superior to both USAL and random selection at selecting instances on which a classifier can be built to achieve high accuracy levels with minimal labelling. These results also suggest that there is little difference between the EGAL variants in this instance, as the Wilcoxon comparison results between EGAL runs are not significant.

5.5 Discussion

This chapter has explored a series of AL approaches to an AH labelling problem. In this case, it has been found that it is possible to classify to 95.5% of maximum model accuracy by labelling only 1.7% of dataset instances using the EGAL selection method. Using a random selection strategy, it is necessary to select 350 instances (11.7% of the total) to surpass this accuracy level. The large variance observed in scores using the random selection strategy makes this an unreliable method in this instance, however. The poor performance of the USAL selection method in this case is surprising given its popularity in other domains. This is possibly due to the low number of confident predictions made by the SVM model, which resulted in many instances with similar uncertainty scores, thereby making the selection of informative instances more difficult.

DL techniques are acknowledged as state-of-the-art in the audio classification domain [27] but are limited in terms of application to specific problems by the existence of suitable, large, appropriately labelled datasets. In a real-world scenario where potentially millions of labelled

instances are required for DL applications, the performance of EGAL in this instance suggests a potential for significant savings on manual labelling effort in both time and money terms for many audio ML problems based on subjective human perception and evaluation of environmental sounds.

This is particularly interesting given the significance accorded to the emergence of large datasets in other domains. For instance, the existence of ImageNet [344], consisting of over 14 million labelled images, is considered an important factor in the success of computer vision techniques and the influence of the DL methods applied to them [20]. While a number of large audio datasets are available [67, 345, 346] they are not labelled in a manner that is universally appropriate for all audio ML problems. Having the ability to label these datasets for other categorisation tasks is a useful contribution to knowledge, particularly for applications where subjective judgement is required in the labelling process. Being able to quickly and efficiently generate new labels for existing sound corpora has the potential to facilitate the study of many more specific questions than would be the case if such datasets required extensive manual labelling for each task. AH applied to the concept of variable asset compression is one example of such a task.

Ultimately, the use of AL in this manner is a trade-off between the manual effort required to label large numbers of instances and the increasing accuracy to be attained by sourcing more manual labels. This work demonstrates that by intelligently selecting informative instances over 95% of total possible accuracy can be reached using 1.7% of all labels. However, scaling datasets to millions of instances even with the use of AL methods is still a challenging logistical task, even if the selection of informative instances for the purposes of labelling greatly reduces the manual workload. For example, to reach the same level of performance on a dataset of 100,000 instances would require a minimum of 1,700 labels. The approach shows promise, however, and for the purposes of this research is suitable for

use in concert with data augmentation techniques and crowd-sourced labelling methods to compile a large audio dataset with hierarchical labels.

5.6 Conclusion

This chapter has presented research which investigates the application of AL techniques to a hierarchical audio labelling task. This process has involved a detailed assessment of the SVM algorithm and associated kernel types, different feature representations and AL selection methods on a hierarchical audio ML task. The work described therefore directly addresses the following OBJ, initially outlined in Section 1.2:

OBJ 3: To assess the performance of supervised ML algorithms when predicting AH.

Investigating methods to efficiently label audio stimuli with hierarchical labels broadens understanding of the phenomenon in a ML context and provides background for an assessment of subsequent algorithm investigations. The work outlined here addresses the following RQ:

RQ 3: Is it possible to accurately predict AH using supervised ML methods?

The RBF and polynomial kernels were noted to perform well in this task. It was also found that high performance was possible using the LPMS feature representation even though the training time was much shorter than when using a concatenated feature representation including MFCC and chroma. EGAL was found to be the most effective selection method in this case, outperforming both USAL and random selections, making it possible to classify to 95.5% of maximum model accuracy while requiring only 1.7% labelled instances.

It is intended to utilise the best methods identified during Experiment 3 to hierarchically label a large corpus of audio data, and to further expand this corpus using data augmentation

techniques. This dataset will also be suitable for use in deeper investigations on the functioning of AH, noted in Chapter 2 to be influenced by a series of factors such as sound context, subject experience level and the physical characteristics of the sound itself. In summary, Chapter 6 will describe a number of investigations, collectively designated Experiment 4 for the sake of clarity, which will:

- Validate the accuracy of predicted labels using a dedicated validation set of manually labelled instances.
- Build a corpus of 100,000 hierarchically labelled instances using crowd-sourced labelling, AL and data augmentations.
- Examine the effect of different sound rating threshold values for determining FG/nonFG instances.
- Compare the performance of DL and SVM algorithms on different dataset configurations and feature representations.

Chapter 6

Deep Learning for Auditory Hierarchy

6.1 Introduction

Work outlined in Chapter 5 has established that the EGAL algorithm can be used to minimise the manual labelling required to label an auditory corpus with hierarchical labels. Chapter 3 has provided an overview of the audio ML domain and highlighted that deep learning algorithms tend have superior performance, once they are supplied with sufficient data, in other ML domains such as computer vision and environmental sound classification. The work outlined in this chapter compares the performance of a popular DL algorithm (CNN) against the SVM used in prior experiments.

This was firstly approached by applying EGAL to a corpus of unlabelled instances to identify those most informative for labelling purposes. These were then manually labelled and used to train a model to predict labels for the remaining unlabelled instances. Secondly, in order to validate the accuracy of predicted labels, a randomly selected validation set was also manually labelled. These manual labels were then compared to those predicted for the validation instances to validate the accuracy of labels predicted for Experiment 4. The methods used for labelling were the same as those employed for Experiments 1 and 3.

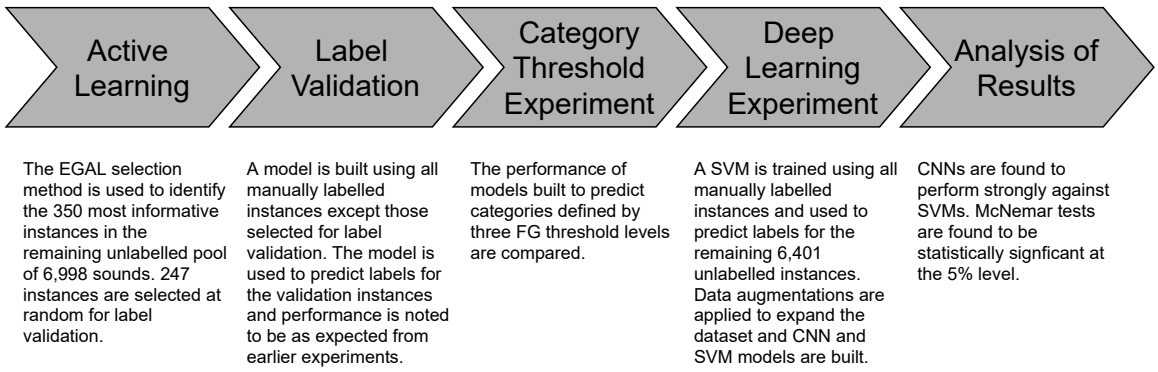

Fig. 6.1 Methodology overview for Experiment 4.

Thirdly, different category threshold values were used to categorise 'FG' and 'nonFG' instances, as it is possible that different audio applications would require differing categorisation criteria. For example, acceptable performance for a variable compression codec may be achieved by focussing on some of the 'most FG' sounds. An auto-mixing application for content creators may need to have a broad definition of 'FG' in order to be effective. Therefore, an investigation of models trained using different thresholds is of interest in the context of audio applications.

Fourthly, the category threshold selected was used to apply labels to all remaining unlabelled instances and data augmentations were applied to bring the total number of instances to 100,000. CNN and SVM algorithms were then trained on a number of dataset and feature representation configurations to assess performance. The chapter concludes with a discussion of results and suitability of each algorithm for the task of predicting AH. An overview of the methodology for this experiment is offered in Figure 6.1.

6.2 Active Learning Experiment

This section outlines an experiment that applies the EGAL selection method to an unlabelled audio corpus to select the optimal instances for manual labelling, such that a model can be

trained to label instances accurately. The accuracy of the labels predicted using this method is checked using a dedicated validation split.

6.2.1 Methodology

The assets and techniques described in the following have been introduced in Chapters 4 and 5. Here they are elaborated upon only if they differ from methods introduced prior.

For this study the ESC datasets [101] are again used as a source of stimuli. In the following work 10,000 sounds selected as described in Section 5.3.1 comprise the initial dataset. These are divided into a corpus L of 3,002 stimuli which have been manually labelled as described in Chapters 4 and 5, and an unlabelled corpus U of 6,998 stimuli.

A Log Power Mel Spectrogram (LPMS) feature representation was used in this investigation, as it was found to be an effective compromise between performance and processing time in Experiment 3. The same settings are used here to perform extraction. The SVM algorithm was used to build models in order to provide a point of comparison with Experiment 3 in which they were found to be effective. ACA, precision and recall scores are used to evaluate model performance.

Active Learning Process

EGAL was applied to U in an iterative manner, selecting 10 instances at a time to select in total 5% (350) of the instances in this corpus as being the most informative for the purposes of predicting labels for all of U. The batch size of 10 and label budget of 350 instances are based on those outlined in Chapter 5, which achieves 95.5% of model accuracy when trained using 1.7% of the most informative instances. For clarity, the 350 instances selected by EGAL and removed from U will henceforth be referred to as 'EGAL Instances'. From the remaining instances in U, 247 were selected at random to form a label validation split and will be referred to as 'Validation Instances'.

The instances thus selected were then deployed to the online labelling environment used for previous experiments described in Chapter 4. Subjects, sourced from the TU Dublin school of media and participants in previous studies, were asked to label the instances as either BG, N or FG. The numerical coding (BG=1, N=2, FG=3) was used to calculate an average rating score for each instance which was used to rank sounds on a BG to FG scale. Presentation order was controlled using random orders sourced from *www.random.org* and every sound was rated at least 3 times. Subjects were asked to listen to sounds using headphones in a quiet environment and not to adjust their volume for the duration of the session, having set it to a comfortable level in an initial training phase.

Validating Predicted Labels

An average score threshold of ≥ 2.5 was initially used to form a binary categorisation schema of FG (≥ 2.5) and nonFG (< 2.5) sounds. Applying a threshold of 2.5 results in the class distributions presented in Table 6.1. It should be noted that there is a significant difference in the class distributions between different sound sets. The instances in *L*, which were not selected using an intelligent selection criteria such as EGAL, contain 20.1% FG instances. EGAL Instances have a greater proportion of FG sounds, suggesting a tendency in the EGAL algorithm to select FG instances as being more informative for labelling purposes. The Validation Instances by comparison, selected at random before labels were gathered, contain only 12.1% FG instances, which has an implication for model accuracy that will be relevant in subsequent discussions. This is possibly because of the propensity of the EGAL algorithm to select a higher proportion of FG instances, therefore leaving fewer in the pool from which the Validation instances were selected at random.

A grid search using a stratified 5-fold CV to fit parameters using the parameter grid outlined in Table 6.2 was then applied to all manually labelled (3,352) instances. The best parameters were then used to build a model trained on all labelled instances, and this model

Table 6.1 Class distribution of the splits used in both Active Learning process and subsequent validation.

Split	Total	FG	nonFG	% FG
L	3,002	602	2,400	20.1%
EGAL Instances	350	128	222	36.6%
Validation Instances	247	30	217	12.1%
Totals	3,599	760	2,839	21.1%

Table 6.2 An outline of grid search parameters used. 'C' is the only parameter varied for the linear kernel. The 'degree' parameter is varied for the polynomial kernel only.

Parameter	Values
kernel	radial basis function, polynomial, linear
C	0.001, 0.01, 0.1, 1, 10, 100, 1000
gamma	'scale', 0.01, 0.1, 1, 10, 100
degree	2, 3, 4

was used to predict labels for the Validation Instances. These labels were then compared to subjective manual labels acquired for these instances in the labelling exercise to generate accuracy scores.

6.2.2 Results

ACA and class accuracy scores for the labels predicted for the Validation Instances are reproduced in Table 6.3, which compares them to the performance noted in Experiment 3 achieved using 3,002 manually labelled instances and the same algorithm, kernel and feature representation. For reference, see the performance noted on the RBF kernel and LPMS feature representation in Table 5.4, Section 5.3.5. Table 6.4 provides a classification report on the accuracy of predicted labels, where the 'f1 Score' is a harmonic mean of the precision and recall scores and 'Support' indicates the number of instances for each class.

These findings demonstrate that the results on Validation Instances are in line with those achieved previously. This indicates that the EGAL algorithm has been successful in selecting

Table 6.3 A comparison of accuracy scores achieved on the Validation Split in the current instance compared to results noted in Experiment 3.

Metric	Experiment 3	Validation Instances
ACA	73.9%	76.4%
FG Class Accuracy	67.1%	66.6%
nonFG Class Accuracy	80.7%	86.2%

Table 6.4 A classification report outlining the accuracy of predicted versus manual labels for the Validation Instances.

Category	Precision	Recall	f1 Score	Support
nonFG	0.95	0.86	0.90	217
FG	0.40	0.67	0.50	30

a minimal number of informative instances that allow a model to be trained to high accuracy to predict labels on an unseen corpus. However, an asymmetry between FG and nonFG class results was noted and is reflected in prior work also. In this case, precision (95%) and recall scores (86%) are strong for the nonFG class but noticeably poorer for the FG class, particularly in the case of the FG Precision score which is only 40%, meaning that of all the instances predicted as FG by the model only 40% are actually of that class. However, FG Recall is better at 67%, meaning two thirds of the FG instances are correctly predicted. The relatively low proportion of FG instances is possibly at fault in this case. To summarise, this means many nonFG instances are incorrectly predicted as FG, but 95% of the nonFG predictions are correct.

SVMs classify instances based on which side of a decision boundary they lie. A negative or positive margin value is generated for each prediction denoting instance distance from the decision boundary, with the sign value denoting class prediction. The margin value for all Validation Instances was analysed for patterns to investigate if it could be used to tune the results of the model trained to predict labels. Implementing a revised boundary of ≤ -0.937 in this instance results in the confusion matrix provided in Table 6.5, though this would obviously come with the risk of overfitting the model to the dataset. This example

Table 6.5 A confusion matrix achieved by tuning the margin information provided by the SVM model. A margin value of -0.937 was used to classify instances.

n=247	Predicted: nonFG	Predicted: FG
Actual: nonFG	100	117
Actual: FG	0	30

demonstrates however that the margin information may be used to optimise classification of FG instances at the expense of incorrectly capturing more nonFG instances as FG.

6.2.3 Discussion

As noted previously, the best performing model achieves 76.4% ACA when comparing the predicted labels to those derived from manual labelling on the validation instances. This is a similar accuracy score to that observed in a previous experiment (73.9%) using similar data and methods, which suggests that Active Learning, specifically EGAL, is an effective method for minimising the number of instances required to achieve high accuracy when using those instances to train a model to predict AH. This in turn suggests EGAL would be an effective method for selecting informative instances with the goal of labelling large corpora of auditory stimuli with minimal manual effort.

Further examination of these results highlights the fact that focussing on FG class accuracy may not be the only option for audio ML tasks. Focussing on correctly identifying nonFG instances may be an effective strategy when seeking to identify less important objects for automated mixing tasks, given the high precision score (95%) achieved on this class (see Table 6.4). While the lower recall score of 86% would mean many nonFG instances are misclassified as FG, the confidence with which nonFG instances are predicted means that a large proportion of audio assets can be identified as being suitable for mixing to less prominent positions. This should however be considered in light of the imbalanced class distribution noted.

Additionally, on examining the margin information used by the SVM algorithm to classify instances, it should be noted that the classification can be manipulated by tuning the decision boundary. This would reduce the number of FG instances that are misclassified as nonFG, at the cost of poor precision in terms of the FG prediction (see the confusion matrix provided in Table 6.5). This approach may be suitable for certain applications as it isolates 46.1% of instances as being definitively nonFG, but it should be stressed that the margin applied in this example could not be considered universally applicable. Simply put, these results suggest that some optimisation of model prediction is possible, but forcing universal capture of all FG instances in this example is only achieved at the cost of incorrectly categorising more than half of all nonFG instances as FG. Also, fine-tuning in this manner would require a large data sample in order to provide confidence the tuning would generalise reasonably to unseen data.

The hierarchical labelling task has been framed in a simplified manner in order to subject it to ML analysis. Given the subjective nature of the task, it is not surprising that considerable variance can be observed in the average rating scores, sourced from human participants, for some sounds. While a certain subset of sounds can be identified as FG, subject to the definition of a suitable threshold, this does not confer unanimity between subjects as to the correct category for every sound. This lack of consensus manifests as noise in the labelling schema, which could be considered to have an adverse effect on the accuracy of any model trained using the data and therefore any labels predicted for unseen instances. In light of this, the ACA of 76.4% achieved on Validation Instances (see Table 6.3) should be considered a strong result, accepting that the performance on FG instances is not ideal.

It has been noted that there is an imbalance in the class distribution of the Validation Instances relative to the other two splits, as outlined in Table 6.1. This was unavoidable, as labels were not known for these instances before they were selected at random. The tendency of the EGAL algorithm to select FG instances as being more informative has also been noted,

meaning fewer FG instances remained in the unlabelled pool for selection in the validation split. It is therefore possible the preponderance of nonFG instances in the validation set has resulted in a misleading performance.

This section has outlined an experiment investigating the accuracy of predicted hierarchical labels in an AL task applied to audio stimuli has been outlined. The following section outlines an experiment examining the effect of threshold value on model performance.

6.3 Category Threshold Experiment

As noted in Section 6.2.1 an average rating score value of ≥ 2.5 has been used to classify manually labelled instances as either FG or nonFG to this point. The value was chosen to strike a balance between the large proportion of sounds participants found difficult to rate (having an average rating of approximately 2) and the much smaller proportion of sounds definitively identified as FG (having an average score of 3). The suitability of this threshold to any given application of hierarchy prediction would require validation via perceptual testing, and other thresholds may potentially be of greater suitability for different applications. In light of this, it would therefore be of interest to examine potential thresholds to investigate the performance of models trained using them.

6.3.1 Methodology

To begin all 3,599 manually labelled instances were pooled and the distribution of all average ratings examined (see Table 6.6). The values of 2.2 and 2.75 were selected for use as additional thresholds to 2.5, as these values provide a roughly even decrease in the proportion of FG instances as the threshold value increases. The resulting class distributions and proportion of FG instances per threshold are noted in Table 6.7. Feature representations, models and metrics are as those outlined in Section 6.2.

Table 6.6 A frequency table showing the count per average rating for all 3,599 manually labelled instances.

Average Rating	Count
1.00	748
1.17	1
1.20	23
1.25	40
1.33	508
1.40	14
1.50	38
1.60	10
1.67	505
1.71	1
1.75	26
1.80	16
2.00	455
2.20	9
2.25	38
2.33	402
2.40	5
2.50	28
2.60	9
2.67	292
2.75	28
2.80	9
2.86	1
3.00	393

Table 6.7 Class distribution per threshold value. An instance is classified as FG for a particular threshold if its average rating value is greater than or equal to the threshold value.

Threshold Value	Total	FG	nonFG	% FG
$T \geq 2.2$	3,599	1,214	2,385	33.7%
$T \geq 2.5$	3,599	760	2,839	21.1%
$T \geq 2.75$	3,599	431	3,168	11.9%

To assess performance 3 random, stratified, 20% test splits were implemented for each threshold level to account for random variation in test data selection. Results reported are therefore averages from the 3 models built for each threshold to account for any abnormal variations in the data, which could lead to misleading performance for a particular selection of data. For each test split, a parameter grid search was then conducted on the training portion of the data using a 5—fold CV across the parameter values outlined in Table 6.2. The best performance observed from the CV was assessed using ACA scores, and from this a model was built using all training data and tested on the test split. This process was repeated for each test split and the scores averaged to arrive at a score for each threshold.

6.3.2 Results and Discussion

Figure 6.2 provides the average scores achieved by models for each threshold. From this it can be observed that the variation in ACA scores achieved across thresholds is small (lowest 73%, highest 77%). FG precision gets progressively worse with higher threshold values and conversely, nonFG precision improves as the threshold increases. Note from Table 6.7 that the classes become quite imbalanced as the FG threshold increases with only 11.9% of the dataset classified as FG when $T \geq 2.75$.

Overall, the results suggest that accurately predicting FG instances is a difficult task where many false positive predictions (nonFG sounds classified as FG) will be made. This is reflected most starkly using the $T \geq 2.75$ threshold, where only 27% of FG predictions will be correct. Set against this is the fact that 97% of nonFG predictions will be correct, meaning if our goal is to simply split assets into two groups, there can be confidence that the nonFG predictions are predominantly correct. On the other hand, the FG grouping contains many incorrect predictions, but captures 4 out of every 5 FG instances correctly. The division of assets in this scenario is not precise, but this may be useful for specific applications.

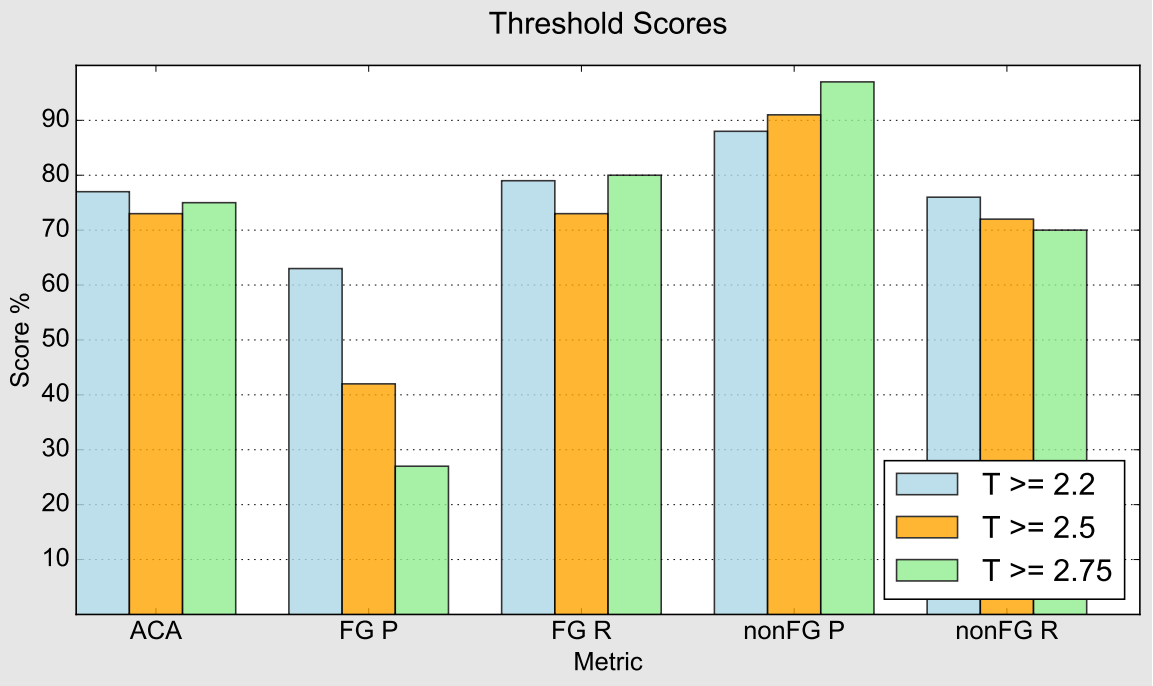

Fig. 6.2 A comparison of scores noted for different thresholds. Note that 'FG P' denotes FG Precision, 'FG R' FG Recall etc.

Using the T $\geq$ 2.5 threshold this division is relatively more accurate, but FG precision is still poor (42%) and FG recall has dropped to 73%, meaning roughly 3 in every 4 FG instances are correctly predicted as such. A similar issue to T $\geq$ 2.75 is observed at T $\geq$ 2.5, where many nonFG instances are incorrectly predicted as FG. However, while more than half of FG predictions at this threshold are wrong, this is actually an improvement from that observed using T $\geq$ 2.75, where almost three quarters of FG predictions are incorrect. It is also worth noting that there are almost twice the number of FG instances using T $\geq$ 2.5 as opposed to T $\geq$ 2.75. This means that, under T $\geq$ 2.5, 73% (555 of 760) FG instances can be correctly identified, whereas at T $\geq$ 2.75, 80% (345 of 431) of FG instances can be correctly identified.

By far, the strongest FG precision score (63%) is observed using the T $\geq$ 2.2 threshold, which contains the largest number of FG instances (1,214, or 33.7% of the total). The FG recall score here is on a par with the best achieved in this exercise and equates to correctly

predicting 79% (959 of 1,214, or 4 out of every 5) FG instances, indicating that almost the same proportion of FG instances can be identified at this threshold as at T $\geq$ 2.75 (80%). Ultimately, the separation of categories is somewhat neater at T $\geq$ 2.2 than at other thresholds. Returning to the example of a variable compression codec, if T $\geq$ 2.2 were used as the threshold to categorise FG instances, this indicates that a lower bitrate could be applied to more than half of auditory assets (2,385, or 66.3% of all those designated as nonFG under this threshold, see Table 6.7). Just as importantly, this could be done while retaining confidence that 79% of FG instances (based on FG recall figure of 79%) could be correctly classified as FG and could therefore be accommodated with a higher quality level.

A discussion around threshold selection would not be complete without accounting for the significance of threshold choice in the audio domain. The T $\geq$ 2.75 threshold focusses on those instances where the greatest consensus exists for nominating a particular instance as FG. For the purposes of this research these should be regarded as the most critical instances to capture as theoretically the consequence for missing one is more likely to be noticed in the context of a variable compression codec. In contrast, the T $\geq$ 2.2 threshold can be considered a much broader categorisation of what constitutes a FG sound, given nearly three times as many sounds are labelled as FG under this threshold as under T $\geq$ 2.75. Given this more inclusive classification includes many instances with a weaker consensus for FG categorisation, it could be considered that miscategorisation of an instance at this threshold would be less likely noticed in the context of a variable compression codec. Use of T $\geq$ 2.5 could be viewed as a compromise choice between the two extremes represented by the other thresholds.

Ultimately, the results do not support a universal optimal threshold for dividing instances into FG and nonFG categories. However, useful classification is achieved to some extent using each of the thresholds investigated. Making a choice of an optimal threshold is therefore dependent on the priorities for the application at hand, there being no clear winner among

those examined in terms of the model accuracies observed in this case. In practical terms, variable compression codecs can still benefit from a non-ideal level of accuracy in either FG or nonFG categorisation. Though FG prediction is more important, higher accuracy in nonFG predictions will also improve codec performance by accurately identifying instances where the heaviest compression can be applied. In this respect, accurate nonFG predictions can thus add value to the overall compression strategy.

The subjective nature of the labelling task has been observed as contributing to noise in the data. The lack of perceptual testing to determine the optimal threshold has also been noted, and in this light it would then seem prudent to adopt a broad threshold to define FG sounds, rather than narrowly constrict the categorisation. For this reason, $T \geq 2.2$ was used in the following Deep Learning experiment.

These results constitute a useful starting point for the construction of a variable compression codec. Given a situation where an ability to manipulate the most important elements of a sound scene is required, it can reasonably be suggested that priority would be given to identifying as many FG instances as possible, with a tolerance for capturing some nonFG instances as part of this process once the majority of FG were identified. The results outlined above indicate capture of a majority of FG cases with a high precision of nonFG predictions, meaning a significant proportion of assets can be isolated as suitable for variable compression treatment. This also suggests that improving the accuracy of these models would be a worthwhile undertaking, given the poor precision of FG predictions observed. To this end, a Deep Learning algorithm is next investigated.

6.4 Deep Learning Experiment

Section 6.2 has described an AL exercise which validates the accuracy of labels predicted by a model trained using minimal manual labelling on instances selected using the EGAL selection method. Section 6.3 has compared different FG threshold values and discussed

classification performance and the implications of threshold choice. The following section outlines an experiment which compares SVM and CNN algorithms trained using different feature representations and data augmentations in order to identify which are most useful for predicting AH.

6.4.1 Methodology

In the following methods are described in brief except where they differ from those used in previous experiments. This section outlines the algorithms chosen for comparison, the data representations and augmentations used to train models and also includes a description of how labels were predicted for all remaining instances in the unlabelled set, U.

Dataset and Feature Representations

Table 6.7 provides a breakdown by threshold for all 3,599 manually labelled instances. The T $\geq$ 2.2 threshold was used to categorise these instances, fit parameters and train an SVM model to predict labels for the remaining 6,401 instances in U giving a total labelled corpus of 10,000 sounds. The class distribution of these 10,000 instances is 39% FG, 61% nonFG meaning the FG proportion is slightly larger than that observed on manually labelled instances only using the T $\geq$ 2.2 threshold (33.7%). This is not surprising given the FG precision score (63%) noted for this threshold in Figure 6.2 which suggested that 37% of instances predicted as FG would be incorrect.

Data augmentations, introduced in Section 3.7, are applied to the 10,000 instances which have a mixture of manually applied and predicted labels to expand the dataset to a total of 100,000 instances. A total of 6 pitch augmentations are extracted using the Python LibROSA [335] sound library employing the following pitch shift values in semitones: -2.5, -2, -1, 1, 2, 2.5. Parameters for 3 DRC augmentations are outlined in Table 6.8 and have

been drawn from standard presets provided with Adobe Audition [347], editing software commonly used in media production environments.

Table 6.8 Parameter settings for the DRC augmentations. In the following, 'ms' are milliseconds, 'dB' are Decibels, 'CSK' refers to 'Classic Soft Knee', 'SL12' to 'Soft Limit -12dB' and 'SL24' to 'Soft Limit -24db'.

Parameter	CSK	SL12	SL24
Look-Ahead Time (ms)	3	3	3
Input Gain (dB)	0	0	0
Attack Time (ms)	1	1	1
Release Time (ms)	300	300	300
Output Gain (dB)	0	8	16
Attack Time (ms)	10	5	5
Release Time (ms)	250	15	15

The feature representation used in this instance was based on that outlined in Section 6.2.1 with the addition of delta and double-delta data for the purposes of comparison. A summary of feature representations is provided in Figure 6.9. Here, '10k' indicates the pool consisted of 10,000 instances (manual and predicted labels only), '100k', means the pool consisted of 100,000 instances, using manual and predicted labels in addition to data augmentations, 'Zero Order' indicates the model was trained using a Zero Order data representation only and 'Delta' indicates that delta and double delta representations were used to train the model in addition to the zero order representation. 'SVM 10k Zero Order' therefore indicates an SVM algorithm trained using 10,000 instances with a zero order representation only. Similarly, 'CNN 100k Delta' indicates a CNN algorithm trained from a pool of 100,000 instances with a zero order and a delta representation. Delta data is employed to capture temporal change effectively and has been successfully used in similar research [106].

Impact of SVM Predicted Labels

It was hypothesised that use of an SVM to predict labels for a dataset then used to compare performance of SVMs and other algorithms would infer an advantage to the SVM. A toy

Table 6.9 A summary of feature representation pools used when building SVM and CNN models.

Model	Dimensions
SVM 10k Zero Order	10,000 x 6,280
SVM 10k Delta	10,000 x 18,840
CNN 10k Zero Order	10,000 x 40 x 157 x 1
CNN 10k Delta	10,000 x 40 x 157 x 3
SVM 100k Zero Order	100,000 x 6,280
SVM 100k Delta	100,000 x 18,840
CNN 100k Zero Order	100,000 x 40 x 157 x 1
CNN 100k Delta	100,000 x 40 x 157 x 3

simulation was used to investigate and attempt to quantify the impact. To begin, 500 instances were selected in a random but stratified manner from the 3,599 manually labelled instances to form the toy dataset using the 'SVM 10k Zero Order' representation and T $\geq$ 2.2 threshold labels. To simulate the same proportion of labelled to unlabelled instances in the entire dataset 180 of the toy instances were used to train a model and predict labels for the remaining 320 instances of the toy dataset. All models built for this exercise employed a 5—fold CV. Finally, models were built on two versions of the toy dataset, one consisting of 500 instances with manually applied labels and the other of 180 instances with manual labels and 320 instances with predicted labels. A graphical representation of this process is offered in Figure 6.3.

