CORA Cork Open Research Archive
Cartlann Taighde Oscailte Chorcaí

Title	Data mining and machine learning techniques for neonatal EEG event recall
Authors	Murphy, Brian M.
Publication date	2019-09-30
Original Citation	Murphy, B. 2019. Data mining and machine learning techniques for neonatal EEG event recall. PhD Thesis, University College Cork.
Type of publication	Doctoral thesis
Rights	© 2019, Brian Murphy. - https://creativecommons.org/licenses/by-nc-nd/4.0/
Download date	2026-11-05 18:58:24
Item downloaded from	https://hdl.handle.net/10468/9974

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

Data Mining and Machine Learning Techniques for Neonatal EEG Event Recall

Brian Murphy

Thesis submitted for the degree of Doctor of Philosophy



NATIONAL UNIVERSITY OF IRELAND, CORK

School of Engineering

Irish Centre for Fetal and Neonatal Translational Research (INFANT)

30 September 2019

Head of School: Prof Liam Marnane

Supervisors: Prof Liam Marnane
Dr Gordon Lightbody

Abstract

Sick neonates admitted to the neonatal intensive care unit (NICU) have their physiological signals monitored. In the case of neonates with brain injury the electroencephalogram (EEG), used to record the electrical activity of the brain, is an important diagnostic tool. The EEG is a non-invasive procedure where electrodes are placed on the skin of the head of the neonate. The EEG signals are difficult to interpret and experienced neurophysiologists are required to interpret the EEG and assess the brain health of a neonate. However, there is a lack of expertise available in the NICU to actively monitor all the patients.

There are a wide variety of neonatal EEG patterns that a neurophysiologist must be able to identify to diagnose an encephalopathy and treat a neonate. Some patterns may be rarer than others and require additional time to identify the meaning or cause of the pattern. Neurophysiologists may see a pattern and realise they have seen it before. The difficulty is that they may not be able to recall where they have previously seen the pattern. Currently, the only option is to search through atlases of EEG or prior patient's EEG records to find a similar pattern, which is a time-consuming process.

The main aim of this thesis is the development of a system that assists experts in finding similar EEG events from a database of previously recorded events. The idea is that the system will speed up the time it takes an expert to find where they have previously seen a particular neonatal EEG pattern.

The current state of the art for automated neonatal EEG analysis tools focus on the classification of the signals. These approaches excel at classifying specific signal types such as seizure or sleep states, but they cannot assist the neurophysiologist in finding a prior patient's records that had the most similar EEG pattern type. There is a requirement for a system that will assist experts in locating similar events that have previously occurred. A system like this could speed up the diagnosis of encephalopathies that have a specific morphology.

The first set of data mining techniques developed mimics experts having to physically search back through old records. To achieve this, systems were developed that look through the entire database of events to find the closest matching event. Distance metrics are used to determine the best match. Two distance metric systems were developed, the first was the fixed point to point Euclidean distance and the second was the elastic dynamic time warping (DTW) distance.

The second set of data mining techniques developed move towards systems that do not need to examine every event in the database, while maintaining the recall accuracy. This is of particular interest as the amount of data grows because it becomes infeasible to compare the query event to every event in the database. The particular systems developed, generate hashes from the data and these hashes are then used to find a match. A hash is an alternative and compressed representation of the original data. Three different hashing techniques were developed for use with neonatal EEG.

The final section of the thesis is in the area of machine learning and it focuses on the development of two multi-class classifiers to classify different neonatal EEG

event classes. As it is expensive and time consuming for a neurophysiologist to evaluate neonatal EEG, a proxy system was developed to evaluate the approaches developed in this thesis. As opposed to finding the nearest matching event, the proxy used was that of a multi-class classifier problem.

The work in this thesis shows that neonatal EEG recall systems are possible. They can be quicker than having a neurophysiologist physically search for the most similar signal. This thesis highlights the importance of compression and shows why brute force search strategies will not scale well. The strength of hashing systems in terms of recall accuracy, query speed and memory requirements are also shown.

Contents

Abstract ii

Declaration viii

Acknowledgements ix

List of figures xi

List of tables xvi

List of abbreviations xvii

List of symbols xx

1 Introduction 1
- 1.1 Introduction . . . 1
- 1.2 Prior work in the area of automated analysis of neonatal EEG . . . 3
  - 1.2.1 Seizure detection . . . 3
  - 1.2.2 Sleep states . . . 5
  - 1.2.3 HIE grading . . . 6
  - 1.2.4 Prior work summary . . . 6
- 1.3 Aims and scope of the thesis . . . 7
- 1.4 Thesis layout . . . 9
- 1.5 Publications arising from this thesis . . . 11

2 Neonatal EEG 12
- 2.1 Introduction . . . 12
- 2.2 Electroencephalogram . . . 12
- 2.3 Normal and abnormal EEG . . . 16
  - 2.3.1 Normal characteristics . . . 17
  - 2.3.2 Abnormal characteristics . . . 19
- 2.4 Artifact EEG . . . 21
- 2.5 Database . . . 25
- 2.6 Summary . . . 27

3 Neonatal EEG feature extraction 28
3.1 Introduction . . . 28
3.2 Pre-processing of neonatal EEG . . . 29
3.3 Neonatal EEG feature extraction . . . 29
3.3.1 Time domain . . . 32
3.3.2 Frequency domain . . . 42
3.3.3 Information theory domain . . . 48
3.4 Principal component analysis . . . 54
3.5 Summary . . . 59

4 Brute force nearest neighbour EEG recall 60
4.1 Introduction . . . 60
4.2 Background . . . 60
4.2.1 Data mining . . . 61
4.2.2 Time series data . . . 61
4.2.3 Transforming time series data . . . 64
4.3 Euclidean distance . . . 65
4.3.1 Extending to feature domain . . . 68
4.4 Dynamic time warping . . . 70
4.4.1 Bounding . . . 75
4.4.2 Early abandoning . . . 77
4.4.3 Extending to feature domain . . . 77
4.5 Bag of patterns . . . 80
4.5.1 SAX . . . 81
4.5.2 Transformation . . . 82
4.6 Results . . . 86
4.6.1 Accuracy . . . 88
4.6.2 Query run time . . . 93
4.6.3 Memory requirements . . . 96
4.7 Summary . . . 98

5 EEG self identification using hashing 100
5.1 Introduction . . . 100
5.2 What is hashing? . . . 100
5.3 Review of hashing systems . . . 106
5.3.1 Audio hashing . . . 106
5.3.2 Text similarity . . . 110
5.3.3 Review summary . . . 113
5.4 Spectrogram hashing of neonatal EEG . . . 113
5.4.1 Data compression and hash creation . . . 114
5.4.2 Database development . . . 122
5.4.3 Search algorithm . . . 123
5.4.4 Optimisation . . . 125
5.5 Feature hashing . . . 127
5.5.1 Data compression and hash creation . . . 129
5.5.2 Database development . . . 134

5.5.3 Search algorithm . . . 134
5.6 Locality sensitive hashing . . . 136
5.6.1 Data compression . . . 137
5.6.2 Hashing . . . 140
5.6.3 Database development . . . 142
5.6.4 Search algorithm . . . 142
5.7 Results . . . 144
5.7.1 Optimisation . . . 145
5.7.2 Accuracy . . . 147
5.7.3 Query run time . . . 150
5.7.4 Memory . . . 153
5.7.5 Database hits . . . 155
5.8 Summary . . . 158

6 Multi-class classification of neonatal EEG 159
6.1 Introduction . . . 159
6.2 Classifier background . . . 159
6.2.1 Classifiers . . . 161
6.2.2 Classifiers summary . . . 166
6.3 K-nearest neighbour . . . 167
6.4 Gaussian mixture model . . . 169
6.4.1 K-means clustering . . . 171
6.4.2 Expectation-maximization . . . 175
6.5 Performance . . . 176
6.5.1 Model selection . . . 176
6.5.2 Accuracy . . . 182
6.5.3 Query run time . . . 183
6.5.4 Memory . . . 184
6.5.5 Patient independent analysis . . . 186
6.6 Summary . . . 188

7 Conclusions and future work 190
7.1 Introduction . . . 190
7.2 Conclusion . . . 191
7.2.1 Brute force distance metric . . . 192
7.2.2 Hash based algorithms . . . 194
7.2.3 Scalable multi-class classifiers . . . 197
7.2.4 Summary . . . 198
7.3 Future work . . . 198

Bibliography 201

This thesis was typeset using LaTeX with the `book` class. Images were edited and processed in Inkscape* and imported using the `svg`† package. Times New Roman acts as both the text and display typeface. Courier is used for formatting hyperlinks and code snippets throughout the text.

Data Mining and Machine Learning Techniques for Neonatal EEG Event Recall

30/09/2019, Brian Murphy vii

*Available at `https://inkscape.org`

†Available at `https://ctan.org/pkg/svg`

Declaration

I hereby declare that I am the sole author of this thesis and all of the work undertaken in this thesis is original in content and was carried out by the author. Work carried out by others has been duly acknowledged in the thesis.

Brian Murphy
30/09/2019

Acknowledgements

PhDs are often considered to be a solitary experience, where a person focuses solely on a unique project. In reality this is true but my PhD would not have been possible without the help and support of many others who deserve acknowledgement.

First and foremost I would like to thank Prof Liam Marnane. If not for his initial encouragement for me to undertake this PhD, who knows where I would be. He showed great patience and provided invaluable advice as my supervisor and I'm extremely grateful for the time and effort he put into this thesis.

I would also like to thank Dr Gordon Lightbody for being my second supervisor. He always made time available when I needed it and shared a wealth of knowledge at each meeting. I am very thankful for all the thesis and paper reviewing throughout the years and it was a pleasure to demonstrate the control labs.

This PhD was undertaken with the INFANT research centre with funding provided by Science Foundation Ireland, without this financial aid this PhD would not have happened. INFANT focuses on maternal and child health, it is led by Prof Geraldine Boylan who I would like to thank for providing insights and showing genuine interest in the project. All the staff and students in INFANT were great throughout the few years. The INFANT post-grad group allowed me socialise with people in a similar boat but vastly different research areas.

I would like to thank my external examiner Prof Christopher James and internal examiner Dr Barry Hayes for examining my work. Most likely, the last people who will look at this thesis in great detail.

I undertook my PhD in the Electrical and Electronic building in UCC and I want to thank all the staff and in particular Ralph O'Flaherty for many interesting conversations and answering queries I had, Niamh O'Sullivan, Mary O'Leary and Claudia Cashman for their administrative support and Hillary Mansfield for all his help through my years of demonstrating control labs.

I would like to thank all the current and past PhD students that I have known, who sacrificed so much of their precious time in the fulfilment of PhDs. There were many good social times, trips away and "intellectual" conversations during tea, which I am grateful for, particularly during the writing stage of the thesis. These include George, Alison, Oksana, Haiyang, Rehan, Megan, James, Kevin, Brendan, Adrian, Ed, Rob,

Diarmaid, Conor, Alex, Mark and Sergi.

To my family, thanks for all the support and always being there if I ever needed anything. My parents were particularly supportive in any decision I made and were very encouraging throughout the PhD. My nieces and nephew provided fun and silliness whenever I got the opportunity to see them. The energy they possess is electric, when they are in good form anyway.

Finally and most importantly I would like to thank Niamh. She suffered throughout the PhD, from beginning to end, listening to me talk about the same PhD stuff over and over again. She has the patience of a saint. I am also grateful for her proof reading of this thesis, which I am sure she regretted agreeing to. The support and encouragement she provided really helped me throughout the PhD, I am truly indebted to her.