It was observed that the models built with a mix of manual and predicted labels outperformed the models built solely with manual labels by an average of 15.01% ACA per fold. The scores were compared using a Kruskal Wallace H-test and were found to be significant at the 5% level. This indicates that the SVM would enjoy an advantage in any algorithm comparison experiment, and thus any direct comparison would be confounded. Speculatively, it is possible that the small size of the dataset used in the assessment compounds the difference in scores noted and that this disparity would not be as stark with larger datasets. It was decided to proceed with building SVM and CNN algorithms while noting this exercise, as

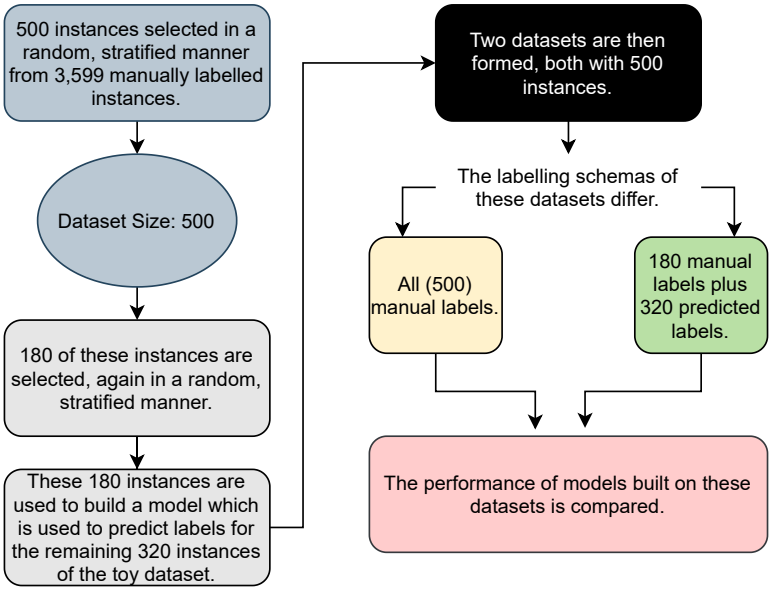

Fig. 6.3 The process and dataset details for comparing the performance of SVMs trained using all manual and a mixture of manual and predicted labels.

useful conclusions may still be drawn, particularly if CNN performance is noted as being similar to, or surpassing, that of an SVM.

Building Models

When building SVMs and CNNs, models were evaluated using ACA, precision and recall scores as outlined in Section 6.2.1. Three randomly selected, stratified, hold-out training/test sets of size 80/20 were implemented for representations having 10,000 instances: those without augmentations. Care was taken with representations having 100,000 instances to ensure that augmentations of instances in the test set were removed from the training set, as otherwise this would constitute data leakage. As the nature of the dataset would not allow a simple 80/20 split because of this consideration, three randomly selected test sets were formed from non-augmented instances, each of these consisting of approximately 3,333 instances. The training set was then formed for each test split by removing augmentations of the test set instances from the training set. This resulted in train set sizes of approximately

Table 6.10 Train and test split sizes used to train SVM and CNN models.

Split Number	No. of Instances	Train Set Size	Test Set Size
All Splits	10,000	7,999	2,001
1	100,000	60,003	3,333
2	100,000	60,003	3,333
3	100,000	59,994	3,334

Table 6.11 An outline of the smaller parameter grid used for large data representations.

Parameter	Values
kernel	radial basis function
C	0.01, 0.1, 1
gamma	'scale', 0.1, 1

60,003 instances. The metric scores reported for each model are therefore averages across three test splits. Train and test split sizes are presented in Table 6.10.

A 3—fold CV was performed on the training data for SVM models to fit parameters using a validation split. Those built on 10,000 instances used the same parameter grid as that presented in Table 6.2. Those built on 100,000 instances utilised a smaller parameter grid (presented in Table 6.11) due to the memory and time required to fit parameters on the larger representation. In selecting a reduced parameter grid, it was decided to focus on the RBF kernel only, due to the performance of this kernel in previous exercises. Furthermore, the 'SVM 100k Delta' model required a slightly different approach, as the memory requirement was still too large to fit parameters on all training instances at once given the size of this feature representation. As a result, the training set was split into 3 to fit parameters for this model.

CNN models were built by experimenting with model architecture and parameters using the training and validation data. Number and level of dropout, size of filter, numbers of filters, stride and padding parameters, activation function type, optimisers and numbers of layers were varied to examine their effect on model performance. The Adam optimiser was ultimately used on all CNN models. The loss metric used was binary cross entropy.

Examples of finalised CNN architectures used are provided in Table 6.12, for the 'CNN 10k Zero Order' model and Table 6.13 for the 'CNN 100k Delta' model.

Where applicable, the McNemar test was used to compare model predictions for statistical significance as it is recommended by Dietterich [348] in cases where it is prohibitive to run many iterations of a model using different randomly selected test splits, as was found here in the case of the 'SVM 100k Zero Order' model. While 3 random splits were implemented, this does not provide enough samples to make meaningful use of the Wilcoxon and Friedman statistical tests used in previous experiments. To make a comparison of models using McNemar's test, an 2 x 2 contingency table must be compiled from the predictions made by two models, as outlined in Table 6.14. The test can then be conducted by comparing the classification errors made by each model using Equation 6.1. A statistically significant result indicates that the two models have different performance, as the errors they make are in different proportions Dietterich [348]. In this case, the test was applied for each random hold-out test split, giving three iterations for each comparison. Therefore, ACA, precision and recall scores averaged across hold-out test splits are used to evaluate model performance and the McNemar test is used to determine if there is a statistically significant difference between the errors made in each test split.

$$\frac{(|AnotB - BnotA| - 1)^2}{AnotB + BnotA} \tag{6.1}$$

Algorithm Choice

For this exercise, SVM and CNN models were evaluated. SVMs, as outlined in Section 6.2, were used to provide consistency with prior research outlined in Chapters 4 and 5 where they were selected due to extensive use in the audio domain [47, 224, 255, 277, 339]. As noted in Section 6.4.1 however, they enjoy an advantage in this context because the model used to predict labels for a portion of the dataset was an SVM.

Table 6.12 An example of CNN architecture applied in this research, in this instance the final configuration for the 'CNN 10k Zero Order' model. An initial architecture based on that described by Chen *et al.* [7] was implemented and adapted for each CNN outlined in Table 6.9. The notation '5 x 5 Conv2D(pad=2, stride=2) x 12 - BN - ReLU - DO(0.3)' denotes a 2D convolutional layer with 12 filters of size 5 x 5 followed by batch normalisation, ReLU activation function and Dropout where p=0.3.

Layer Name	**Settings**
Input	10,000 x 40 x 157 x 1
Convolution 1	5 x 5 Conv2D(pad=2, stride=2) x 12 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 24 - BN - ReLU 2 x 2 MaxPooling
Convolution 2	3 x 3 Conv2D(pad=1, stride=1) x 48 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 48 - BN - ReLU 3 x 3 Conv2D(pad=1, stride=1) x 48 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 48 - BN - ReLU 2 x 2 MaxPooling
Convolution 3	3 x 3 Conv2D(pad=1, stride=1) x 56 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 56 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 56 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 56 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 96 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 96 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 96 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 96 - BN - ReLU - DO(0.3) 2 x 2 MaxPooling
Convolution 4	3 x 3 Conv2D(pad=1, stride=1) x 128 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 128 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 128 - BN - ReLU - DO(0.3) 3 x 3 Conv2D(pad=1, stride=1) x 128 - BN - ReLU - DO(0.3) 2 x 2 MaxPooling
Pooling	Flatten() Dense() x 128 - BN - ReLU
Output	Dense() x 2 - Sigmoid

Table 6.13 An example of CNN architecture applied in this research, in this instance the final configuration for the 'CNN 100k Delta' model. The notation '5 x 5 Conv2D(pad=2, stride=2) x 12 - BN - ReLU - DO(0.2)' denotes a 2D convolutional layer with 12 filters of size 5 x 5 followed by batch normalisation, ReLU activation function and Dropout where p=0.2.

Layer Name	Settings
Input	100,000 x 40 x 157 x 3
Convolution 1	5 x 5 Conv2D(pad=2, stride=2) x 12 - BN - ReLU - DO(0.2) 3 x 3 Conv2D(pad=1, stride=1) x 24 - BN - ReLU 2 x 2 MaxPooling
Convolution 2	3 x 3 Conv2D(pad=1, stride=1) x 48 - BN - ReLU 3 x 3 Conv2D(pad=1, stride=1) x 48 - BN - ReLU 2 x 2 MaxPooling
Convolution 3	3 x 3 Conv2D(pad=1, stride=1) x 56 - BN - ReLU 3 x 3 Conv2D(pad=1, stride=1) x 56 - BN - ReLU 3 x 3 Conv2D(pad=1, stride=1) x 56 - BN - ReLU 3 x 3 Conv2D(pad=1, stride=1) x 56 - BN - ReLU 3 x 3 Conv2D(pad=1, stride=1) x 96 - BN - ReLU 3 x 3 Conv2D(pad=1, stride=1) x 96 - BN - ReLU 3 x 3 Conv2D(pad=1, stride=1) x 96 - BN - ReLU 3 x 3 Conv2D(pad=1, stride=1) x 96 - BN - ReLU 2 x 2 MaxPooling
Convolution 4	3 x 3 Conv2D(pad=1, stride=1) x 128 - BN - ReLU 3 x 3 Conv2D(pad=1, stride=1) x 128 - BN - ReLU 2 x 2 MaxPooling
Pooling	Flatten() Dense() x 128 - BN - ReLU - DO(0.5)
Output	Dense() x 2 - Sigmoid

Table 6.14 Composition of a contingency table based on the results of two models, A and B.

Number of instances misclassified by both A and B (AandB)	Number of instances misclassified by A but not by B (AnotB)
Number of instances misclassified by B but not by A (BnotA)	Number of instances misclassified by neither A nor B (AnorB)

Table 6.15 Summary of average ACA, precision and recall scores noted across three randomly selected hold-out test sets. Note that 'P' indicates Precision and 'R' indicates Recall in the following.

MODEL	ACA	FG P	FG R	nonFG P	nonFG R
SVM Manual Labels Only (3,599)	77.5%	63.3%	78.7%	88.0%	76.3%
SVM 10k Zero Order	81.1%	76.0%	78.0%	86.0%	84.0%
SVM 10k Delta	77.6%	74.7%	70.7%	81.7%	84.3%
SVM 100k Zero Order	78.2%	74.3%	72.3%	83.0%	84.0%
SVM 100k Delta	78.6%	75.0%	73.0%	83.0%	84.7%
CNN 10k Zero Order	82.2%	78.0%	79.0%	86.3%	85.7%
CNN 10k Delta	81.4%	78.3%	76.0%	85.0%	86.7%
CNN 100k Zero Order	80.9%	76.0%	77.3%	85.0%	84.3%
CNN 100k Delta	80.6%	78.7%	74.0%	84.0%	87.0%

CNNs were chosen as they are extensively used in audio DL tasks including environmental sound classification where Sailor *et al.* [104] have derived the most successful model for sound categorisation at time of writing based on the ESC-50 dataset [101] with a classification accuracy of 86.5%. Indeed, CNNs feature strongly throughout this leaderboard [105, 295]. CNNs are also well represented in successful solutions to the DCASE 2017 [15, 349] environmental sound classification challenges [7, 21, 300].

6.4.2 Results and Discussion

The average scores achieved by each classifier are summarised in Table 6.15, which corresponds in terms of feature representations used to Table 6.9. The only exception is the first model, an SVM trained using manually sourced labels only, which was used to predict labels as described in Section 6.4.1. This is provided as a baseline to give context for the scores achieved by the other models, which are all trained using a mixture of manual and predicted labels.

Performance of the SVM models trained using a mixture of manual and predicted labels is marginally ahead of that noted for the SVM trained on manual labels only. The performance improvement is much smaller than that noted in Section 6.4.1, where it was observed that the

models built with a mix of manual and predicted labels outperformed models built solely with manual labels by an average of 15.01% ACA per fold. The difference in scores is much smaller here, being of the order of 0.1 - 4.7%. Indeed, when examining nonFG precision (88.0% versus 86.3%) and FG recall (78.7% versus 79.0%) the model trained entirely on manual labels either surpasses or is only very marginally behind the best score noted on models trained using manual and predicted labels. This aligns with the speculation that the advantage enjoyed by the SVM in this respect may decrease as dataset size grows.

Little benefit appears to be derived from the use of data augmentations for SVMs. The best ACA score noted on SVMs is trained without augmentations, 81.1%. This contrasts with 78.6%, the best score noted using augmentations. This is not particularly surprising, given there is no context in the literature for using data augmentations to train SVMs. In terms of feature representations, it again appears that there is little benefit in this case to using delta data, with the best ACA score noted for SVMs being achieved using a zero order representation. Note that the SVM trained on 10,000 instances of zero order data outperforms the equivalent model trained using a delta representation in all metrics except nonFG recall, and there only marginally, by 0.3%. The opposite is the case with SVMs trained using augmentations, with the delta variant performing either equally or marginally better than the zero order model. The scale difference is larger in the models trained without augmentations. For example, FG recall on the zero order model outperforms the delta model by 78.0% to 70.7%. The largest difference between metrics noted in models trained with augmentations is 0.7%. McNemar tests carried out on the three splits comparing SVMs trained on zero order versus delta representations were statistically significant to the 5% level for all three splits for models trained on 10,000 instances. However, for SVMs trained using a pool of 100,000 instances only one split found a statistically significant split on the proportion of errors made by each model.

In all, this indicates that practitioners should strongly question the utility of using delta data representations and data augmentations to build SVM models to predict AH. The larger representations are considerably more cumbersome to train in terms of the time and compute required, and thus could not be advised based on the results noted in this case. Furthermore, the lack of a statistically significant result in two of the splits for the larger feature representations indicates that there is little difference in the information derived between feature representations at this scale.

Considering CNN models, we note the highest scores in all metrics are achieved using this model type. The highest ACA score noted is 82.2%, achieved on 10,000 instances using a zero order representation. This model also achieves the best FG recall score noted (79.0%) and the best nonFG precision score (86.3%). The best scores achieved in FG precision (78.7%) and nonFG recall (87.0%) are on a CNN using data augmentations with a delta representation. The fact that these are the best scores noted strongly suggests that CNNs are a better choice than SVM as they are achieved in a configuration where the SVM is acknowledged to have an advantage. Also, interestingly, when comparing the CNN scores with the SVM trained solely on manual labels with 3,599 instances we find that the CNN model trained on a zero order representation with 10,000 instances outperforms the SVM across a number of metrics. These include ACA (82.2% versus 77.5%), FG precision (78.0% versus 63.3%), FG recall (79.0% versus 78.7%) and nonFG recall (85.7% versus 76.3%) with the sole exception being nonFG precision, where the SVM outperforms the CNN (88.0% versus 86.3%). This is interesting because the CNN is trained on a small dataset relative to the size of most deep learning datasets, which suggests that greater performance would be noted once a large dataset is available.

Set against this point is the performance of CNNs trained using data augmentations. Surprisingly, they are generally less accurate than the models trained using 10,000 instances, although the best scores on some metrics of all models trained are noted on CNNs trained

using augmentations: FG precision of 78.7% and nonFG recall of 87.0%. The reason for this can be speculated upon. Only two forms of augmentations are applied in this case: pitch shifting and dynamic range compression. Perhaps a greater variety of augmentations would be more useful. Furthermore, 9 augmentation types were implemented for this experiment. Due to the necessity of avoiding data leakage the training sets for the larger models trained would still be considered small in the context of deep learning, as they consisted of approximately 60,000 instances. By expanding the number of augmentations applied, and also configuring the augmentation process such that different levels of augmentations can be applied (applying pitch shift on top of dynamic range compression, for instance) the size of the training set can be enlarged, and perhaps better performance may be observed.

The difference in performance on CNNs when comparing zero order to delta representations is marginal and on the whole is not statistically significant using the McNemar test. Only one split of the models trained using augmentations is statistically significant, both of the other splits for this model, and all of those for the model trained without augmentations, fail to reject the null hypothesis. Therefore, there is no compelling case here for the use of larger delta representation on CNNs.

On the whole, these are interesting results, particularly in light of the advantage enjoyed by the SVM due to the method used to predict labels for a portion of the dataset used. It suggests that CNNs are a better choice than the SVM, even when trained on a relatively small number of instances. Certainly SVMs and CNNs are in general making errors in different proportions, as the McNemar tests comparing models trained on the same data and representations are all statistically significant apart from one split each on the 10,000 instance zero order and 100,000 instance delta representations. It is noted, however, that the best performing CNN trained using manual and predicted labels is only marginally better than an SVM trained using instances which have all been manually labelled. While this cannot be considered definitive, it still suggests that use of SVMs should not be discarded out of hand,

as they are capable of strong performance on small datasets. Equally, these results outline that CNNs are a good option, and it would be reasonable to hypothesise that, should a large dataset be available, the performance noted here would be surpassed.

6.5 Conclusions

This chapter contains a comparison of algorithms and feature representations that include manual and predicted labels in addition to data augmentations. This is relevant to OBJ 3:

OBJ 3: To assess the performance of supervised ML algorithms when predicting AH.

The models built in Experiment 4 are a step in the development of the proof-of-concept model to predict AH outlined in Chapter 4. They suggest that CNN models can be trained to high accuracy levels to predict AH on a dataset consisting of instances which have a mixture of manually applied and predicted labels. However, the performance noted is only slightly better than that of an SVM trained using manual labels alone. These findings directly address RQ 3.

RQ 3: Is it possible to accurately predict AH using supervised ML methods?

The work outlined in this chapter has underlined previous work introduced in Chapters 4 and 5 which presented models capable of high accuracy levels ($> 80\%$) across a number of metrics. In this instance, models trained using manual and predicted labels achieve an ACA of 82.2% ('CNN 10k Zero Order'). In addition, an SVM trained using manual labels only achieved an ACA of 77.5%. These accuracy levels are comparable with other audio ML problems, and collectively suggest that it is possible to predict hierarchy between audio objects using ML methods.

Section 6.2 has presented an AL experiment which validates the accuracy of labels predicted by an SVM model trained using instances selected with the EGAL selection

method. As the accuracy achieved on the validation set (ACA 76.4%) is comparable to that noted in Experiment 3 (ACA 73.9%) this suggests that EGAL is an effective method for selecting instances in a hierarchical audio ML context and is therefore suitable for application to the task of labelling large numbers of audio instances with hierarchical labels.

The work described in this chapter has utilised LPMS feature representations, having found them useful in Experiment 3. Experiment 4 has compared LPMS representations which utilise zero order data alone, and those implementing zero order data in addition to delta and double delta representations. In this case, it has been found that delta data has not aided the accuracy of SVMs. The utility of delta data for CNN models is debatable.

Similarly, little difference is observed in the scores of SVM models trained with and without data augmentations, making their use questionable for this algorithm. As can be observed from Table 6.15 there is a similar finding with regard to CNNs, with marginal differences only observed between models trained with and without augmentations.

As outlined in Section 6.4.2 the 'CNN 10k Zero Order' model has been found to achieve the highest ACA score of the models trained for this experiment. However, the performance is only superior to an SVM trained on a smaller dataset by 4.7% ACA — not so large a result that would rule out usage of SVMs in future work. The performance of CNNs trained using data augmentations is quite surprising given their success in other works, and should be the focus of further investigation given the successful application of the technique on other audio ML problems. Nevertheless, strong results appear to be possible should the requirement to build a suitable dataset for deep learning be deemed excessive. This finding is very useful as it can be used as guidance in the formation of any real-world implementations which take account of the FIAH outlined in Chapter 2.

This chapter has described three exercises which investigate the application of ML techniques to the problem of predicting AH. The following chapter will offer a summary of

the central work of this thesis and a recap of research objectives and questions, which will lead to a final statement of thesis contributions.

Chapter 7

Machine Learning Methods Applied to Auditory Hierarchy

This work has reviewed research relevant to Auditory Scene Analysis (ASA) and audio Machine Learning (ML). It has also described an investigation of sound perception and the performance of models trained to predict Auditory Hierarchy (AH). This chapter will review the work completed with reference to the research objectives, research questions and thesis contributions introduced in Chapter 1.

The research objectives (OBJ) around which this work has been structured were as follows:

OBJ 1: To develop an understanding of ASA with particular attention to the concepts of object-based audio, AH and modern media consumption paradigms.

OBJ 2: Informed by perceptual audio research, to propose a machine learning approach for the task of predicting AH.

OBJ 3: To assess the performance of supervised ML algorithms when predicting AH.

These OBJs were formative in the derivation of a number of research questions (RQ) which have structured and defined the scope of the work completed in this thesis:

RQ 1: What factors are involved in the perception of AH?

RQ 2: Does a hierarchy of importance exist between sounds isolated from context?

RQ 3: Is it possible to accurately predict AH using supervised ML methods?

Section 2.2 has identified auditory perception as a multi-faceted task which is influenced by a series of factors. This in turn has informed the proposal of a 'map' of AH based on a perceptual theory of hierarchical classification as influenced by a number of factors, designated Factors Influencing Auditory Hierarchy (FIAH) for the purposes of this work. These FIAH, introduced in Section 2.6, include physical properties of sounds, such as pitch and timbre, and biases that vary by individual, such as experience, training and expectation. In turn, characteristics of sounds hypothesised to indicate AH, such as the presence or absence of humans and abstract versus semantically loaded sounds, designated Potential Hierarchical Indicators (PHI) for this research, were outlined.

This conception of AH as a non-trivial system subject to multiple influences, both endogenous and exogenous to the person, emphasises the desirability of studying individual factors in isolation. By approaching the problem this way, a complete picture can be built of the degree to which each influences categorisation. Use of a dataset which removes the presence of these factors to the greatest extent possible is therefore mandated. The desire to focus on the modern media production paradigm has motivated the use of a broad palette of environmental sounds as stimuli because they are considered to be ecologically valid in the context of drama, current affairs, reality television and game audio content.

Chapter 3 has offered an overview of supervised ML research with particular attention to the audio domain. The methodology for building ML models was outlined in Section 3.3 and additionally the significance of feature extraction and selection methods were discussed. Evaluation measures used to measure the performance of models were also summarised. A number of different audio feature types were then reviewed in Section 3.4, and an overview

of common supervised ML algorithms employed on audio prediction tasks was then offered in Section 3.5. Active Learning (AL) theory was introduced in Section 3.6 and data augmentation methods were reviewed in Section 3.7. This overview concludes that supervised ML is a complex task which features many variables requiring adjustment in order to optimise the accuracy of final models, concurring with the *No Free Lunch Theorem* [191], which outlines the lack of unified approach that consistently outperforms all others. This mandates considerable experimentation for which this thesis offers efforts in the area of AH, which it is hoped will constitute a useful roadmap for both machine learning and perceptual audio researchers interested in the area.

The review of literature presented in Chapters 2 and 3 has been followed by an investigation of the RQs detailed in Chapters 4, 5 and 6. This has in turn lead to the following contributions, organised into major and minor:

Major Contributions

Maj. Contrib. 1: ***A roadmap for research into ML methods for AH.*** AH has received relatively little attention in terms of ML research. This work explores perceptual audio theory and applies a number of common ML methods to the domain and the findings are offered in the shape of a roadmap which can inform future research in the area.

Maj. Contrib. 2: ***A published working theory of AH.*** AH is theorised to vary due to the influence of factors such as the physical properties of sounds and individual biases. Sounds are proposed to be characterised hierarchically in terms of a number of indicators such as whether they indicate the presence of humans or not, whether the sound contains semantic information or not, and others.

Maj. Contrib. 3: ***Evidence of a hierarchy of importance between sounds isolated from context is presented.*** The understanding of AH is enhanced by conducting a perceptual experiment where the hierarchical relationship between sounds isolated from context is investigated.

Maj. Contrib. 4: ***Validation of the use of ML methods to predict AH with competitive performance. Average Class Accuracy of 82.2% is noted using a Convolutional Neural Network (CNN).*** A series of experiments are described which address the problem of hierarchical prediction in an audio context. Performance comparable with other audio ML applications is noted using Random Forest (RF), Support Vector Machine (SVM) and CNN algorithms.

Maj. Contrib. 5: ***Applied to AH, the Exploration Guided Active Learning (EGAL) algorithm can be used to select a minimal number of labels (in this case 1.7% of the total) to achieve 95.5% of possible model accuracy, outperforming other selection methods.*** In an assessment of Active Learning (AL) selection methods, EGAL is found to be most effective in selecting informative instances to reduce manual labelling effort, outperforming Uncertainty Sampling Active Learning (USAL). Use of EGAL is more computationally efficient and less time consuming than USAL as it does not require a model to be trained at each iteration of the algorithm.

Minor Contributions

Min. Contrib. 1: ***In the context of AH, the Log Power Mel Spectrogram (LPMS) zero order feature representation is found to be an effective compromise for predicting AH, providing comparable performance to larger representations which are considerably more expensive in terms of computation***

time. Delta representations are found to provide performance improvement in some, but not all cases. A number of feature representations have been utilised in the course of this research. While it is noted that in certain cases superior performance is possible from larger data representations it is debatable as to whether the increase in performance is justified by the computation cost entailed.

Min. Contrib. 2: ***The development of a hierarchically labelled corpus of 10,000 sounds consisting of both manual and predicted labels.*** Future investigations of AH are facilitated via the corpus developed during the experiments conducted for this thesis. To our knowledge, this corpus represents the largest audio database of hierarchically labelled audio instances.

These contributions will be discussed in the next section, summarised by the research questions identified.

7.1 Summary of Research Questions

The following sections discuss each RQ outlined previously in turn, noting relevant findings and limitations where necessary. Additionally, these sections highlight the thesis contributions relevant to each RQ.

7.1.1 RQ1: What factors are involved in the perception of Auditory Hierarchy?

Material covered in Sections 2.2, 2.3 and 2.6 has outlined a working theory of how sounds are sorted hierarchically on a continuous basis and identified a series of FIAH which are hypothesised as having relevance to hierarchical sound categorisation. These include the

physical properties of sounds, such as pitch, timbre, loudness, sound transients and onsets. A number of individual biases are also hypothesised to have an influence. These are noted as anticipation and expectation, sound proximity, attention, context, prior experience and training and also senses other than hearing such as olfaction, touch and sight. This literature review has also highlighted a series of PHIs hypothesised to be useful in identifying hierarchical position. These are the presence or absence of people, sounds which are abstract in nature versus those that are rich in semantic content, speech and non-speech sounds, pleasant versus unpleasant sounds and discrete or continuous sounds.

This review motivated a desire to enable a study of these factors in isolation to establish a firm foundation for real-world implementations. Existing sound taxonomies, stimuli selection methods and sound datasets were reviewed in Section 2.4 and this revealed that, while a number of datasets feature hierarchical information, the labels are either not empirically derived, the sounds are selected to accommodate the presence of a situational context (the stimuli are sound 'scenes' rather than isolated sounds) or the stimuli are limited in scope in that they feature urban sounds only, and thus do not provide a broad palette of different sounds. It was therefore determined that in order to study FIAH identified as being relevant to hierarchical classification a dataset would be required which permitted such study, featuring sounds which are isolated from context to the greatest extent possible.

Of interest in this respect is the investigation of sound importance hierarchy between stimuli isolated from context to the extent that this is possible in an experimental scenario (RQ 2) and additionally, whether this hierarchy can be accurately predicted using ML methods (RQ 3). Once these questions are answered, investigation of FIAH can logically proceed, or not, armed with greater knowledge regarding the nature of AH and the likely ability of ML algorithms to predict the phenomenon.

Addressing this question was formative in three contributions of this thesis. It provided the necessary theoretical grounding for a roadmap of research into AH, motivated a theory

of AH, and established the requirement for a dataset of sounds isolated from context. The relevant contributions are as follows, firstly major:

Maj. Contrib. 1: A roadmap for research into ML methods for AH.

Maj. Contrib. 2: A published working theory of AH.

Secondly, minor:

Min. Contrib. 2: The development of a hierarchically labelled corpus of 10,000 sounds consisting of both manual and predicted labels.

7.1.2 RQ2: Does a hierarchy of importance exist between sounds isolated from context?

This RQ was formulated in Chapter 2 when the requirement for a dataset of sounds isolated from context in order to study individual FIAH was identified. A number of the stimuli datasets reviewed in Section 2.4 contain hierarchical information which was deemed unsuitable for study in this case for reasons outlined in the previous section. While these studies established the existence of a hierarchical organisation between the sounds utilised, an extensive effort was deployed to generate a corpus of sounds which were isolated from context in order to facilitate the study of FIAH. For this reason, it was decided to investigate the existence hierarchy between the sounds selected to study the interrelationships of FIAH.

This was addressed in Experiment 1, described in Section 4.2, where 40 sounds were labelled by 112 participants. While there were no unanimous categorisations, a clear continuum was observed in this experiment, ranking sounds from BG to FG. The 'Clock Alarm' sound was considered the most FG sound in this experiment, receiving 104 FG, 7 N and 1 BG ratings. The 'Birds' sound received an emphatic BG rating, collecting 5 FG, 12 N and 95 BG ratings. The other sounds presented in this experiment ranked between these two

examples, with many sounds characterised by a lack of consensus between subjects as to the correct hierarchical category. This suggests the position of a classification boundary should be open to debate. These issues are possibly due to a number of factors such as the subjective nature of the task, the difficulty inherent in attempting to isolate sounds fully from context and the artificial nature of the experimental paradigm. Nevertheless, the results observed here suggest the existence of a hierarchical spectrum between sounds, which is of interest for object-based audio applications.

Answering this RQ forms another of the contributions to this thesis as it is a useful case study for future researchers in the domain and as it offers evidence of hierarchical importance between sounds which have been isolated from context to the extent this is possible in an experimental paradigm. It therefore forms part of the following contributions:

Maj. Contrib. 1: A roadmap for research into ML methods for AH.

Maj. Contrib. 3: Evidence of a hierarchy of importance between sounds isolated from context is presented.

7.1.3 RQ3: Is it possible to accurately predict AH using supervised ML methods?

This section is divided into a number of sub-sections, as the investigation of ML algorithms has also involved a number of exercises which compare AL methods, feature representations and data augmentations. Algorithm comparisons are first reviewed.

ML Algorithms

The work presented in Chapters 4, 5 and 6 has investigated the accuracy of RF, SVM and CNN algorithms to assess which is most appropriate for the task of predicting AH. Chapter 4 has presented work comparing RF and SVM models on a task to predict AH using statistical

summaries of audio low level descriptors as a feature representation. The SVM in this case was observed to perform more strongly than the RF, both in terms of FG class accuracy (93.3% versus 73.3%) and average class accuracy (ACA) (88.1% versus 80.3%). This indicates that, while high accuracy is possible with both algorithms, the SVM is particularly good at capturing almost all FG instances, albeit on a small dataset in this case. The performance of SVM versus RF is not surprising, given that it is in line with other ML studies in the audio domain noted in Section 3.5. This work motivated the desire to form a larger dataset of sounds to render a more complete comparison of ML algorithms on the problem. This in turn motivated the investigation of AL techniques in Experiment 3 to minimise the manual effort required to build large corpora of hierarchically labelled sounds.

Chapter 5 has presented work which compared a number of kernels for the SVM algorithm to investigate which would be best for use in an Active Learning (AL) exercise to label audio data hierarchically. Experiments conducted for linear, Radial Basis Function (RBF) and polynomial kernels across four feature representations revealed that the linear kernel was outperformed by the RBF and polynomial kernels. Little difference in performance was observed between these last two kernels. For example, the RBF kernel outperformed the polynomial in terms of ACA score on the Log Power Mel Spectrogram (LPMS) representation (73.9% versus 73.4%) but in turn was slightly less accurate (ACA of 72.2% versus 72.4%) on the Mel Frequency Cepstral Coefficient (MFCC) representation. Comparing the other metrics, there is generally very little to choose between RBF and polynomial kernels in this case, indicating that either would be a reasonable choice for applications to predict AH.

Chapter 6 outlines the performance of SVM and CNN algorithms while predicting AH. Each algorithm was trained on a number of feature representations, some of which include augmented data. Superior scores are achieved using CNN models in spite of the advantage SVMs are acknowledged to have because of the method used to predict labels. The best ACA score noted is 82.2% from a CNN trained with a zero order representation, compared

to the best noted ACA from an SVM of 81.1%. It is interesting to note however that the performance of an SVM model trained on a smaller dataset with manual labels is competitive with this result, and indeed surpasses CNN performance on certain metrics. The conclusion therefore is that neither algorithm can be ruled out of consideration for AH tasks.

The work outlined in this section is interesting, as it indicates that competitive performance is possible using both SVM and CNN models. Given the performance improvement noted on other audio ML tasks, it is surprising that the use of data augmentations for CNNs has been ineffective in this case, and this deserves further investigation. These are useful insights for the study of AH and are thus contribute to the roadmap of research offered in this work. Collectively, these investigations indicate that AH can successfully be predicted using ML methods. They thus form part of the following thesis contributions:

Maj. Contrib. 1: A roadmap for research into ML methods for AH.

Maj. Contrib. 4: Validation of the use of ML methods to predict AH with competitive performance (Average Class Accuracy of 82.2% is noted using a CNN).

Active Learning

Chapter 5 outlined a simulated AL exercise using a labelled corpus of 3,002 instances. Two AL selection methods, Uncertainty Sampling Active Learning (USAL) and EGAL were compared, with random selection also implemented as a baseline. Noting the ACA of an SVM model built on 3,002 instances of 76.9%, the EGAL selection method was found to have a statistically significant performance benefit over the other methods, achieving 95.5% of possible model accuracy from 1.7% of all labels. This confirms that selecting the instances to train a model intelligently is an effective way to minimise the manual effort required to train a model to near maximal performance. It furthermore suggests the EGAL selection

method works well on an audio domain problem, outperforming USAL and random selection methods in this instance.

To our knowledge, this work constitutes the first application of AL to AH and of the EGAL selection method to an audio problem. The results are particularly interesting when contrasted with the USAL method, given its popularity in many other domains. EGAL offers some advantages over USAL in that it selects instances for labelling based on their proximity both to each other and to labelled instances in the feature space. EGAL does not require a model to be trained at each iteration of the selection algorithm, as USAL does. It is therefore more computationally efficient in addition to not being subject to the biases inherent in any model used to predict as required when using USAL. The work presented here suggests that AL is an effective method for reducing the manual effort required to label audio instances with hierarchical labels.

These findings were formative in assessing the viability and implementation of AL to audio object hierarchy, establishing a research context for using these methods to form a large dataset of hierarchically labelled audio instances. This work has therefore added to the following contributions of this thesis:

Maj. Contrib. 1: A roadmap for research into ML methods for AH.

Maj. Contrib. 5: Applied to AH, the EGAL algorithm can be used to select a minimal number of labels (in this case 1.7% of the total) to achieve 95.5% of possible model accuracy, outperforming other selection methods.

Min. Contrib. 2: The development of a hierarchically labelled corpus of 10,000 sounds consisting of both manual and predicted labels.

Feature Representations

A number of different feature representations have been implemented for the experiments described in this thesis. These range from the statistical summary features employed in

Experiment 2 when comparing RF and SVM algorithms to the spectrogram based LPMS features utilised in Experiment 4 which compared SVM and CNN algorithms. Varying degrees of success have been observed using different representations and due to the specific approaches of each experiment it is difficult to draw definitive conclusions. However, a number of observations can be made and some general recommendations for suitable feature representations when predicting AH are appropriate.

Experiment 2, which compares the performance of RF and SVM algorithms when predicting AH, has utilised statistical summary representations drawn from a number of objective Low Level Descriptors (LLD) of sounds. These included data on spectral spread, entropy, rolloff and flux, in addition to MFCC and chroma measures. Zero order representations were extracted in addition to first and second order delta variants, and a number of standard statistical summaries (mean, median, min, max etc.) were drawn from the sounds. The summary vector was then subjected to recursive feature elimination to select the most informative elements. In this case, the double-delta representation was found to be disproportionately more useful than other representations — 57% of the features chosen as more informative were of this type, whereas 23% were zero order with 20% being first order delta. An SVM trained using these methods achieved high FG recall (93.3%), however, due to the dataset size (a total of 40 sounds) caution must be advised in terms of recommending this approach generally.

The AL exercise described in Experiment 3 utilised MFCC, chroma and LPMS feature representations consisting of zero order and first, second and fifth order delta data. The cross validation experiment described in Section 5.3.5 explicitly compared a number of different representations for use in the simulated labelling task. In this case, the delta representations were found not to be of use and were thus discarded. Marginal differences were found in the performance of an SVM model with an RBF kernel trained using zero order MFCC and LPMS representations (72.2% versus 73.9% ACA) with a chroma representation somewhat

poorer (65.7% ACA). It should be noted that the best performing representation (74.3%) for this SVM configuration used a zero order feature representation which was a concatenation of MFCC, chroma and LPMS representations which took approximately 4 times as long to train as the LPMS representation alone. Given the marginal performance improvement noted in this case, it was decided to utilise the LPMS feature representation given the considerable time saving involved.

The exploration of SVM and CNN algorithms described in Chapter 6 compares models trained using zero order and delta feature representations of LPMS data. Here it was established that for certain performance metrics a CNN trained using a zero order representation outperforms CNNs trained on delta data and is competitive in the other metrics used. Table 7.1 summarises these results which outline superior CNN zero order performance on 10,000 instances in ACA (82.2%) FG recall (79.0%) and nonFG precision (86.3%). The CNN trained using a delta representation on 100,000 instances is superior on both FG precision (78.7%) and nonFG recall (87.0%). Performance of SVMs trained with and without delta representations is similarly close.

Table 7.1 Summary of average ACA, precision and recall scores noted across three randomly selected hold-out test sets. Note that 'P' indicates Precision and 'R' indicates Recall in the following.

MODEL	ACA	FG P	FG R	nonFG P	nonFG R
CNN 10k Zero Order	82.2%	78.0%	79.0%	86.3%	85.7%
CNN 10k Delta	81.4%	78.3%	76.0%	85.0%	86.7%
CNN 100k Zero Order	80.9%	76.0%	77.3%	85.0%	84.3%
CNN 100k Delta	80.6%	78.7%	74.0%	84.0%	87.0%

In summary, statistical summaries of LLDs have been observed to provide representations on which high accuracy levels were noted in Experiment 2, an investigation of RF and SVM algorithms predicting AH. Double-delta representations proved particularly useful in this case, although it should be noted that these results were observed on a small dataset.

The results of exercises investigating feature representations for Experiment 3 indicate that, of the representations utilised, models built using LPMS and MFCC data perform to similar levels, with the LPMS being slightly superior to the MFCC and those using chroma representations somewhat behind both of these. A representation combining all of those extracted was found to provide marginally superior performance to both MFCC and LPMS representations on their own at the cost of significantly increased training time. Delta representations were not found to be useful in this experiment.

Finally, Experiment 4 presented SVM and CNN algorithms trained using LPMS representations, utilising both zero order and delta data. A number of the best scores observed were noted on a CNN trained using zero order information, with generally little to choose in terms of performance between the smaller zero order and larger delta representations. This suggests that bigger feature vectors are not always the optimal choice in the context of AH prediction, and that use of delta information in addition to zero order data is not always necessary or optimal. This work is relevant to the following contributions of this thesis:

Maj. Contrib. 1: A roadmap for research into ML methods for AH.

Min. Contrib. 1: In the context of AH, the LPMS zero order feature representation is found to be an effective compromise for predicting AH, providing comparable performance to larger representations which are considerably more expensive in terms of computation time. Delta representations are found to provide performance improvement in some, but not all cases.

7.2 Future Work

These results build the case for predicting AH using ML methods, indicating the feasibility of doing so via the performance of models compared in the experiments detailed in Chapters 4,

5 and 6. This is not to suggest that development in the area is in any sense complete. One potential application of a these models is within a variable asset compression codec, prioritising the most important audio elements by encoding them at high bitrates, while less important elements are delivered at poorer quality. A perceptual investigation of this idea is an obvious choice for further investigations, as it will give some insight as to the use of hierarchical prediction in this respect. Additionally, the complexity of perceiving and categorising sounds hierarchically has been summarised as being subject to a number of FIAH which are hypothesised to have an effect on hierarchical categorisation. These factors are another logical course of further study so that their degree of influence, or lack thereof, may be accounted for in future work.

The dataset developed for this study is intended to be a starting point for such research, providing as it does a ground-truth for AH which can be further manipulated in order to study FIAH. In order for this work to benefit the research community at large, it is intended to make this dataset publicly available once formatting and other considerations are addressed. Additionally, the models outlined in this work explore AH within the framework of sound categorisation. Exploring regression techniques in this domain is also worthy of investigation, given the potential for a more nuanced interpretation of hierarchy this would afford.

Results from the application of AL to AH are promising, and there are a number of additional areas for future work. The variance in scores achieved using different feature representations noted in Section 5.3.2 suggests an avenue for future investigation given the abundance of choice in terms of possible feature representations outlined in Section 3.4. Many possibilities for experimentation exist in this respect, such as altering the frame-level extraction parameters, or incorporating different window lengths to provide a more temporal context. Also, recent audio ML work has focussed on the use of raw waveforms as input to an 'end-to-end' deep learning classifier, which both learns a representation and classifies sounds [106]. Furthermore, in the case of AL methods, techniques with *Self Learning*

elements, where labels are assigned to instances based on predictions from a model, or the concept of *Co-Training*, where labels are derived via a combination of prediction and selection methods on different feature representations, are also worthy of investigation.

The work outlined in Chapter 6 has investigated SVM and CNN algorithms. An investigation of other deep learning architectures and algorithms, such as Recurrent Neural Networks, Long Short Term Memory structures, transfer learning and attention mechanisms as outlined in Section 3.5.5 would also be of interest. Capsule networks are another example, having been introduced by Sabour *et al.* [350] as an alternative to the use of dropout, and successfully used in an audio context by Vesperini *et al.* [351] and Iqbal *et al.* [352]. The focus on investigation of deep learning algorithms in the foregoing has also resulted in use of feature representations known to be effective in this context, such as the LPMS features implemented in Experiments 3 and 4. An FG recall rate of 93.3% was noted in Experiment 2 when comparing the performance of RF and SVM algorithms. This promising result, using statistical summary features as input, optimised with recursive feature elimination to find those most useful for classification, suggests that experimentation with such feature types in addition to LPMS and other combinations subjected to feature selection techniques as in Experiment 2 may be of interest.

AL techniques involve intelligent selection of instances to provide the maximum information to a model being trained to predict. They have been found to be effective in this case, but still involve a trade-off between prediction model accuracy and the cost of acquiring more labels. Considering media consumption applications, it would be worthwhile investigating the prospect of sourcing stimuli directly from broadcast content if possible. Using multi-track mixes from drama programmes, for example, might offer a method of labelling assets which are panned centrally and therefore objectively placed in the sound scene FG, in addition to giving insight into the variance of object hierarchy as proposed in Section 2.6. This may

give access to large volumes of data labelled for broadcast applications, therefore potentially rendering use of deep learning algorithms more attractive, given the results noted previously.

While this research has ultimately found little benefit in the implementation of data augmentations, in that CNN models trained using them are slightly outperformed by models trained without them in ACA, they are still an interesting concept which is worthy of further investigation. As noted in Section 3.7, the authors are unaware of any perceptual research investigating the effect of different audio augmentations on the semantic meaning of auditory stimuli, but this is another interesting topic for future research given the potential applications for other audio ML problems. The research described here has attempted to minimise bias in this respect by applying augmentations selectively and in a minimal manner. Providing knowledge of the extremes to which augmentations can be applied, before an effect on semantic meaning is caused, would be of use in many audio applications where a scarcity of data exists in tandem with potential for superior performance from ML algorithms.

In conclusion, this research has investigated AH as it pertains to modern media consumption and found that, while complex, the phenomenon can be predicted to high accuracy levels using ML methods, though with some caveats. The lack of a suitable dataset has been addressed using a number of techniques. Among these, sourcing labels manually should be considered the optimal method of building datasets. In certain situations it is impractical to do so at the scales required for best prediction performance, however, and in such cases AL using the EGAL selection method has been found to be most effective at selecting informative audio instances for labelling purposes. A number of algorithms have been investigated and found to classify auditory instances to high accuracy levels comparable to other audio ML applications. Taken together, these findings comprise a useful roadmap for both audio and machine learning practitioners to predict AH and thus provide a launch pad for further research.

References

[1] Brown, A. L., Kang, J. and Gjestland, T. Towards Standardization in Soundscape Preference Assessment. *Applied Acoustics*, vol. 72(6):pp. 387–392 [2011]. doi: 10.1016/j.apacoust.2011.01.001.

[2] Raimbault, M. and Dubois, D. Urban Soundscapes: Experiences and Knowledge. *Cities*, vol. 22(5):pp. 339–350 [2005]. doi:10.1016/j.cities.2005.05.003.

[3] Gemmeke, J. F., W Ellis, D. P., Freedman, D., Jansen, A., Lawrence, W., Channing Moore, R., Plakal, M. and Ritter, M. Audio Set: An Ontology and Human-Labeled Dataset for Audio Events. In *Proc. IEEE ICASSP 2017, New Orleans, LA (to appear)*. New Orleans, LA, USA; March 5-9 [2017].

[4] Ahirwar, K. Everything you need to know about Neural Networks [2017]. *https://hackernoon.com/everything-you-need-to-know-about-neural-networks-8988c3ee4491* [Accessed: 2020-02-08].

[5] Nielsen, M. A. *Neural Networks and Deep Learning*. Determination Press [2015]. URL http://neuralnetworksanddeeplearning.com/.

[6] Simonyan, K. and Zisserman, A. Very Deep Convolutional Networks for Large-scale Image Recognition. In *International Conference on Learning Representations (ICLR)*. San Diego, CA, USA; May 7 - 9 [2015].

[7] Chen, H., Liu, Z., Liu, Z., Zhang, P. and Yan, Y. Integrating the Data Augmentation Scheme with Various Classifiers for Acoustic Scene Modeling. Tech. rep., DCASE2019 Challenge [2019].

[8] Churnside, T. Object-Based Broadcasting [2013]. *http://www.bbc.co.uk/rd/blog/2013-05-object-based-approach-to-broadcasting* [Accessed: 2019-11-13].

[9] Woodcock, J., Davies, W. J., Melchior, F., Cox, T. J. and Member, A. Elicitation of Expert Knowledge to Inform Object-Based Audio Rendering to Different Systems. *Journal of the Audio Engineering Society*, vol. 66(1/2):pp. 44–59 [2018]. doi:10.17743/jaes.2018.0001.

[10] Woodcock, J., Davies, W. J. and Cox, T. J. A Cognitive Framework for the Categorisation of Auditory Objects in Urban Soundscapes. *Applied Acoustics*, vol. 121(2017):pp. 56–64 [2017]. doi:10.1016/j.apacoust.2017.01.027.

References

[11] Bregman, A. S. *Auditory Scene Analysis: The Perceptual Organisation of Sound*. Cambridge, MA: The MIT Press [1990].

[12] Guastavino, C. Everyday Sound Categorization. In T. Virtanen, M. D. Plumbley and D. Ellis (eds.), *Computational Analysis of Sound Scenes and Events*, pp. 183–213. Cham: Springer [2018]. doi:10.1007/978-3-319-63450-0_7.

[13] Bishop, C. M. *Pattern Recognition and Machine Learning*. New York: Springer-Verlag, 1 edn. [2006]. doi:10.1117/1.2819119.

[14] Virtanen, T., Plumbley, M. D. and Ellis, D. (eds.). *Computational Analysis of Sound Scenes and Events*. Springer International Publishing [2018]. doi: 10.1007/978-3-319-63450-0.

[15] Virtanen, T., Mesaros, A. and Heittola, T. [2017]. *http://www.cs.tut.fi/sgn/arg/dcase2017/index* [Accessed: 2019-11-13].

[16] Virtanen, T., Mesaros, A. and Heittola, T. [2018]. *http://dcase.community/challenge2018/index* [Accessed: 2019-11-13].

[17] Aytar, Y., Vondrick, C. and Torralba, A. Soundnet: Learning Sound Representations from Unlabeled Video. In *30th Conference on Neural Information Processing Systems (NIPS 2016)*, pp. 892–900. Barcelona, Spain; December 5 - 10 [2016].

[18] Schuller, B., Zhang, Y. and Weninger, F. Three Recent Trends in Paralinguistics on the way to Omniscient Machine Intelligence. *Journal on Multimodal User Interfaces*, vol. 12:pp. 273–283 [2018]. doi:10.1007/s12193-018-0270-6.

[19] Li, J., Dai, W., Metze, F., Qu, S. and Das, S. A Comparison of Deep Learning methods for Environmental Sound Detection. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 126–130. New Orleans, LA, USA; 5 - 9 March [2017]. doi:10.1109/ICASSP.2017.7952131.

[20] Purwins, H., Li, B., Virtanen, T., Schlüter, J., Chang, S.-y. and Sainath, T. Deep Learning for Audio Signal Processing. *Journal of Selected Topics of Signal Processing*, vol. 13(2):pp. 206–219 [2019]. doi:10.1109/JSTSP.2019.2908700.

[21] Mun, S., Park, S., Han, D. and Ko, H. Generative Adversarial Network Based Acoustic Scene Training Set Augmentation and Selection Using {SVM} Hyper-Plane. In *Detection and Classification of Acoustic Scenes and Events (DCASE 2017)*. Munich, Germany; 16th November: DCASE2017 Challenge [2017].

[22] Coleman, W., Adams, L., Cullen, C. and Yan, M. Perception of Auditory Objects in Complex Scenes: Factors and Applications. In *Institute of Acoustics - 21st Century Developments in Musical Sound Production, Presentation and Reproduction*, pp. 1–16. Nottingham, UK; November 21st [2017].

[23] Coleman, W., Cullen, C. and Yan, M. Categorisation of Isolated Sounds on a Background - Neutral - Foreground Scale. In *Proceedings of the 144th Convention of the Audio Engineering Society*, pp. 1–9. Milan, Italy; May 23-26 [2018].

[24] Coleman, W., Delany, S. J., Yan, M. and Cullen, C. A Machine Learning Approach to Hierarchical Categorisation of Auditory Objects. *Journal Audio Eng. Soc.*, vol. 68(1/2):pp. 48–56 [2020].

[25] Coleman, W., Delany, S. J., Yan, M. and Cullen, C. Active Learning for Auditory Hierarchy. In *Cross Domain Conference for Machine Learning and Knowledge Extraction (CD-MAKE)*, pp. 1–20. Dublin, Ireland; 25-28 August [2020].

[26] Chen, X.-W. and Lin, X. Big Data Deep Learning: Challenges and Perspectives. *IEEE Access*, vol. 2:pp. 514–525 [2014]. doi:10.1109/ACCESS.2014.2325029. URL http://ieeexplore.ieee.org/document/6817512/.

[27] Cakir, E., Parascandolo, G., Heittola, T., Huttunen, H. and Virtanen, T. Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 25(6):pp. 1291–1303 [2017].

[28] McAdams, S., Goodchild, M. and Thoret, E. [2018]. *https://www.mcgill.ca/timbre2018/* [Accessed: 2019-11-13].

[29] Bregman, A. S. Auditory Scene Analysis. In *International Encyclopedia of the Social and Behavioral Sciences*. Amsterdam: Pergamon (Elsevier) [2004].

[30] Guastavino, C. Categorization of Environmental Sounds. *Canadian Journal of Experimental Psychology*, vol. 61(1):pp. 54–63 [2007]. doi:10.1037/cjep2007006.

[31] Alain, C., Arnott, S. R. and Picton, T. W. Bottom-up and Top-down Influences on Auditory Scene Analysis: Evidence from Event-Related Brain Potentials. *Journal of Experimental Psychology: Human Perception and Performance*, vol. 27(5):pp. 1072–1089 [2001]. doi:10.1037//TO96-1523.27.

[32] Wisniewski, A. B. and Hulse, S. H. Auditory Scene Analysis in European Starlings (Sturnus Vulgaris): Discrimination of Song Segments, their Segregation from Multiple and Reversed Conspecific Songs, and Evidence for Conspecific Song Categorisation. *Journal of Comparative Psychology*, vol. 111(4):pp. 337–350 [1997].

[33] Darwin, C. J. and Carlyon, R. P. Auditory Grouping. In B. C. J. Moore (ed.), *Handbook of perception and cognition: Hearing.*, chap. 11, pp. 387–424. London, UK: Academic Press, 2nd edn. [1995].

[34] Rummukainen, O., Radun, J., Virtanen, T., Pulkki, V. and Murray, M. M. Categorization of Natural Dynamic Audiovisual Scenes. *PLoS ONE*, vol. 9(5):p. 14 [2014]. doi:10.1371/.

[35] Truax, B. *Acoustic Communication*. Norwood, NJ, USA: Ablex Publishing Corporation, 1st edn. [1984].

[36] Kotzé, H. and Möller, A. Effect of Auditory Subliminal Stimulation on GSR. *Psychological Reports*, vol. 67:pp. 931–934 [1990]. doi:10.2466/PR0.67.7.931-934.

[37] Norman, L. J., Heywood, C. A. and Kentridge, R. W. Exogenous Attention to Unseen Objects? *Consciousness and Cognition*, vol. 35:pp. 319–329 [2015].

[38] Linzarini, A., Houdé, O. and Borst, G. Cognitive Control Outside of Conscious Awareness. *Consciousness and Cognition*, vol. 53(June):pp. 185–193 [2017]. doi: 10.1016/j.concog.2017.06.014.

[39] Gregg, M. K. and Snyder, J. S. Enhanced Sensory Processing Accompanies Successful Detection of Change for Real-world Sounds. *NeuroImage*, vol. 62(2012):pp. 113–119 [2012]. doi:10.1016/j.neuroimage.2012.04.057.

[40] Demany, L., Bayle, Y., Puginier, E. and Semal, C. Detecting Temporal Changes in Acoustic Scenes: The Variable Benefit of Selective Attention. *Hearing Research*, vol. 353:pp. 17–25 [2017]. doi:10.1016/j.heares.2017.07.013.

[41] Dupoux, E., de Gardelle, V. and Kouider, S. Subliminal Speech Perception and Auditory Streaming. *Cognition*, vol. 109(2):pp. 267–273 [2008]. doi:10.1016/j.cognition.2008.06.012. URL http://dx.doi.org/10.1016/j.cognition.2008.06.012.

[42] Van Valkenburg, D. and Kubovy, M. From Gibson's fire to Gestalts: A Bridge-building Theory of Perceptual Objecthood. In *Ecological psychoacoustics*, pp. 113–147. Elsevier Science [2004].

[43] Cherry, C. E. Some Experiments on the Recognition of Speech, with One and with Two Ears. *Journal of the Acoustical Society of America*, vol. 25(5):pp. 975–979 [1953].

[44] Winkler, I. N., Denham, S. L. and Nelken, I. Modeling the Auditory Scene: Predictive Regularity Representations and Perceptual Objects. *Trends in Cognitive Sciences*, vol. 13(12):pp. 532–540 [2009]. doi:10.1016/j.tics.2009.09.003.

[45] Hausfeld, L., Riecke, L. and Formisano, E. Acoustic and Higher-level Representations of Naturalistic Auditory Scenes in Human Auditory and Frontal Cortex. *NeuroImage*, vol. 173:pp. 472–483 [2018]. doi:10.1016/j.neuroimage.2018.02.065.

[46] Pressnitzer, D., Graves, J., Chambers, C., de Gardelle, V. and Egré, P. Auditory Perception: Laurel and Yanny Together at Last. *Current Biology*, vol. 28(13):pp. R739 – R741 [2018]. doi:https://doi.org/10.1016/j.cub.2018.06.002.

[47] Thorogood, M., Fan, J. and Pasquier, P. Soundscape Audio Signal Classification and Segmentation Using Listener's Perception of Background and Foreground Sound. *Journal of the Audio Engineering Society*, vol. 64(7/8):pp. 484–492 [2016]. doi: 10.17743/jaes.2016.0021.

[48] World Soundscape Project Tape Library [2019]. *https://www.sfu.ca/~truax/wsp.html* [Accessed: 2019-11-13].

[49] Chion, M. *Audio-Vision: Sound on Screen*. New York, NY, USA: Columbia University Press [1994]. doi:10.7202/1025555ar.

[50] Wolvin, A. D. and Coakley, C. G. A Listening Taxonomy. In A. D. Wolvin and C. G. Coakley (eds.), *Perspectives on Listening*, pp. 15–22. Norwood, NJ, USA: Ablex Publishing Corporation [1993].

[51] Augoyard, J.-F. and Torgue, H. *Sonic Experience: A Guide to Everyday Sounds*. London, UK: McGill-Queen's University Press [2005].

[52] Woods, K. J. P. and McDermott, J. H. Attentive Tracking of Sound Sources. *Current Biology*, vol. 25(17):pp. 2238–2246 [2015].

[53] Winkler, I. and Schröger, E. Auditory Perceptual Objects as Generative Models: Setting the Stage for Communication by Sound. *Brain and Language*, vol. 148:pp. 1–22 [2015]. doi:https://doi.org/10.1016/j.bandl.2015.05.003.

[54] Eargle, J. *Audio Transmission Systems*, pp. 112–161. Dordrecht: Springer Netherlands [1986]. doi:10.1007/978-94-010-9366-8_4.

[55] Mann, M., Churnside, A., Bonney, A. and Melchior, F. Object-Based Audio Applied to Football Broadcasts: The 5 live Football Experiment. Tech. rep., BBC Research & Development, BBC Research & Development White Paper [2013]. URL http://downloads.bbc.co.uk/rd/pubs/whp/whp-pdf-files/WHP272.pdf.

[56] Davies, W. J., Adams, M. D., Bruce, N. S., Cain, R., Carlyle, A., Cusack, P., Hall, D. A., Hume, K. I., Irwin, A., Jennings, P., Marselle, M., Plack, C. J. and Poxon, J. Perception of Soundscapes: An Interdisciplinary Approach. *Applied Acoustics*, vol. 74(2):pp. 224–231 [2013]. doi:10.1016/j.apacoust.2012.05.010.

[57] Guastavino, C. The Ideal Urban Soundscape: Investigating the Sound Quality of French Cities. *Acta Acustica united with Acustica*, vol. 92(2006):pp. 945–951 [2006].

[58] Woodcock, J., Davies, W. J., Cox, T. J., Member, A. and Melchior, F. Categorization of Broadcast Audio Objects in Complex Auditory Scenes. *Journal of the Audio Engineering Society*, vol. 64(6) [2016]. doi:10.17743/jaes.2016.0007.

[59] Salamon, J., Jacoby, C. and Bello, J. P. A Dataset and Taxonomy for Urban Sound Research. In *Proceedings of the ACM International Conference on Multimedia - MM '14*, pp. 1041–1044. Orlando, Florida, USA; November 3-7 [2014]. doi:10.1145/2647868.2655045.

[60] Lindborg, P. A Taxonomy of Sound Sources in Restaurants. *Applied Acoustics*, vol. 110:pp. 297–310 [2016]. doi:10.1016/j.apacoust.2016.03.032.

[61] Sussman-Fort, J. and Sussman, E. The Effect of Stimulus Context on the Buildup to Stream Segregation. *Frontiers in Neuroscience*, vol. 8(8 APR):pp. 1–8 [2014]. doi:10.3389/fnins.2014.00093.

[62] Gygi, B., Kidd, G. R. and Watson, C. S. Similarity and Categorization of Environmental Sounds. *Perception & Psychophysics*, vol. 69(6):pp. 839–855 [2007].

[63] Lewis, J. W., Talkington, W. J., Tallaksen, K. C. and Frum, C. a. Auditory Object Salience: Human Cortical Processing of Non-Biological Action Sounds and their Acoustic Signal Attributes. *Frontiers in Systems Neuroscience*, vol. 6(May):pp. 1–15 [2012]. doi:10.3389/fnsys.2012.00027.

[64] Truax, B. World Soundscape Project Tape Library [2015]. *http://www.sfu.ca/sonic-studio/srs/index2.html* [Accessed: 2017-03-07].

[65] Gaver, W. W. What in the World do we Hear?: An Ecological Approach to Auditory Event Perception. *Ecological Psychology*, vol. 5(1):pp. 1–29 [1993].

[66] Schafer, R. M. *The Soundscape: Our Sonic Environment and the Tuning of the World*. Rochester, Vermont: Destiny Books [1994].

[67] Gemmeke, J. F., Ellis, D. P. W. and Freedman, D. [2019]. *https://research.google.com/audioset/ontology/index.html* [Accessed: 2019-11-13].

[68] YouTube [2019]. *https://www.youtube.com/* [Accessed: 2019-11-13].

[69] Ekeroot, J., Berg, J. and Nykänen, A. Selection of Audio Stimuli for Listening Tests. In *Audio Engineering Society Convention 130*, pp. 1–7. London, UK: AES [2011].

[70] International Telecommunication Union. ITU-R BS.1534-3, Method for the Subjective Assessment of Intermediate Quality Level of Audio Systems. *ITU-R Recommendation*, vol. 1534-3 [2015].

[71] Olive, S. E. A Method For Training Listeners and Selecting Program Material For Listening Tests. In *Proceedings of the 97th Convention of the Audio Engineering Society*. San Francisco, CA, USA; November 10-13 [1994].

[72] Lagrange, M., Lafay, G., Défréville, B. and Aucouturier, J.-J. The Bag-of-frames Approach: A Not So Sufficient Model for Urban Soundscapes. *The Journal of the Acoustical Society of America*, vol. 138(5):pp. EL487–EL492 [2015]. doi:10.1121/1.4935350.

[73] Sturm, B. L. Classification Accuracy is Not Enough. *Journal of Intelligent Information Systems*, vol. 41(3):pp. 371–406 [2013]. doi:10.1007/s10844-013-0250-y. URL https://doi.org/10.1007/s10844-013-0250-y.

[74] Bates, E., Gorzel, M., Ferguson, L., O'Dwyer, H. and Boland, F. M. Comparing Ambisonic Microphones: Part 1. In *2016 AES International Conference on Sound Field Control*, 6-3, pp. 1–10. Guildford, UK: AES [2016].

[75] Wüstenhagen, U., Feiten, B., Kroll, J., Raake, A. and Wältermann, M. Evaluation of Super-Wideband Speech and Audio Codecs. In *Proceedings of the 129th Convention of the Audio Engineering Society (2010)*, p. Paper 8205. San Francisco, CA, USA: AES [2010].

[76] Feiten, B., Raake, A., Garcia, M.-N., Wüstenhagen, U. and Kroll, J. Subjective Quality Evaluation of Audio Streaming Applications on Absolute and Paired Rating Scales. In *Proceedings of the 126th Convention of the Audio Engineering Society (2009)*, p. Paper 7787. Munich, Germany: AES [2009].

[77] Barbour, J. L. Subjective Consumer Evaluation of Multi-Channel Audio Codecs. In *Proceedings of the 119th Convention of the Audio Engineering Society (2005)*, p. Paper 6558. New York, NY, USA: AES [2005].

[78] George, S., Zielinski, S., Rumsey, F. and Bech, S. Evaluating the Sensation of Envelopment Arising from 5-channel Surround Sound Recordings. In *Proceedings of the 124th Convention of the Audio Engineering Society (2008)*, p. Paper 7382. Amsterdam, The Netherlands [2008].

[79] Schinkel-Bielefeld, N., Lotze, N. and Nagel, F. Audio Quality Evaluation by Experienced and Inexperienced Listeners. In *Proceedings of Meetings on Acoustics*, vol. 19. Montreal, Canada [2013]. doi:10.1121/1.4799190. URL http://acousticalsociety.org/.

[80] Quackenbush, S. and Gross, A. M. Analysis of Subjective Data From the Mpeg Unified Speech and Audio Coding Call for Proposals. In *Proceedings of the 38th International Conference of the Audio Engineering Society*, pp. 7–3. Pitea, Sweden: AES [2010].

[81] Stoll, G. and Kozamernik, F. EBU listening tests on Internet audio codecs. *EBU Technical Review*, (June):p. 24 [2000]. doi:10.1049/cp:19971266.

[82] De Man, B. and Reiss, J. D. A Pairwise and Multiple Stimuli Approach to Perceptual Evaluation of Microphone Types. In *Proceedings of the 134th Convention of the Audio Engineering Society (2013)*. Rome, Italy; May 4-7: May 4-7 [2013].

[83] Sun, S., Shen, Y., Liu, Z. and Feng, X. The Effects of Recording and Playback Methods in Virtual Listening Tests. *Journal of the Audio Engineering Society*, vol. 63(7/8):pp. 570–582 [2015]. doi:10.17743/jaes.2015.0058.

[84] Collett, E., Marx, M., Gaillard, P., Roby, B., Fraysse, B., Deguine, O. and Barone, P. Categorization of Common Sounds by Cochlear Implanted and Normal Hearing Adults. *Hearing Research*, vol. 335:pp. 207–219 [2016]. doi:10.1016/j.heares.2016.03.007.

[85] Gygi, B. and Shafiro, V. The Incongruency Advantage for Sounds in Natural Scenes. In *Proceedings of the 125th Convention of the Audio Engineering Society*, p. 6. San Francisco, CA, USA; October 2-5 [2008].

[86] Lewis, J. W., Brefczynski, J. A., Phinney, R. E., Janik, J. J. and Deyoe, E. A. Distinct Cortical Pathways for Processing Tool versus Animal Sounds. *Journal of Neuroscience*, vol. 25(21):pp. 5148–5158 [2005]. doi:10.1523/JNEUROSCI.0419-05.2005.

[87] Wilson, A. and Fazenda, B. M. Perception of Audio Quality in Productions of Popular Music. *AES: Journal of the Audio Engineering Society*, vol. 64(1-2):pp. 23–34 [2016]. doi:10.17743/jaes.2015.0090.

[88] Hold, C., Nagel, L., Wierstorf, H. and Raake, A. Positioning of Musical Foreground Parts in Surrounding Sound Stages. In *Proceedings of the 2016 AES International Conference on Audio for Virtual and Augmented Reality*, pp. 1–7. Los Angeles, CA, USA; Sept 30 - Oct 1 [2016].

[89] International Telecommunication Union. ITU-R BS.1116-3, Methods for the Subjective Assessment of Small Impairments in Audio Systems. *ITU-R Recommendation*, vol. 1116(3) [2015].

[90] Gygi, B., Kidd, G. R. and Watson, C. S. Spectral-temporal Factors in the Identification of Environmental Sounds. *The Journal of the Acoustical Society of America*, vol. 115(1252):pp. 1252–1265 [2004]. doi:10.1121/1.1635840ÍŤ.