List of figures

1.1 General outline of the pattern identification system 9

2.1 The structure of a neuron 13
2.2 The 10-20 system of electrode placement. The upper montage is for channel pairs F4-C4, C4-O2, F3-C3, C3-O1, T4-C4, C4-Cz, Cz-C3 and C3-T3. The lower montage is for the channel pars F4-P4, F3-P3, T4-Cz and Cz-T3 15
2.3 Incubator with EEG machine and other monitoring equipment with permission from the INFANT Research Centre 16
2.4 Example showing 30 seconds of the active sleep stage 18
2.5 Example showing 30 seconds of the TA pattern in the quiet sleep stage. 18
2.6 Seizure in all EEG channels of a neonate 20
2.7 Example of burst suppression activity. 21
2.8 Example of the frequency spectrum showing $50Hz$ mains interference 22
2.9 Example of the mains artifact on all EEG channels 23
2.10 Example of the bad electrode artifact on channel T4-C4 23
2.11 Example of the pulsatile artifact on channels C4-Cz and Cz-C3 . . . 24
2.12 Example of the respiration artifact on channels F4-C4, C4-Cz and Cz-C3 25
2.13 Neonate with a short seizure on channel T4-C4. 27

3.1 Pre-processing of the EEG 30
3.2 Line length PDF plot for the six neonatal EEG event types 33
3.3 RMS amplitude PDF plot for the six neonatal EEG event types . . . 34
3.4 Skewness PDF plot for the six neonatal EEG event types 35
3.5 Kurtosis PDF plot for the six neonatal EEG event types 36
3.6 Hjorth parameters PDF plots for the six neonatal EEG event types. (a) is activity. (b) is mobility. (c) is complexity 38
3.7 Zero crossing PDF plots for the six neonatal EEG event types. (a) Time series. (b) First derivative of time series. (c) second derivative of time series 39
3.8 Nonlinear energy PDF plot for the six neonatal EEG event types . . 40

3.9 2^{nd} order autoregressive model fit error PDF plot for the six neonatal EEG event types 41
3.10 (a) 8 second EEG epoch (b) PSD of 8 second epoch 43
3.11 PDF plot for the total power contained within the region of $0-12Hz$ for the six neonatal EEG event types 44
3.12 PDF plots for the power in sub-bands (a) $0-2Hz$ and (b) $3-5Hz$ for the six neonatal EEG event types 45
3.13 PDF plots for the normalised power in sub-bands (a) $0-2Hz$ and (b) $3-5Hz$ for the six neonatal EEG event types 46
3.14 Peak frequency PDF plot for the six neonatal EEG event types . . . 47
3.15 PDF plots for the spectral edge frequency at (a) 80% and (b) 95% for the six neonatal EEG event types 48
3.16 Example PMF plot for a random epoch from a background event . . 50
3.17 Shannon entropy PDF plot for the six neonatal EEG event types . . 51
3.18 SVD entropy PDF plot for the six neonatal EEG event types 53
3.19 Fisher entropy PDF plot for the six neonatal EEG event types 54
3.20 Spectral entropy PDF plot for the six neonatal EEG event types . . . 55
3.21 Example of the PCA process. (a) correlated raw feature data. (b) normalised feature data and eigenvectors. (c) orthogonally separated data 57
3.22 Cumulative variance retained as the number of principal components is increased 58

4.1 Example demonstrating the need for elastic approaches (a) two beats in an ECG, (b) same beats with 16 sample delay. 63
4.2 Example demonstrating how the Euclidean distance is computed for (a) signals with different frequencies, (b) signals with similar frequencies. 67
4.3 Euclidean distance with constant phase delay. 68
4.4 Euclidean flowchart for when a query is being carried out 69
4.5 Cost matrix for 2 example signals. 70
4.6 Accumulated cost matrix with optimal warp path example. 72
4.7 Example demonstrating DTW mapping between time series signals x and q. 73
4.8 (a) Signals before warping, (b) signals warped according to the optimal warping path. 74
4.9 Sakoe-Chiba band and Itakura parallelogram bounding. 75
4.10 Example of the DTW process using a set of features extracted from the EEG signals. 79
4.11 DTW flowchart showing the steps involved in the DTW search algorithm. 79

4.12 SAX computation example using a subsequence p with a length n_s of 100 samples, word length ω of 8 and alphabet size α of 4. 83
4.13 Distribution of data for all subsections of length n_s for each event group. (a) Background, (b) Short seizure, (c) Tracé Alternant, (d) Long seizure, (e) Respiration artifact, (f) Pulsatile artifact. 83
4.14 SAX flowchart 85
4.15 Example time series signals. (a) Background event. (b) Pulsatile artifact event 86
4.16 BOP transformation examples for (a) background event and (b) Pulsatile artifact event 87
4.17 BOP flowchart for when a query is being carried out. 87
4.18 Performance accuracy as the number of PCs used is increased and when the full 55 features are used 92
4.19 Matches obtained when testing both the (a) Euclidean distance (correct match) and (b) DTW distance (incorrect match) on the same query 92
4.20 Database timing graphs for increasing database size for each system evaluated in this chapter 95
4.21 Database memory graphs for increasing database size for each system evaluated in this chapter 97

5.1 Example demonstrating general hashing systems. 102
5.2 Example showing how a key-value pair is added to a hash table. . . 104
5.3 (a) Basic hashing based query idea. (b) Hashing when events create multiple hashes. 105
5.4 An example 8 second window of an EEG time series signal at sampling rate 32 Hz 115
5.5 The spectrogram of the 8 second EEG time series signal seen in Figure 5.4 116
5.6 Spectrogram after application of a peak removal threshold of 71.4% 117
5.7 Spectrogram after application of a peak removal threshold of 100% . 118
5.8 Binary dilation example showing the original image, structuring element used and the resulting dilated image 119
5.9 Spectrogram after the dilation was applied showing the dominating higher amplitude components 120
5.10 Spectrogram with landmarks overlaid 122
5.11 Spectrogram with landmarks overlaid for a longer signal consisting of three cascaded chirp signals from $0Hz$ to $16Hz$ 123
5.12 Spectrogram based method query flow graph showing the steps involved in the hashing process 124

5.13 PSO on a two parameter mathematical function using constriction with 10 particles and 15 iterations. (a) showing after the first iteration. (b) Showing after the 15^{th} iteration. 129
5.14 Frequency hashing algorithm shows the multiple stages involved . . 130
5.15 Example of the hashes generated from the EEG signal and how they are converted into integer numbers 131
5.16 Example showing the difference between two seizure events (a) and (b) which were converted into hash blocks (c) and (d) with the differences between the events shown by (e). 135
5.17 Quantisation based query flow graph for the frequency band quantisation hash method 136
5.18 Example of the LSH sketching process 138
5.19 Example of the LSH shingling process with query that was misaligned 139
5.20 Example of the LSH data transformations. (a) is the sketching and shingling approach. (b) is the BOP approach. (c) is the features set approach 143
5.21 LSH based method query flow graph 144
5.22 Database timing graphs for increasing database size 151
5.23 Database timing graphs for LSH SS and LSH BOP systems 152
5.24 Database timing graph for the LSH feature set approach 152
5.25 Database memory graphs for increasing database size 155
5.26 Database size graph for the LSH feature set approach to show fit . . 156

6.1 KNN example plot. 169
6.2 Example showing how a GMM model is created. 172
6.3 kmeans example plot showing the clustering at multiple iterations with the cluster centres highlighted. The data naturally has four clusters but for this example the aim was to locate three cluster centres. (a) Iteration 1, (b) Iteration 3, (c) Iteration 5, (d) Iteration 9. 174
6.4 EM plot on example data for demonstration. (a) is the initial Gaussian centres positioned close together. (b) is after one iteration of the EM algorithm. (c) is after the algorithm has converged and there now exists a large separation between the Gaussian centres. 177
6.5 Confusion matrix. 179
6.6 Classifier scoring example. 179
6.7 ROC and AUC example plot. 181
6.8 Database timing graphs for increasing database size. 184
6.9 Database memory graphs for increasing database size 185

7.1 General flowgraph for NN algorithms showing the main stages . . . 191
7.2 Taxonomy tree of the brute force approaches showing the main results for the developed algorithms 193

7.3 Taxonomy tree of hashing approaches showing the different areas where algorithms were developed 195
7.4 Taxonomy tree of classifiers showing the different areas 196

List of tables

2.1 Table showing the different EEG bands 16
2.2 Event database 26

3.1 Extracted features 31

4.1 Tables showing optimum (a) and pessimal (b) warping path. 74
4.2 BOP example showing how the *N* events of a fixed size are stored. . 81
4.3 Group majority results for each system evaluated in this chapter . . 90
4.4 Performance accuracy for each system evaluated in this chapter . . . 90
4.5 Slopes of the query timing plots depicted in Figure 4.20 94
4.6 Slopes of the database memory plots depicted in Figure 4.21 98

5.1 Optimised spectrogram parameters 146
5.2 Optimised frequency parameters 146
5.3 Optimised LSH sketch and shingle parameters 147
5.4 Optimised LSH BOP parameters 147
5.5 Group majority results 149
5.6 Performance accuracy 150
5.7 Query time for a database of 10000 events and time difference between a database of 2 and 10000 events 153
5.8 Average number of hashes generated per event 154
5.9 Predicted database size for when there are 10000 events 156
5.10 Average number of database event hits when carrying out a query . . 157

6.1 High level classifier summary 166
6.2 AUC results for the GMM classifier 181
6.3 Classifier group majority results 182
6.4 Classifier accuracy results 183
6.5 AUC results for KNN and GMM classifiers 186
6.6 Classifier group majority results - leave one patient out 187
6.7 Classifier accuracy results - leave one patient out 187

List of abbreviations

AS . . . . . Active Sleep

AUC . . . . . Area Under the Curve

BCI . . . . . Brain Computer Interface

BER . . . . . Bit Error Rate

BOP . . . . . Bag of Patterns

BOW . . . . . Bag of Words

BSF . . . . . Best so Far

CNN . . . . . Convolutional Neural Network

CT . . . . . Computerised Tomography

CUMH . . . . . Cork University Maternity Hospital

DRM . . . . . Digital Rights Management

DTW . . . . . Dynamic Time Warping

ECG . . . . . Electrocardiogram

EEG . . . . . Electroencephalograph

EM . . . . . Expectation Maximisation

EMG . . . . . Electromyography

EOG . . . . . Electrooculogram

FFT . . . . . Fast Fourier Transform

FN . . . . . False Negative

FP . . . . . False Positive

GA . . . . . Gestational Age

GMM . . . . . Gaussian Mixture Model

HIE . . . . . Hypoxic-Ischemic Encephalopathy

HMM . . . . . Hidden Markov Model

IBI . . . . . Interburst Interval

IS . . . . . Indeterminate Sleep

KDD . . . . . Knowledge Discovery in Datasets

KNN . . . . . K-Nearest Neighbours

LDA . . . . . Linear Discriminate Analysis

LOO . . . . . Leave One Out

LSH . . . . . Locality Sensitive Hashing

MAP . . . . . Maximum a Posteriori

MFCC . . . . . Mel Frequency Cepstral Coefficients

MIR . . . . . Music Information Retrieval

MRI . . . . . Magnetic Resonance Imaging

NICU . . . . . Neonatal Intensive Care Unit

NN . . . . . Nearest Neighbour

PAA . . . . . Piecewise Aggregate Approximation

PC . . . . . Principal Component

PCA . . . . . Principal Component Analysis

PDF . . . . . Probability Density Function

PMF . . . . . Probability Mass Function

PSD . . . . . Power Spectral Density

PSO . . . . . Particle Swarm Optimisation

QS . . . . . Quiet Sleep

RMS . . . . . Root Mean Square

ROC . . . . . Receiver Operator Characteristic Curve

SAX . . . . . Symbolic Aggregate Approximation

SEF . . . . . Spectral Edge Frequency

SSA . . . . . Singular Spectrum Analysis

STFT . . . . . Short-Time Fourier Transform

SVD . . . . . Singular Value Decomposition

SVM . . . . . Support Vector Machine

TA . . . . . Tracé Alternant

TN . . . . . True Negative

TP . . . . . True Positive

UBM . . . . . Universal Background Model

List of symbols

Sign	Description	Unit
Ac	Accumulative cost matrix	-
B_x	Bit string	-
D_w	Warping path distance	-
E	Energy band	-
F	Number of features	-
F_c	Batch containing all feature sequences belonging to class c	-
F_n	Number of feature epochs	-
F_{anc}	Anchor point frequency	Hz
F_{eat}	Feature vector	-
G_b	Global best particle position	-
H	Hash family	-
I_w	Inertia weight	-
K	Constriction factor	-
L	Warping path length	-
L_f	Warping path for feature f	-
L_s	Length of shingle	-
M	Number of Gaussian components	-
N	Number of file in the database	-
P	Probability mass function	-
P_{b_p}	Personal best particle position for particle p	-
Pr	Probability	-
Q	Feature representation of q	-
Q_x	Query bitstring	-
S	Number of quantisation hash bits	-
$S\omega_i$	Weight/ count of shingle i	-
S_x	Weighted set of shingles	-
S_x^i	The i^{th} shingle	-
T	Number of spectrogram time windows	-
U	Left singular vectors	-