[91] Novello, A., Mckinney, M. F. and Kohlrausch, A. Perceptual Evaluation of Music Similarity. In *ISMIR 2006, 7th International Conference on Music Information Retrieval*. Victoria, Canada; 8-12 October 2006 [2006].

[92] Francombe, J., Mason, R., Dewhirst, M. and Bech, S. Investigation of a Random Radio Sampling Method for Selecting Ecologically Valid Music Programme Material. In *Proceedings of the 136th AES Convention, Berlin, Germany*, vol. 4277. Berlin, Germany; April 26-29 [2014].

[93] Thomassen, S. and Bendixen, A. Subjective Perceptual Organization of a Complex Auditory Scene. *J. Acous. Soc. Am.*, vol. 141(1):pp. 265–276 [2017]. doi:10.1121/1.4973806.

[94] Denham, S., Bõhm, T. M., Bendixen, A., Szalárdy, O., Kocsis, Z., Mill, R. and Winkler, I. Stable Individual Characteristics in the Perception of Multiple Embedded Patterns in Multistable Auditory Stimuli. *Frontiers in Neuroscience*, (8 FEB) [2014]. doi:10.3389/fnins.2014.00025.

[95] Axelsson, O., Nilsson, M. E. and Berglund, B. A Principal Components Model of Soundscape Perception. *Journal of the Acoustic Society of America*, vol. 128(5):pp. 2836–2846 [2010]. doi:10.1007/978-1-4419-0561-1_48.

[96] Sudarsono, A. S., Lam, Y. W. and Davies, W. J. The Effect of Sound Level on Perception of Reproduced Soundscapes. *Applied Acoustics*, vol. 110:pp. 53–60 [2016]. doi:10.1016/j.apacoust.2016.03.011.

[97] Peltonen, V. T. K., Eronen, A. J., Parviainen, M. P. and Klapuri, A. P. Recognition of Everyday Auditory Scenes: Potentials, Latencies and Cues. In *Proceedings of the 110th Convention of the Audio Engineering Society (2001)*. Amsterdam, The Netherlands; May 12-15 [2001].

[98] Davies, W. J., Bruce, N. S. and Murphy, J. E. Soundscape Reproduction and Synthesis. *Acta Acustica united with Acustica*, vol. 100(2):pp. 285–292 [2014]. doi:10.3813/AAA.918708.

[99] Brefczynski-Lewis, J. A. and Lewis, J. W. Auditory Object Perception: A Neurobiological Model and Prospective Review. *Neuropsychologia*, (In Press):pp. 1–20 [2017]. doi:10.1016/j.neuropsychologia.2017.04.034.

[100] Roma, G., Janer, J., Kersten, S., Schirosa, M., Herrera, P. and Serra, X. Ecological Acoustics Perspective for Content-Based Retrieval of Environmental Sounds. *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2010:pp. 1–11 [2010]. doi:10.1155/2010/960863.

[101] Piczak, K. J. ESC: Dataset for Environmental Sound Classification. In *Proceedings of the 23rd ACM International Conference on Multimedia - MM '15*, pp. 1015–1018. Brisbane, Australia; October 26-28: Harvard Dataverse [2015]. doi:10.1145/2733373.2806390.

[102] Salamon, J., Jacoby, C. and Bello, J. P. [2019]. *https://urbansounddataset.weebly.com/* [Accessed: 2019-11-13].

[103] FreeSound [2019]. *https://freesound.org/* [Accessed: 2019-11-13].

[104] Sailor, H. B., Agrawal, D. M. and Patil, H. A. Unsupervised Filterbank Learning Using Convolutional Restricted Boltzmann Machine for Environmental Sound Classification. In *Proceedings of Interspeech 2017*, pp. 3107–3111. Stockholm, Sweden; 20 - 24 August [2017].

[105] N. Tak, R., Agrawal, D. and Patil, H. Novel Phase Encoded Mel Filterbank Energies for Environmental Sound Classification. In *International Conference on Pattern Recognition and Machine Intelligence 2017*, pp. 317–325. Kolkata, India; December 5 - 8 [2017].

[106] Piczak, K. J. Environmental Sound Classification with Convolutional Neural Networks. In *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6. Boston, USA; 17 - 20 September: IEEE [2015].

[107] Mesaros, A., Heittola, T. and Virtanen, T. DCASE2018 Challenge [2018]. *http://dcase.community/challenge2018/index* [Accessed: 2018-05-10].

[108] Fonseca, E., Plakal, M., Font, F., Ellis, D. P. W., Favory, X., Pons, J. and Serra, X. [2018]. *https://www.kaggle.com/c/freesound-audio-tagging* [Accessed: 2019-11-13].

[109] Heittola, T. [2018]. *http://www.cs.tut.fi/~heittolt/datasets* [Accessed: 2018-08-27].

[110] European Broadcasting Union Members [2019]. *https://www.ebu.ch/about/members* [Accessed: 2019-11-13].

[111] Marston, D., Kozamernik, F., Stoll, G. and Spikofski, G. Further EBU Tests of Multi-channel Audio Codecs. In *Proceedings of the 126th Convention of the Audio Engineering Society (2009)*, p. 10. Munich, Germany: AES [2009].

[112] Bech, S. and Zacharov, N. *Perceptual Audio Evaluation - Theory, Method and Application*. London, UK: John Wiley & Sons [2006].

[113] Liebetrau, J., Nagel, F., Zacharov, N., Watanabe, K., Colomes, C., Crum, P., Sporer, T. and Mason, A. Revision of Rec. ITU-R BS.1534. In *Proceedings of the 137th Convention of the Audio Engineering Society (2014)*. Los Angeles, CA, USA: Audio Engineering Society [2014].

[114] Jillings, N., De Man, B., Moffat, D. and Reiss, J. D. Web Audio Evaluation Tool: A Browser-Based Listening Test Environment. In *12th Sound and Music Computing Conference*. Maynooth, Ireland; July 26th - August 1st [2015].

[115] European Broadcasting Union. EBU – TECH 3324 EBU Evaluations of Multichannel Audio Codecs: Phase 1 & 2. *EBU*, vol. 3324(September):pp. 1–88 [2007]. URL https://tech.ebu.ch/docs/tech/tech3324.pdf.

[116] European Broadcasting Union. EBU – TECH 3339 EBU Evaluations of Multichannel Audio Codecs: Phase 3. *EBU*, vol. 3324(March):pp. 1–37 [2010]. URL https://tech.ebu.ch/docs/tech/tech3339.pdf.

[117] Mason, A., Marston, D., Kozamernik, F. and Stoll, G. EBU tests of multi-channel audio codecs. In *Proceedings of the 122nd Convention of the Audio Engineering Society (2007)*, pp. 1–9. Vienna, Austria: AES [2007].

[118] Cartwright, M., Pardo, B., Mysore, G. J. and Hoffman, M. Fast and Easy Crowdsourced Perceptual Audio Evaluation. Tech. rep., Adobe Research [2015].

[119] Lipshitz, S. and Vanderkooy, J. The Great Debate: Subjective Evaluation. *Journal of the Audio Engineering Society*, vol. 29(7/8):pp. 482–491 [1981].

[120] Toole, F. E. Subjective Measurements of Loudspeaker Sound Quality. In *Proceedings of the 72nd Convention of the Audio Engineering Society (1982)*. Anaheim, CA, USA: Audio Engineering Society [1982].

[121] Gabrielsson, A. and Lindström, B. Perceived Sound Quality of High-Fidelity Loudspeakers. *Journal of the Audio Engineering Society*, vol. 33(1/2):pp. 33–53 [1985].

[122] Lavandier, M., Herzog, P. and Meunier, S. Comparative Measurements of Loudspeakers in a Listening Situation. *Journal of the Acoustical Society of America*, vol. 123(1):pp. 77–87 [2008]. doi:10.1121/1.2816571.

[123] Soren, B. Listening Tests on Loudspeakers: A Discussion of Experimental Procedures and Evaluation of the Response Data. In *Proceedings of the 8th International Conference of the Audio Engineering Society*. Washington D.C., USA: May [1990].

[124] Wustenhagen, U., Feiten, B. and Hoeg, W. Subjective Listening Test of Multichannel Audio Codecs. In *Proceedings of the 105th Convention of the Audio Engineering Society (1998)*, p. 8. San Francisco, CA, USA; September 26-29 [1998].

[125] Soulodre, G., Grusec, T., Lavoie, M. and Thibault, L. Subjective Evaluation of State-of-the-art Two-channel Audio Codecs. *Journal of the Audio Engineering Society*, vol. 5489(June):pp. 0–24 [1998]. doi:10.1016/S0920-5489(99)90991-1.

[126] European Broadcasting Union. EBU Tech 3276s1-2004 Supplement 1 - Listening conditions for the assessment of sound programme material: Multichannel Sound. *EBU*, vol. 3276s1-200(May):p. 13 [2004].

[127] Wickelmaier, F. and Choisel, S. Selecting Participants for Listening Tests of Multichannel Reproduced Sound. In *Proceedings of the 118th Convention of the Audio Engineering Society*. Barcelona, Spain; May 28-31 [2005].

[128] Lavoie, M. and Soulodre, G. Stereo and Multichannel Loudness Perception and Metering. In *Proceedings of the 119th Convention of the Audio Engineering Society (2005)*. New York, USA; October 7-10 [2005].

[129] Hynninen, J. and Zacharov, N. GuineaPig A generic subjective test system for multichannel audio. In *Proceedings of the 106th Convention of the Audio Engineering Society (1999)*. Munich, Germany [1999].

[130] O'Toole, B., O'Sullivan, L., Kelly, I., Boland, F., Gorzel, M. and Kearney, G. Virtual 5.1 Surround Sound Localization using Head-Tracking Devices. *Proc. of 25th IET Irish Signals & Systems Conference 2014 and 2014 China-Ireland International Conference on Information and Communities Technologies (ISSC 2014/CIICT 2014)*, (JUNE):pp. 41–46 [2014]. doi:10.1049/cp.2014.0656.

[131] Zieliński, S. K., Rumsey, F., Kassier, R. and Bech, S. Comparison of Basic Audio Quality and Timbral and Spatial Fidelity Changes caused by Limitation of Bandwidth and by Down-mix Algorithms in 5.1 Surround Audio Systems. *AES: Journal of the Audio Engineering Society*, vol. 53(3):pp. 174–192 [2005].

[132] Marins, P., Rumsey, F. and Zielinski, S. Unravelling the Relationship between Basic Audio Quality and Fidelity Attributes in Low Bit-rate Multi-channel Audio Codecs. In *Proceedings of the 124th Convention of the Audio Engineering Society (2008)*. Amsterdam, The Netherlands: Audio Engineering Society [2008].

[133] Bagousse, S. L., Colomes, C., Paquier, M., Le Bagousse, S., Colomes, C. and Paquier, M. State of the Art on Subjective Assessment of Spatial Sound Quality. In *AES 38th International Conference: Sound Quality Evaluation*, vol. 2. Pitea, Sweden; June 13-15 [2010].

[134] Zielinski, S., Rumsey, F. and Bech, S. On some biases encountered in modern listening tests. *Journal of the Audio Engineering Society*, vol. 56(6):pp. 427–451 [2008]. doi:10.17743/jaes.2015.0094.

[135] Zieliński, S., Hardisty, P., Hummersone, C. and Rumsey, F. Potential Biases in MUSHRA Listening Tests. In *Proceedings of the 123rd Convention of the Audio Engineering Society (2007)*, p. Paper 7179. New York, NY, USA: AES [2007].

[136] Poulton, E. C. *Bias in Quantifying Judgments*. Exeter, UK: Lawrence Erblaum Associates [1989]. doi:10.2307/1423161. URL http://www.jstor.org/stable/1423161?origin=crossref.

[137] Mizrahi, A., Shalev, A. and Nelken, I. Single Neuron and Population Coding of Natural Sounds in Auditory Cortex. *Current Opinion in Neurobiology*, vol. 24:pp. 103–110 [2014].

[138] Lavandier, C. and Defréville, B. The Contribution of Sound Source Characteristics in the Assessment of Urban Soundscapes. *Acta Acustica united with Acustica*, vol. 92:pp. 912–921 [2006].

[139] Zhang, M. and Kang, J. Towards the Evaluation, Description, and Creation of Soundscapes in Urban Open Spaces. *Environment and Planning B: Planning and Design*, vol. 34(1):pp. 68–86 [2007]. doi:10.1068/b31162.

[140] Hume, K. and Ahtamad, M. Physiological responses to and subjective estimates of soundscape elements. *Applied Acoustics*, vol. 74(2):pp. 275–281 [2013]. doi:10.1016/j.apacoust.2011.10.009.

[141] Olive, S. E. and Welti, T. The Relationship between Perception and Measurement of Headphone Sound Quality. In *Proceedings of the 133rd Convention of the Audio Engineering Society (2012)*, pp. 1–17. San Francisco, CA, USA: October 26-29 [2012].

[142] Lemaitre, G., Houix, O., Misdariis, N. and Susini, P. Listener Expertise and Sound Identification Influence the Categorization of Environmental Sounds. *Journal of Experimental Psychology: Applied*, vol. 16(1):pp. 16–32 [2010]. doi:10.1037/a0018762.

[143] Lorho, G., Le Ray, G. and Zacharov, N. eGauge - A Measure of Assessor Expertise in Audio Quality Evaluations. In *Proceeding of the Audio Engineering Society 38th International Conference on Sound Quality Evaluation*, pp. 13–15 June. Pitea, Sweden: AES [2010].

[144] Pérez-Martínez, G., Torija, A. J. and Ruiz, D. P. Soundscape Assessment of a Monumental Place: A Methodology Based on the Perception of Dominant Sounds. *Landscape and Urban Planning*, vol. 169:pp. 12–21 [2017]. doi:10.1016/j.landurbplan.2017.07.022.

[145] Brambilla, G., Gallo, V., Asdrubali, F. and D'Alessandro, F. The Perceived Quality of Soundscape in Three Urban Parks in Rome. *The Journal of the Acoustical Society of America*, vol. 134(1):pp. 832–839 [2013]. doi:10.1121/1.4807811.

[146] Craig, A., Moore, D. and Knox, D. Experience Sampling: Assessing Urban Soundscapes using In-situ Participatory Methods. *Applied Acoustics*, vol. 117:pp. 227–235 [2017]. doi:10.1016/j.apacoust.2016.05.026.

[147] Bech, S. Selection and Training of Subjects for Listening Tests on Sound-reproducing Equipment. *Journal Audio Eng. Soc.*, vol. 40(7/8):pp. 590–610 [1992]. URL http://www.aes.org/e-lib/browse.cfm?elib=7040.

[148] Bellman, R. *Adaptive Control Processes: A Guided Tour*. Princeton University Press [1961].

[149] Disley, A. C., Howard, D. M. and Hunt, A. D. Timbral Description of Musical Instruments. In *9th International Conference on Music Perception and Cognition*. Bologna, Italy; August 22-26 [2006].

[150] McGraw, K. O., Tew, M. D. and Williams, J. E. The Integrity of Web-delivered Experiments: Can You Trust the Data? *Psychological Science : A Journal of the American Psychological Society / APS*, vol. 11(6):pp. 502–506 [2000]. doi:10.1111/1467-9280.00296.

[151] Amazon Mechanical Turk [2019]. *https://www.mturk.com/* [Accessed: 2019-11-13].

[152] Vaughan, B. *Naturalistic Emotional Speech Corpora with Large Scale Emotional Dimension Ratings. Doctoral thesis*. Doctoral, Technological University Dublin [2011]. doi:10.21427/D7GK59.

[153] Ellis, D. AUDITORY list home page [2020]. *http://www.auditory.org/* [Accessed: 2020-02-19].

[154] Xperi Website. Home - Xperi [2020]. *https://www.xperi.com/* [Accessed: 2020-02-19].

[155] Shamma, S., Elhilali, M., Ma, L., Micheyl, C., Oxenham, A. J., Pressnitzer, D., Yin, P. and Xu, Y. Temporal Coherence and the Streaming of Complex Sounds. In B. C. J. Moore, R. D. Patterson, I. M. Winter, R. P. Carlyon and H. E. Gockel (eds.), *Basic Aspects of Hearing: Physiology and Perception*, vol. 787, chap. 59, pp. 109–118. New York, USA: Springer-Verlag [2013]. doi:10.1007/978-1-4614-1590-9.

[156] Lehmann, A. and Schönwiesner, M. Selective Attention Modulates Human Auditory Brainstem Responses: Relative Contributions of Frequency and Spatial Cues. *PLoS ONE*, vol. 9(1):pp. 1–10 [2014]. doi:10.1371/journal.pone.0085442.

[157] Webster, J. C. and Thompson, P. O. Responding to Both of Two Overlapping Messages. *The Journal of the Acoustical Society of America*, vol. 26(3):p. 396 [1954]. doi:10.1121/1.1907348.

[158] Pollack, I. and Pickett, J. Cocktail Party Effect. *Journal of the Acoustical Society of America*, vol. 29(11):pp. 1262–1262 [1957].

[159] Huron, D. *Sweet Anticipation: Music and the Psychology of Expectation*. Cambridge, MA, USA: The MIT Press [2006].

[160] Bigand, E. and Poulin-Charronnat, B. Are We "Experienced Listeners"? A Review of the Musical Capacities that do not Depend on Formal Musical Training. *Cognition*, vol. 100(2006):pp. 100–130 [2006]. doi:10.1016/j.cognition.2005.11.007.

[161] McAdams, S. Recognition of Sound Sources and Events. In S. McAdams and E. Bigand (eds.), *Thinking in Sound: The Cognitive Psychology of Human Audition*, chap. 6, pp. 146–198. Oxford, UK: Clarendon Press [1993].

[162] Udesen, J., Piechowiak, T. and Gran, F. Vision Affects Sound Externalization. In *Proceedings of the 55th International Conference of the Audio Engineering Society*, pp. 1–4. Helsinki, Finland; August 27-29 [2014].

[163] Gruters, K. G., Murphy, D. L. K., Smith, D. W., Shera, C. A. and Groh, J. M. The Eardrum Moves when the Eyes Move: A Multisensory Effect on the Mechanics of Hearing. *bioRxiv*, vol. 156570 [2017]. doi:10.1101/156570.

[164] Yong Jeon, J., Jik Lee, P., Young Hong, J. and Cabrera, D. Non-auditory Factors Affecting Urban Soundscape Evaluation. *The Journal of the Acoustical Society of America*, vol. 130(6):pp. 3761–3770 [2011]. doi:10.1121/1.3652902.

[165] Steffens, J., Steele, D. and Guastavino, C. Situational and Person-related Factors Influencing Momentary and Retrospective Soundscape Evaluations in Day-to-day Life. *The Journal of the Acoustical Society of America*, vol. 141(3):pp. 1414–1425 [2017]. doi:10.1121/1.4976627.

[166] Giordano, B. L., McDonnell, J. and McAdams, S. Hearing Living Symbols and Nonliving Icons: Category Specificities in the Cognitive Processing of Environmental Sounds. *Brain and Cognition*, vol. 73(1):pp. 7–19 [2010]. doi:10.1016/j.bandc.2010.01.005.

[167] Dubois, D., Guastavino, C. and Raimbault, M. A Cognitive Approach to Urban Soundscapes: Using Verbal Data to Access Everyday Life Auditory Categories. *Acta Acustica united with Acustica*, vol. 92(2006):pp. 865–874 [2006].

[168] Handel, S. *Listening: An Introduction to the Perception of Auditory Events*. Cambridge, MA, USA: The MIT Press [1989].

[169] Handel, S. Timbre Perception and Auditory Object Identification. In B. C. J. Moore (ed.), *Listening*, chap. 12. London, UK: Academic Press [1995].

[170] Experience in Multimedia Systems and Services (COST Action IC 1003). Tech. Rep. Version 1.2, Lausanne, Switzerland [2013].

[171] Moller, S. *Quality Engineering - Quality of Communication Systems*. Berlin: Springer-Verlag Berlin Heidelberg, 1 edn. [2010]. doi:10.1007/978-3-642-11548-6. URL https://www.springer.com/de/book/9783642115486.

[172] Dave, N. Feature Extraction Methods LPC , PLP and MFCC In Speech Recognition. *International Journal for Advance Research in Engineering and Technology*, vol. 1(Vi):pp. 1–5 [2013].

[173] Hermansky, H. and Morgan, N. RASTA Processing of Speech. *IEEE Transactions on Speech and Audio Processing*, vol. 2(4):pp. 578–589 [1994].

[174] Rabiner, L. R. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, vol. 77(2):pp. 257–286 [1989]. doi:10.1109/5.18626.

[175] Yang, Y.-H. and Chen, H. H. Machine Recognition of Music Emotion: A Review. *ACM Transactions on Intelligent Systems and Technology*, vol. 3(3):pp. 1–30 [2012]. doi:10.1145/2168752.2168754.

[176] Virtanen, T., Plumbley, M. D. and Ellis, D. P. W. Introduction to Sound Scene and Event Analysis. In T. Virtanen, M. D. Plumbley and D. P. W. Ellis (eds.), *Computational Analysis of Sound Scenes and Events*, chap. 1, pp. 3–12. Springer International Publishing, 1 edn. [2018].

[177] Joder, C., Essid, S. and Richard, G. Temporal Integration for Audio Classification With Application to Musical Instrument Classification. *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, vol. 17(1):pp. 174–186 [2009]. doi:10.1109/TASL.2008.2007613.

[178] Google Home. Google Home - Smart Speaker & Home Assistant - Google Store [2019]. *https://store.google.com/product/google{_}home* [Accessed: 2018-08-27].

[179] Echo Alexa 2019. Amazon Echo (2nd generation) — Alexa Speaker [2019]. *https://www.amazon.com/all-new-amazon-echo-speaker-with-wifi-alexa-dark-charcoal/dp/B06XCM9LJ4* [Accessed: 2018-08-27].

[180] Kelleher, J. D., Mac Namee, B. M. and D'Arcy, A. *Fundamentals of Machine Learning for Predictive Data Analytics*. 1. London, England: The MIT Press [2015]. doi:10.1007/s13398-014-0173-7.2.

[181] VanderPlas, J. *Python Data Science Handbook: Essential Tools for Working with Data*. O'Reilly Media, Inc. [2016].

[182] Boser, B. E., Guyon, I. M. and Vapnik, V. N. A Training Algorithm for Optimal Margin Classiiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning*. Pittsburgh, PA, USA; July 27-29 [1992].

[183] Ng, A. Y. and Jordan, M. I. On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. In T. G. Dietterich, S. Becker and Z. Ghahramani (eds.), *Advances in Neural Information Processing Systems 14*, pp. 841–848. MIT Press [2002].

[184] Noda, J., Travieso, C. and Sánchez-Rodríguez, D. Automatic Taxonomic Classification of Fish Based on Their Acoustic Signals. *Applied Sciences*, vol. 6(12):p. 443 [2016]. doi:10.3390/app6120443.

[185] Hinton, G. E. and Salakhutdinov, R. R. Reducing the Dimensionality of Data with Neural Networks. *Science*, vol. 313(5786):pp. 504–507 [2006]. doi:10.1126/science.1127647.

[186] Alías, F., Socoró, J. C. and Sevillano, X. A Review of Physical and Perceptual Feature Extraction Techniques for Speech, Music and Environmental Sounds. *Applied Sciences*, vol. 6(143):p. 44 [2016]. doi:10.3390/app6050143.

[187] Agrawal, D. M., Sailor, H. B., Soni, M. H. and Patil, H. A. Novel TEO-based Gammatone Features for Environmental Sound Classification. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pp. 1809–1813. Kos Island, Greece; 28 August - 2 September [2017]. doi:10.23919/EUSIPCO.2017.8081521.

[188] Mitrovic, D., Zeppelzauer, M. and Breiteneder, C. Discrimination and Retrieval of Animal Sounds. In *Proceedings of the IEEE 2006 12th International Multi-Media Modelling Conference*, pp. 1–5. Beijing, China; January 4 - 6 [2006].

[189] Stowell, D. and Plumbley, M. D. Automatic Large-scale Classification of Bird Sounds is Strongly Improved by Unsupervised Feature Learning. *PeerJ*, vol. 2:p. e488 [2014].

[190] Tang, J., Alelyani, S. and Liu, H. Feature Selection for Classification: A Review. In C. C. Aggarwal (ed.), *Data Classification: Algorithms and Applications*, chap. 2, pp. xxvii, 667. London: CRC Press [2014].

[191] Wolpert, D. H. The Lack of A Priori Distinctions between Learning Algorithms. *Neural computation*, vol. 8(7):pp. 1341–1390 [1996].

[192] Zeiler, M. D. and Fergus, R. Visualizing and Understanding Convolutional Networks. In *European Conference on Computer Vision (ECCV 2014)*, pp. 818–833. Zurich, Switzerland; September 6 - 12: Springer [2014].

[193] Ruvolo, P., Fasel, I. and Movellan, J. R. A Learning Approach to Hierarchical Feature Selection and Aggregation for Audio Classification. *Pattern Recognition Letters*, vol. 31(12):pp. 1535–1542 [2010].

[194] Barrington, L., Chan, A., Turnbull, D. and Lanckriet, G. Audio Information Retrieval using Semantic Similarity. In *Proceedings IEEE Int. Conf. on Acoustics, Speech and Signal Processing. ICASSP 2007*, vol. 2, pp. II–725. Honolulu, USA; April 15-20: IEEE [2007].

[195] Grimm, M., Kroschel, K., Mower Provost, E. and Narayanan, S. Primitives-Based Evaluation and Estimation of Emotions in Speech. *Speech Communication*, vol. 49:pp. 787–800 [2007].

[196] McFee, B. Statistical Methods for Scene and Event Classification. In T. Virtanen, M. D. Plumbley and D. P. W. Ellis (eds.), *Computational Analysis of Sound Scenes and Events*, chap. 5, pp. 103–146. Springer International Publishing, 1 edn. [2018].

[197] Eronen, A. J., Peltonen, V. T., Tuomi, J. T., Klapuri, A. P., Fagerlund, S., Sorsa, T., Lorho, G. and Huopaniemi, J. Audio-Based Context Recognition. *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, vol. 14(1):pp. 321–329 [2006]. doi:10.1109/TSA.2005.854103.

[198] Matlab. MATLAB [2019]. *https://uk.mathworks.com/discovery/what-is-matlab.html* [Accessed: 2018-08-27].

[199] Zhivomirov, H. Sound Analysis with Matlab Implementation [2017]. *https://uk.mathworks.com/matlabcentral/fileexchange/38837-sound-analysis-with-matlab-implementation* [Accessed: 2018-06-22].

[200] Giannakopoulos, T. and Pikrakis, A. *Introduction to Audio Analysis: A MATLAB Approach*. Oxford, UK: Elsevier Academic Press [2014]. doi:10.1016/C2012-0-03524-7.

[201] MIRtoolbox [2019]. *https://www.jyu.fi/hytk/fi/laitokset/mutku/en/research/materials/mirtoolbox* [Accessed: 2018-08-27].

[202] Ligges, U. Package 'tuneR' [2018]. *https://cran.r-project.org/web/packages/tuneR/tuneR.pdf* [Accessed: 2018-08-27].

[203] Anikin, A. 'soundgen': 'R' audio feature extraction package. [2018]. *https://cran.r-project.org/web/packages/soundgen/soundgen.pdf* [Accessed: 2018-08-27].

[204] Giannakopoulos, T. PyAudioAnalysis: An open-source python library for audio signal analysis. *PLoS ONE*, vol. 10(12)(e0144610):pp. 1–17 [2015]. doi:10.1371/journal.pone.0144610.

[205] R Stats Environment. R: The R Project for Statistical Computing [2019]. *https://www.r-project.org/* [Accessed: 2018-08-27].

[206] Python. Python.org [2019]. *https://www.python.org/* [Accessed: 2018-08-27].

[207] MARSYAS. MARSYAS: Music Analysis, Retrieval and Synthesis for Audio Signals [2019]. *http://marsyas.info/index.html* [Accessed: 2018-08-27].

[208] PRAAT. PRAAT: Doing Phonetics by Computer [2019]. *http://www.fon.hum.uva.nl/praat/* [Accessed: 2018-08-27].

[209] Eyben, F., Wöllmer, M. and Schuller, B. Opensmile. In *Proceedings of the International Conference on Multimedia - MM '10*, p. 1459. New York, New York, USA: ACM Press [2010]. doi:10.1145/1873951.1874246.

[210] James, G., Witen, D., Hastie, T. and Tibshirani, R. *An Introduction to Statistical Learning with Applications in R*. New York: Springer [2014]. doi:10.1016/j.peva.2007.06.006.

[211] Özseven, T. A Novel Feature Selection Method for Speech Emotion Recognition. *Applied Acoustics*, vol. 146:pp. 320–326 [2019]. doi:10.1016/J.APACOUST.2018.11.028.

[212] Breiman, L. Random Forests. *Machine Learning*, vol. 45:pp. 5–32 [2001].

[213] Himberg, J., Mantyjarvi, J. and Korpipaa, P. Using PCA and ICA for Exploratory Data Analysis in Situation Awareness. In *International Conference on Multisensor Fusion and Integration for Intelligent Systems, 2001 (MFI 2001)*, pp. 127–131. Baden-Baden, Germany; 20 - 22 August: IEEE [2001].

[214] Hastie, T., Tibshirani, R. and Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, second edi edn. [2009]. doi:10.1007/b94608.

[215] Mesaros, A., Heittola, T. and Ellis, D. Datasets and Evaluation. In T. Virtanen, M. D. Plumbley and D. P. W. Ellis (eds.), *Computational Analysis of Sound Scenes and Events*, chap. 6, pp. 147–179. Springer International Publishing, 1 edn. [2018].

[216] Mitrovic, D., Zeppelzauer, M. and Breiteneder, C. Features for Content-Based Audio Retrieval. *Advances in Computers*, vol. 78:pp. 71–150 [2010].

[217] Widmer, G., Dixon, S., Knees, P., Pampalk, E. and Pohle, T. From Sound to 'Sense' via Feature Extraction and Machine Learning: Deriving High-Level Descriptors for Characterising Music. In P. Polotti and D. Rocchesso (eds.), *Sound to Sense, Sense to Sound A State of the Art in Sound and Music Computing*, chap. 5. Berlin: Logos Verlag [2008].

[218] Mijić, M., Mašović, D., Petrović, M. and Šumarac-Pavlović, D. Statistical Properties of Music Signals. In *Proceedings of the 126th Convention of the Audio Engineering Society (2009)*. Munich, Germany; May 7th - 10th [2009].

[219] Aletta, F., Axelsson, Ö. and Kang, J. Towards Acoustic Indicators for Soundscape Design. In *Forum Acusticum*, c, pp. 1–6. Krakow, Poland; 7 - 12 September [2014]. doi:10.13140/2.1.1461.3769.

[220] Gubka, R. and Kuba, M. A Comparison of Audio Features for Elementary Sound Based Audio Classification. *International Conference on Digital Technologies 2013, DT 2013*, (1):pp. 14–17 [2013]. doi:10.1109/DT.2013.6566278.

[221] Bountourakis, V., Vrysis, L. and Papanikolaou, G. Machine Learning Algorithms for Environmental Sound Recognition. In *Proceedings of the Audio Mostly 2015 on Interaction With Sound - (AM15)*, pp. 1–7. Thessaloniki, Greece; October 07-09 [2015]. doi:10.1145/2814895.2814905.

[222] Panagiotakis, C. and Tziritas, G. A Speech/Music Discriminator based on RMS and Zero-crossings. *IEEE Transactions on multimedia*, vol. 7(1):pp. 155–166 [2005].

[223] Torija, A. J., Ruiza, D. P. F., Ramos-Ridao, B., Ruiz, D. P. and Ramos-Ridao, A. F. Application of a Methodology for Categorizing and Differentiating Urban Soundscapes using Acoustical Descriptors and Semantic-differential Attributes. *The Journal of the*

Acoustical Society of America, vol. 134(1 Pt. 2):pp. 791–802 [2013]. doi:10.1121/1.4807804.

[224] Jiang, H., Bai, J., Zhang, S. and Xu, B. SVM-based audio scene classification. In *Proceedings of 2005 IEEE International Conference on Natural Language Processing and Knowledge Engineering, IEEE NLP-KE'05*, vol. Oct 30 - N, pp. 131–136. Wuhan, China [2005]. doi:10.1109/NLPKE.2005.1598721.

[225] Chen, G. B. and Zhao, Y. Z. Study on the Correlation Between the Temporal Factors and the Soundscape Excerpts Assessment. *Acta Acustica united with Acustica*, vol. 102(4):pp. 663–674 [2016].

[226] Pace, F., Benard, F., Glotin, H., Adam, O. and White, P. Subunit Definition and Analysis for Humpback Whale Call Classification. *Applied Acoustics*, vol. 71(11):pp. 1107–1112 [2010].

[227] Yang, M. and Kang, J. Psychoacoustical Evaluation of Natural and Urban Sounds in Soundscapes. *The Journal of the Acoustical Society of America*, vol. 134(1):pp. 840–851 [2013]. doi:10.1121/1.4807800.

[228] Ogg, M., Slevc, L. R. and Idsardi, W. J. The Time Course of Sound Category Identification: Insights from Acoustic Features. *The Journal of the Acoustical Society of America*, vol. 142(6):p. 3459 [2017].

[229] Fagerlund, S. Bird Species Recognition Using Support Vector Machines. *EURASIP Journal on Advances in Signal Processing*, vol. 2007(1):p. 38637 [2007]. doi:10.1155/2007/38637.

[230] Tzanetakis, G., Essl, G. and Cook, P. Automatic Musical Genre Classification Of Audio Signals. *IEEE Transactions on Speech and Audio Processing*, vol. 10(5):pp. 293–302 [2002].

[231] Yang, M. and Kang, J. Identification of Sound Sources in Soundscape using Acoustic, Psychoacoustic, and Music Parameters. *The Journal of the Acoustical Society of America*, vol. 136(4):p. 2164 [2014]. doi:10.1121/1.4899831.

[232] Shepard, R. N. Circularity in Judgments of Relative Pitch. *The Journal of the Acoustical Society of America*, vol. 36(12):pp. 2346–2353 [1964].

[233] Bartsch, M. A. and Wakefield, G. H. Audio Thumbnailing of Popular Music using Chroma-based Representations. *IEEE Transactions on multimedia*, vol. 7(1):pp. 96–104 [2005].

[234] Müller, M., Kurth, F. and Clausen, M. Audio Matching via Chroma-Based Statistical Features. In *6th International Conference on Music Information Retrieval (ISMIR2005)*, vol. 2005. London, UK; 11 - 15 September: ISMIR [2005].

[235] Collins, T., Tillmann, B., Barrett, F. S., Delbé, C. and Janata, P. A Combined Model of Sensory and Cognitive Representations Underlying Tonal Expectations in Music: From Audio Signals to Behavior. *Psychological Review*, vol. 121(1):pp. 33–65 [2014].

[236] Pikrakis, A., Giannakopoulos, T. and Theodoridis, S. A Speech/Music Discriminator of Radio Recordings Based on Dynamic Programming and Bayesian Networks. *IEEE Transactions on Multimedia*, vol. 10(5):pp. 846–857 [2008]. doi:10.1109/TMM.2008.922870.

[237] DCASE 2017 Results. Acoustic Scene Classification Results - DCASE2017 [2017]. *http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-acoustic-scene-classification-results* [Accessed: 2018-08-27].

[238] DCASE 2018 Results. DCASE 2018 Challenge [2018]. *http://dcase.community/challenge2018/index* [Accessed: 2018-06-08].

[239] DCASE 2019 Results. DCASE 2019 Challenge [2019]. *http://dcase.community/challenge2019/task-acoustic-scene-classification-results-a{#}Chen2019* [Accessed: 2020-02-08].

[240] Noll, A. M. Short-Time Spectrum and & 'Cepstrum' Techniques for Vocal-Pitch Detection. *Journal of the Acoustical Society of America*, vol. 36(2):pp. 296–302 [1964]. doi:10.1121/1.2143271.

[241] Noll, A. M. Cepstrum Pitch Determination. *The Journal of the Acoustical Society of America*, vol. 41(2):pp. 293–309 [1967]. doi:10.1121/1.1910339.

[242] Barchiesi, D., Giannoulis, D. D., Stowell, D. and Plumbley, M. D. Acoustic Scene Classification: Classifying Environments from the Sounds they Produce. *IEEE Signal Processing Magazine*, vol. 32(3):pp. 16–34 [2015]. doi:10.1109/MSP.2014.2326181.

[243] Tyagi, H., Hegde, R. M., Murthy, H. A. and Prabhakar, A. Automatic Identification of Bird Calls using Spectral Ensemble Average Voice Prints. In *Proceedings of the IEEE 2006 14th European Signal Processing Conference*, pp. 1–5. Florence, Italy; September 4-8 [2006].

[244] Chu, S., Narayanan, S. and Kuo, J. Environmental Sound Recognition With Time–Frequency Audio Features. *IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING*, vol. 17(6) [2009]. doi:10.1109/TASL.2009.2017438. URL http://mcl.usc.edu/wp-content/uploads/2014/01/200908-Environmental-sound-recognition-with-time-frequency-audio-features.pdf.

[245] Cowling, M. and Sitte, R. Comparison of Techniques for Environmental Sound Recognition. *Pattern Recognition Letters*, vol. 24(15):pp. 2895–2907 [2003]. doi:10.1016/S0167-8655(03)00147-8.

[246] Davis, S. and Mermelstein, P. Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28(4):pp. 357–366 [1980]. doi:10.1109/TASSP.1980.1163420.

[247] Serizel, R., Bisot, V., Essid, S. and Richard, G. Acoustic Features for Environmental Sound Analysis. In T. Virtanen, M. D. Plumbley and D. P. W. Ellis (eds.), *Computational Analysis of Sound Scenes and Events*, chap. 4, pp. 71–101. Springer International Publishing, 1 edn. [2018].

[248] Lyon, R. F. *Human and Machine Hearing: Extracting Meaning from Sound*. Cambridge, UK: Cambridge University Press [2017].

[249] Salamon, J. and Bello, J. P. Unsupervised Feature Learning for Urban Sound Classification. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 171–175. Brisbane, Australia; April 19-24: IEEE [2015]. doi:10.1109/ICASSP.2015.7177954.

[250] Salamon, J. and Bello, J. P. Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification. *IEEE Signal Processing Letters*, vol. 24(3):pp. 279–283 [2017].

[251] Dennis, J. W. *Sound Event Recognition in Unstructured Environments using Spectrogram Image Processing*. Doctoral, Nanyang Technological University [2014].

[252] Dennis, J., Tran, H. D. and Li, H. Combining Robust Spike Coding with Spiking Neural Networks for Sound Event Classification. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 176–180. Brisbane, Australia; April 19-24: IEEE [2015].

[253] Shao, Y., Srinivasan, S., Jin, Z. and Wang, D. A Computational Auditory Scene Analysis System for Speech Segregation and Robust Speech Recognition. *Computer Speech & Language*, vol. 24(1):pp. 77–93 [2010].

[254] Socoro, J. C., Ribera, G., Sevillano, X. and Alías, F. Development of an Anomalous Noise Event Detection Algorithm for Dynamic Road Traffic Noise Mapping. In *22nd International Congress on Sound and Vibration (ICSV22)*. Florence, Italy; 12-16 July [2015].

[255] Sawata, R., Ogawa, T. and Haseyama, M. Human-centered Favorite Music Estimation: EEG-based Extraction of Audio Features Reflecting Individual Preference. In *Proceedings of the 2015 IEEE International Conference on Digital Signal Processing (DSP)*, pp. 818–822. Singapore; 21-24 July, 2015 [2015].

[256] McKinney, M. and Breebaart, J. Features for Audio and Music Classification. In *4th International Conference on Music Information Retrieval (ISMIR2003)*. Baltimore, USA; 26 - 30 October [2003].

[257] Bai, L., Hu, Y., Lao, S., Chen, J. and Wu, L. Feature Analysis and Extraction for Audio Automatic Classification. In *2005 IEEE International Conference on Systems, Man and Cybernetics*, vol. 1, pp. 767–772. Hawaii, USA; October 10-12 [2005]. doi:10.1109/ICSMC.2005.1571239.

[258] Hermansky, H. Perceptual Linear Predictive (PLP) Analysis of Speech. *The Journal of the Acoustical Society of America*, vol. 87(4):pp. 1738–1752 [1990]. doi:10.1121/1.399423.

[259] Ntalampiras, S. Audio Pattern Recognition of Baby Crying Sound Events. *J. Audio Eng. Soc*, vol. 63(5):pp. 358–369 [2015].

[260] Mao, S., Ching, P. C. and Lee, T. Deep Learning of Segment-Level Feature Representation with Multiple Instance Learning for Utterance-Level Speech Emotion Recognition. In *Interspeech 2019*. Graz, Austria; 15-19 September [2019]. doi: 10.21437/Interspeech.2019-1968.

[261] Abdoli, S., Cardinal, P. and Koerich, A. L. End-to-End Environmental Sound Classification using a 1D Convolutional Neural Network. *Preprint* [2019].

[262] Yang, L. and Su, F. Auditory Context Classification using Random Forests. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2012*, pp. 2349–2352. Kyoto, Japan; March 25-30: IEEE [2012].

[263] Cai, R., Lu, L., Hanjalic, A., Zhang, H. J. and Cai, L. H. A Flexible Framework for Key Audio Effects Detection and Auditory Context Inference. *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14(3):pp. 1026–1038 [2006]. doi: 10.1109/TSA.2005.857575.

[264] Malfante, M., Mars, J. I., Dalla Mura, M. and Gervaise, C. Automatic Fish Classification. *Journal of the Acoustical Society of America*, vol. 143(5):pp. 2834–2846 [2018]. doi:10.1121/1.5036628.

[265] Esfahanian, M., Zhuang, H. and Erdol, N. Sparse Representation for Classification of Dolphin Whistles by Type. *The Journal of the Acoustical Society of America*, vol. 136(1):pp. EL1–EL7 [2014]. doi:10.1121/1.4881320.

[266] Han, N. C., Muniandy, S. V. and Dayou, J. Acoustic Classification of Australian Anurans based on Hybrid Spectral-entropy Approach. *Applied Acoustics*, vol. 72(9):pp. 639–645 [2011].

[267] Wang, J.-C., Wang, J.-F., He, K. W. and Hsu, C.-S. Environmental Sound Classification using Hybrid SVM/KNN Classifier and MPEG-7 Audio Low-level Descriptor. In *International Joint Conference on Neural Networks, 2006. IJCNN'06.*, pp. 1731–1735. IEEE [2006].

[268] Ghahramani, Z. An Introduction to Hidden Markov Models and Bayesian Networks. *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15(1):pp. 9–42 [2001].

[269] Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. and Kingsbury, B. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, vol. 29(6):pp. 82–97 [2012]. doi:10.1109/MSP.2012.2205597.

[270] Dahl, G. E., Yu, D., Deng, L. and Acero, A. Context-dependent Pre-trained Deep Neural Networks for Large-vocabulary Speech Recognition. *IEEE Transactions on audio, speech, and language processing*, vol. 20(1):pp. 30–42 [2012].

[271] Sharan, R. V. and Moir, T. J. An Overview of Applications and Advancements in Automatic Sound Recognition. *Neurocomputing*, vol. 200:pp. 22–34 [2016]. doi: 10.1016/j.neucom.2016.03.020.

[272] Gencoglu, O., Virtanen, T. and Huttunen, H. Recognition of Acoustic Events using Deep Neural Networks. In *Proceedings of the 22nd European Signal Processing Conference (EUSIPCO)*, pp. 506–510. Lisbon, Portugal; September 1-5: IEEE [2014].

[273] Schroder, J., Moritz, N., Anemuller, J., Goetze, S. and Kollmeier, B. Classifier Architectures for Acoustic Scenes and Events: Implications for DNNs, TDNNs, and Perceptual Features from DCASE 2016. *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 25(6):pp. 1304–1314 [2017]. doi:10.1109/TASLP.2017.2690569.

[274] Pietila, G. and Lim, T. C. Intelligent Systems Approaches to Product Sound Quality Evaluations - A Review. *Applied Acoustics*, vol. 73(10):pp. 987–1002 [2012]. doi:10.1016/j.apacoust.2012.04.012.

[275] Härmä, A., Park, M. and Kohlrausch, A. Data-driven modeling of the spatial sound experience. In *Proceedings of the 136th AES Convention, Berlin, Germany*, pp. April 26–29. Berlin, Germany: AES [2014].

[276] Gozalo, G. R., Carmona, J. T., Barrigón Morillas, J. M., Vílchez-Gómez, R. and Gómez Escobar, V. Relationship between Objective Acoustic Indices and Subjective Assessments for the Quality of Soundscapes. *Applied Acoustics*, vol. 97:pp. 1–10 [2015]. doi:10.1016/j.apacoust.2015.03.020.

[277] McLoughlin, I., Zhang, H., Xie, Z., Song, Y. and Xiao, W. Robust Sound Event Classification Using Deep Neural Networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23(3):pp. 540–552 [2015]. doi:10.1109/TASLP.2015.2389618.

[278] Wang, J.-C., Wang, J.-F., Lin, C.-B., Jian, K.-T. and Kuok, W.-H. Content-Based Audio Classification Using Support Vector Machines and Independent Component Analysis [2006]. doi:10.1109/ICPR.2006.407.

[279] Zeiler, M. D., Taylor, G. W. and Fergus, R. Adaptive Deconvolutional Networks for Mid and High Level Feature Learning. In *IEEE International Conference on Computer Vision (ICCV), 2011*, pp. 2018–2025. Barcelona, Spain; November 6 - 13: IEEE [2011].

[280] Ciresan, D. C., Meier, U., Gambardella, L. M. and Schmidhuber, J. Deep, Big, Simple Neural Nets for Handwritten Digit Recognition. *Neural Computation*, vol. 22(12):pp. 3207–3220 [2010].

[281] Jones, N. Computer Science: The Learning Machines. *Nature News*, vol. 505(7482):p. 146 [2014].

[282] Efrati, A. How 'Deep Learning' Works at Apple [2013]. *https://www.theinformation.com/articles/How-Deep-Learning-Works-at-Apple-Beyond* [Accessed: 2018-07-05].

[283] Kirk, J. Universities, IBM Join Forces to Build a Brain-like Computer | PCWorld [2013]. URL https://www.pcworld.com/article/2051501/universities-join-ibm-in-cognitive-computing-research-project.html.

[284] Rosenblatt, F. The Perceptron: a Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, vol. 65(6):p. 386 [1958].

[285] Goodfellow, I., Bengio, Y. and Courville, A. *Deep Learning*. MIT Press [2016]. URL http://www.deeplearningbook.org/.

[286] Duchi, J., Hazan, E. and Singer, Y. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, vol. 12:pp. 2121–2159 [2011].

[287] Kingma, D. P. and Ba, J. L. Adam: A Method for Stochastic Optimization. In *Proc. 3rd Int. Conf. Learn. Representations*. San Diego, USA; May 7-9 [2015].

[288] Bengio, Y., Simard, P. and Frasconi, P. Learning Long-Term Dependencies with Gradient Descent is Difficult. *IEEE Transactions on Neural Networks*, vol. 5(2):pp. 157–166 [1994]. doi:10.1109/72.279181.

[289] Parascandolo, G., Huttunen, H. and Virtanen, T. Recurrent Neural Networks for Polyphonic Sound Event Detection in Real Life Recordings. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6440–6444. Shanghai, China; March 20-25 [2016]. doi:10.1109/ICASSP.2016.7472917.

[290] LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P. Gradient-based Learning Applied to Document Recognition. *Proceedings of the IEEE*, vol. 86(11):pp. 2278–2324 [1998].

[291] Hershey, S., Chaudhuri, S., Ellis, D. P. W., Gemmeke, J. F., Jansen, A., Moore, R. C., Plakal, M., Platt, D., Saurous, R. A., Seybold, B., Slaney, M., Weiss, R. J. and Wilson, K. CNN Architectures for Large-scale Audio Classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 131–135. New Orleans, LA, USA; March 5 - 9: IEEE [2017].

[292] Ibrahim, A. K., Zhuang, H., Chérubin, L. M., Schärer-Umpierre, M. T. and Erdol, N. Automatic Classification of Grouper Species by their Sounds using Deep Neural Networks. *Citation: The Journal of the Acoustical Society of America*, vol. 144:p. 196 [2018]. doi:10.1121/1.5054911.

[293] Sharan, R. V. and Moir, T. J. Acoustic Event Recognition using Cochleagram Image and Convolutional Neural Networks. *Applied Acoustics*, vol. 148:pp. 62–66 [2019].

[294] Dai, W., Dai, C., Qu, S., Li, J. and Das, S. Very Deep Convolutional Neural Networks for Raw Waveforms. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 421–425. New Orleans, LA, USA; March 5 - 9 [2017]. doi:10.1109/ICASSP.2017.7952190.

[295] Kumar, A., Khadkevich, M. and Fugen, C. Knowledge Transfer from Weakly Labeled Audio using Convolutional Neural Network for Sound Events and Scenes. *arXiv preprint*, vol. arXiv:1711 [2017].

[296] Krizhevsky, A., Sutskever, I. and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, pp. 1097–1105. Lake Tahoe, NV, USA; 3 - 6 December: Curran Associates Inc. [2012].

[297] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 2818–2826. Las Vegas, NV, USA; June 27 - 30: IEEE Computer Society [2016]. doi:10.1109/CVPR.2016.308.

[298] He, K., Zhang, X., Ren, S. and Sun, J. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, pp. 770–778. Las Vegas, NV, USA; June 27 - 30: IEEE Computer Society [2016]. doi:10.1109/CVPR.2016.90.

[299] Mesaros, A., Heittola, T., Benetos, E., Foster, P., Lagrange, M., Virtanen, T. and Plumbley, M. D. Detection and Classification of Acoustic Scenes and Events: Outcome of the DCASE 2016 Challenge. *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 26(2) [2018]. doi:10.1109/TASLP.2017.2778423.

[300] Han, Y. and Park, J. Convolutional Neural Networks with Binaural Representations and Background Subtraction for Acoustic Scene Classification. In *Detection and Classification of Acoustic Scenes and Events (DCASE 2017)*. Munich, Germany; 16th November: DCASE2017 Challenge [2017].

[301] Hayashi, T., Watanabe, S., Toda, T., Hori, T., Roux, J. L. and Takeda, K. Duration-Controlled LSTM for Polyphonic Sound Event Detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25(11):pp. 2059–2070 [2017]. doi:10.1109/TASLP.2017.2740002.

[302] Krstulovic, S. Audio Event Recognition in the Smart Home. In T. Virtanen, M. D. Plumbley and D. P. W. Ellis (eds.), *Computational Analysis of Sound Scenes and Events*, chap. 12, pp. 335–371. Springer International Publishing, 1 edn. [2018].

[303] Settles, B. Active Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 6(1):pp. 1–114 [2012].

[304] Dagan, I. and Engelson, S. P. Committee-Based Sampling For Training Probabilistic Classifiers. In *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 150–157. Tahoe City, CA, USA; 9 - 12 July: Elsevier [1995]. doi: 10.1016/b978-1-55860-377-6.50027-x.

[305] Abe, N. and Mamitsuka, H. Query Learning Strategies Using Boosting and Bagging. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998)*, pp. 1–9. Madison, WI, USA; 24 - 27 July [1998].

[306] Seung, H. S., Opper, M. and Sompolinsky, H. Query by Committee. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pp. 287–294. Pittsburgh, PA, USA; July 27-29: ACM [1992]. doi:10.1145/130385.130417.

[307] Melville, P. and Mooney, R. J. Diverse Ensembles for Active Learning. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pp. 74—-. Banff, Canada; 4 - 8 July: ACM [2004]. doi:10.1145/1015330.1015385.

[308] Roy, N. and McCallum, A. Toward Optimal Active Learning Through Sampling Estimation of Error Reduction. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pp. 441–448. San Francisco, CA, USA; June 28 - July 1: Morgan Kaufmann Publishers Inc. [2001].

[309] Han, W., Coutinho, E., Ruan, H., Li, H., Schuller, B., Yu, X. and Zhu, X. Semi-supervised Active Learning for Sound Classification in Hybrid Learning Environments. *PloS one*, vol. 11(9) [2016].

[310] Hu, R., Delany, S. J. and Mac Namee, B. EGAL: Exploration Guided Active Learning for TCBR. In *Proceedings of ICCBR*, pp. 156–170. Alessandria, Italy; 19-22 July [2010]. doi:10.1007/978-3-642-14274-1_13.

[311] Mandel, M. I., Poliner, G. E. and Ellis, D. P. W. Support Vector Machine Active Learning for Music Retrieval. *Multimedia Systems*, vol. 12(1):pp. 3–13 [2006]. doi: 10.1007/s00530-006-0032-2.

[312] Chang, E. Y., Tong, S., Goh, K. and Chang, C.-W. Support Vector Machine Concept-Dependent Active Learning for Image Retrieval. *IEEE Transactions on Multimedia*, vol. 2:pp. 1–35 [2005].

[313] Gulluni, S., Essid, S., Buisson, O. and Richard, G. Interactive Classification of Sound Objects for Polyphonic Electro-Acoustic Music Annotation. In *Audio Engineering Society Conference: 42nd International Conference: Semantic Audio*. Ilmenau, Germany; 22 - 24 July [2011].

[314] Zhang, Z. and Schuller, B. Active Learning by Sparse Instance Tracking and Classifier Confidence in Acoustic Emotion Recognition. In *13th Annual Conference of the International Speech Communication Association (INTERSPEECH 2012)*, pp. 362–365. Portland, OR, USA; September 9 - 13 [2012].

[315] Aggarwal, C. C., Kong, X., Gu, Q., Han, J. and Yu, P. S. Active Learning: A Survey. In *Data Classification*, chap. 22, pp. 572–605. Chapman and Hall/CRC [2014].

[316] Zhao, S., Heittola, T. and Virtanen, T. An Active Learning Method Using Clustering and Committee-Based Sample Selection for Sound Event Classification. In *16th International Workshop on Acoustic Signal Enhancement (IWAENC)*. Tokyo, Japan; 17 - 20 September [2018]. doi:10.1109/IWAENC.2018.8521336.

[317] McFee, B., Humphrey, E. J. and Bello, J. A Software Framework for Musical Data Augmentation. In F. Wiering and M. Muller (eds.), *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015*, Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, pp. 248–254. Malaga, Spain; 26 - 30 October: International Society for Music Information Retrieval [2015].

[318] Mikołajczyk, A. and Grochowski, M. Data Augmentation for Improving Deep Learning in Image Classification Problem. In *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pp. 117–122. Swinoujscie, Poland; 9 - 12 May [2018]. doi: 10.1109/IIPHDW.2018.8388338.

[319] Kaggle. Kaggle: Your Home for Data Science [2019]. *https://www.kaggle.com/* [Accessed: 2018-08-27].

[320] LeCun, Y., Weinberger, K., Coruana, R. and Simardi, P. The Great AI Debate - NIPS2017 - Yann LeCun, Killian Weinberger, Rich Coruana, Patrice Simardi - YouTube [2017]. URL https://www.youtube.com/watch?v=93Xv8vJ2acI{&}feature=youtu.be.

[321] Rahimi, A. and Recht, B. An Addendum to Alchemy [2017]. URL http://www.argmin.net/2017/12/11/alchemy-addendum/, *http://www.argmin.net/2017/12/11/alchemy-addendum/* [Accessed: 2018-08-07].

[322] Haahr, M. and Haahr, S. Random.org [2018]. *https://www.random.org/media/* [Accessed: 2018-01-04].

[323] Giannakopoulos, T. Matlab Audio Analysis Library [2014]. *https://uk.mathworks.com/matlabcentral/fileexchange/45831-matlab-audio-analysis-library?s{_}tid=prof{_}contriblnk* [Accessed: 2019-01-30].

[324] Muller, A. C. and Guido, S. *Introduction to Machine Learning with Python*. Sebastopol, United States: O'Reilly Media [2017]. doi:10.1017/CBO9781107415324.004.

[325] Cunningham, P., Cord, M. and Delany, S. J. Supervised Learning. In M. Cord and P. Cunningham (eds.), *Machine Learning Techniques for Multimedia*, pp. 21–49. Berlin, Heidelberg: Springer Berlin Heidelberg [2008]. doi:10.1007/978-3-540-75171-7_2.

[326] Cunningham, P. Dimension Reduction. In M. Cord and P. Cunningham (eds.), *Machine Learning Techniques for Multimedia.*, chap. 4, pp. 1–24. Dublin: Springer, Berlin, Heidelberg [2008].

[327] Kohavi, R. and John, G. H. Wrappers for Feature Subset Selection. *Artificial Intelligence*, vol. 97(1-2):pp. 273–324 [1997]. doi:10.1016/S0004-3702(97)00043-X.

[328] Corder, G. W. and Foreman, D. I. *Nonparametric Statistics for Non-Statisticians: A Step-by-step Approach*. Hoboken, NJ, USA: John Wiley & Sons, Inc. [2009]. doi:10.1002/9781118165881.

[329] Tong, S. and Chang, E. Support Vector Machine Active Learning for Image Retrieval. In *Proceedings of the 9th ACM international Conference on Multimedia*, pp. 107–118. Ottawa, ON, Canada; September 30 - October 05: ACM [2001].

[330] Qian, K., Zhang, Z., Baird, A. and Schuller, B. Active Learning for Bird Sound Classification via a Kernel-based Extreme Learning Machine. *The Journal of the Acoustical Society of America*, vol. 142(4):pp. 1796–1804 [2017]. doi:10.1121/1.5004570.

[331] O'Neill, J., Delany, S. J. and Macnamee, B. Model-free and Model-based Active Learning for Regression. In P. Angelov, A. Gegov, J. C. and Q. Shen (eds.), *Advances in Intelligent Systems and Computing*, vol. 513, pp. 375–386. Springer Verlag [2017]. doi:10.1007/978-3-319-46562-3_24.

[332] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, É. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, vol. 12(Oct):pp. 2825–2830 [2011].

[333] Jones, E., Oliphant, T., Peterson, P. and Others. SciPy: Open source scientific tools for Python [2001]. *http://www.scipy.org/* [Accessed: 2019-11-13].

[334] Mckinney, W. Data Structures for Statistical Computing in Python. In *PROC. OF THE 9th PYTHON IN SCIENCE CONF. (SCIPY 2010)*, p. 51. Austin, USA; June 28 - July 3 [2010].

[335] Mcfee, B., Raffel, C., Liang, D., Ellis, D. P. W., Mcvicar, M., Battenberg, E. and Nieto, O. librosa: Audio and Music Signal Analysis in Python. In *Proceedings of the 14th Python in Science Conference (SciPy 2015)*. Austin, USA; July 6-12 [2015].

[336] Acevedo, M. A., Corrada-Bravo, C. J., Corrada-Bravo, H., Villanueva-Rivera, L. J. and Aide, T. M. Automated Classification of Bird and Amphibian Calls using Machine Learning: A Comparison of Methods. *Ecological Informatics*, vol. 4(4):pp. 206–214 [2009]. doi:https://doi.org/10.1016/j.ecoinf.2009.06.005.

[337] Yenigalla, P., Kumar, A., Tripathi, S., Singh, C., Kar, S. and Vepa, J. Speech Emotion Recognition Using Spectrogram & Phoneme Embedding. In *Interspeech 2018*. Hyderabad, India; 2-6 September [2018]. doi:10.21437/Interspeech.2018-1811.

[338] Espi, M., Fujimoto, M., Kinoshita, K. and Nakatani, T. Exploiting Spectro-temporal Locality in Deep Learning Based Acoustic Event Detection. *Eurasip Journal on Audio, Speech, and Music Processing* [2015]. doi:10.1186/s13636-015-0069-2.

[339] Torija, A. J., Ruiz, D. P. and Ramos-Ridao, Á. F. A Tool for Urban Soundscape Evaluation Applying Support Vector Machines for Developing a Soundscape Classification Model. *Science of the Total Environment*, vol. 482-483(1):pp. 440–451 [2014]. doi:10.1016/j.scitotenv.2013.07.108.

[340] Nisbet, R., Miner, G., Yale, K., Nisbet, R., Miner, G. and Yale, K. Advanced Algorithms for Data Mining. In *Handbook of Statistical Analysis and Data Mining Applications*, pp. 149–167. Academic Press [2018]. doi:10.1016/B978-0-12-416632-5.00008-6.

[341] Hu, R., Mac Namee, B. and Delany, S. J. Off to a Good Start: Using Clustering to Select the Initial Training set in Active Learning. In *Twenty-Third International FLAIRS Conference*. Florida; 19-21 May [2010]. doi:10.21427/D7Q89W.

[342] Field, A. *Discovering Statistics Using SPSS*, vol. 58. London, UK: SAGE Publications, 3rd edn. [2009]. doi:10.1234/12345678.

[343] Demšar, J. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, vol. 7:pp. 1–30 [2006].

[344] Imagenet. ImageNet [2019]. *http://www.image-net.org/* [Accessed: 2019-09-23].

[345] Linguistic Data. Linguistic Data Consortium [2019]. *https://catalog.ldc.upenn.edu/* [Accessed: 2019-09-23].

[346] Million Song Dataset [2019]. URL http://millionsongdataset.com/, *http://millionsongdataset.com/* [Accessed: 2019-09-23].

[347] Adobe Audition. Adobe Audition: Audio Recording, Editing, and Mixing Software [2019]. *https://www.adobe.com/ie/products/audition.html* [Accessed: 2019-11-13].

[348] Dietterich, T. G. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, vol. 10(7):pp. 1895–1923 [1998]. doi:10.1162/089976698300017197.

[349] Virtanen, T., Mesaros, A. and Heittola, T. [2019]. *http://dcase.community/challenge2019/* [Accessed: 2020-01-06].

[350] Sabour, S., Frosst, N. and Hinton, G. E. Dynamic Routing Between Capsules. In *31st Conference on Neural Information Processing Systems (NIPS 2017)*. Long Beach, USA; 4-9 December [2017].

[351] Vesperini, F., Gabrielli, L., Principi, E. and Squartini, S. Polyphonic Sound Event Detection by Using Capsule Neural Networks. *IEEE Journal of Selected Topics in Signal Processing*, vol. 13(2):pp. 310–322 [2019]. doi:10.1109/JSTSP.2019.2902305. URL https://ieeexplore.ieee.org/document/8654643/.

[352] Iqbal, T., Xu, Y., Kong, Q. and Wang, W. Capsule Routing for Sound Event Detection. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 2255–2259. Rome, Italy; September 3-7: IEEE [2018]. doi:10.23919/EUSIPCO.2018.8553198.