Sign	Description	Unit
V	Right singular vectors	-
W	Length of random filter	-
X	Feature representation of x	-
$X(j)$	Feature value for the j^{th} epoch	-
Δ	First derivative of time series data	-
ΔF	Difference in frequency	Hz
ΔT	Difference in time	s
Δ^2	Second derivative of time series data	-
Γ	Gamma distributed variable	-
Σ	Covariance matrix	-
Θ	Diagonal matrix of singular values	-
$\aleph_p$	Particle position for the p^{th} particle	-
α	Alphabet size	-
$\bar{p}$	Normalized time series data	-
$\bar{x}_j$	Mean of $\frac{1}{2}$ the j^{th} epoch	-
β	Uniformly distributed variable	-
δ	Sketch step size	-
F	Fingerprint	-
η	Singular values	-
$\hat{\eta}$	Normalized singular values	-
$\hat{x}(k)$	Autoregressive predicted value	-
λ	Eigenvalue	-
μ	Mean	-
μ_f	Template mean	-
ω	Number of symbols in SAX computation	-
$\overline{X_f}$	Normalized feature vector for feature f	-
ϕ_g	Global acceleration constant	-
ϕ_p	Personal acceleration constant	-
σ	Standard deviation	-
σ^2	Activity	-
σ_f	Template standard deviation	-
$\tilde{a}_j$	Normalized frequency spectrum for the j^{th} epoch	-
φ	Autoregressive coefficient	-
$\vec{v}$	Eigenvector	-
a_j	Frequency spectrum for the j^{th} epoch	-
b	Gamma distributed variable	-
c	Cost matrix	-
d_E	Number of singular values	-
e	Warping constraint reach	-

Sign	Description	Unit
f	Feature number	-
f_s	Sampling frequency	Hz
$h(x)$	Hash function for variable x	-
l	DTW lower bound envelope	-
m	Length of q	-
n	Length of x	-
n_T	Number of LSH hash tables	-
n_c	Number of classes	-
n_f	Number of frequency bins	-
n_p	Autoregressive order	-
n_s	Sliding window length	-
p	Time series signal	-
q	Time series signal	-
r	Uniformly distributed random numbers	-
s	Spectrogram	-
t	Time	s
u	DTW upper bound envelope	-
v_p	Particle velocity	-
w	Warping path	-
x	Time series signal	-
x_j	Time series data for the j^{th} epoch	-
y	Normalized time series data	-

Chapter 1

Introduction

1.1 Introduction

In Ireland, there were approximately $62,053$ babies born in 2017 [1], of these, $11,000$ were admitted to the neonatal intensive care unit (NICU) [2]. In the NICU monitoring equipment is attached to the neonate that can check heart rate, blood oxygen saturation, respiration and brain function. There are numerous reasons for babies to be admitted to the NICU, e.g. being born prematurely, having a low blood sugar, infection, trouble breathing or suffering from some form of encephalopathy. An encephalopathy is a disease or disorder of the brain. There are a wide variety of possible causes such as lack of oxygen at birth, infection or genetics [3]. These encephalopathies often result in altered brain activity with seizures commonly occurring for different conditions [4].

Babies in the NICU with a higher risk of encephalopathy have their electroencephalograph (EEG) recorded. EEG records the electrical activity of the brain, the signals are complex and require a neurophysiologist for interpretation. The neurophysiologists need to be available on demand to interpret EEGs in the NICU which is a time consuming and difficult task and can involve searching through old patients' recordings or searching through books such as [4–6]. The work presented in this thesis aims to develop a support tool to assist neurophysiologists in identifying where they may have previously seen a particular EEG morphology. The tool would ideally reduce the search time by returning the medical records of previous patients that had the most similar EEGs morphologies.

The EEG signals enable the brain health and quality to be assessed and it is an excellent diagnostic tool to help identify the health of the neonatal brain [7].

Seizures are often a symptom of some form of encephalopathy and are typically associated with clinical signs such as limb movements and convulsions, although seizures in neonates are challenging to detect as they often do not show clinical signs [8]. When a baby has a sub-clinical seizure, the EEG is the only way to detect the seizure. The most common cause of seizure in neonates is due to Hypoxic-Ischemic Encephalopathy (HIE) [9]. HIE occurs when there is a lack of oxygen to the brain. Strokes are the second most common cause of neonatal seizures and have specific characteristics [10]. A stroke occurs when there is bleeding into brain tissue or when there is an interruption in the blood supply to the brain.

Rare neurological disorders can also cause seizures and irregular brain patterns that can be initially detected from EEG analysis. These rare condition may be difficult to identify and a support tool may assist with identification. Examples of rare epileptic encephalopathies that present early in neonates include Ohtahara syndrome and early myoclonic encephalopathy [11]. The outlook for neonates with these conditions is poor with a high mortality rate, and surviving children are severely incapacitated. Benign familial neonatal seizures are a rare seizure type that typically occurs during the first few weeks of life on an otherwise healthy neonate. They are associated with a previous family history or neonatal seizures [12]. The outlook for neonates with benign familial neonatal seizures is positive with a normal developmental outcome [13]. Pyridoxine-dependent seizures occur in neonates as a form of seizure that does not come under control with conventional treatment via anticonvulsant drugs [14]. In neonates with this rare condition, they are treated by administering doses of pyridoxine which is a type of vitamin B6. It is crucial to treat these neonates fast in order to prevent chronic encephalopathy [14]. It was also stated in [14] that the neonatal EEG might have features that can aid in the diagnosis of this condition. Lissencephaly is a rare brain malformation that can result in a smooth brain surface [15], and which has a very poor prognosis [16]. It is reported that the EEG can be used in order to make a diagnosis of lissencephaly as there are three known patterns associated with the condition [17]. Another neonatal disorder that results in an unusual EEG pattern is Zellweger syndrome. Neonates suffering from this syndrome typically develop severe neurological deficits or death [18]. This syndrome has a particular EEG morphology associated with it that could be used to diagnose the syndrome [19].

There are a wide variety of encephalopathies that exist and many of them have a general EEG morphology. Some of these encephalopathies result in seizure activity

and some result in little brain activity. The EEG captures this brain activity and the neurophysiologists interpret it. There have been attempts at automated EEG analysis tools, however they focus on detecting specific patterns such as seizures and provide no guidance in identifying the underlying encephalopathy.

1.2 Prior work in the area of automated analysis of neonatal EEG

The work presented in this thesis is in the area of automated neonatal EEG analysis. This is an exciting area with many advancements in recent years. These advancements are made while tackling multiple problem areas within the neonatal EEG space. The following will introduce and briefly discuss the areas that are most focused on, which are seizure detection, sleep stage detection and HIE grading.

1.2.1 Seizure detection

As previously mentioned seizures occur frequently in neonates with encephalopathy. It is vital that neonatal seizures are identified as soon as possible so the neonates can be treated accordingly, to prevent any further neurological damage. EEG is the best approach for identifying neonatal seizures [7], though this requires an expert neurophysiologist for interpretation. As neonatal seizures often do not show clinical signs, neurophysiologists are required to monitor the EEG to look for seizure activity. The problem is that the neurophysiologists are often not able to monitor EEG recordings in real time and may identify the seizure activity after it has occurred.

This problem has lead to the development of automated neonatal seizure detection algorithms. Initially, techniques were developed to distinguish neonatal seizures from background activity [20, 21]. Spectral analysis at the cotside has been used [20]. This approach enabled people with little or no experience with EEG to notice changes in the electrical activity of the brain. By using spectral analysis the paper [20] states that seizures are easily confirmed due to increased amplitudes in certain energy bands. This data transformation provided an initial decision support for the detection of seizures. Another approach uses autocorrelation analysis to distinguish background data from seizure activity [21]. As seizure activity in neonates consists of rhythmic discharges of the electrical activity of the brain, they are distinctive from the stochastic normal neonatal EEG. The approach used took advantage of this fact

and used an autocorrelation scoring technique to identify rhythmic seizure activity. A high autocorrelation value indicated the presence of rhythmic seizure activity and a threshold could be applied for the detection of seizures.

In 1997 Gotman proposed an automatic neonatal seizure detection algorithm [22]. This technique was developed to detect a wide range of patterns. The system consisted of three types of EEG analysis which included spectral analysis to find discharges at different frequencies, spike detection and lowpass filtered EEG to locate slow discharges. The features were extracted using 10 second epochs. This approach set the standard for neonatal seizure detection algorithms. Seizures were predicted based on thresholds applied to the extracted features.

A newer method for the detection of seizures in infants which uses extracted features (10 second epochs) and thresholds to determine if the EEG data contains seizure activity was presented in [23]. The method is based on singular spectrum analysis (SSA). The goal of SSA is to decompose the time series signals into oscillatory components and noise. This idea of SSA performing well on quasi-periodic signals was used in order to detect seizure activity. The approach uses singular value decomposition and minimum description length [24] to determine if the singular values are from seizure data or more complex background data.

As there were increases in computational power, the approaches used for neonatal seizure detection became more advanced. The previous approaches described applied data transformations and extracted features from the EEG, and then applied thresholds to distinguish between seizure and non-seizure. The more advanced approaches use extracted features and different types of machine learning classifiers for seizure detection such as convolutional neural networks (CNN) [25, 26], Gaussian mixture models (GMM) [27], hidden Markov models (HMM) [28], support vector machines (SVM) [29] and random forests [30].

Of these approaches, the SVM approach [29] was demonstrated to offer a performance that would be clinically useful [31]. This neonatal seizure detection algorithm is currently seen as state of the art for neonatal seizure detection. It has undergone testing as part of a randomised clinical evaluation called ANSeR [32]. The algorithm initially extracts a set of features from overlapping 8 second epochs of neonatal EEG [33] and then an SVM is used to generate a probability that a seizure is present on any one EEG channel.

All of the previously mentioned classification systems require domain knowledge to manually extract the meaningful features that are used to train the classifiers.

However, recently through the use of CNN, the manually hand-crafted feature extraction step has been removed. This deep learning approach automatically finds features in the data that distinguishes between seizure and non-seizure. It has been shown that deep learning outperforms traditional feature extraction and classification approaches [25]. The caveat is that there needs to be a sufficient amount of training data available and it can take time to identify the best system architecture.

1.2.2 Sleep states

Neonatal sleep analysis is used to assess brain organisation and maturation [34, 35]. The sleep structure for neonates is different from that of adults [36]. In neonates there are two main sleep stages; active sleep and quiet sleep [37]. There are several examples in the literature that developed automated sleep state detection algorithms. Different visual pattern recognition techniques for an automated detection system were investigated for sleep state detection in [38]. The pattern they were aiming to detect was tracé alternant (TA) which is a pattern that alternates between high amplitude mixed frequency activity and low amplitude mixed frequency activity, and it occurs in the quiet sleep stage in term neonates. The discrete wavelet transform was used to identify the TA pattern. This approach was based on using feature extraction to identify the TA pattern. This study was limited in that it had only six neonates. Then in 2004, the idea of using non-linear analysis for the study of sleep phases was presented [39], where the fractal dimension and the mean power of the different spectral bands were used to distinguish sleep states. In [35] an automatic sleep stage detection algorithm that used the neonatal EEG along with other physiological signals was presented. They extracted frequency-based features from the EEG signals and then used a HMM to classify the sleep stages. Another neonatal sleep stage detection algorithm using change point detection and cluster analysis in order to classify brain states was presented in [36].

More recently in 2017 and 2018 automated sleep stage detection algorithms that extracted features and used machine learning classifiers to automatically classify different sleep states [40, 41]. In [40], a system that uses a set of features from the time, frequency and spatial domains which were extracted from 10 minute epochs of the neonatal EEG was presented. The extracted features were used in conjunction with an SVM to classify the sleep states. GMMs and HMMs were evaluated for the classification of sleep states on features extracted from 30 second epochs across

the time, information theory and frequency domains with HMMs having the greater performance [41].

1.2.3 HIE grading

When a neonate suffers a lack of oxygen at birth, this may result in HIE. HIE can cause long term neurological damage and severe cases of HIE can result in death [42]. The background EEG of a neonate with HIE is important as it enables the monitoring of brain recovery [43]. HIE is graded into four different grades, these determine the severity of the brain injury which helps in deciding the course of treatment.

Quantitative EEG features were used to grade the severity of HIE [44]. The quantitative features that were extracted from 64 second epochs and include: amplitude, relative delta power, spectral edge frequency, fractal dimension, skewness, kurtosis, discontinuity, interhemispheric symmetry and synchrony. The features were used with multiple linear least square analysis to grade the EEG. The performance accuracy of the grading system was improved by Stevenson et al. [45] when using features extracted from the amplitude modulated and instantaneous frequency components of the EEG. The features were extracted using windows of 64 seconds with a 50% overlap. The automated grading system used multi-class linear discriminant classifier for classification.

A more recent approach by Ahmed et al. [46] uses a combination of 55 feature vectors from the time, frequency and information theory domains, each extracted from 8 second epochs. These short term features were used to create long term features, 20 non-overlapping short term features which corresponds to approximately 80 seconds are used adapt a GMM EEG background model. The means of the adapted model become the long term features and SVM classifiers are then used to automatically grade the severity of HIE.

1.2.4 Prior work summary

The prior work has shown that the earlier techniques described in this section may have been carried out on smaller amounts of data as proof of concept work such as [21, 38] where limited amounts of EEG were used, 58.5 minutes and 1080 minutes respectively. Following this, large datasets and more advanced systems were developed overtime that were then able to achieve better generalisation performance.

In all of the approaches discussed, features are used to classify signals. Initial approaches use features and thresholds to classify events and then features are used with classifiers to automatically classify events. An observation was made in relation to the length of a signal used to make a classification decision. They range from 8 seconds to multiple minutes and depend on the specific classification problem.

The works described in Section 1.2 emphasised the classification of signals. When classifiers are developed, they use large quantities of data to make classification models for the particular problems. The classifiers aim is to separate the different classes as much as possible. When developing these methods, the unique traits of the particular events may be lost. When the classifiers are trained, the individuality of each event is lost and there is no way to retrieve it as the output of the classifiers are for example a binary decision between two event classes.

1.3 Aims and scope of the thesis

While classifiers are valuable decision support tools, they cannot assist in identifying the underlying cause of some neonatal EEG patterns. The work in this thesis is focused on capturing and representing the individuality of neonatal EEG events. The purpose of such a system is to aid in the faster diagnosis of certain neonatal conditions that may have associated rare EEG patterns such as the conditions which were discussed in Section 1.1. Of these rare conditions, seizures may be common. A seizure detection algorithm may be good at identifying when there is a seizure. The problem is that the seizure detection algorithms will give no indication as to what is the underlying encephalopathy.

A neurophysiologist may see a particular morphology and relate this to something similar from a previous case. The difficulty is that there are a vast number of possible patterns that it could be. They could search through books describing a range of neonatal EEG signals such as [4–6]. They could also search through the large quantity of previously saved EEG recordings to look for the familiar pattern. Both of these approaches are considered as brute force and they are time consuming approaches. For sick neonates, time can be critical especially for seizing neonates as the seizures may cause further brain damage. It is important to try to identify the cause of the different brain conditions so the appropriate treatment can be administered.

The focus of this thesis is the development of a support tool to aid neurophysiolo-

gists in automatically identifying where they may have seen a specific pattern before. The goal is to find the nearest matching signal of a query to a database of EEG events. The idea is that such an approach could reduce the search time for clinicians in finding a similar event. When the most similar event is found, the records belonging to that patient can be examined. These records may contain information about the diagnosis, medication received and other notes that may aid in the treatment of the current patient.

This type of tool has been previously developed and tested on other time series domains [47, 48] but has not been implemented for neonatal EEG. There has been work that focused on different physiological signals [49–52]. The work in this thesis is novel as it focuses on the development and application of algorithms with the primary target being neonatal EEG signals.

Initially, this thesis will investigate brute force systems (searching the query event against every other event in the database) in order to achieve a baseline performance for the closest matching result. The idea is that these approaches are the next step on from having a neurophysiologist perform a brute force search as the computer can search faster. Following a baseline performance, more advanced data mining and machine learning techniques, such as hash-based approaches and multi-class classifiers are investigated to determine if a system with higher accuracy can be achieved while lowering the query time and memory requirements.

Figure 1.1 gives a high-level flowchart of the systems that are developed. There are two stages to each system. The first system involves creating a database of neonatal EEG events. This step involves taking the raw neonatal EEG time series data, pre-processing this data to clean the signals, applying some data compression techniques to convert the neonatal time series EEG data into a different representation. These compressed events are then added to a database in the database generation stage. In the query stage, the same pre-processing and compression steps are applied. Instead of just being added to a database, the query event searches the databases for potentially similar events. Following this, these potentially similar events are sorted from highest to lowest similarity. The closest match would then be returned to the user.

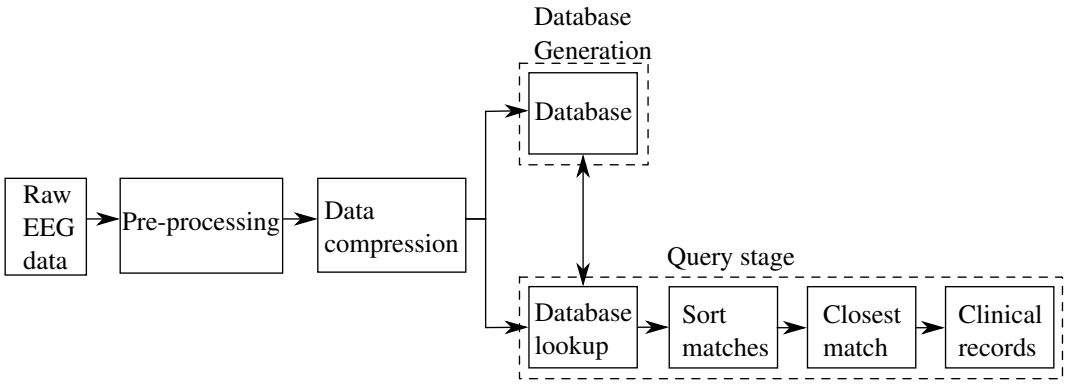


Figure 1.1: General outline of the pattern identification system

1.4 Thesis layout

Chapter 2 aims to introduce and familiarise the reader with neonatal EEG. The EEG, how it is recorded and the normal characteristics will be discussed. The normal healthy EEG of a neonate will then be examined. Following this, the characteristics of abnormal EEG will be detailed, and how these abnormal characteristics indicate that there is a neonatal encephalopathy. EEG that contains artifacts along with the causes of the artifacts will be discussed. The database of neonatal events will then be introduced along with examples of EEG.

Chapter 3 introduces the idea of preparing the EEG signals and extracting meaningful features from the data after the EEG signals are recorded. This preprocessing removes some unwanted higher frequency noise and filters the signals in the frequency range of interest. Once the signals are prepared, the features are extracted to compress the time series representation into a more compressed and meaningful representation. There are 55 features in total and they come from the time, frequency and information theory domains. A dimensionality reduction technique is then discussed.

Chapter 4 explores the novel idea of applying brute force techniques for the task of nearest neighbour EEG recall. In this chapter, novel recall systems are developed using the Euclidean distance and dynamic time warping distance methods. This chapter carries out initial testing on the preprocessed time series data. This chapter then shows that compression techniques are important when working with the stochastic EEG signals. This is done by the development and evaluation of novel neonatal EEG recall systems that are carried out in the feature domain and on the dimensionality-reduced feature set. An alternative data transformation technique to feature extraction was also developed and tested in this chapter for neonatal EEG.

This technique transforms the neonatal EEG into a symbolic representation and then computes the Euclidean distance metric between different compressed events. National and international published peer-reviewed conference paper arose from this work [53, 54].

Chapter 5 introduces the idea of using hashing for neonatal EEG event recall. This chapter describes in detail the novel coupling of neonatal EEG compression strategies and hashing techniques which in turn leads to the development of new neonatal EEG recall systems. These techniques have the advantage of not requiring a brute force search, thus time is saved as the database grows. There are three variations of hashing techniques presented. The first technique uses spectrograms from the preprocessed data to generate hashes from the event. The technique developed to prepare the spectrogram prior to hashing is detailed. The second technique uses the quantification of feature data in order to make hashes. Two quantification approaches are tested. The first approach quantizes the energy in multiple frequency bands and the second approach quantises using a dimensionality reduced feature sebt. The final technique developed in this chapter is an approximate nearest neighbour hashing method that is carried out on data that has been compressed from its original representation using one of three compression techniques. The optimisation strategy is used to ensure the optimal parameters are found for each system detailed in this chapter. Two internationally published peer-reviewed conference papers arose from this work [55, 56].

Chapter 6 introduces machine learning classifiers. A high-level overview of five different classifiers is provided. Following this, a detailed description of the K-nearest neighbours and GMM classifiers is provided. The methods used to develop and expand these classifiers to deal with the multi-class problem is then detailed. There are two types of evaluations carried out on these classifiers, patient specific and patient independent. Both of these evaluations are explained in detail. A one page paper was submitted to and presented as a poster at an international conference, detailing this work.

Chapter 7 presents a summary of the chapters. This chapter includes the main findings for each chapter. Then the best results for the three previous chapters will be detailed and compared. With the results obtained, an overall best system is proposed. Finally, potential areas for some future work are presented.

1.5 Publications arising from this thesis

1. Murphy, B.M. , Boylan, G.B., Lighbody, G., Marnane, W., "Bag of Patterns for Nearest Neighbour Neonatal EEG Recall", *Proceedings of the IEEE Engineering in Medicine and Biology Society Conference, (EMBC)*, IEEE, 2019.

2. Murphy, B.M. , Boylan, G.B., Lighbody, G., Marnane, W., "An Approximate Nearest Neighbour System For Neonatal EEG Recall", *Proceedings of the IEEE Engineering in Medicine and Biology Society Conference, (EMBC)*, pp. 283-286. IEEE, 2018.

3. Murphy, B.M. , Boylan, G.B., Lighbody, G., Marnane, W., "Distance Metric Approach for Nearest Neighbour Recall of Neonatal EEG", *Proceedings of the IEEE Irish Signals and Systems Conference, (ISSC)*, pp. 1-6. IEEE, 2018,

4. Murphy, B.M. , O'Driscoll, C., Korotchikova, I., Boylan, G.B., Lighbody, G., Marnane, W., "Application of Audio Fingerprinting to Neonatal EEG", *Proceedings of the IEEE Engineering in Medicine and Biology Society Conference, (EMBC)*, pp. 912-915. IEEE, 2016,

Chapter 2

Neonatal EEG

2.1 Introduction

The work presented in this thesis focuses on applying different techniques to the EEG recordings from neonates for the task of nearest neighbour (NN) neonatal EEG recall. The EEG detects and records the electrical activity of the brain, which is used to assess and diagnose whether any neurological problems exist. The neonatal EEG recordings used in this thesis are from term neonates that have a gestational age (GA) ranging between 38-42 weeks [4].

An overview of the literature surrounding the different EEG events types along with the medical equipment and international standards will be presented in this chapter. This chapter will start with a high-level overview of how the brain produces electrical signals, the EEG signal and how it is recorded is then discussed. Healthy term neonatal EEG will be discussed. The idea of abnormal EEG is then introduced, and artifact EEG is later described. These should give the reader a high-level overview of the EEG signals. The specific EEG events used in this thesis are then discussed along with why they were chosen.