Appendix A

Computer Code

A.1 Experiment 1

A.1.1 Experiment 1 - R Data Exploration Code

R code written to analyse the findings of Experiment 1 and produce plots.

```
# Read in dataset
data_demographics <- read.csv('/Volumes/GoogleDrive/My Drive/DIT-PhD/EXP_2/R_Analysis/soundFeatures.csv',
                              header=TRUE, stringsAsFactors=TRUE)

# Use this to read the dataset into R memory so you can refer directly to variables
attach(data_demographics)

class(gender)
levels(gender)
names(data_demographics)
summary(data_demographics)
head(data_demographics)
```

```
13
14 summ_data <- summary(data_demographics)
15 View(summ_data)
16
17 sound_ratings <- data_demographics[,4:43]
18 View(sound_ratings)
19
20 dim(sound_ratings)
21 head(data_demographics)
22 summary(data_demographics)
23
24 # Mean of female scores for Thunderstorm
25 mean(Thunderstorm[gender=="female"])
26
27 # Mean of male scores for Thunderstorm
28 mean(Thunderstorm[gender=="male"])
29
30 # Mean of all scores for Thunderstorm
31 mean(Thunderstorm)
32
33 # Subset Female data - everything in data_demographics where gender =
       female
34 FemData <- data_demographics[gender=="female", ]
35
36 # Subset Male data
37 MaleData <- data_demographics[gender=="male", ]
38
39 dim(FemData)
40 dim(MaleData)
41
42 # calculate SDs for all data by columns
43 allSD <- apply(data_demographics, 2, sd)
```

```
# join SDs to other summary stats for all tests
summ_data <- rbind(summ_data, allSD)
View(summ_data)
head(summ_data)

# Repeat for Female respondents
sdFem <- apply(FemData, 2, sd)
View(sdFem)
names(FemData)
summFem <- rbind(summFem, sdFem)
View(summFem)

# Repeat for Male respondents
sdMale <- apply(MaleData, 2, sd)
View(sdMale)
summMal <- rbind(summMal, sdMale)
View(summMal)

# Join all with female and male subsets for comparison
summStats <- rbind(summ_data, summFem, summMal)
View(summStats)

# Export this table to .csv
write.csv(summStats, "summStats.csv")

# Calculate mean sound rating
sound_rating_Means <- apply(sound_ratings, 2, mean)
head(sound_rating_Means)

# Calculate standard deviation of sound ratings
sound_rating_SDs <- apply(sound_ratings, 2, sd)

```

```
# Calculate summary stats
sound_rating_Summary <- apply(sound_ratings, 2, summary)
head(sound_rating_Summary)
View(sound_rating_Summary)
View(sound_rating_SDs)

# Bind the mean and SD values into one table
mean_sd <- rbind(sound_rating_Means, sound_rating_SDs)
View(mean_sd)

# Bind Summary Stats to mean and SD values - compare means to see if
    they match
summary_mean_sd <- rbind(sound_rating_Summary,mean_sd)
View(summary_mean_sd)

# Bind mean & SD with originl ratings
allratings_meanSD <- rbind(sound_ratings, mean_sd)
View(allratings_meanSD)

# Transpose a data frame
sort_mean_sd <- t(mean_sd)
View(sort_mean_sd)

# Attach sound class data
classFactor <- c("Natural","Natural","Household","Animals",
          "Household","Household","Household",
                 "Animals","Animals","Exterior","Exterior",
                 "Natural","Exterior","Animals",
                 "Human","Natural","Human","Animals",
                 "Natural","Exterior","Natural","Animals",
                 "Household","Human","Human","Household",
                 "Household","Natural","Exterior","Human",
```

```
                  "Exterior","Human","Exterior","Household",
                  "Animals","Animals","Natural","Exterior",
                  "Human","Human")

View(sound_rating_Means)

meanClass <- cbind(classFactor, sound_rating_Means)
View(meanClass)

sdClass <- cbind(classFactor, sound_rating_SDs)
View(sdClass)

# Plot means and SDs on the same scatter
plot(sound_rating_Means, sound_rating_SDs,
     xlab="Sound Mean Score",
     ylab="Sound Score Standard Deviation",
     main="Summary of Sound Scores and Standard Deviations by Class",
     col=as.factor(classFactor), pch = 19, cex = 1, lty = "solid",
    lwd = 2)

# add sort_mean_sd row.names as data labels
     text(sound_rating_Means, sound_rating_SDs,
          labels = row.names(sort_mean_sd), cex=0.7, pos = 1)

     # legend
     legend("bottom", legend=c("Exterior", "Human", "Household", "
    Animals", "Natural"),
            col = c("red","blue", "green", "black", "cyan"), pch=c
    (15), bg="white", border="black")

#######################################################
     names(data_demographics)
```

```
136
137     boxplot(Baby_Crying ~ gender, data=data_demographics,
138             main="Gender Comparison",
139             xlab="Score by Gender", ylab="Sound Rating")
140
141     tapply(Thunderstorm, gender, mySummary)
142
143     mySummary <- function(x) {
144       theSD <- sd(x) #Standard deviation
145       # fiveNumSumm <- fivenum(x) #Tukey's five number summary,
    usefull for boxplots
146       # IQR(x) #Interquartile range
147       # quantile(x) #Compute sample quantiles
148       # range(x) # Get minimum and maximum
149       # result <- list(fiveNumSumm,theSD)
150       data <- cbind(min(x), median(x), max(x), theSD)
151       result <- as.data.frame(data, col.names=c("MIN", "MEDIAN", "
    MAX", "SD"))
152       return(result)
153     }
154
155     tapply(Thunderstorm, gender, mySummary)
156
157     ################## MUNGED DATA PLOTTING #############
158     munged_data <- read.csv('/Volumes/GoogleDrive/My Drive/DIT-PhD/
    STATS/EXP_1_R_ANALYSIS/Munged_all_male_female.csv',
159                             header=TRUE)
160 munged_data
161
162 # Plot means and SDs on the same scatter
163 plot(munged_data$ALL.MEAN, munged_data$ALL.SD,
164     xlab="Sound Mean Score",
```

```
165      ylab="Sound Score Standard Deviation",
166      main="Summary of Sound Scores and Standard Deviations",
167      col= "red", pch = 19, cex = 1, lty = "solid", lwd = 2)
168
169 points(munged_data$FEMALE.MEAN, munged_data$FEMALE.SD,
170        col= "blue", pch = 20, cex = 1, lty = "solid", lwd = 2)
171
172 points(munged_data$MALE.MEAN, munged_data$MALE.SD,
173        col= "orange", pch = 18, cex = 1, lty = "solid", lwd = 2)
174
175 # add sort_mean_sd row.names as data labels
176 text(munged_data$ALL.MEAN, munged_data$ALL.SD,
177      labels = munged_data$GROUP, cex=0.7, pos = 1)
178
179 # Plot FEMALE mean and SDs on the same scatter
180 plot(munged_data$FEMALE.MEAN, munged_data$FEMALE.SD,
181      xlab="Sound Mean Score",
182      ylab="Sound Score Standard Deviation",
183      main="Female Sound Scores and Standard Deviations",
184      col= "blue", pch = 20, cex = 1, lty = "solid", lwd = 2)
185
186 # add data labels
187 text(munged_data$FEMALE.MEAN, munged_data$FEMALE.SD,
188      labels = munged_data$GROUP, cex=0.7, pos = 1)
189
190 # Plot MALE mean and SDs on the same scatter
191 plot(munged_data$MALE.MEAN, munged_data$MALE.SD,
192      xlab="Sound Mean Score",
193      ylab="Sound Score Standard Deviation",
194      main="Male Sound Scores and Standard Deviations",
195      col= "brown", pch = 18, cex = 1, lty = "solid", lwd = 2)
196
```

```
197 # add data labels
198 text(munged_data$MALE.MEAN, munged_data$MALE.SD,
199      labels = munged_data$GROUP, cex=0.7, pos = 1)
200
201 labelsVector <- munged_data$GROUP
202 labelsVector
203
204 # What are the graph margins?
205 par('mar')
206
207 # reset graph margins to see sound labels
208 par(mar=c(9.1,4.1,4.1,2.1))
209
210 # Plot all means on the scatter
211 plot(munged_data$ALL.MEAN,
212      xlab="",
213      xaxt="n", # prevents the drawing of tick marks and numbers on
    the x axis
214      ylab="Mean Sound Rating",
215      main="Mean Sound Ratings - Comparison by Gender",
216      # xlim=c(1,40),
217      ylim=c(1,3),
218      col= "red", pch = 15, cex = 1, lty = "solid", lwd = 2, las=2)
219
220 # Draw gridlines
221 grid(nx = NULL, ny = NULL, col = "lightgray", lty = 5,
222      lwd = par("lwd"), equilogs = TRUE)
223
224 # Draw sound names on x axis
225 axis(1,
226      at=c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,
227      21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40),
```

```
228       labels=labelsVector,
229       pos=0.919, # vertical placement of new axis
230       lty="solid",
231       col="black",
232       las=2,
233       tck=-0.01,
234       outer=0)
235 
236 # Draw female mean points to graph
237 points(munged_data$FEMALE.MEAN,
238         col= "blue", pch = 16, cex = 1, lty = "solid", lwd = 2)
239 
240 # Draw male mean points to graph
241 points(munged_data$MALE.MEAN,
242         col= "green", pch = 17, cex = 1, lty = "solid", lwd = 2)
243 
244 # Add a legend, 'legend' is for the test int he legend, 'fill' is for
      colours,
245 # which has been replaced below with 'col=' and 'pch=' to specify
     colours and
246 # shapes for the legend
247 legend("topleft", legend=c("All", "Female", "Male"),
248         col = c("red","blue", "green"), pch=c(15,16,17), bg="white",
     border="black")
249 
250 names(munged_data)
251 
252 # A numerical vector of the form c(bottom, left, top, right) which
     gives the
253 # number of lines of margin to be specified on the four sides of the
     plot.
254 # The default is c(5, 4, 4, 2) + 0.1.
```

```
255 par(mar=c(9.1,4.1,4.1,2.1))
256
257 # Plot all SDs on the scatter
258 plot(munged_data$ALL.SD,
259      xlab="",
260      xaxt="n", # prevents the drawing of tick marks and numbers on
    the x axis
261      ylab="Standard Deviation of Sound Ratings",
262      main="Standard Deviation of Sound Ratings - Comparison by Gender
    ",
263      # xlim=c(1,40),
264      ylim=c(0.1,1),
265      col= "red", pch = 15, cex = 1, lty = "solid", lwd = 2, las=2)
266
267 # Draw gridlines
268 grid(nx = NULL, ny = NULL, col = "lightgray", lty = 5,
269      lwd = par("lwd"), equilogs = TRUE)
270
271 # Draw sound names on x axis
272 axis(1,
273      at=c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,
274      21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40),
275      labels=labelsVector,
276      pos=0.065, # vertical placement of new axis
277      lty="solid",
278      col="black",
279      las=2,
280      tck=-0.01,
281      outer=0)
282
283 # Draw female SD points to graph
284 points(munged_data$FEMALE.SD,
```

```
285         col= "blue", pch = 16, cex = 1, lty = "solid", lwd = 2)
286 
287 # Draw male SD points to graph
288 points(munged_data$MALE.SD,
289         col= "green", pch = 17, cex = 1, lty = "solid", lwd = 2)
290 
291 # Add a legend, 'legend' is for the test in the legend, 'fill' is for
        colours,
292 # which has been replaced below with 'col=' and 'pch=' to specify
       colours and
293 # shapes for the legend
294 legend("topleft",
295         legend=c("All", "Female", "Male"),
296         col = c("red","blue", "green"), pch=c(15,16,17), bg="white",
       border="black")
```

A.2 Experiment 2

A.2.1 Experiment 2 - Random Forest Python Code

Python code (originally written in a Jupyter Notebook) for Random Forest parameter grid search and recursive feature elimination.

```
1 #!/usr/bin/env python
2 # coding: utf-8
3 # In[61]:
4 import pandas as pd
5 import numpy as np
6 import pprint as pp # pretty printer
7 
8 from sklearn.model_selection import StratifiedKFold,
      RandomizedSearchCV, ParameterGrid, GridSearchCV
```

```
# Recursive feature elimination -  starts with all features, leaves one out
from sklearn.feature_selection import RFECV, RFE
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import make_scorer, recall_score, precision_score, classification_report, confusion_matrix

df = pd.read_csv('data/mtResults_AllStim_ROWS.csv') # Read in data from external excel sheet

# In[22]:
# indexing rows and columns by name, note 'loc' and square brackets
CAT_target = pd.DataFrame(df.loc[:, "EXP2_FGnotFG_CAT"])
# indexing rows and columns by number, note 'iloc' and square brackets - : gives all rows here
# This gives access to all the soundNames
soundNames = pd.DataFrame(df.iloc[:,3])

# all numerical data inc. deltas
allData = pd.DataFrame(df.loc[:, "maxval":"DBLDELTA_Chroma_Vector_12_STDdivMEAN"])
print("CAT_target Shape is: ", CAT_target.shape)
print("soundNames Shape is: ", soundNames.shape)
print("39th entry in soundNames is: ", soundNames.iloc[39])
print("allData Shape is: ", allData.shape)

# In[981]:
# DELTA_MFCC_9_MIN
# allData.loc[:, "DELTA_MFCC_9_MIN"]
# Which dataset to analyse
# datasetFocus = 2
# DataFrame of sound information - includes counts from EXP1
```

```
# dataSetInfo = pd.DataFrame(df.iloc[:,0:22])

# In[23]:
#################
### STEP 1.1 ###
#################
# First split the data up into 5 stratified folds
# Stratified k Folds - outer split of 5
skf_Outer = StratifiedKFold(n_splits=5, random_state=3, shuffle=True)

'''
On the outer split // trainval/test:
Do a parameter search on trainval for what works best with this fold
Choose model, then use this to build a series of feature sets.
Then on the inner split // train/val:
Do CV fitting across all feature sets and inner folds to find best
    performing model/feature set combination
Then evaluate this model using the outer split trainval/test
'''

i = 1 # just for counting
# dicts to hold outer splits and indexes
X_trainval_dict = {} # trainval data
y_trainval_dict = {} # trainval labels
X_test_dict = {} # test data
y_test_dict = {} # test labels

trainval_indices = {}
test_indices = {}
# Stepping through the Outer folds one at a time
for trainval_index, test_index in skf_Outer.split(allData,
                            CAT_target):
```

```
    # keep track of different indices for later
    trainval_indices[i] = trainval_index
    test_indices[i] = test_index
    # Store the actual data and corresponding labels for later access
    X_trainval_dict[i], X_test_dict[i] = allData.loc[trainval_index],
    allData.loc[test_index]
    y_trainval_dict[i], y_test_dict[i] = CAT_target.loc[
   trainval_index], CAT_target.loc[test_index]
    # make sure shapes of data are correct
    print("X_TRAINVAL_SHAPE", [i], ":", X_trainval_dict[i].shape, "
   X_TEST_SHAPE", [i], ":", X_test_dict[i].shape)
    print("y_TRAINVAL_SHAPE", [i], ":", y_trainval_dict[i].shape, "
   y_TEST_SHAPE", [i], ":", y_test_dict[i].shape)
    i += 1

# In[982]:
# SAVING THE TRAIN/TEST SPLITS TO A SEPERATE EXCEL FILE IN CASE I
   NEED THEM LATER.
filename = "METHOD_3_trainval_test_splits.xlsx"
# print(filename)
# Write data to excel
writer = pd.ExcelWriter(filename)

dataSetInfo.to_excel(writer, 'dataSetInfo')

j = 1

for j in trainval_indices:
    X_trainval = 'X_trainval_data_%d' % (j)
    X_trainval_dict[j].to_excel(writer, X_trainval)
    y_trainval = 'y_trainval_labels_%d' % (j)
    y_trainval_dict[j].to_excel(writer, y_trainval)
```

```
94      X_test = 'X_test_data_%d' % (j)
95      X_test_dict[j].to_excel(writer, X_test)
96      y_test = 'y_test_labels_%d' % (j)
97      y_test_dict[j].to_excel(writer, y_test)
98
99  writer.save()
100
101 # In[1326]:
102 ################
103 ### STEP 2.1 ###
104 ################
105
106 # Randomised parameter search across trainval data
107 # Defining Parameters used in RANDOM grid search
108 # Number of trees in random forest
109 n_estimators = [50, 200, 500]
110 # Number of features to consider at every split
111 max_features = [2, 5, 10, 20, 50]
112 # Maximum number of levels in tree
113 max_depth = [2, 3, 5, None]
114 # Minimum number of samples required to split a node
115 min_samples_split = [2, 3, 5]
116 # Minimum number of samples required at each leaf node
117 min_samples_leaf = [1, 2]
118 # Method of selecting samples for training each tree
119 bootstrap = [True, False]
120 # Create the random grid
121 random_grid = {'n_estimators': n_estimators,
122                'max_features': max_features,
123                'max_depth': max_depth,
124                'min_samples_split': min_samples_split,
125                'min_samples_leaf': min_samples_leaf,
```

```
126                'bootstrap': bootstrap}
127
128 pp.pprint(random_grid)
129
130 # In[1327]:
131 #################
132 ### STEP 2.2 ###
133 #################
134
135 ##########################################################
136 ###### RANDOMISED PARAMETER SEARCH ON FOLD DATA ######
137 ##########################################################
138
139 # Automating text output
140 # f = open('random_param_search.txt','w')
141 # sys.stdout = f
142
143 ##########################################################
144 ################ CHANGE FOLDFOCUS HERE ################
145 # change this to move through the outer folds ########
146 foldFocus = 4 ##########################################
147 ##########################################################
148
149 modelRows = {} # to track different model parameters and scores
150 allmodelDF = pd.DataFrame()
151
152 # Change dataset here \/ \/ \/
153 useThisDataset = X_trainval_dict[foldFocus] # look at this fold only
154                                             # searching for
    parameters that work best on this fold
155
156 # classifier to use in parameter search
```

```
157 rf = RandomForestClassifier()
158
159 scorers = { # setting up recall and precision as the metrics we'll
        use
160     'precision': make_scorer(precision_score, pos_label="FG"),
161     'recall': make_scorer(recall_score, pos_label="FG")
162 }
163
164 X_train = useThisDataset
165 y_train = y_trainval_dict[foldFocus]
166
167 # Random search of parameters, using 4 fold cross validation,
168 # search across 100 different combinations, and use all available
        cores
169 clf = RandomizedSearchCV(estimator = rf,
170                          param_distributions = random_grid,
171                          n_iter = 100, cv = 4, verbose=2,
172                          random_state=42, scoring=scorers,
173                          refit=False, n_jobs = -1)
174
175 # Fit model
176 clf.fit(X_train, y_train.values.ravel())
177
178 # Reporting which parameters perform best on this inner fold
179 #print()
180 #print("Grid scores on development set:")
181 #print()
182 means = clf.cv_results_['mean_test_precision']
183 stds = clf.cv_results_['mean_test_recall']
184
185 for mean, std, params in zip(means, stds, clf.cv_results_['params']):
186     #print("%0.3f (+/-%0.03f) for %r"
```

```
187                 #% (mean, std * 2, params))
188
189     modelRows.update({'Precision': means, 'Recall': stds, 'Params':
        clf.cv_results_['params']})
190
191 modelDF = pd.DataFrame(modelRows)
192 allmodelDF = pd.concat([allmodelDF, modelDF], axis=0, join='outer')
193
194 # In[1148]:
195 ################
196 ### STEP 2.3 ###
197 ################
198
199 # Define column order
200 colOrd=['Precision', 'Recall', 'Params']
201 # Reorder columns
202 allmodelDF = allmodelDF[colOrd]
203 # Filename with model params & Dataset
204 filename = "RF_FG_RandParamSearch.xlsx"
205 # print(filename)
206 # Write data to excel
207 writer = pd.ExcelWriter(filename)
208 allmodelDF.to_excel(writer, 'Sheet1')
209 writer.save()
210
211
212 # In[1150]:
213 ################
214 ### STEP 3.1 ###
215 ################
216
217 ################################
```

```
###### RFECV ON FOLD DATA ######
### GENERATE FEATURE SETS ######
################################

# Automating text output
f = open('workings.txt','w')
sys.stdout = f
bestDatasets = {} # stores all feature sets
noFeatures = [50, 50, 50, 50, 100] # min numbers of features to
    extract
# applies a RF classifier to feature selection
i = 0

########################################################
CHANGE VALUES OF estimator MODEL TO GENERATE FEATURE
SUBSETS TUNED TO BEST PARAMETERS FOR THIS FOLD ######
#### \/ \/ \/ \/ \/ \/ \/ ############################
estimator = RandomForestClassifier(n_estimators=500,
                                   min_samples_split=2,
                                   min_samples_leaf=2,
                                   max_features=50,
                                   max_depth=3,
                                   bootstrap=False)
##### /\ /\ /\ /\ /\ /\ /\ ########################
########################################################
########################################################

# constructing best features sets with different numbers of features
for feat in noFeatures:

    select = RFECV(estimator,
                   step=1, # remove one feature at a time
```

```
                    n_jobs=-1, # use all cores
                    cv=4,
                    # how many features to select
                    min_features_to_select=feat)

    # change the fold number here in the square brackets to move on
    to the next one
    # fit the model to the correct outer fold
    select.fit(X_trainval_dict[foldFocus],
      y_trainval_dict[foldFocus].values.ravel())

    # visualize the selected features:
    # mask becomes an index to the best features
    mask = select.get_support(indices=True)
    print(mask)

    # save the index to features for future access
    bestDatasets[i] = mask
    print("Computed best features: ", feat)
    print("Optimal number of features : %d" % select.n_features_)

    i += 1

for key in bestDatasets:
    print(bestDatasets[key])

print("BEST FEATURES")
print("Ignore the numbers - just the values from the first row.")
print(allData.iloc[0, bestDatasets[0]])

# remember when looking at each fold to pull data from the trainval
   subset, NOT allData
```

```
# print(X_trainval[1].iloc[0, mask])

#select.ranking_ # prints the ranking of each feature in order

# In[1151]:
################
### STEP 3.2 ###
################

#################################################################
###### TAKE KEY FROM DATASETS AND ASSOCIATE WITH DATA ######
#################################################################

# Automating text output
f = open('workings.txt','w')
sys.stdout = f

FG_F4b_bestFeatures = {}
FG_F4b_testSounds = {}

for key in bestDatasets:
    FG_F4b_bestFeatures[key] = X_trainval_dict[foldFocus].iloc[:,
   bestDatasets[key]]
    FG_F4b_testSounds[key] = X_test_dict[foldFocus].iloc[:,
   bestDatasets[key]]
    print('FG_F4b_bestFeatures', [key],
      ' shape', FG_F4b_bestFeatures[key].shape)
    print('FG_F4b_testSounds', [key],
      ' shape', FG_F4b_testSounds[key].shape)

# In[1295]:
################
```

```
309 ### STEP 4.1 ###
310 ################
311
312 ############################################################
313 ########## PARAMETER GRID FOR GRID SEARCH ###############
314 ######## USE MAX 15 OPTIONS HERE FOR BREVITY ############
315 ######## IF USING 200, 1000, 2000 ESTIMATORS ############
316 ############################################################
317
318 # Automating text output
319 f = open('workings.txt','w')
320 sys.stdout = f
321 # Number of trees in random forest
322 n_estimators = [10, 50, 100, 500]
323 # Number of features to consider at every split
324 max_features = [2, 10, 50]
325 # Maximum number of levels in tree
326 max_depth = [2, 3]
327 # Minimum number of samples required to split a node
328 min_samples_split = [2, 3, 4]
329 # Minimum number of samples required at each leaf node
330 min_samples_leaf = [1, 2] #[1, 2]
331 # Method of selecting samples for training each tree
332 bootstrap = [True, False]
333
334 # Create the random grid
335 param_grid = {'n_estimators': n_estimators,
336               'max_features': max_features,
337               'max_depth': max_depth,
338               'min_samples_split': min_samples_split,
339               'min_samples_leaf': min_samples_leaf,
340               'bootstrap': bootstrap}
```

```
341
342 # classifier to use in parameter search
343 rf_GridSearch = RandomForestClassifier()
344
345 # In[1296]:
346 ####################################
347 ### Change dataset here \/ \/ \/ ###
348 datasetFocus = 4
349 ####################################
350
351 FG_F4b_bestFeatures[datasetFocus].shape
352 #FG_F4b_testSounds[datasetFocus].shape
353
354 # In[1297]:
355 ################
356 ### STEP 4.2 ###
357 ################
358
359 ##########################################################
360 ############## PARAMETER GRID SEARCH ####################
361 ##########################################################
362
363 # Automating text output
364 f = open('parameter_grid_search.txt','w')
365 sys.stdout = f
366 modelRows = {} # to track different model parameters and scores
367 allmodelDF = pd.DataFrame()
368
369 useThisDataset = FG_F4b_bestFeatures[datasetFocus]
370
371 scorers = {
372     'precision': make_scorer(precision_score, pos_label="FG"),
```

```
    'recall': make_scorer(recall_score, pos_label="FG")
}

X_train = useThisDataset
y_train = y_trainval_dict[foldFocus]

# PARAMETER SEARCH
# Grid search of parameters, using 4 fold cross validation,
# search across 100 different combinations, and use all available
    cores
clf = GridSearchCV(estimator = rf_GridSearch,
                   param_grid = param_grid, cv = 4,
                   n_jobs = -1, # using all cores
                   scoring=scorers, refit=False, iid=False,
                   verbose = 2)

# Fit model
clf.fit(X_train, y_train.values.ravel())

# Reporting which parameters perform best on this inner fold
means = clf.cv_results_['mean_test_precision']
stds = clf.cv_results_['mean_test_recall']

for mean, std, params in zip(means, stds, clf.cv_results_['params']):
    modelRows.update({'Precision': means, 'Recall': stds, 'Params':
    clf.cv_results_['params']})

modelDF = pd.DataFrame(modelRows)
allmodelDF = pd.concat([allmodelDF, modelDF], axis=0, join='outer')

# In[1298]:
################
```

```
### STEP 4.3 ###
################

############################################################
#################### FIX PARAMETERS ########################
############################################################

# Define column order
colOrd=['Precision', 'Recall', 'Params']
# Reorder columns
allmodelDF = allmodelDF[colOrd]
# Filename with model params & Dataset
filename = "RF_FG_ParamGridSearch.xlsx"
# Write data to excel
writer = pd.ExcelWriter(filename)
allmodelDF.to_excel(writer, 'Sheet1')
writer.save()

# In[1330]:
################
### STEP 4.4 ###
################

#######################################################
########## INITIAL FIXED PARAMETERS ###############
#######################################################

# Number of trees in random forest
n_estimators = [10]
# Number of features to consider at every split
max_features = [10]
# Maximum number of levels in tree
```

```
max_depth = [3]
# Minimum number of samples required to split a node
min_samples_split = [3]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1]
# Method of selecting samples for training each tree
bootstrap = [False]

# In[1333]:
################
### STEP 4.5 ###
################

#################################################
## RUNNING INITIAL PARAMETERS ON INNER CV FOLDS ##
#################################################
# Stratified folds - inner split
skf_Inner = StratifiedKFold(n_splits=4, random_state=5, shuffle=True)
# Automating text output
f = open('dimensionality_reduction.txt','w')
sys.stdout = f
# Summary Results table - for comparing models
allSummaryData = pd.DataFrame(columns=['FG_Prec', 'FG_Recall',
                   'FG_F1', 'notFG_Prec',
                   'notFG_Recall', 'notFG_F1',
                                      'Confusion Matrix', 'FG_Support',
                                      'notFG_Support', 'Inner Fold',
                                      'Acc_Train', 'Acc_Val',
                                      'no_estimators_Param',
                                      'max_depth_Param',
                                      'max_features_Param',
                                      'min_samples_leaf_Param',
```

```
                                    'min_samples_split_Param',
                                    'Bootstrap', 'featSet'])

# to track classification success by sound - are some sounds more
    difficult to categorise than others?
allCatResults = pd.DataFrame(columns=['Sound_Name', 'EXP2_FGnotFG_CAT
    ', 'Prediction'])

j = 1 # to count inner CV folds

# change this to flip between different feature sets
#featSetNumber = datasetFocus
thisFeatureSet = FG_F4b_bestFeatures[datasetFocus] # <<<<<<<

# for matching names/index to important features
names = thisFeatureSet.columns
# will add importances in loop
featimportance = pd.DataFrame([names])

for train_index, val_index in skf_Inner.split(thisFeatureSet,
    y_trainval_dict[foldFocus]):
    print('################### INNER ########################')
    print('INNER FOLD {} of KFold {}'.format(j, skf_Inner.n_splits))
    print('################### INNER ########################')

    # To pull data from the trainval split
    # by finding the original index (for the outer CV) from the index
    for the inner CV
    X_train =\
    thisFeatureSet.loc[trainval_indices[foldFocus][train_index]]
    X_val =\
    thisFeatureSet.loc[trainval_indices[foldFocus][val_index]]
```

```
 y_train = y_trainval_dict[foldFocus].loc[trainval_indices[foldFocus][train_index]]
 y_val = y_trainval_dict[foldFocus].loc[trainval_indices[foldFocus][val_index]]

 # TARGETED PARAMETER AND FEATURE SET SEARCH
 for est in n_estimators:
     for feat in max_features:
         for dep in max_depth:
             for split in min_samples_split:
                 for leaf in min_samples_leaf:
                     for boot in bootstrap:
                             # Train a random forest
                             clf = RandomForestClassifier(
                             n_estimators=est,
                             max_features=feat,
                             max_depth=dep,
                             min_samples_split=split,
                             min_samples_leaf=leaf,
                             bootstrap=boot,
                             random_state=0)

                             # Fit model
                             clf.fit(X_train,
                             y_train.values.ravel())

                             # Get model parameters
                             ModPar = clf.get_params()
                             print('Model parameters are: ',
                             ModPar)

                             # Predict on the validation set
```

```
                                y_true, y_pred = y_val,
                                    clf.predict(X_val)
                                class_report =\
                                classification_report(y_true, y_pred,
                                output_dict=True)
                                class_report_ =\
                                classification_report(y_true, y_pred)

                                print('## CLASS REPORT ##')
                                print(class_report_)

                                # Generate accuracy scores
                                acc = clf.score(X_train,
                                  y_train.values.ravel())
                                acc_2 = clf.score(X_val,
                                  y_val.values.ravel())
                                print('Mean model accuracy on
    training set: ',
                                acc)
                                print('Mean model accuracy on
    validation set: ',
                                acc_2)

                                print('############################')
                                print('############################')

                                # Generate confusion matrix
                                conMat = confusion_matrix(y_true,
                                            y_pred)
                                print("Confusion matrix:\n{}".
                                  format(conMat))
                                print("y_true.shape is:",
```

```
                                y_true.shape)
print("y_pred.shape is:",
                                y_pred.shape)

print('## ##### ##')

# How predictor matches with actual
newDf = pd.DataFrame(y_val,
                                columns=['EXP2_FGnotFG_CAT'])
predDf = pd.DataFrame(y_pred,
                                index=y_val.index,
                                columns=['Prediction'])
result =\
pd.concat([soundNames.
                                iloc[y_val.index],
                                newDf, predDf], axis=1,
                                join='outer')
print(result)

print('##########################')

foldData =\
pd.DataFrame({'FG_Prec':
[class_report['FG']['precision']],
                                'FG_Recall':
[class_report['FG']['recall']],
                                'FG_F1':
[class_report['FG']['f1-score']],
                                'FG_Support':
[class_report['FG']['support']],
                                'notFG_Prec':
[class_report['notFG']['precision']],
```

```
                                        'notFG_Recall':
                                [class_report['notFG']['recall']],
                                        'notFG_F1':
                                [class_report['notFG']['f1-score']],
                                        'notFG_Support':
                                [class_report['notFG']['support']],
                                        'Confusion Matrix':
                                [conMat],
                                        'Acc_Train': [acc],
                                        'Acc_Val': [acc_2],
                                        'no_estimators_Param':
                                clf.n_estimators,
                                        'max_depth_Param':
                                [clf.max_depth],
                                        'max_features_Param':
                                [clf.max_features],
                                        'min_samples_leaf_Param
    ':
                                [clf.min_samples_leaf],
                                 'min_samples_split_Param':
                                [clf.min_samples_split],
                                        'Bootstrap':
                                [clf.bootstrap],
                                        'Inner Fold':
                                [j],
                                        'featSet':
                                [datasetFocus]})

                                # Add data from this fold to that
                                # from previous folds
                                allSummaryData = pd.concat([
                                  allSummaryData, foldData],
```

```
                                    axis=0, join='outer')

                                # gather all the categorisations
                                # for each sound
                                allCatResults = pd.concat([
                                  allCatResults, result],
                                  axis=0, join='outer')

                                # gather information on features
                                imps = pd.DataFrame(
                                  clf.feature_importances_)
                                featimportance = pd.concat(
                                [featimportance, imps.T],
                                axis=0, join='outer')

    j += 1

# In[1301]:
################
### STEP 4.6 ###
################

#####################################################
## EVALUATE INITIAL PARAMETERS ON INNER CV FOLDS ##
#####################################################

# Define column order
defCols = ['FG_Prec', 'FG_Recall', 'FG_F1', 'FG_Support',
  'notFG_Prec', 'notFG_Recall', 'notFG_F1', 'notFG_Support',
    'Acc_Train', 'Acc_Val', 'Confusion Matrix', 'Inner Fold',
    'no_estimators_Param', 'max_depth_Param',
    'max_features_Param', 'min_samples_leaf_Param',
```

```
650     'min_samples_split_Param', 'Bootstrap', 'featSet']
651
652 # Reorder columns
653 allSummaryData = allSummaryData[defCols]
654 filename = "RF_FG_Dimensionality_Reduction.xlsx"
655 # Write data to excel
656 writer = pd.ExcelWriter(filename)
657 allSummaryData.to_excel(writer, 'Sheet1')
658 # Comment in if you want to write to the same excel file
659 allCatResults.to_excel(writer, 'Sheet2')
660 writer.save()
661
662
663 # In[1332]:
664 filename = "featimportance.xlsx"
665 # print(filename)
666 # Write data to excel
667 writer = pd.ExcelWriter(filename)
668 featimportance.to_excel(writer, 'Sheet1')
669 writer.save()
670
671 # In[1211]:
672 ################
673 ### STEP 5.1 ###
674 ################
675
676 ##############################################
677 ## PARAMETERS FOR DIMENSIONALITY REDUCTION ##
678 ##############################################
679 # Number of trees in random forest
680 n_estimators = [50]
681 # Number of features to consider at every split
```

```
682 max_features = [10]
683 # Maximum number of levels in tree
684 max_depth = [3]
685 # Minimum number of samples required to split a node
686 min_samples_split = [2]
687 # Minimum number of samples required at each leaf node
688 min_samples_leaf = [2]
689 # Method of selecting samples for training each tree
690 bootstrap = [False]
691
692 # In[1303]:
693 #################
694 ### STEP 5.2 ###
695 #################
696
697 ############################################################
698 ############## DIMENSIONALITY REDUCTION ##################
699 ########## GENERATING NEW (SUB) DATASETS ###############
700 ############################################################
701 # Automating text output
702 f = open('subFeatureSets.txt','w')
703 sys.stdout = f
704
705 # change this to flip between different feature sets
706 # featSetNumber = 1
707 thisFeatureSet = FG_F4b_bestFeatures[datasetFocus] # <<<<<<<
708
709 subDatasets = {} # stores all feature set indices
710 #thisFeatureSet.shape[1]] # numbers of features to extract
711 noFeatures = [2, 5, 10, 20, 100, 200]
712
713 i = 0
```

```
714
715 for est in n_estimators:
716     for feat in max_features:
717         for dep in max_depth:
718             for split in min_samples_split:
719                 for leaf in min_samples_leaf:
720                     for boot in bootstrap:
721                         for feat in noFeatures:
722
723                             select = RFE(RandomForestClassifier(
724                               n_estimators=est,
725                               bootstrap=boot,
726                               max_depth=dep,
727                               max_features=feat,
728                               min_samples_leaf=leaf,
729                               min_samples_split=split,
730                               random_state=42),
731                               n_features_to_select=feat)
732
733                             # change the fold number here in the
734                             # square brackets to move on to the
735                             # next one
736                             # fit the model to the correct outer fold
737                             select.fit(thisFeatureSet,
738                             y_trainval_dict[foldFocus].values.ravel()
    )
739                             # visualize the selected features:
740                             # mask becomes an index to the best
741                             # features
742                             mask = select.get_support(indices=True)
743                             print(mask)
744                             # save the index to features for future
```

```
                                # access
                                subDatasets[i] = mask

                                print("BEST FEATURES", i)
                                print("Ignore the numbers - just the
    values from the first row.")
                                print(FG_F4b_bestFeatures[datasetFocus].
    iloc[0, subDatasets[i]])
                                print("RANKING")
                                # prints the ranking of each feature in
                                # order
                                print(select.ranking_)
                                print("Computed best features: ", feat)
                                print("i: ", i)
                                i += 1

# In[1304]:
#################
### STEP 5.3 ###
#################

#########################################################
## HOUSEKEEPING SO DATA AND LABELS CAN BE ACCESSED ##
############ FOR EACH NEW DATASET ###################
#########################################################

newDatasets = {}
newTestSets = {}
# use the indices in the subDatasets dict to pull all those features
    into new dicts so they can be called
for key in subDatasets:
```

```
773     newDatasets[key] = FG_F4b_bestFeatures[datasetFocus].iloc[:,
    subDatasets[key]]
774     newTestSets[key] = FG_F4b_testSounds[datasetFocus].iloc[:,
    subDatasets[key]]
775
776 # In[1305]:
777 ################
778 ### STEP 6.1 ###
779 ################
780
781 ##############################################
782 ### RUN FIXED PARAMETERS AGAINST EACH OF ###
783 ######## THESE NEW SUB DATASETS #############
784 ## COMPARE THE RESULTS TO CHOOSE THE BEST ##
785 ########## WORKING DATASET #################
786 ##############################################
787
788 # Stratified folds - inner split
789 skf_Inner = StratifiedKFold(n_splits=4,
790         random_state=5,
791         shuffle=True)
792
793 # Automating text output
794 f = open('train_validation.txt','w')
795 sys.stdout = f
796
797 # Summary Results table - for comparing models
798 allSummaryData = pd.DataFrame(columns=['FG_Prec', 'FG_Recall',
799       'FG_F1', 'notFG_Prec', 'notFG_Recall', 'notFG_F1',
800       'Confusion Matrix', 'FG_Support', 'notFG_Support',
801             'Inner Fold', 'Acc_Train', 'Acc_Val',
802             'no_estimators_Param', 'max_depth_Param',
```

```
            'max_features_Param', 'min_samples_leaf_Param',
            'min_samples_split_Param', 'Bootstrap', 'featSet'])

# to track classification success by sound - are some sounds more
    difficult to categorise than others?
allCatResults = pd.DataFrame(columns=['Sound_Name',
            'EXP2_FGnotFG_CAT', 'Prediction'])

# for matching names/index to important features
names = thisFeatureSet.columns
# will add importances in loop
featimportance = pd.DataFrame([names])

# change this to flip between different feature sets
for featSetNumber in newDatasets:
    thisFeatureSet = newDatasets[featSetNumber] # <<<<<<<
    # to count inner CV folds
    j = 1
    for train_index, val_index in skf_Inner.split(thisFeatureSet,
                    y_trainval_dict[foldFocus]):
        print('################### INNER ########################')
        print('INNER FOLD {} of KFold {}'.format(j,
                        skf_Inner.n_splits))
        print('################### INNER ########################')

        # To can pull data from the trainval split
        # by finding the original index (for the outer CV)
        # from the index for the inner CV
        X_train =\
        thisFeatureSet.loc[trainval_indices[foldFocus][train_index]]
        X_val =\
        thisFeatureSet.loc[trainval_indices[foldFocus][val_index]]
```

```
834         y_train =\
835         y_trainval_dict[foldFocus].loc[trainval_indices[foldFocus][
    train_index]]
836         y_val =\
837         y_trainval_dict[foldFocus].loc[trainval_indices[foldFocus][
    val_index]]
838
839         # TARGETED PARAMETER AND FEATURE SET SEARCH
840         for est in n_estimators:
841             for feat in max_features:
842                 for dep in max_depth:
843                     for split in min_samples_split:
844                         for leaf in min_samples_leaf:
845                             for boot in bootstrap:
846
847                                 # if max_features parameter >
848                                 # no. features in the dataset,
849                                 # let max_features = no. features
850                                 # in the dataset as
851                                 if feat > int(thisFeatureSet.
852                                           shape[1]):
853                                     useThis = int(thisFeatureSet.
854                                             shape[1])
855                                     print("working!!!")
856                                 else:
857                                     useThis = feat
858
859                                 # Train a random forest
860                                 clf = RandomForestClassifier(
861                                   n_estimators=est,
862                                   max_features=useThis,
863                                     max_depth=dep,
```

```
                                    min_samples_split=split,
                                    min_samples_leaf=leaf,
                                    bootstrap=boot,
                                    random_state=0)

                                # Fit model
                                clf.fit(X_train,
                                y_train.values.ravel())

                                # Get model parameters
                                ModPar = clf.get_params()
                                print('Model parameters are: ',
                                ModPar)

                                # How does this model do on the
                                # validation set?
                                y_true, y_pred = y_val,
                                  clf.predict(X_val)
                                class_report =\
                                classification_report(y_true, y_pred,
                                output_dict=True)
                                class_report_ =\
                                classification_report(y_true, y_pred)

                                print('## CLASS REPORT ##')
                                print(class_report_)
                                # Generate accuracy scores
                                acc = clf.score(X_train,
                                  y_train.values.ravel())
                                acc_2 = clf.score(X_val,
                                  y_val.values.ravel())
                                print('Mean model acc.train:',acc)
```

```
                                print('Mean model acc.valid:',acc_2)
                                print('##########################')

                                # Generate confusion matrix
                                conMat = confusion_matrix(y_true,
                                  y_pred)
                                print("Confusion matrix:\n{}".
                                  format(conMat))
                                print("y_true.shape is:",
                                  y_true.shape)
                                print("y_pred.shape is:",
                                  y_pred.shape)
                                print('## ##### ##')

                                # Printing how the predictor matches
                                #  with actual
                                newDf = pd.DataFrame(y_val,
                                  columns=['EXP2_FGnotFG_CAT'])
                                predDf = pd.DataFrame(y_pred,
                                  index=y_val.index,
                                  columns=['Prediction'])
                                result =\
                                pd.concat([soundNames.
                                  iloc[y_val.index],
                                  newDf, predDf],
                                axis=1,
                                join='outer')
                                print(result)
                                print('###########################')


foldData =\
```

```
                                pd.DataFrame({'FG_Prec':
                                [class_report['FG']['precision']],
                                'FG_Recall':
                                [class_report['FG']['recall']],
                                'FG_F1':
                                [class_report['FG']['f1-score']],
                                'FG_Support':
                                [class_report['FG']['support']],
                                'notFG_Prec':
                                [class_report['notFG']['precision']],
                                'notFG_Recall':
                                [class_report['notFG']['recall']],
                                'notFG_F1':
                                [class_report['notFG']['f1-score']],
                                'notFG_Support':
                                [class_report['notFG']['support']],
                                'Confusion Matrix':
                                [conMat],
                                'Acc_Train': [acc],
                                'Acc_Val': [acc_2],
                                'no_estimators_Param':
                                clf.n_estimators,
                                'max_depth_Param':
                                [clf.max_depth],
                                'max_features_Param':
                                [clf.max_features],
                                'min_samples_leaf_Param':
                                [clf.min_samples_leaf],
                                'min_samples_split_Param':
                                [clf.min_samples_split],
                                'Bootstrap':
                                [clf.bootstrap],
```

```
                                    'Inner Fold':
                                    [j],
                                    'featSet':
                                    [featSetNumber]})

                                    # Add data from this fold to that
                                    # from previous folds
                                    allSummaryData =\
                                    pd.concat([allSummaryData, foldData],
                                      axis=0, join='outer')

                                    # gather all the categorisations for
                                    # each sound
                                    allCatResults =\
                                    pd.concat([allCatResults, result],
                                      axis=0, join='outer')

                                    # gather information on features
                                    imps =\
                                    pd.DataFrame(clf.feature_importances_
    )
                                    featimportance =\
                                    pd.concat([featimportance, imps.T],
                                      axis=0, join='outer')

            j += 1

# In[1306]:
################
### STEP 6.2 ###
################