2.2 Electroencephalogram

Figure 2.1 depicts the structure of a neuron and at birth, the brain of a full-term neonate has billions of neurons. These neurons are responsible for transmitting signals to other neurons and different parts of the body such as muscles. The dendrites on the cell body of the neuron seen in Figure 2.1 receive the signal and it then transfers this signal to other neurons via the terminal branches. This process of signal

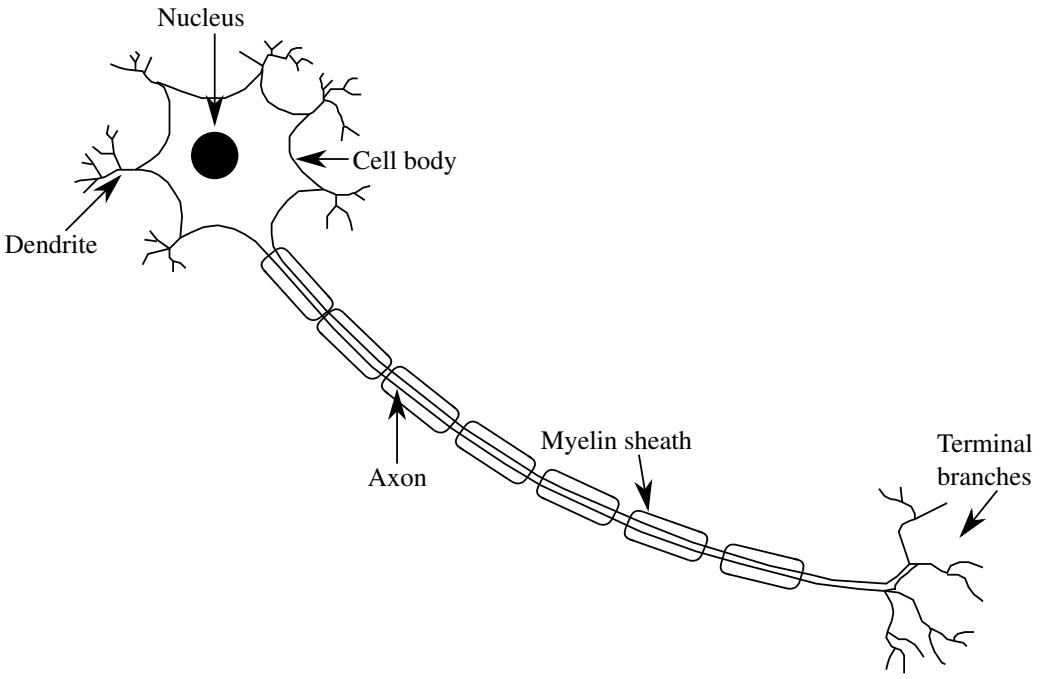


Figure 2.1: The structure of a neuron. The dendrites on the cell body of the neuron receive the signal and it then transfers this signal to other neurons via the terminal branches.

transmission occurs across approximately 100 billion neurons which are present in a term neonate [57].

The EEG captures the continuous voltage fluctuations that are present at the scalp, which occur due to signal transmission from the large number of neurons within the brain [58]. Other methods of examining brain health such as magnetic resonance imaging (MRI) and computerised tomography (CT) scans only provide a snapshot of the brain and cannot be used for the continuous monitoring of brain function. The EEG is a powerful tool that can be used to aid in the diagnosis of different neurological disorders such as seizures [26, 59–61] and HIE [45, 46] along with the rarer conditions such as Pyridoxine-dependent seizures [14], lissencephaly [17] and Zellweger syndrome [19].

The EEG uses electrodes to capture the electrical activity of the brain. The standard approach is to use non-invasive electrodes that are placed on the scalp. Initially, the scalp is prepared by using a paste to remove the upper layer of dead skin from the epidermis to reduce the skin impedance followed by applying a conducting gel to reduce the impedance further so the electrodes can detect the electrical activity. The amplitude of the recorded EEG is typically in the region of μV and varies from patient to patient due to factors including bone density, skin thickness and resistance.

The averaged electrical activity of the neurons are projected onto the scalp. This represents averaged activity from particular regions of the brain. An electrode placement system was developed to record the electrical activity from the different regions of the brain which is known as the 10-20 system of electrode placement [62]. This system sets principals to be followed when positioning electrodes around the head to cover different cortical areas, it is presented in Figure 2.2 where each number and letter used has a specific purpose. The lettering convention comes from the lobes of the brain **F**rontal **p**ole, **F**rontal, **C**entral, **A**uricular, **T**emporal, **P**arietal and **O**ccipital. The left and right hemispheres of the brain have an odd and even numbering convention respectively. The numbers then refer to specific locations within the lobes. The **z** is used for electrodes that are placed on the mid-line between the hemisphere. The 10-20 system was modified for use with neonates as depicted in Figure 2.2.

When evaluating an EEG, it is the potential difference between electrodes that is examined where one of the electrodes is used as a reference electrode. The electrodes are arranged in configurations known as montages. Several different types of montages exist with the most common being the bipolar and referential montages [63]. The bipolar montage arranges the channels in a traverse arrangement in which the second lead in the first channel will then be the first lead in the second channel [63], such as $T4 - \mathbf{C4}$ then $\mathbf{C4} - Cz$. The referential montage then uses a constant reference electrode that will be common in all the electrode pairs. The montages shown in Figure 2.2 are bipolar montages.

The work in this thesis used a bipolar montage as the database of EEG events was annotated using the bipolar montage by the neurophysiologist. The primary reason for using the bipolar montage is because the data used in this thesis was annotated by neurophysiologists who used the bipolar montage. This means the annotations were specifically for the bipolar montage. The channel pairs that were used for the vast majority of the work were F4-C4, C4-O2, F3-C3, C3-O1, T4-C4, C4-Cz, Cz-C3 and C3-T3. This montage is presented in Figure 2.2 (a). In this configuration, it is clear that the electrodes are traversing from the frontal to the occipital lobes and then from the right hemisphere to the left hemisphere in a connected manner. There are some EEG events recorded in this thesis that had fewer electrodes and used the following bipolar montage F4-P4, F3-P3, T4-Cz and Cz-T3. This montage is presented in Figure 2.2 (b). The data used in this thesis was recorded at the Cork University Maternity Hospital (CUMH). The EEG machines that are most commonly used in CUMH are the NicoletOne ICU Monitor, Nihon Kohden Neurofax EEG-

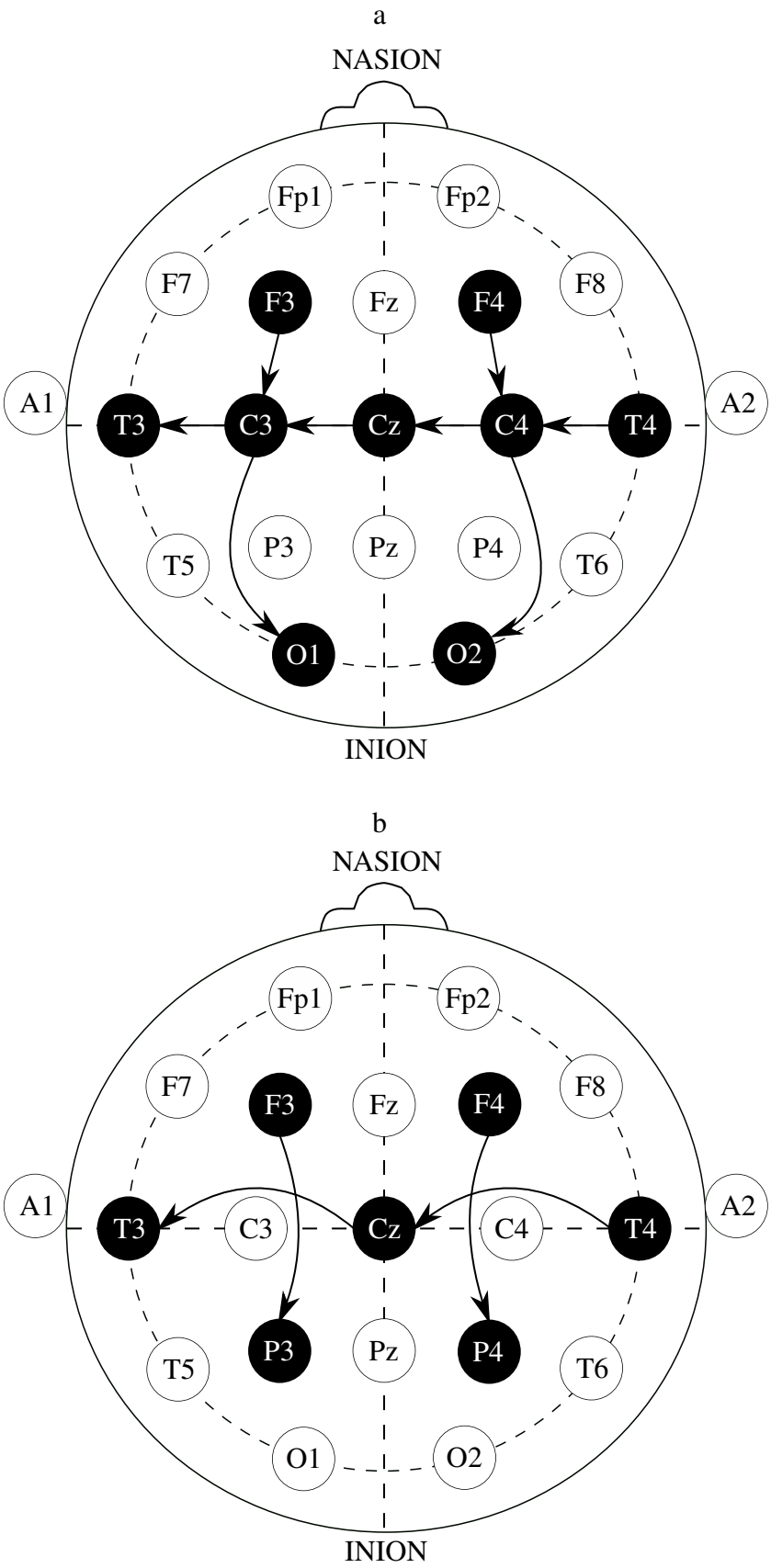


Figure 2.2: The 10-20 system of electrode placement. The upper montage is for channel pairs F4-C4, C4-O2, F3-C3, C3-O1, T4-C4, C4-Cz, Cz-C3 and C3-T3. The lower montage is for the channel pars F4-P4, F3-P3, T4-Cz and Cz-T3

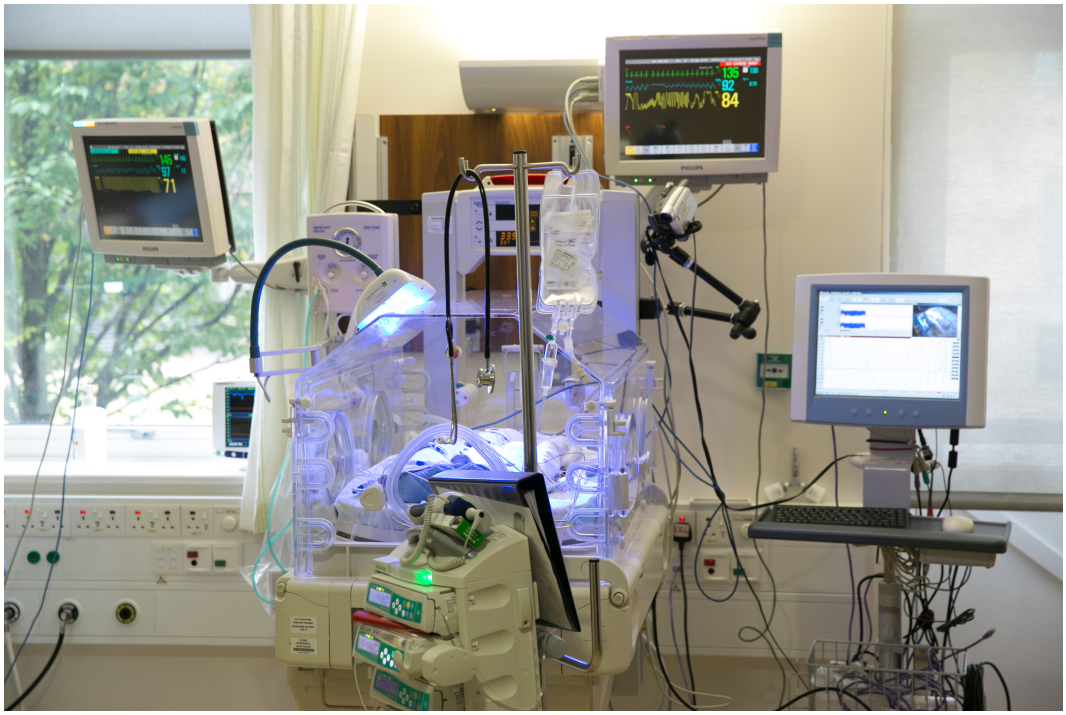


Figure 2.3: Incubator with EEG machine and other monitoring equipment with permission from the INFANT Research Centre

Table 2.1: Table showing the different EEG bands

Name	Frequency range (Hz)
Delta	0 - 4
Theta	5 - 7
Alpha	8 - 13
Beta	13 - 30

1200 Diagnostic and Monitoring Platform and Xltek EEG machine. The commonly used sampling frequency for the EEG signals is $256Hz$. Figure 2.3 shows a baby connected to multiple monitoring machines including an EEG machine which is located at the right.