```

```
990 ##################################################################
991 ## EVALUATE PARAM PERFORMANCE ON EACH SUB DATASET, TWEAK PARAMS ##
992 ##################### IF NECESSARY ################################
993 ##################################################################
994 # Define column order
995 defCols = ['FG_Prec', 'FG_Recall', 'FG_F1', 'FG_Support',
996   'notFG_Prec', 'notFG_Recall', 'notFG_F1', 'notFG_Support',
997     'Acc_Train', 'Acc_Val', 'Confusion Matrix', 'Inner Fold',
998     'no_estimators_Param', 'max_depth_Param', 'max_features_Param',
999     'min_samples_leaf_Param', 'min_samples_split_Param', 'Bootstrap',
1000    'featSet']
1001
1002 # Reorder columns
1003 allSummaryData = allSummaryData[defCols]
1004 filename = "RF_FG_Train_Validation.xlsx"
1005 # Write data to excel
1006 writer = pd.ExcelWriter(filename)
1007 allSummaryData.to_excel(writer, 'Sheet1')
1008 # Comment in if you want to write to the same excel file
1009 allCatResults.to_excel(writer, 'Sheet2')
1010 writer.save()
1011
1012 # In[1310]:
1013 ########################################################
1014 ########## CHANGE SUBSET NUMBER HERE!!! #############
1015 ########################################################
1016
1017 subSetNo = 0 # <<<<<< BEST PERFORMING DATASET FROM 6.2
1018
1019 newDatasets[subSetNo].shape
1020
1021
```

```
# In[1311]:
################
### STEP 6.3 ###
################

##########################################
## PARAMETERS FOR TRAIN/VALIDATION TWEAK ##
##########################################
# Number of trees in random forest
n_estimators = [10, 25, 50, 200]
# Number of features to consider at every split
max_features = [2]
# Maximum number of levels in tree
max_depth = [2, 3, 4] #
# Minimum number of samples required to split a node
min_samples_split = [2, 3, 5]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2]
# Method of selecting samples for training each tree
bootstrap = [True, False]

# In[1312]:
################
### STEP 6.4 ###
################

#########################################################
## RUNNING A SLIGHTLY EXPANDED GRID SEARCH ON THE BEST ##
######### PERFORMING DATASET FROM 6.2 ###################
## USE RESULTS TO FINALISE BEST MODEL ON THIS DATASET ###
#########################################################
# Stratified folds - inner split
```

```
skf_Inner = StratifiedKFold(n_splits=4, random_state=5,
  shuffle=True)
# Automating text output
f = open('train_validation.txt','w')
sys.stdout = f

# Summary Results table - for comparing models
allSummaryData = pd.DataFrame(columns=['FG_Prec', 'FG_Recall',
  'FG_F1', 'notFG_Prec', 'notFG_Recall', 'notFG_F1',
    'Confusion Matrix', 'FG_Support', 'notFG_Support',
    'Inner Fold', 'Acc_Train', 'Acc_Val',
    'no_estimators_Param', 'max_depth_Param',
    'max_features_Param', 'min_samples_leaf_Param',
    'min_samples_split_Param', 'Bootstrap', 'featSet'])

# to track classification success by sound - are some sounds
# more difficult to categorise than others?
allCatResults = pd.DataFrame(columns=['Sound_Name',
  'EXP2_FGnotFG_CAT', 'Prediction'])

thisFeatureSet = newDatasets[subSetNo]

# to count inner CV folds
j = 1

for train_index, val_index in skf_Inner.split(thisFeatureSet,
   y_trainval_dict[foldFocus]):
    print('################### INNER ########################')
    print('INNER FOLD {} of KFold {}'.format(j, skf_Inner.n_splits))
    print('################### INNER ########################')

    # To pull data from the trainval split
```

```
    # by finding the original index (for the outer CV)
    # from the index for the inner CV
    X_train =\
    thisFeatureSet.loc[trainval_indices[foldFocus][train_index]]
    X_val =\
    thisFeatureSet.loc[trainval_indices[foldFocus][val_index]]
    y_train =\
    y_trainval_dict[foldFocus].
    loc[trainval_indices[foldFocus][train_index]]
    y_val =\
    y_trainval_dict[foldFocus].
    loc[trainval_indices[foldFocus][val_index]]

    # TARGETED PARAMETER AND FEATURE SET SEARCH
    for est in n_estimators:
        for feat in max_features:
            for dep in max_depth:
                for split in min_samples_split:
                    for leaf in min_samples_leaf:
                        for boot in bootstrap:
                                # Train a random forest
                                clf = RandomForestClassifier(
                                  n_estimators=est,
                                    max_features=feat,
                                    max_depth=dep,
                                    min_samples_split=split,
                                    min_samples_leaf=leaf,
                                    bootstrap=boot,
                                    random_state=0)

                                # Fit model
                                clf.fit(X_train,
```

```
                                    y_train.values.ravel())
                                # Get model parameters
                                ModPar = clf.get_params()
                                print('Model parameters are: ',
                                    ModPar)
                                # How does this model do on the
                                # validation set?
                                y_true, y_pred = y_val,
                                    clf.predict(X_val)
                                class_report =\
                                    classification_report(y_true,
                                        y_pred, output_dict=True)
                                class_report_ =\
                                    classification_report(y_true,
                                        y_pred)

                                print('## CLASS REPORT ##')
                                print(class_report_)

                                # Generate accuracy scores
                                acc = clf.score(X_train,
                                    y_train.values.ravel())
                                acc_2 = clf.score(X_val,
                                    y_val.values.ravel())
                                print('Mean model acc. train: ', acc)
                                print('Mean model acc, val: ', acc_2)
                                print('############################')

                                # Generate confusion matrix
                                conMat = confusion_matrix(y_true,
                                    y_pred)
                                print("Confusion matrix:\n{}".
```

```
    format(conMat))
print("y_true.shape is:",
    y_true.shape)
print("y_pred.shape is:",
    y_pred.shape)

print('## ##### ##')

# Printing how the predictor matches
# with actual
newDf = pd.DataFrame(y_val,
    columns=['EXP2_FGnotFG_CAT'])
predDf = pd.DataFrame(y_pred,
    index=y_val.index,
    columns=['Prediction'])
result = pd.concat([soundNames.
    iloc[y_val.index],
      newDf, predDf],
    axis=1, join='outer')
print(result)
print('######################')

foldData =\
pd.DataFrame({'FG_Prec':
[class_report['FG']['precision']],
'FG_Recall':
[class_report['FG']['recall']],
'FG_F1':
[class_report['FG']['f1-score']],
'FG_Support':
[class_report['FG']['support']],
'notFG_Prec':
```

```
                                [class_report['notFG']['precision']],
                                'notFG_Recall':
                                [class_report['notFG']['recall']],
                                'notFG_F1':
                                [class_report['notFG']['f1-score']],
                                'notFG_Support':
                                [class_report['notFG']['support']],
                                'Confusion Matrix':
                                [conMat],
                                'Acc_Train': [acc],
                                'Acc_Val': [acc_2],
                                'no_estimators_Param':
                                clf.n_estimators,
                                'max_depth_Param':
                                [clf.max_depth],
                                'max_features_Param':
                                [clf.max_features],
                                'min_samples_leaf_Param':
                                [clf.min_samples_leaf],
                                'min_samples_split_Param':
                                [clf.min_samples_split],
                                'Bootstrap':
                                [clf.bootstrap],
                                'Inner Fold':
                                [j],
                                'featSet':
                                [subSetNo]})

                                # Add data from this fold to that
                                # from previous folds
                                allSummaryData =\
                                  pd.concat([allSummaryData,
```

```
                                        foldData],
                                      axis=0, join='outer')

                                    # gather all the categorisations for
                                    # each sound
                                    allCatResults =\
                                      pd.concat([allCatResults, result],
                                      axis=0, join='outer')

                                    # gather information on features
                                    imps = pd.DataFrame(clf.
                                        feature_importances_)
                                    featimportance =\
                                      pd.concat([featimportance, imps.T],
                                      axis=0, join='outer')

    j += 1

# In[1313]:
#################
### STEP 6.5 ###
#################

################################################
## EVALUATE PARAM PERFORMANCE ON BEST DATASET, ##
########### CHOOSE FINAL MODEL ##################
################################################

# Define column order
defCols = ['FG_Prec', 'FG_Recall', 'FG_F1', 'FG_Support',
  'notFG_Prec', 'notFG_Recall', 'notFG_F1', 'notFG_Support',
  'Acc_Train', 'Acc_Val', 'Confusion Matrix', 'Inner Fold',
```

```
  'no_estimators_Param', 'max_depth_Param',
  'max_features_Param', 'min_samples_leaf_Param',
  'min_samples_split_Param', 'Bootstrap', 'featSet']

# Reorder columns
allSummaryData = allSummaryData[defCols]
filename = "RF_FG_Train_Validation_FIX.xlsx"
# Write data to excel
writer = pd.ExcelWriter(filename)
allSummaryData.to_excel(writer, 'Sheet1')
# Comment in if you want to write to the same excel file
allCatResults.to_excel(writer, 'Sheet2')
writer.save()

# In[1324]:
#################
### STEP 7.1 ###
#################

########################################
####### FINALISED FEATURE SET - ########
####### TRAINING ON TRAIN/TEST #########
########################################

# Automating text output
f = open('final_train_test.txt','w')
sys.stdout = f

# Summary Results table - for comparing models
allSummaryData = pd.DataFrame(columns=['FG_Prec', 'FG_Recall',
  'FG_F1', 'notFG_Prec', 'notFG_Recall', 'notFG_F1',
  'Confusion Matrix', 'FG_Support', 'notFG_Support',
```

```
  'Acc_Train', 'Acc_Val', 'no_estimators_Param',
  'max_depth_Param', 'max_features_Param',
  'min_samples_leaf_Param', 'min_samples_split_Param',
  'Bootstrap', 'featSet'])

# to track classification success by sound - are some sounds
# more difficult to categorise than others
allCatResults = pd.DataFrame(columns=['Sound_Name',
  'EXP2_FGnotFG_CAT', 'Prediction'])

# change this to flip between different feature sets
featSetNumber = subSetNo
thisFeatureSet = newDatasets[featSetNumber] # <<<<<<<
# also changes data for test sounds
testSoundsData = newTestSets[featSetNumber]
trainval_index = trainval_indices[foldFocus]
test_index = test_indices[foldFocus]
X_train = thisFeatureSet.loc[trainval_index]
X_test = testSoundsData.loc[test_index]
y_train = y_trainval_dict[foldFocus]
y_test = y_test_dict[foldFocus]

print('trainval_index: ', trainval_index)
print('test_index: ', test_index)
print('X_train.shape: ', X_train.shape)
print('X_test.shape: ', X_test.shape)
print('y_train.shape: ', y_train.shape)
print('y_test.shape: ', y_test.shape)

# Train a random forest
clf = RandomForestClassifier(n_estimators=10,
                             max_depth=3,
```

```
                                max_features=2,
                                min_samples_leaf=2,
                                min_samples_split=5,
                                bootstrap=False,
                                random_state=0)

# Fit model
clf.fit(X_train, y_train.values.ravel())

# Get model parameters
ModPar = clf.get_params()
print('Model parameters are: ', ModPar)

# How does this model do on the validation set?
y_true, y_pred = y_test, clf.predict(X_test)
class_report = classification_report(y_true,
                    y_pred,
                    output_dict=True)
class_report_ = classification_report(y_true, y_pred)

#with open('text_filename.txt', 'a') as f:
print('################ CLASS REPORT ###############')
print(class_report_)

# Generate accuracy scores
acc = clf.score(X_train, y_train.values.ravel())
acc_2 = clf.score(X_test, y_test.values.ravel())
print('Mean model accuracy on training set: ', acc)
print('Mean model accuracy on test set: ', acc_2)
print('#################################################')

# Generate confusion matrix
```

```
conMat = confusion_matrix(y_true, y_pred)
print("Confusion matrix:\n{}".format(conMat))
print("y_true.shape is:", y_true.shape)
print("y_pred.shape is:", y_pred.shape)

print('################## ##### ########################')
print('################## ##### ########################')
print('################## ##### ########################')

# Printing how the predictor matches with actual
newDf = pd.DataFrame(y_test, columns=['EXP2_FGnotFG_CAT'])
predDf = pd.DataFrame(y_pred, index=y_test.index,
          columns=['Prediction'])
result = pd.concat([soundNames.iloc[y_test.index], newDf, predDf],
        axis=1, join='outer')
print(result)

print('#################################################')
print('#################################################')


foldData = pd.DataFrame({
  'FG_Prec': [class_report['FG']['precision']],
  'FG_Recall': [class_report['FG']['recall']],
  'FG_F1': [class_report['FG']['f1-score']],
  'FG_Support': [class_report['FG']['support']],
  'notFG_Prec': [class_report['notFG']['precision']],
  'notFG_Recall': [class_report['notFG']['recall']],
  'notFG_F1': [class_report['notFG']['f1-score']],
  'notFG_Support': [class_report['notFG']['support']],
  'Confusion Matrix': [conMat],
  'Acc_Train': [acc],
```

```
  'Acc_Val': [acc_2],
  'no_estimators_Param': clf.n_estimators,
  'max_depth_Param': [clf.max_depth],
  'max_features_Param': [clf.max_features],
  'min_samples_leaf_Param': [clf.min_samples_leaf],
  'min_samples_split_Param': [clf.min_samples_split],
  'Bootstrap': [clf.bootstrap],
  'featSet': [featSetNumber]})

# Add data from this fold to that from previous folds
allSummaryData = pd.concat([allSummaryData, foldData], axis=0,
    join='outer')
# gather all the categorisations for each sound
allCatResults = pd.concat([allCatResults, result], axis=0,
    join='outer')

i += 1

# In[1325]:
#################
### STEP 7.2 ###
#################

#################################################################
## PRINT MODEL RESULTS AND DATASETS - TRAINING ON TRAIN/TEST ##
########### PRINT CONTAINS ALL SUB DATASETS USED ##############
#################################################################

# Define column order
defCols = ['FG_Prec', 'FG_Recall', 'FG_F1', 'FG_Support',
  'notFG_Prec', 'notFG_Recall', 'notFG_F1', 'notFG_Support',
    'Acc_Train', 'Acc_Val', 'Confusion Matrix',
```

```
    'no_estimators_Param', 'max_depth_Param', 'max_features_Param',
    'min_samples_leaf_Param', 'min_samples_split_Param',
    'Bootstrap', 'featSet']

# Reorder columns
allSummaryData = allSummaryData[defCols]

filename = "RF_FG_Final_Train_Test.xlsx"

# Write data to excel
writer = pd.ExcelWriter(filename)
allSummaryData.to_excel(writer, 'Model')
# Comment in if you want to write to the same excel file
allCatResults.to_excel(writer, 'Cat_bySound')

for key in newDatasets:
    dSet_fname = 'FeatSet_%d' % (key)
    newDatasets[key].to_excel(writer, dSet_fname)

writer.save()
```

A.3 Experiment 3

A.3.1 Experiment 3 - EGAL Python Code

Python code to implement EGAL algorithm for Experiment 3.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Jun  4 17:15:03 2019

```

```
@author: billcoleman

################################################
########### EGAL IMPLEMENTATION ###############
################################################

CALCULATE DENSITY MEASURE FOR EVERYTHING
The sum of the distances from each instance to all other instances
    within radius alpha

CALCULATE DIVERSITY MEASURE FOR EVERY INSTANCE NOT IN S1
The distance between each unlabelled instance and its closest
    labelled neighbour
(member of S1)

CALCULATE CANDIDATE SET (CS)
All unlabelled instances where diversity measure is greater than beta
"""

import trainTest_functions

import numpy as np
import pandas as pd
from sklearn.metrics.pairwise import euclidean_distances#,
    cosine_similarity
from sklearn.svm import SVC # SVM model

#################
#### DENSITY #### "The sum of the distances from the instance to all
#################        other instances within radius alpha."

```

```
# returns the sum of all the similarities where the similarity value <= alpha
def get_sum_of_density_values(similarities_i, _alpha_):
    """
    similarities_i: similarity matrix - my equivalent of this is pairwiseDist
    alpha: threshold of the density radius
    """
    # check the instances conform to the rule first
    # return similarities_i[similarities_i <= alpha]
    # then return the density for the instance
    return np.sum(similarities_i[similarities_i <= _alpha_])

def get_sorted_density_values(_pairwiseDist, _U, _alpha_):
    # construct a dict to hold density values for each instance its required for
    _density_dict = {}

    for i in range(len(_U)):

        # feed each row in pairwiseDist into a function that filters
        # for values <= alpha
        _density_dict[_U[i]] =\
            get_sum_of_density_values(_pairwiseDist[i, :], _alpha_)

    # may not actually need this because these may get filtered out in
    # find_candidate_set()
    _denseVals_sort = dict(sorted(_density_dict.items(),
                                  key=lambda kv: kv[1],
                                  reverse=False))

```

```
    return _denseVals_sort, _density_dict


####################
#### DIVERSITY #### "The distance between the unlabelled instance and
####################    its nearest labelled neighbour."

# Implementing functionality to make the batch sizes consistent
def find_candidate_set_vIII(L_,
                            U_,
                            density_dict,
                            U,
                            beta_,
                            beta_old,
                            NoL,
                            w,
                            stop_all,
                            loop_counter,
                            S1_index,
                            chroma_X_df):

    # find distance values between labelled and unlabelled sets
    s = euclidean_distances(L_, U_)

    # not dividing 1 by the result here because I can take care of
    # that in the next step taking the minimum here because I need
    # the distance to the NEAREST labelled neighbour
    div = np.min(s, axis=0)

    # making a dataframe that holds density and diversity values
    # per instance in U, we can use this to sort and pick
    # instances by density/diversity values
```

```
94      filtered_dict = list(filter(lambda item: item[0] in U,
95                                  density_dict.items()))
96      # strips away index
97      filtered_dict_dense = [i[1] for i in filtered_dict]
98      filtered_dict_idx = [i[0] for i in filtered_dict]
99
100     make_dict = {'dense':filtered_dict_dense,
101                  'check_idx':filtered_dict_idx,
102                  'U':U,
103                  'diverse':div}
104     _df_denseDiv = pd.DataFrame(make_dict, index=U)
105
106     # if w = 0 then selection is purely diversity based, so take the
107   # largest diversity values
108     if w == 0:
109         print("w = 0 $$$$$$$$$$$$$$$$$$$$$$$$ PURE DIVERSITY")
110
111         cs_df = _df_denseDiv
112         cs_df_idx = cs_df.nlargest(NoL, 'diverse', keep='first')
113
114     # if w is between 0 and 1 then selection is controlled by the
115     # value of w
116     elif w > 0 and w < 1:
117         print("w = ", w, " $$$$$$$$ BTW 0 and 1 $$$$$$$$")
118
119         # Filtering to Candidate Set where diverse value falls within
120         # bounds controlled by beta_
121         cs_df = _df_denseDiv[(_df_denseDiv.diverse > beta_) &\
122                              (_df_denseDiv.diverse <= beta_old)]
123
124         # Find the NoL instances with the largest density values
125         # the index of this dataframe is the index for instances to
```

```
126         # take from U and put into L
127         if cs_df.shape[0] >= NoL:
128             cs_df_idx = cs_df.nlargest(NoL, 'dense', keep='first')
129 
130         # This is therefore the last batch of instances to be
131         # labelled, So get a mini batch that is this size
132         # this is to keep the batch sizes uniform, so if we set NoL
133         # to 10, all the batches for the run will be of size 10
134         # (except the last one)
135         else:
136             mini_NoL = NoL - cs_df.shape[0]
137             incomplete_batch = cs_df
138 
139             if mini_NoL > len(U):
140                 mini_NoL = len(U)
141                 print("LAST BATCH OF INSTANCES TO BE LABELED")
142 
143             # there are no more instances in the candidate set, so we
144             # need to update beta_ before we continue
145             beta_old = beta_
146             beta_ = update_beta(beta_, NoL, w, div)
147 
148             print("beta_ changed to: ", beta_, "<<<<<")
149             print("Mini batch size = ", mini_NoL, "< < < < < <")
150 
151             # minibatch to add to cs_df - keep batch sizes uniform
152             # beta is updated so:
153             # remove instances in incomplete_batch from what you
154             # supply to next function
155             # setup temporary L_ and U_ so we can fill this batch
156             S1_temp = S1_index.copy()
157             S1_temp.extend(incomplete_batch.index)
```

```
158             mini_L_data = chroma_X_df.loc[S1_temp]
159
160             mini_U_idx = U.copy()
161             # remove these instances from U
162             for j in incomplete_batch.index:
163                 # delete the element by value NOT index
164                 mini_U_idx.remove(j)
165
166             mini_U_data = chroma_X_df.loc[mini_U_idx]
167
168             print("Shape of mini_L_data = ", mini_L_data.shape,
169                   "Shape of mini_U_data = ", mini_U_data.shape,
170                   "Length of mini_U_idx = ", len(mini_U_idx))
171
172             # Send mini_L_ and mini_U_ to function to find candidate
173             # set for this minibatch if required
174             if len(mini_U_idx) < 1:
175                 cs_df_idx = incomplete_batch
176             else:
177                 mini_batch_instances =\
178                 complete_this_batch(mini_L_data,
179                                     mini_U_data,
180                                     density_dict,
181                                     mini_U_idx,
182                                     beta_,
183                                     beta_old,
184                                     mini_NoL)
185
186                 cs_df_idx =\
187                 pd.DataFrame(pd.concat([incomplete_batch,
188                             mini_batch_instances],
189                                         # make sure no duplicate
```

```
                                        # index values
                                        verify_integrity=True))

            # check to make sure beta is changing, if it's not then
            # stop the run
            if beta_old == beta_:
                print("Beta_ isn't changing. %%%%%%%%%")
                stop_all = True

    # when w = 1 we don't need to control for beta_
    # so just grab the next NoL instances with the smallest density
    # values in _df_denseDiv
    else:
        print("w = ", w, " $$$$$$$$$$$ PURE DENSITY")
        cs_df = _df_denseDiv
        cs_df_idx = cs_df.nlargest(NoL, 'dense', keep='first')

    return div, s, _df_denseDiv, cs_df, cs_df_idx, beta_, stop_all,\
        loop_counter

# Implementing minibatch functionality
def complete_this_batch(L_, U_, density_dict, U, beta_, beta_old,
                        NoL):
    # find distance values between labelled and unlabelled sets
    s = euclidean_distances(L_, U_)
    # not dividing 1 by the result here because I can take care
    # of that in the next step taking the minimum here because
    # I need the distance to the NEAREST labelled neighbour
    div = np.min(s, axis=0)

    # making a dataframe that holds density and diversity values
    # per instance in U
```

```
    filtered_dict = list(filter(lambda item: item[0] in U,
                                density_dict.items()))
    # strips away index
    filtered_dict_dense = [i[1] for i in filtered_dict]
    filtered_dict_idx = [i[0] for i in filtered_dict]

    make_dict = {'dense':filtered_dict_dense,
                 'check_idx':filtered_dict_idx,
                 'U':U,
                 'diverse':div}
    _df_denseDiv = pd.DataFrame(make_dict, index=U)

    cs_df = _df_denseDiv[(_df_denseDiv.diverse > beta_) &\
                         (_df_denseDiv.diverse <= beta_old)]

    mini_batch_instances = cs_df.nlargest(NoL, 'dense', keep='first')

    return mini_batch_instances

'''
After running this function CS_sort_filt will hold an index to the
next instances to be used for labelling. Take the NoL first
instances as these are the densest, most diverse (outside radius of
beta_) there are.

Update S1. Retrain model. Log score for plot. Get more labels.

Recalculate CS - because we have new instances which will mean the
diversity measure for possible candidates will change. Take the
densest instances of the new CS for labelling.