2.3 Normal and abnormal EEG

The EEG varies with both time and frequency. Four frequency groups of the EEG are used to describe the activity of the signals, and these are presented in Table 2.1 [64]. These frequency bands are used in the literature and by experts to describe characteristics of the EEG [5, 64, 65].

2.3.1 Normal characteristics

Normal healthy background EEG is stochastic with a relatively constant amplitude consisting of mixed frequencies [64] such as that shown in Figures 2.4 and 2.5. Figure 2.4 shows an example of healthy neonatal EEG from the active sleep (AS) stage of a neonate and Figure 2.5 shows an example of healthy neonatal EEG from the quiet sleep (QS) stage. Although healthy EEG is stochastic, it does follow some general pattern types as seen from Figures 2.4 and 2.5, and more examples can be found in [5]. Normal EEG typically has symmetry, meaning the left and right hemispheres should roughly be similar to each other [65]. The range of voltages and frequencies should be roughly similar on both hemispheres. Figure 2.5 shows symmetry across both hemispheres, bursts of activity which have similar amplitudes occur across both hemispheres simultaneously. In a healthy term neonate, the EEG activity should be synchronous [65], meaning for example when a burst of activity happens it should happen across all channels simultaneously. An example of this is also from Figure 2.5 as there are bursts of activity across all channels during the quiet sleep stage. The burst appears to start and finish around the same time in all channels. These bursts of higher amplitude activity followed by the low amplitude sections of EEG are a pattern known as the tracé alternant (TA) pattern which was discussed in Chapter 1. The TA pattern is typical for normal healthy term neonates, and this pattern occurs during the QS stage.

Healthy term neonates sleep for approximately two-thirds of their time spent as a neonate [66]. This sleep consists of four main sleep stages and the amount of time spent in the different sleep stages can assist in determining maturation [35]. These four states were described in [37] and are summarised here;

1. Active sleep - combined theta and delta frequency waveforms with intermixed low-amplitude irregular segments (alpha frequency and faster rhythms) (Figure 2.4)

2. Quiet sleep - a combination of high amplitude slow waves, predominantly in the delta band, and TA segments (beta and theta activity alternating with delta activity) (Figure 2.5)

3. Indeterminate sleep (IS) - features of both AS and QS

4. Wakefulness

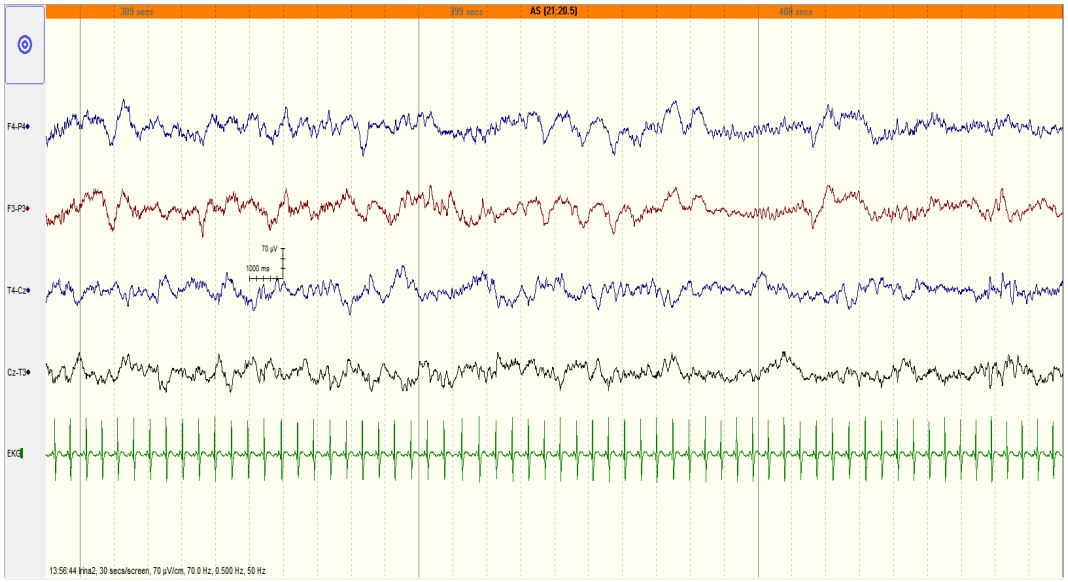


Figure 2.4: A 30 second example of healthy neonatal EEG from the active sleep stage of a term neonate. The first four traces are EEG in the bipolar montage and the last trace is ECG.

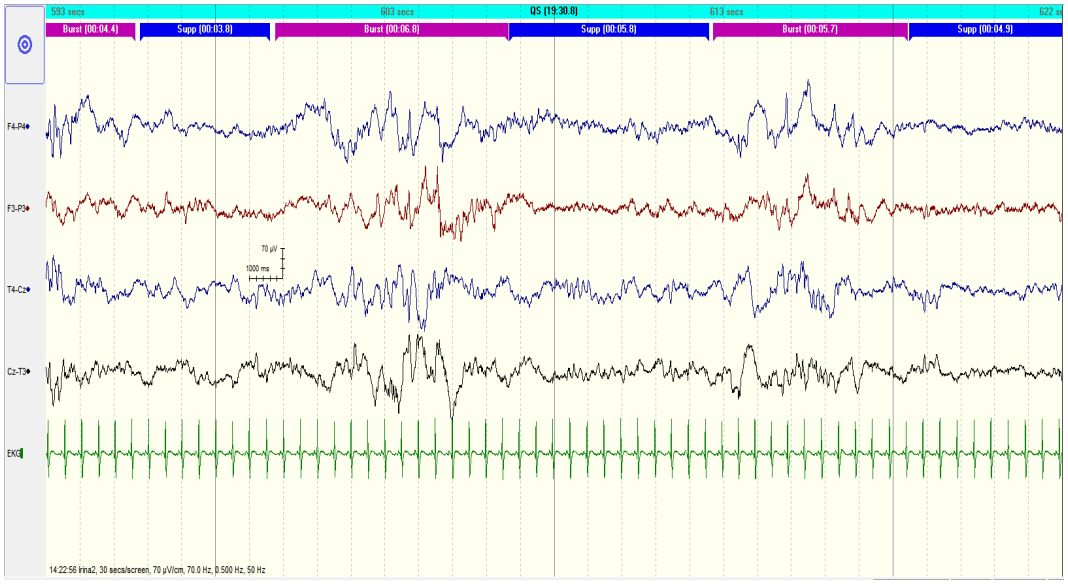


Figure 2.5: A 30 second example of healthy neonatal EEG from the quiet sleep stage of a term neonate. The pattern present is the TA pattern. The first four traces are EEG and the last trace is ECG. On the EEG traces high amplitude bursts of multi-frequency activity interspersed with periods of more suppressed activity can be seen.

Figures 2.4 and 2.5 show examples of the active sleep and quiet sleep stages respectively wherein Figure 2.5 the TA pattern is seen. The "Burst" and "Supp" annotations represent the high and low amplitude mixed frequency activity and do not refer to the burst and suppression activity found in the abnormal EEG of sick neonates. The two classes of EEG used in this thesis that are not abnormal or corrupted with artifact are the classic normal background EEG and the TA pattern. The following section will detail elements of abnormal EEG and provide examples where necessary.

2.3.2 Abnormal characteristics

The first abnormal characteristic to be discussed is the presence of neonatal seizures. These signals represent the most distinctive sign that neurological disorder is present in the neonate [4]. Neonatal seizures are associated with an adverse effect on the neurodevelopmental outcome [67]. Seizures occur when there is an excessive synchronous electrical discharge of neurons within the central nervous system [4]. In neonates that have seizures, the majority of the time these seizures show no clinical signs (66.67%), which is known as subclinical seizures [8]. These seizures need to be identified through examining the electrical activity of the brain using EEG.

The EEG is known as the gold standard for identifying seizures and can be used to assist in quantifying the level of brain damage [7]. The EEG is very complex, difficult to analyse and changes overtime, therefore an experienced neurophysiologist is needed to diagnose the seizures using the EEG recordings [4]. The seizure activity seen on the EEG can show a repetitive and evolving pattern of electrical activity that can be in a specific location (localised) or across the entire EEG (global). The neurophysiologist can locate the start and end time of the seizure activity from the EEG recordings. Figure 2.6 shows an example of a seizure in a neonate; the repetitive discharges of electrical activity are clear. For this particular example, there is seizure activity present across all channels of the montage and this is therefore known as a global seizure.

The next abnormal brain activity to be discussed for term neonates is burst suppression. This activity is normal when seen in preterms and how it evolves overtime is usually an indication of brain health. Term neonates that suffer from HIE also may have this pattern and it can occur in term neonates with rare disorders such as Ohtahara syndrome and early myoclonic encephalopathy [68]. The activity has EEG



Figure 2.6: A 30 second example of rhythmic seizure activity present on all 8 bipolar channels.

bursts separated by abnormally low voltage ($< 5\mu V$) sections called an interburst interval (IBI) [65]. The burst suppression activity has a resemblance to the TA pattern in the QS stage of a term neonate [69]. This resemblance is due to the characteristics of TA pattern which contain high amplitude bursts of multi-frequency activity ranging from 50 - 150 μV, interspersed with periods of more suppressed activity at 25 – 50 μV [65]. Term neonates with an occurring burst suppression pattern typically have a bad projected outcome although some neonates may only have a mild to moderate deficit [70]. Comparing Figures 2.5 and 2.7 the resemblance between the burst suppression activity and the TA pattern can be be seen, although the amplitude of the suppressed sections is much higher for the TA pattern.

There are other shorter duration abnormal characteristics such as sharp wave transients, which are a combination of spikes and sharp waves. Abnormal sharp waves are characterised as high voltage, generally focal, periodic or semirhythmic activity and would appear as a sudden change from the background activity [4]. Sharp waves typically last from 100 to 200 milliseconds and sharp transients that last less than 100 milliseconds are commonly called spikes [65].

There are other indicators of abnormal EEG such as abnormal symmetry and a lack of synchronisation. As opposed to the normal symmetry mentioned in Section 2.3.1, when abnormal symmetry occurs there is a clear difference in the background activity between the two hemispheres. This difference could be an average voltage

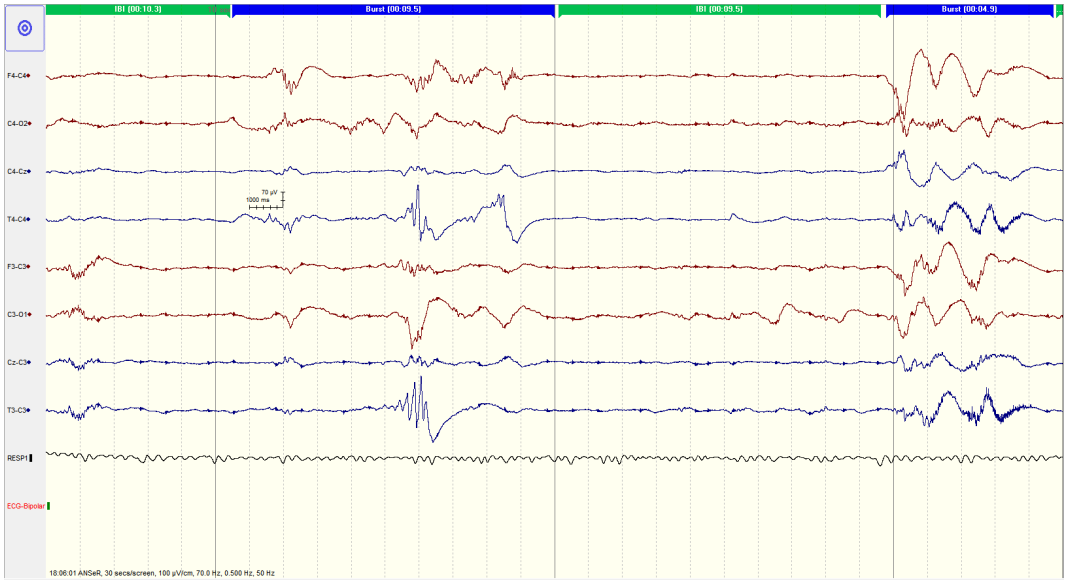


Figure 2.7: Example of burst suppression activity. The bursts are of a short duration and present across all channels. The respiration and ECG traces were not recorded for this patient.

difference between the hemispheres greater than 2 : 1 [65]. The asymmetry events are used in the diagnosis of brain problems as they can occur due to the presence of focal lesions. Asynchrony is when there is a lack of synchronisation. Healthy term neonates should have synchronised activity; the presence of asynchronous activity in a term neonate is abnormal. If the brain activity occurs in some channels and does not occur in other channels within 1.5 seconds, it is considered as asynchronous activity [6, 65]. The asynchronous activity would be present in conditions that cause diffuse encephalopathy [6] where encephalopathy is a disease of the brain that alters the structure or function.

2.4 Artifact EEG

When an EEG is being analysed the experts are concerned mainly for the unexpected behaviour of the EEG, and thus they need to investigate unfamiliar EEG patterns. Some of the anomalies present in neonatal EEG are due to the presence of disturbances in the recording of the EEG. Some of these are obvious although some require more attention to classify as they may resemble seizure or another abnormal event [27, 29, 59].

EEG is recorded in a hospital setting using equipment powered by mains elec-

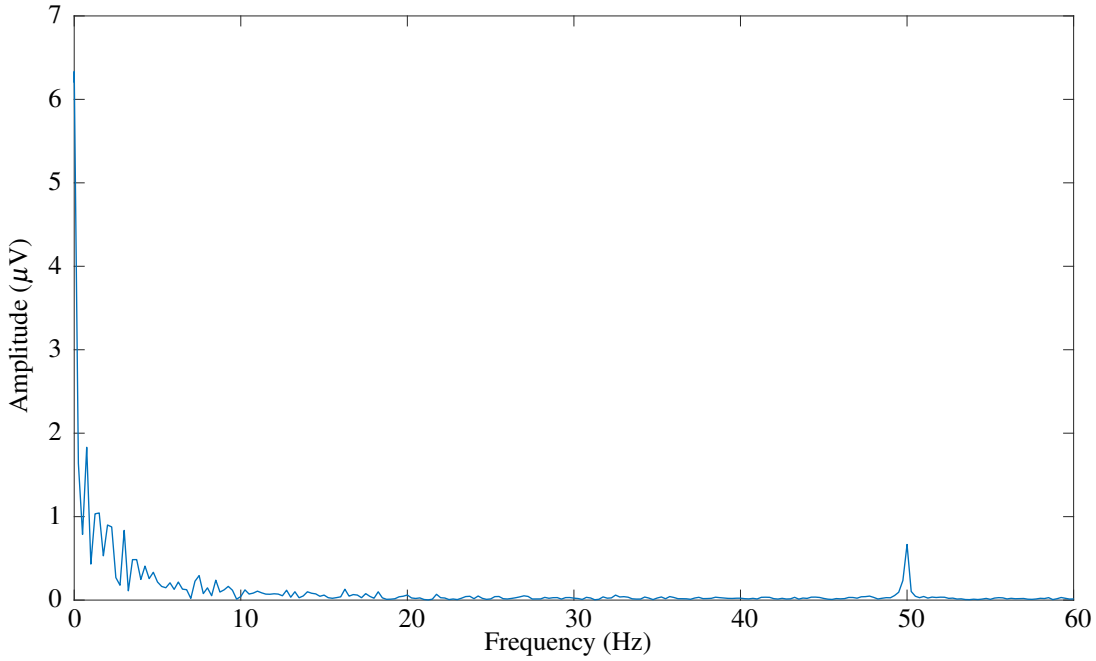


Figure 2.8: Example of the frequency spectrum showing $50Hz$ mains interference which was computed from 60 seconds of EEG. The delta band $(0-4Hz)$ has a high amplitude for this example.

tricity. The frequency of the mains electricity is a common source of electrical interference [71] and may be more prominent on the EEG if there is poor contact between the electrode and the head. This reduced contact then appears as a thickening of the EEG trace. As the mains frequency is known to be $50Hz$ the artifact can be easily removed using a notch filter which is inbuilt in EEG recording machines. When the mains interference is present, it can be seen from the frequency spectrum as seen in Figure 2.8. An interesting observation from Figure 2.8 is that the delta band, described in Table 2.1, has a high amplitude for this particular example. Figure 2.9 shows an example of the mains frequency interference across all the EEG channels. This figure is 5 seconds in duration, and the disturbance is high-frequency activity.

When an electrode detaches or "pops", an artifact is introduced into the EEG. Often the resulting artifact has a high amplitude and is unpredictable as seen in Figure 2.10 where electrode T4-C4 was annotated as a bad electrode. An electrode pop or loose electrode will also pick up the mains frequency more than electrodes with good contact. These electrode detachments have caused false detections in seizure classification systems [27].

When a baby is being patted or stroked an artifact may occur on the EEG as a

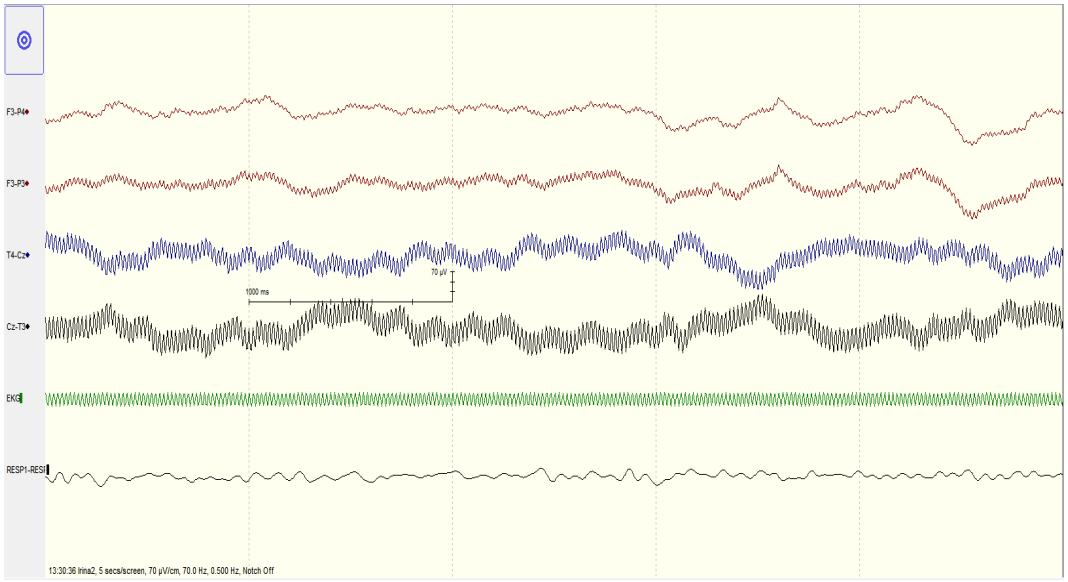


Figure 2.9: A 5 second example of the mains artifact on all EEG channels. The high frequency activity has appeared as a thickening of the EEG trace. The ECG trace (green) was not connected and picked up the mains artifact clearly.

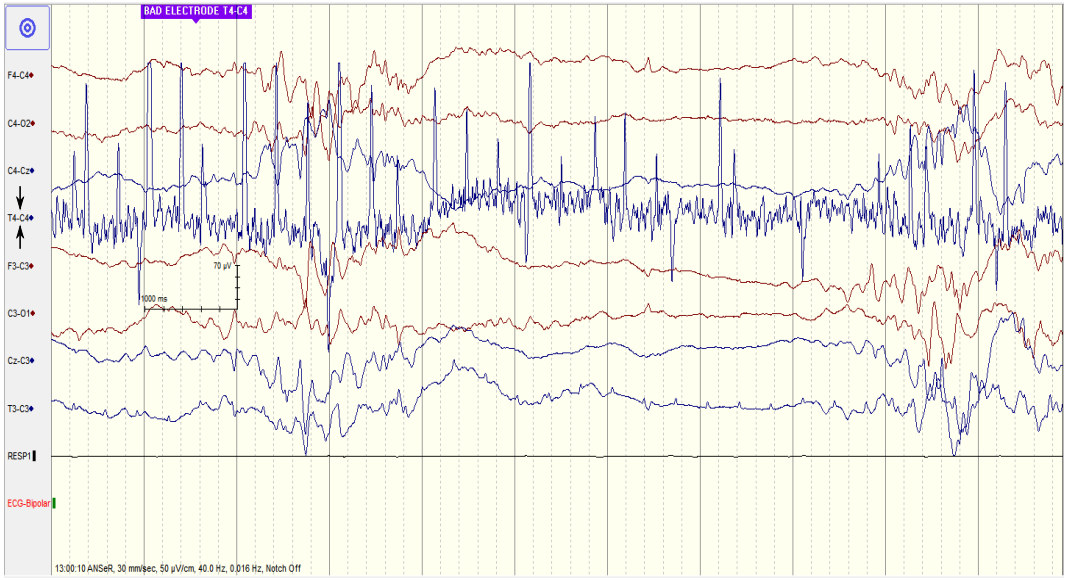


Figure 2.10: Example of the bad electrode artifact on channel T4-C4. This artifact has caused high amplitude and high frequency interference on T4-C4.

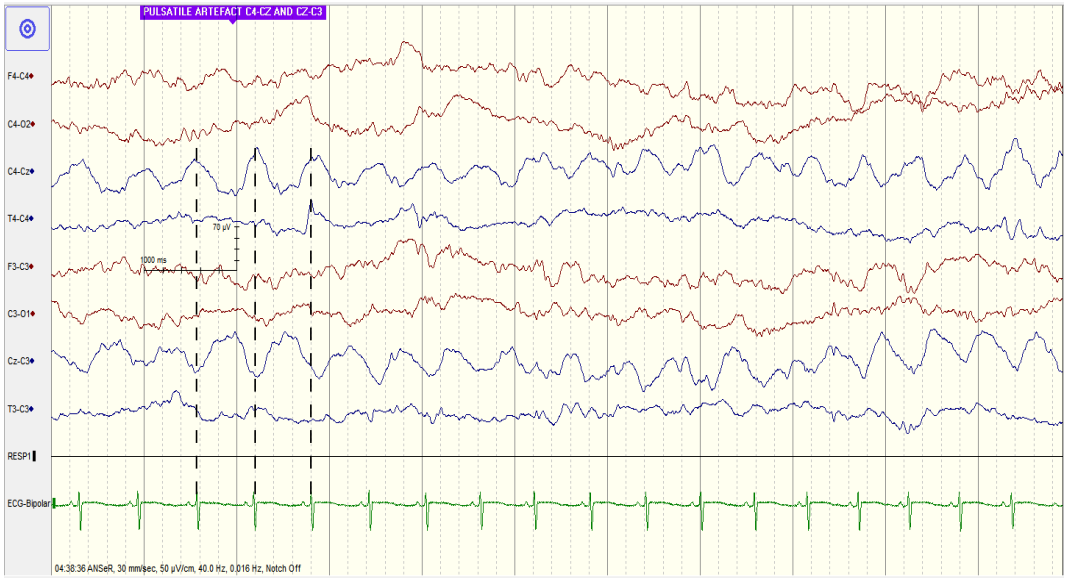


Figure 2.11: Example of the pulsatile artifact on channels C4-Cz and Cz-C3. The pulses can be time locked with the ECG trace as depicted by the overlaying dashed lines.

result. This pattern can look rhythmic and a video recording can be the best way to distinguish this artifact from a seizure event. The muscle activity that occurs when a baby is chewing on a pacifier or crying may cause a fast irregular low amplitude artifact. This type of activity is not normally misclassified as seizure activity.