Keep repeating this until there are no instances in CS.
```

```
254
255 Adjust beta_. Recalculate CS.
256
257 Repeat until beta_stops changing. Stop process.
258 '''
259
260 def update_beta(beta_, NoL, w, diversity_values_U_L):
261     '''
262     When w = 0, EGAL defaults to pure diversity based
263     When w = 1, EGAL defaults to pure density based
264     '''
265     # the numeric in the [::1] controls the order of the sort
266     # slice notation [start here, end here, step (or order)]
267     # change numeric to -1 for reverse order
268     # adding .sort() on the end sorts
269     diversity_values_U_L[::-1].sort()
270
271     # sw roughly gives the index for the slot in the structure that
272     # will have  new beta_
273
274   # Letting _sw = the index relevant to the list of diversity values
275   # This value is controlled by w
276   # It's the index to the new value for beta
277     _sw = w * (len(diversity_values_U_L) + 1)
278
279     # If the index value is < NoL, let beta_ equal to 0
280   # because there are very few instances left in U
281     if _sw < NoL:
282         beta_ = 0
283
284     # let beta_ equal to the value in the slot that splits instances
285     # in U to the proportion dictated by w
```

```
286     else:
287         beta_ = diversity_values_U_L[round(_sw)]
288 
289         if beta_ is None:
290             beta_ = 0
291             print("## beta_ POPPED TO None, RESET to 0 ##")
292 
293     return beta_
294 
295 # Use EGAL algorithm to select instances for labelling
296 def egal_loop_vII(loop_counter, stop_all, U, S1_index, chroma_X_df,
297                   D, L, y_true_list, y_pred_list, trainAcc, testAcc,
298                   labelled_instances, density_dict, beta_, beta_old,
299                   NoL, w, ft, al, y_pred_dec_list, test_data):
300 
301     # convert U to a list so I can use .remove()
302     U = U.tolist()
303 
304     # while there are instances in diversity_values_U_L and stop_all
    is True
305     while len(U) > 0 and not stop_all:
306 
307         print("EGAL LOOP: ", loop_counter, " ::: Feature Set: ", ft,
    " ::: Alpha Method: ", al)
308         print("-------------------------------------")
309 
310         # update diversity values
311         diversity_values_U_L, eucl_distanceS_U_L, CS, CS_sort,\
312         CS_sort_filt, beta_, stop_all, loop_counter =\
313         # vIII maintains cohesive batch sizes
314         find_candidate_set_vIII(chroma_X_df.loc[S1_index],
315                     chroma_X_df.loc[U], density_dict, U, beta_,
```

```
                    beta_old, NoL, w, stop_all, loop_counter,
                                S1_index, chroma_X_df)

        # update S1
        S1_index.extend(CS_sort_filt.index)
        print("Number of labels being added = ",
                CS_sort_filt.shape[0])
        print("Size of S1_index now = ", len(S1_index))

        # remove these instances from U
        for j in CS_sort_filt.index:
            # delete the element by value NOT index
            U.remove(j)

        print("Number of unlabelled instances, U = ", len(U))

        # Sanity stop check
        if len(U) < 1:
            stop_all = True

        if stop_all:
            print("STOP ALL - Getting one last score - loop = ",
                        loop_counter)

        print("----------------------------------------")
        print("::::::: Getting a score ::::::::::::::::")
        print("----------------------------------------")

        # train a model using selected instances to track progress
        U, L, loop_counter, y_true_list, y_pred_list, trainAcc,\
            testAcc, labelled_instances, y_pred_dec_list = \
            trainTest_functions.train_Test_EGAL(chroma_X_df,
```

```
                                S1_index, D, U, L,
                                                loop_counter, y_true_list,
                                                y_pred_list, trainAcc,
                                                testAcc,
                                                labelled_instances,
                                                y_pred_dec_list,
                                                test_data)

        if stop_all:
            break

        loop_counter = loop_counter + 1

    return U, L, loop_counter, y_true_list, y_pred_list, trainAcc,\
         testAcc, labelled_instances, diversity_values_U_L,\
         eucl_distanceS_U_L, CS, CS_sort, CS_sort_filt, beta_,\
         y_pred_dec_list

    print("########## EGAL Loop ENDED #############")
```

A.4 Experiment 4

A.4.1 Experiment 4 - SVM Python Code

Python code to fit parameters to train/validation splits for the SVM algorithm in Experiment 4.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Oct  8 15:14:09 2019
@author: billcoleman
```

```
"""

import pickle
import pandas as pd
import numpy as np
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC # SVM model
from sklearn.metrics import make_scorer, recall_score
from sklearn.metrics import f1_score, accuracy_score, precision_score
from sklearn.metrics import balanced_accuracy_score
# execution time
import sys
import timeit

start_time = timeit.default_timer()

'''
PSEUDOCODE:
    Load in svm zero order data
    Load in different labels (all_labels)
    Load in 3 x random stratified test splits
    Declare scorers - include precision and recall for nonFG
    Declare function to run grid search
    Run grid search in each split
    Generate DF with results of grid search per split
'''

# load lpms data (flattened and scaled)
# local location: '/Volumes/COLESLAW_1TB/ESC/
   LPMS_flat_scaled_EGAL_data_10000.data'
# Kevin Street: '/data/d15126149/datasets/
   LPMS_flat_scaled_EGAL_data_10000.data'
```

```
with open('/Volumes/COLESLAW_1TB/ESC/LPMS_flat_scaled_EGAL_data_10000.data', 'rb') as filehandle:
        # read the data as binary data stream
        allData = pickle.load(filehandle)

allData = pd.DataFrame(allData)
print("Shape of allData: ", allData.shape)

# labels
# local location: '/Volumes/COLESLAW_1TB/scaled_data/all_labels.data'
with open('/data/d15126149/datasets/all_labels.data',
          'rb') as filehandle:
        # read the data as binary data stream
        all_labels = pickle.load(filehandle)

print("Shape of all_labels: ", all_labels.shape)

# indices for 3 random stratified test splits
# local location: '/Users/billcoleman/NOTEBOOKS/EXPERIMENT_4/backup/svm_10000_train_val_indices.csv'
train_val_indices = \
pd.read_csv('/data/d15126149/datasets/svm_10000_train_val_indices.csv')
train_val_indices = pd.DataFrame(train_val_indices)
print("Shape of train_val_indices: ", train_val_indices.shape)
print("Columns of train_val_indices: ", train_val_indices.columns)

# local location: '/Users/billcoleman/NOTEBOOKS/EXPERIMENT_4/backup/svm_10000_test_indices.csv'
test_indices = \
pd.read_csv('/data/d15126149/datasets/svm_10000_test_indices.csv')
test_indices = pd.DataFrame(test_indices)
```

```
print("Shape of test_indices: ", test_indices.shape)
print("Columns of test_indices: ", test_indices.columns)

'''
Storage and scorers
'''
# objects to track scores
modelRows = {} # to track different model parameters and scores
allmodelDF = pd.DataFrame()

# classifier to use in parameter search
svmMod = SVC(class_weight='balanced')

scorers = { # setting up recall and precision as the metrics
    'precision': make_scorer(precision_score, pos_label=1),
    'recall': make_scorer(recall_score, pos_label=1),
    'accuracy': make_scorer(accuracy_score),
    'balanced_accuracy': make_scorer(balanced_accuracy_score),
    'f1': make_scorer(f1_score, pos_label=1)
}

# define function for randomised grid search
def do_svm_GridSearch(train_data, train_labels):

    '''
    Function to execute grid search. The train data and labels need
    to be fed to the function indexed from the allData and
    all_labels objects
    '''
    # Make this available in local scope
    global allmodelDF

```

```
96      # Assign data and labels, so we can feed different splits
97      X_train = train_data
98      y_train = train_labels
99      y_train = y_train.astype('int')
100
101     # Grid search of parameters, using 5 fold cross validation,
102     clf = GridSearchCV(estimator = svmMod, param_grid = param_grid,
103                        cv = 5, n_jobs = -1, # using all cores
104                        scoring=scorers, refit=False, iid=False,
105                        return_train_score=True, verbose = 2)
106
107     # Fit model
108     clf.fit(X_train, y_train)
109
110     test_Prec_means = clf.cv_results_['mean_test_precision']
111     test_Prec_stds = clf.cv_results_['std_test_precision']
112     test_Rec_means = clf.cv_results_['mean_test_recall']
113     test_Rec_stds = clf.cv_results_['std_test_recall']
114     test_f1s = clf.cv_results_['mean_test_f1']
115     test_accs = clf.cv_results_['mean_test_accuracy']
116     test_balAccs = clf.cv_results_['mean_test_balanced_accuracy']
117     train_accs = clf.cv_results_['mean_train_accuracy']
118     train_balAccs = clf.cv_results_['mean_train_balanced_accuracy']
119
120     for te_P_m, te_P_s, te_R_m, te_R_s, te_f1, te_ac, te_bAc, tr_ac,\
121         tr_bAc, params in zip(test_Prec_means, test_Prec_stds,
122                               test_Rec_means, test_Rec_stds,
123                               test_f1s, test_accs,
124                               test_balAccs, train_accs,
125                               train_balAccs,
126                               clf.cv_results_['params']):
127
```

```
        modelRows.update({'Test_Precision': test_Prec_means,
                          'Test_Prec_STD': test_Prec_stds,
                          'Test_Recall': test_Rec_means,
                          'Test_Rec_STD': test_Rec_stds,
                          'Test_F1_Score': test_f1s,
                          'Test_Accuracy': test_accs,
                          'Test_Bal_Accuracy': test_balAccs,
                          'Train_Accuracy': train_accs,
                          'Train_Bal_Accuracy': train_balAccs,
                          'Params': clf.cv_results_['params']})

    modelDF = pd.DataFrame(modelRows)
    allmodelDF = pd.concat([allmodelDF, modelDF], axis=0, join='outer
    ')

    return allmodelDF

'''
Assign data and labels
'''
splits = ['0', '1', '2']
for i in splits:
    print("Beginning split: ", i)
    '''
    Set data and labels
    '''
    train_val_data = allData.loc[train_val_indices[i]]
    train_val_labels = all_labels.loc[train_val_indices[i]]

    '''
    Set Linear Grid
    '''
```

```
    param_grid = {'kernel': ['linear'],
                  'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000]}
    '''
    Call Function
    '''
    pSearch_svm_10000 = do_svm_GridSearch(train_val_data,
                                          train_val_labels)
    # Save File
    pSearch_svm_10000.to_csv('results/psearch_svm_10000.csv')
    print("Linear kernel complete for split: ", i)

    '''
    Set RBF Grid
    '''
    param_grid = {'kernel': ['rbf'],
                  'gamma': [1, 'scale', 1e-1, 1e-2],
                  'C': [0.001, 0.01, 0.1, 1]}
    '''
    Call Function
    '''
    pSearch_svm_10000 = do_svm_GridSearch(train_val_data,
                                          train_val_labels)
    # Save File
    pSearch_svm_10000.to_csv('results/psearch_svm_10000.csv')
    print("RBF kernel complete for split: ", i)

    '''
    Set Poly Grid
    '''
    param_grid = {'kernel': ['poly'],
                  'gamma': [1, 'scale', 1e-1, 1e-2],
                  'C': [0.001, 0.01, 0.1, 1],
```

```
                    'degree': [3, 4]}
    '''
    Call Function
    '''
    pSearch_svm_10000 = do_svm_GridSearch(train_val_data,
                                          train_val_labels)
    # Save File
    pSearch_svm_10000.to_csv('results/psearch_svm_10000.csv')
    print("Poly kernel complete for split: ", i)

    # Track time taken per split
    now_time = timeit.default_timer()
    split_time = now_time - start_time
    # output running time in a nice format.
    mins, secs = divmod(split_time, 60)
    hours, mins = divmod(mins, 60)
    print("Time for this split from start: %d:%d:%d.\n" % (hours,
   mins, secs))

'''
Export final result
'''
pSearch_svm_10000.to_csv('results/psearch_svm_10000.csv')

'''
Timing script
'''
# Track the time it took to run the script
stop_time = timeit.default_timer()
total_time = stop_time - start_time

# output running time in a nice format.
```

```
mins, secs = divmod(total_time, 60)
hours, mins = divmod(mins, 60)

sys.stdout.write("Total running time: %d:%d:%d.\n" % (hours, mins,
    secs))
print("(print)Total running time: %d:%d:%d.\n" % (hours, mins, secs))
```

A.4.2 Experiment 4 - CNN Python Code

Python code to experimenting with architecture and other model parameters for the CNN algorithm in Experiment 4.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Fri Sep 13 08:54:26 2019
@author: billcoleman
CNN for Auditory Hierarchy
Local: /Users/billcoleman/NOTEBOOKS/EXPERIMENT_4/
    EXP_4_CNN10k_DELTA_testSplits_slurm.py
Implemented to train and test the CNN on 100k instances on the same
    splits used
on SVM for comparative purposes.
"""

from __future__ import absolute_import, division, print_function,
    unicode_literals
import tensorflow as tf
from sklearn.metrics import classification_report, confusion_matrix,
    balanced_accuracy_score
import numpy as np
import pandas as pd
```

```
# file management
import pickle
# execution time
import sys
import timeit
# for making directories
import os
from os import path

# train/test splits
from sklearn.model_selection import train_test_split

start_time = timeit.default_timer()

print("This is a script to run CNN models!")

# For 100_000 instances - includes augmented data
# load data (flattened and scaled)
# local location: '/Volumes/COLESLAW_1TB/scaled_data/
    data_cnn_ALLbatches_zOrder.data'
# Kevin Street: '/data/d15126149/datasets/data_cnn_ALLbatches_zOrder.
    data'
with open('/data/d15126149/datasets/data_cnn_ALLbatches_zOrder.data',
     'rb') as filehandle:
        # read the data as binary data stream
        CNN_tensor = pickle.load(filehandle)

# Testing by augmentation batch - index to zero order data
CNN_tensor = CNN_tensor[:,:,:,0]
print("Shape of input data, CNN_tensor, is:", CNN_tensor.shape)

# labels
```

```
# local location: '/Volumes/COLESLAW_1TB/scaled_data/all_labels_100k.data'
# Kevin Street: '/data/d15126149/datasets/all_labels_100k.data'
with open('/data/d15126149/datasets/all_labels_100k.data',
          'rb') as filehandle:
        # read the data as binary data stream
        all_labels = pickle.load(filehandle)
print("Shape of input labels, all_labels, is:", all_labels.shape)

'''
Use this for optimising models - vary the random_state if desired
Switch to .csv indices for results generation
'''
# TRAIN/TEST SPLIT
X_train, X_test, y_train, y_test = train_test_split(CNN_tensor,
                                                    all_labels,
                                                    test_size=0.2,
                                                    random_state
    =3799,
                                                    shuffle=True,
                                                    stratify=
    all_labels)

# ============================================================
# # comment back in to generate results
# # indices for 3 random stratified test splits
# # local location: '/Users/billcoleman/NOTEBOOKS/EXPERIMENT_4/backup
    /svm_100k_train_val_indices.csv'
# # Kevin Street: '/data/d15126149/datasets/
    svm_100k_train_val_indices.csv'
# train_val_indices = \
```

```
# pd.read_csv('/data/d15126149/datasets/svm_100k_train_val_indices.csv',
#             index_col=[0])
# train_val_indices = pd.DataFrame(train_val_indices)
# print("Shape of train_val_indices: ", train_val_indices.shape)
# print("Columns of train_val_indices: ", train_val_indices.columns)
#
# # local location: '/Users/billcoleman/NOTEBOOKS/EXPERIMENT_4/backup/svm_100k_test_indices.csv'
# # Kevin Street: '/data/d15126149/datasets/svm_100k_test_indices.csv'
# test_indices = \
# pd.read_csv('/data/d15126149/datasets/svm_100k_test_indices.csv',
#             index_col=[0])
# test_indices = pd.DataFrame(test_indices)
# print("Shape of test_indices: ", test_indices.shape)
# print("Columns of test_indices: ", test_indices.columns)
#
# # to index to different splits
# splits = ['0', '1', '2']
#
# this_split = 0
#
# X_train = CNN_tensor[train_val_indices[splits[this_split]]]
# y_train = all_labels.loc[train_val_indices[splits[this_split]]]
# X_test = CNN_tensor[test_indices[splits[this_split]]]
# y_test = all_labels.loc[test_indices[splits[this_split]]]
# ============================================================

'''
We will print training sample shape, test sample shape and total
number of classes present. There are 2 classes.
```

```
'''

print('Training data shape : ', X_train.shape, y_train.shape)
print('Testing data shape : ', X_test.shape, y_test.shape)

# Find the unique numbers from the train labels
classes = np.unique(y_train)
nClasses = len(classes)
print('Total number of outputs : ', nClasses)
print('Output classes : ', classes)

'''
Find the shape of input image then reshape it into input format for
training and testing sets. After that change datatypes into floats.
'''

# reshaping to provide right shape to model
nRows,nCols,nDims = 40, 157, 1
train_data = X_train.reshape(X_train.shape[0], nRows, nCols, nDims)
test_data = X_test.reshape(X_test.shape[0], nRows, nCols, nDims)
input_shape = (nRows, nCols, nDims)

train_data = train_data.astype('float32')
test_data = test_data.astype('float32')

# My data is already categorical so probably don't need this
train_labels_one_hot = tf.keras.utils.to_categorical(y_train)
test_labels_one_hot = tf.keras.utils.to_categorical(y_test)
print('Original label 0 : ', y_train.iloc[0])
print('After conversion to categorical ( one-hot ) : ',
      train_labels_one_hot[0])

```

```
# Create Model
def createModel():

    '''
    Now create our model. We will add up Convo layers followed by
    pooling layers. Then we will connect Dense(FC) layer to predict
    the classes. Input data fed to first Convo layer, output of
    that Convo layer acts as input for next Convo layer and so on.
    Finally data is fed to FC layer which try to predict the
    correct labels.

    Initial architecture based on Chen2019, a CNN which achieved
    first place in the DCASE Acoustic Scene Classification
    challenge 2019.
    '''

    model = tf.keras.models.Sequential()

    # Convolution 1
    # The first layer with 14 filters of window size 5x5
    model.add(tf.keras.layers.Conv2D(12, (5, 5), padding='same',
      activation='relu', strides=(2,2), input_shape=input_shape))
    model.add(tf.keras.layers.BatchNormalization())
    model.add(tf.keras.layers.Dropout(0.2))
    model.add(tf.keras.layers.Conv2D(24, (3, 3), padding='same',
      activation='relu', strides=(1,1)))
    model.add(tf.keras.layers.BatchNormalization())
    model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))

    # Convolution 2
    model.add(tf.keras.layers.Conv2D(48, (3, 3), padding='same',
      activation='relu', strides=(1,1)))
```

```
165     model.add(tf.keras.layers.BatchNormalization())
166     model.add(tf.keras.layers.Dropout(0.0))
167     model.add(tf.keras.layers.Conv2D(48, (3, 3), padding='same',
168       activation='relu', strides=(1,1)))
169     model.add(tf.keras.layers.BatchNormalization())
170     model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
171
172     # Convolution 3
173     model.add(tf.keras.layers.Conv2D(56, (3, 3), padding='same',
174       activation='relu', strides=(1,1)))
175     model.add(tf.keras.layers.BatchNormalization())
176     model.add(tf.keras.layers.Dropout(0.0))
177     model.add(tf.keras.layers.Conv2D(56, (3, 3), padding='same',
178       activation='relu', strides=(1,1)))
179     model.add(tf.keras.layers.BatchNormalization())
180     model.add(tf.keras.layers.Dropout(0.0))
181     model.add(tf.keras.layers.Conv2D(56, (3, 3), padding='same',
182       activation='relu', strides=(1,1)))
183     model.add(tf.keras.layers.BatchNormalization())
184     model.add(tf.keras.layers.Dropout(0.0))
185     model.add(tf.keras.layers.Conv2D(56, (3, 3), padding='same',
186       activation='relu', strides=(1,1)))
187     model.add(tf.keras.layers.BatchNormalization())
188     model.add(tf.keras.layers.Dropout(0.0))
189     model.add(tf.keras.layers.Conv2D(96, (3, 3), padding='same',
190       activation='relu', strides=(1,1)))
191     model.add(tf.keras.layers.BatchNormalization())
192     model.add(tf.keras.layers.Dropout(0.0))
193     model.add(tf.keras.layers.Conv2D(96, (3, 3), padding='same',
194       activation='relu', strides=(1,1)))
195     model.add(tf.keras.layers.BatchNormalization())
196     model.add(tf.keras.layers.Dropout(0.0))
```

```
197     model.add(tf.keras.layers.Conv2D(96, (3, 3), padding='same',
198       activation='relu', strides=(1,1)))
199     model.add(tf.keras.layers.BatchNormalization())
200     model.add(tf.keras.layers.Dropout(0.0))
201     model.add(tf.keras.layers.Conv2D(96, (3, 3), padding='same',
202       activation='relu', strides=(1,1)))
203     model.add(tf.keras.layers.BatchNormalization())
204     model.add(tf.keras.layers.Dropout(0.0))
205     model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
206 
207     # Convolution 4
208     model.add(tf.keras.layers.Conv2D(128, (3, 3), padding='same',
209       activation='relu', strides=(1,1)))
210     model.add(tf.keras.layers.BatchNormalization())
211     model.add(tf.keras.layers.Dropout(0.0))
212     model.add(tf.keras.layers.Conv2D(128, (3, 3), padding='same',
213       activation='relu', strides=(1,1)))
214     model.add(tf.keras.layers.BatchNormalization())
215     model.add(tf.keras.layers.Dropout(0.0))
216     model.add(tf.keras.layers.MaxPooling2D(pool_size=(2, 2)))
217 
218     # Pooling
219     model.add(tf.keras.layers.Flatten())
220     model.add(tf.keras.layers.Dense(128, activation='relu'))
221     model.add(tf.keras.layers.BatchNormalization())
222     # model.add(tf.keras.layers.Dropout(0.5)) # added in for 210
223     model.add(tf.keras.layers.Dense(nClasses, activation='sigmoid'))
224 
225     return model
226 
227 # Checking GPU
228 print("Num GPUs Available: ",
```

```
229       len(tf.config.experimental.list_physical_devices('GPU')))
230
231 # To print diagnostics in slurm output
232 # tf.debugging.set_log_device_placement(True)
233
234 # Create model and set some parameters
235 model1 = createModel()
236 batch_size = 128
237 lr = 0.01
238 epochs = 1
239
240 # assign a name to this model - to keep them seperate
241 name = "CNN100k_zORDER_optimising_no100_"  # + str(this_split)
242 namepath = name
243
244 # create the folder to hold model objects if it doesn't already exist
245 if not os.path.exists(os.path.join('models', namepath)):
246         os.mkdir(os.path.join('models', namepath))
247
248 # Declare optimiser - remember 'rmsprop' worked well for 10_000
249 optimiser = tf.keras.optimizers.Adam(learning_rate=lr,
250                                      beta_1=0.9,
251                                      beta_2=0.999,
252                                      amsgrad=False)
253
254 # Compile the model
255 model1.compile(optimizer=optimiser,
256                # use 'categorical_crossentropy' for multi-class
257                loss='binary_crossentropy',
258                metrics=['accuracy'])
259
260 '''
```

```
Checkpoint
'''
# filepath="weights-improvement-{epoch:02d}-{val_accuracy:.2f}.hdf5"
weightspath = os.path.join('models',
        namepath,
        namepath + '.best.hdf5')
checkpoint = tf.keras.callbacks.ModelCheckpoint(weightspath,
        monitor='val_accuracy',
        verbose=1,
        save_best_only=True,
        mode='max',
        save_freq='epoch')
callbacks_list = [checkpoint]

'''
model.summary() is used to see all parameters and shapes in each
   layers in our
models
'''
model1.summary()

'''
After compiling our model, we will train our model by fit() method,
   then
evaluate it.
'''
mod_history = model1.fit(train_data,
                     train_labels_one_hot,
                     batch_size=batch_size,
                     epochs=epochs,
                     verbose=1,
                     validation_split=(0.2),
```

```
                    callbacks=callbacks_list)

mod_evaluate = model1.evaluate(test_data,
          test_labels_one_hot,
          verbose=2)

# save history object for plotting loss and accuracy
with open(os.path.join('models', namepath,
                       namepath + '_hist.data'), 'wb') as file_hi:
    pickle.dump(mod_history.history, file_hi)

print("History object saved")

# serialize model to JSON
model_json = model1.to_json()
with open(os.path.join('models', namepath,
                       namepath + '_model.json'), 'w') as json_file:
    json_file.write(model_json)
print("Model saved to json")

print("Loading best model weights from training run, to evaluate...")
# https://machinelearningmastery.com/save-load-keras-deep-learning-
    models/
# just want to seperate these to load best weights to best_model
best_model = model1
best_model.load_weights(weightspath)
print("Loaded model from disk")

# evaluate loaded model on test data
best_model.compile(optimizer=optimiser,
               loss='binary_crossentropy',
               metrics=['accuracy'])
```

```
score = best_model.evaluate(test_data,
          test_labels_one_hot,
          verbose=2)
print("Best Validation %s: %.2f%%" % (best_model.metrics_names[1],
                                      score[1]*100))

# save evaluate object
with open(os.path.join('models', namepath,
                       namepath + '_eval.data'), 'wb') as file_ev:
    pickle.dump(score, file_ev)
print("Evaluate object saved")

# predict using model and measure precision, recall etc...
y_pred = model1.predict(test_data, batch_size=64, verbose=2)
y_pred_bool = np.argmax(y_pred, axis=1)
print(classification_report(y_test, y_pred_bool))

# Confusion Matrix
print("Confusion matrix:\n{}".format(confusion_matrix(y_test,
                           y_pred_bool)))
bal_acc = balanced_accuracy_score(y_test, y_pred_bool)

print("------------------------------------------------------------")
print('Balanced accuracy on validation set (y_true Vs y_pred): %.2f%%
    ' % (bal_acc * 100))
print("------------------------------------------------------------")

print("This model is: ", namepath)
# Export true and predicted labels for Mcnemar statistical test
y_pred = pd.DataFrame(y_pred, index=y_test.index)
y_pred_bool_ = pd.Series(y_pred_bool, index=y_test.index)
truePred = pd.DataFrame(pd.concat([y_test, y_pred, y_pred_bool_],
```

```
                                axis=1,
                                join='outer'))
truePred.columns=['true', 'pred0', 'pred1', 'pred_bool']
truePred.to_csv(os.path.join('models', namepath,
                        namepath + '_truePred.csv'))
print("Predicted Labels Saved")

# Track the time it took to run the script
stop_time = timeit.default_timer()
total_time = stop_time - start_time

# output running time in a nice format.
mins, secs = divmod(total_time, 60)
hours, mins = divmod(mins, 60)

print("(print)Total running time: %d:%d:%d.\n" % (hours, mins, secs))
```

Appendix B

Model Evaluation

In a supervised learning categorisation task such as that proposed in this work, the classification of isolated sounds on a BG — N — FG scale, the end product of the ML process will be a table that outlines categorisation success and failure which is known as a confusion matrix. This table will compare predicted labels with actual labels and identify whether the model has successfully categorised the test instances or not. An example of a confusion matrix for a binary classifier for 165 instances is provided in Figure B.1 for elucidation.

The relevant values are known as:

- True Positive (TP) — Prediction of YES values for actual YES instances
- True Negative (TN) — Prediction of NO values for actual NO instances
- False Positive (FP) — Prediction of YES values for instances that are actually NO
- False Negative (FN) — Prediction of NO values for instances that are actually YES

Various metrics can then be calculated from these values which give insight into the strengths and weaknesses of the model. This involves the computation of the row and column

n = 165	**Predicted: NO**	**Predicted: YES**	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Fig. B.1 An example of a confusion matrix for a binary classifier.

totals which are used in tandem with the categorisation scores. The following scores equate to the example figures given in Figure B.1.

- **Accuracy** — In total, what percentage of predictions made by the model are correct?:

$$\frac{TP+TN}{Total\ Instances} = \frac{100+50}{165} = 91\%$$

- **Average Class Accuracy (ACA)** — Sometimes referred to as 'balanced' accuracy, where individual class accuracies are averaged.

$$\left(\frac{\frac{TP}{Total\ YES} + \frac{TN}{Total\ NO}}{2}\right) = \left(\frac{\frac{100}{105} + \frac{50}{60}}{2}\right) = 89\%$$

- **Precision** — What percentage of instances predicted as YES are correct?

$$\frac{TP}{Predicted\ YES} = \frac{100}{110} = 91\%$$

- **Recall/True Positive Rate** — What percentage of YES instances are correctly predicted as YES?

$$\frac{TP}{Actual\ YES} = \frac{100}{105} = 95\%$$

- **False Positive Rate** — What percentage of NO instances are incorrectly predicted as YES?

$$\frac{FP}{Actual\ NO} = \frac{10}{60} = 17\%$$

- **True Negative Rate** — What percentage of NO instances are correctly predicted as NO?

$$\frac{TN}{Actual\ NO} = \frac{50}{60} = 83\%$$

- **False Negative Rate** — What percentage of YES instances are incorrectly predicted as NO?

$$\frac{FN}{Actual\ YES} = \frac{5}{105} = 5\%$$

- **Misclassification Rate/Error Rate** — What percentage of model predictions are incorrect?

$$\frac{FP+FN}{Total} = \frac{10+5}{165} = 9\%$$

- **F Score** — A weighted average of the Recall and Precision.

$$\frac{2}{\frac{1}{Recall}+\frac{1}{Precision}} = \frac{2}{\frac{1}{0.91}+\frac{1}{0.95}} = \frac{2}{2.1515} = 93\%$$

Other popular evaluation approaches include the Receiver Operating Characteristic (ROC) and the Area Under the Curve (AUC) metrics. Given a model whose output is a probability distribution between 0 and 1, a decision can be made as to which value is used as a threshold to classify positive and negative instances. A ROC curve is a plot of TP rate against FP rate for the range of different thresholds. The ROC curve then demonstrates how TP and FP rates vary for different threshold values and facilitates choice of the optimal threshold for the application. An AUC value summarises the entire ROC into a single number by calculating the area underneath a ROC curve [180]. The strongest models will appear in the top left hand corner of the ROC curve with more pronounced curves indicating better models and larger AUC scores. This facilitates comparison of classifiers by providing a single digit metric. ROC curves are only applicable to binary classification problems, however, this can be tackled by treating each class performance separately as a binary class membership problem [215].

The decision on which evaluation metric to use will vary on the particular application concerned. In certain cases the overall classification accuracy will be most important, while in other cases it may be far more important to have a highly accurate YES prediction score with

less emphasis on other categories. With regard to the application of hierarchical classification for audio sounds it may arguably be the case that successfully predicting FG sounds is the most important task of the classifier as these sounds may fulfil the requirement for any application in terms of variable audio object delivery.

Appendix C

List of Publications

C.1 Journal Papers

The following publications directly exploit work presented in this document.

1. Coleman, W., Delany, S. J., Yan, M., & Cullen, C. (2020). **A Machine Learning Approach to Hierarchical Categorisation of Auditory Objects.** *Journal of the Audio Engineering Society.* 68(1/2), 48–56.

C.2 Conference Papers

The following publications directly exploit work presented in this document.

1. Coleman, W., Delany, S. J., Cullen, C & Yan, M. (In Review). **Active Learning for Auditory Hierarchy.** *Cross Domain Conference for Machine Learning and Knowledge Extraction (CD-MAKE)*, Dublin, Ireland; 25-28 August, 2020.

2. Coleman, W., Cullen, C., & Yan, M. (2018). **Categorisation of Isolated Sounds on a Background - Neutral - Foreground Scale** *Proceedings of the 144th Convention of the Audio Engineering Society*, Milan, Italy; May 23-26, 2018.

3. Coleman, W., Adams, L., Cullen, C., & Yan, M. (2017). **Perception of Auditory Objects in Complex Scenes: Factors and Applications**. *Institute of Acoustics - 21st Century Developments in Musical Sound Production, Presentation and Reproduction (pp. 1–16)*, Nottingham, UK; November 21st, 2017.

C.3 Other Papers

The following publications constitute other work which has informed the context of this research.

1. Coleman, W., O'Sullivan, L., Cullen, C., & Yan, M. (2017). **sonicPainter: Modifications to the Computer Music Sequencer Inspired by Legacy Composition Systems and Visual Art**. *International Festival and Conference on Sound in the Arts. Science and Technology (ISSTA 2017)*, Dundalk, Ireland; 8-9 September, 2017.

2. Coleman, W., O'Sullivan, L., Cullen, C., & Yan, M. (2017). **iPhone FM Tilter: A Frequency Modulation Instrument for Improvisational Performance using iPhone and Arduino**. *International Festival and Conference on Sound in the Arts. Science and Technology (ISSTA 2017)*, Dundalk, Ireland; 8-9 September, 2017.

3. Cullen, C., & Coleman, W. (2016). **Human Pattern Recognition in Data Sonification**. *6th International Workshop on Folk Music Analysis*, Dublin, Ireland; 15th-17th June, 2016.

Appendix D

List of Employability and Discipline Specific Skills

- Semester 2, 2015/16 - PH6022 Reporting Results in Physical Science - 5 ECTS
- Semester 1, 2016/17 - MED9003 Authoring Principles - 10 ECTS
- Semester 1, 2016/17 - MATH 9102 Probability and Statistical Inference - 5 ECTS
- Semester 2, 2016/17 - GRSO1001 Research Methods - 5 ECTS
- Semester 1, 2017/2018 - MENS 9106 Ensemble 1 - 5 ECTS
- Semester 2, 2017/2018 - SPEC 9270 Machine Learning - 10 ECTS
- Semester 2, 2018/2019 - COMP 9001 Deep Learning - 5 ECTS