When an electrode is placed in the vicinity of a pulsing vessel, an artifact may be present on the EEG trace. This artifact is rhythmic and can be identified when looking at the electrocardiogram (ECG) trace [5], which is commonly recorded when a patient is having an EEG recording taken. Figure 2.11 shows an example of the pulsatile artifact. The pulses can be matched to the ECG trace as the overlaying lines indicate.

In some EEG recordings, the movement of a baby when it is breathing can cause an artifact to appear on the EEG trace, which can be hard to detect [72]. This artifact is known as the respiratory artifact and can occur when a baby is on a ventilator or breathing on its own. When the baby is on a ventilator, the pattern is regular and rhythmic. When a baby is breathing on its own, the pattern is less regular but rhythmic and follows the baby's natural breathing pattern. This pattern type is known to be a cause of false detection in seizure detection algorithms [59, 72]. A way in which this artifact is detected in the clinical setting is by adding a respiratory trace, this is done by placing a sensor in the area of the baby's abdomen. Then when there is a suspected rhythmic pattern, it can be compared against this trace [71].

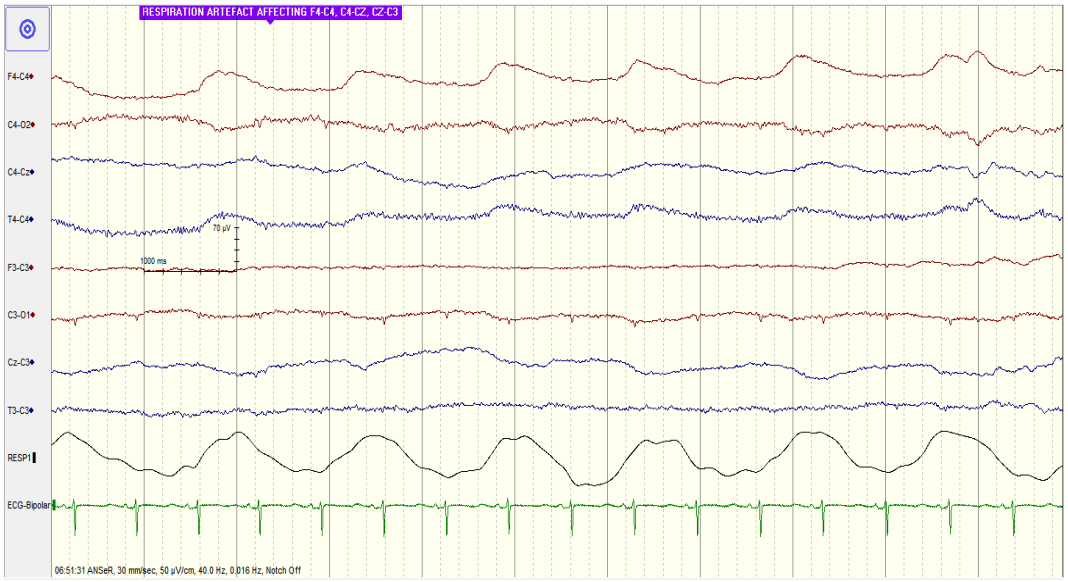


Figure 2.12: Example of the respiration artifact on channels F4-C4, C4-Cz and Cz-C3. The rhythmic pattern of the respiration trace is present on three EEG channels. Channel F4-C4 provides the clearest depiction of the respiration artifact.

Automated approaches for respiration artifact removal have also been developed [72]. The respiration trace and the rhythmic activity of the EEG activity should align if the respiration is the cause of the rhythmic pattern. Figure 2.12 shows an example of this artifact where F4-C4 is the channel that shows the artifact the clearest. The trend of F4-C4 is correlated with the respiration trace (”RESP1”).

2.5 Database

The neonatal EEG data that was used in this thesis came from a variety of different signal patterns including seizure data and sleep data. In seizure classification a decision is made using short duration epochs such as 8 second epochs [29], in sleep state classification a decision is made over longer duration epochs like 10 minute epochs as were used in [40] and in HIE grading a class probability is generated medium duration epochs such as 80 seconds as was discussed in Chapter 1. As different classification systems focus on different fixed time durations to make their decisions, it was decided that the work in this thesis would initially focus on single channel analysis of events that are 60 seconds in duration. Table 2.2 shows the six different pattern event types used along with the number of occurrences of that particular event and the number of patients for each particular event class. There were

50 patients in total with a total of 430 recorded events. Written informed parental consent was obtained, and the study had ethical approval from the Research Ethics Committee of the Cork Teaching Hospitals and the data was fully anonymised.

Table 2.2: Event database

Event number	Event name	Number of events	Number of patients
1	Background	274	18
2	Short seizure	12	7
3	Tracé alternant	40	10
4	Long seizure	23	10
5	Pulsatile artifact	55	2
6	Respiration artifact	26	3

The background events (event 1) came from a mix of patients that were one minute in duration and the annotator deemed them to be normal background activity. The short seizure events (event 2) consists of approximately 25 seconds of annotated seizure activity and 35 seconds of what the annotator deemed to be normal data. Figure 2.13 shows an example of short seizure data, where the seizure is present on channel $T4-C4$ and is highlighted by the pink box located around it. This particular seizure is localised around the area of $T4-C4$ as there is no synchrony with the other channels. This example figure is 60 seconds in duration. The TA events (event 3) is the TA pattern that was previously discussed. Figure 2.5 shows an example of the TA pattern where the high and low amplitude sections are clearly seen. This patient is healthy, and the signals are symmetrical and synchronous. The long seizure events (event 4) come from patients that had annotated seizure episodes that were greater than 60 seconds in duration. Figure 2.6 showed an example of the global seizure activity as it is present across all the recorded channels. The pulsatile artifact events (event 5) and respiration artifact events (event 6) came from only two and three patients respectively. Figures 2.11 and 2.12 are examples of pulsatile and respiration artifacts respectively.

The six different event classes come from three different types of EEG. The background and TA events are considered normal events. The short and long seizures are considered as abnormal events. The respiration and pulsatile events are artifact events. This gives a mix of events across the normal, abnormal and artifact EEG event types which can occur during EEG recordings. Pulsatile and respiration artifacts were chosen as they have the potential to be misclassified as seizures in some seizure detection algorithms [59]. Adding these artifacts tests each system's ability

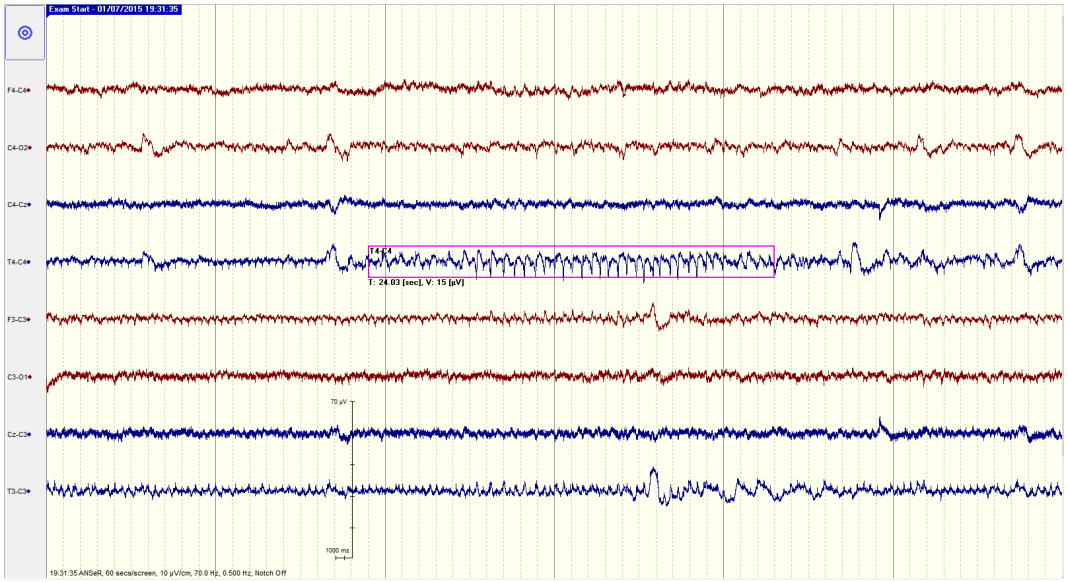


Figure 2.13: EEG with a localised short seizure. This seizure occurs on channel T4-C4, it is approximately 24 seconds in duration and is highlighted by the box.

to identify the same event type correctly.

The work presented in this thesis is aimed at the development and analysis of NN-of-one recall systems. As it is too expensive and time-consuming, an experienced neurophysiologist was not able to assess the NN matches found using the systems. As a proxy, a multi-class classification problem is used, where the performance of the system is evaluated by checking if the recalled event has the same event class as the query. This is the reason why multiple different annotated event types were used.

2.6 Summary

This chapter introduced and discussed the neonatal EEG, from how it was recorded to the different possible signal types that can occur. The high-level overview was provided to introduce the reader to a variety of EEG signals to show the complex nature of the neonatal EEG. The neonatal EEG events database that is used throughout this thesis was also discussed.

Chapter 3

Neonatal EEG feature extraction

3.1 Introduction

As was seen earlier in Chapter 1, automated analysis systems frequently use features. The advantage of using features is that they have been chosen because they discriminate between different conditions or event classes, such as seizure and non-seizure for example [33]. The success of automated systems that use features is the reason why features were used throughout this thesis.

Neurophysiologists develop skills and domain knowledge from extensive training and practical experience that enable them to easily notice changes in both the time and frequency of the signals. This domain knowledge was used to extract meaningful features that quantify the data [33, 73, 74]. These features are used to help encompass what the experts see and use in their evaluations. Before extracting features from the EEG, the EEG must first be preprocessed to clean it by removing unwanted noise and frequencies. Post feature extraction, the features may have their dimensionality reduced to save computational resources.

This chapter is important as it details how the neonatal EEG time series data is represented in the feature space. This feature space representation along with a dimensionality reduction technique was used in the following three chapters of this thesis.

This chapter will describe all the signal processing steps. Initially, the preprocessing steps are discussed, and an example is presented. Following this, the features that were extracted are described. Finally, the technique used for dimensionality reduction is presented with examples.

3.2 Pre-processing of neonatal EEG

Neonatal EEGs are typically recorded using a sampling frequency f_s of $256Hz$ or higher as noted in [27, 37, 75]. As discussed in the previous chapter there are four main frequency bands that capture the most information for neonatal EEG. These bands are alpha $(8-13Hz)$, beta $(13-30Hz)$, delta $(0-4Hz)$ and theta $(5-7Hz)$ waves [64]. The frequency ranges that are of interest are often far below the recorded frequency of $256Hz$ (Nyquist frequency $128Hz$). Having higher frequency signals increases the computational cost and memory requirements of any operations on the recorded data. Signals with frequencies $> 30Hz$ may also have mains frequency interference $(50Hz)$ as well as muscular artifact [76].

For this reason, the signal undergoes a two step process of filtering and downsampling [77]. Step one is to apply a low pass filter to the signal to eliminate the higher frequency components. The low pass filter had a cutoff frequency set to $12.8Hz$, which is used to eliminate aliasing. The second step is to downsample the signal to $32Hz$ (Nyquist frequency $16Hz$). Following this, the signals are high pass filtered with corner frequency set to $0.5Hz$ to remove any low frequency and DC components. An illustration of the pre-processing stage is given in Figure 3.1. In this image, the output of the low pass and high pass operations are coupled into a single band-pass filter. The downsampling provides a significant reduction in the amount of data needed to represent the data accurately. It is more efficient to downsample the EEG in order to reduce the amount of data stored and processed by each of the methods presented in this thesis.

These signals are now in the form of $x = (x_1, x_2, ..., x_n)$ for time $t = (1, 2, ..., n)$, these signals will either be used directly by algorithms which are discussed in later chapters, or features will be extracted from these signals as is discussed next.

3.3 Neonatal EEG feature extraction

Many algorithms exist that work well when applied directly to physiological time series signals [26, 78]. There are also some systems that would typically work better when the data is compressed into a quantitative representation [33, 60, 79]. The features used in this work were not randomly used quantitative features. The set of 55 features were designed with input from domain experts (neurophysiologists) to capture the meaning associated with the EEG signals. This input gave the signals a

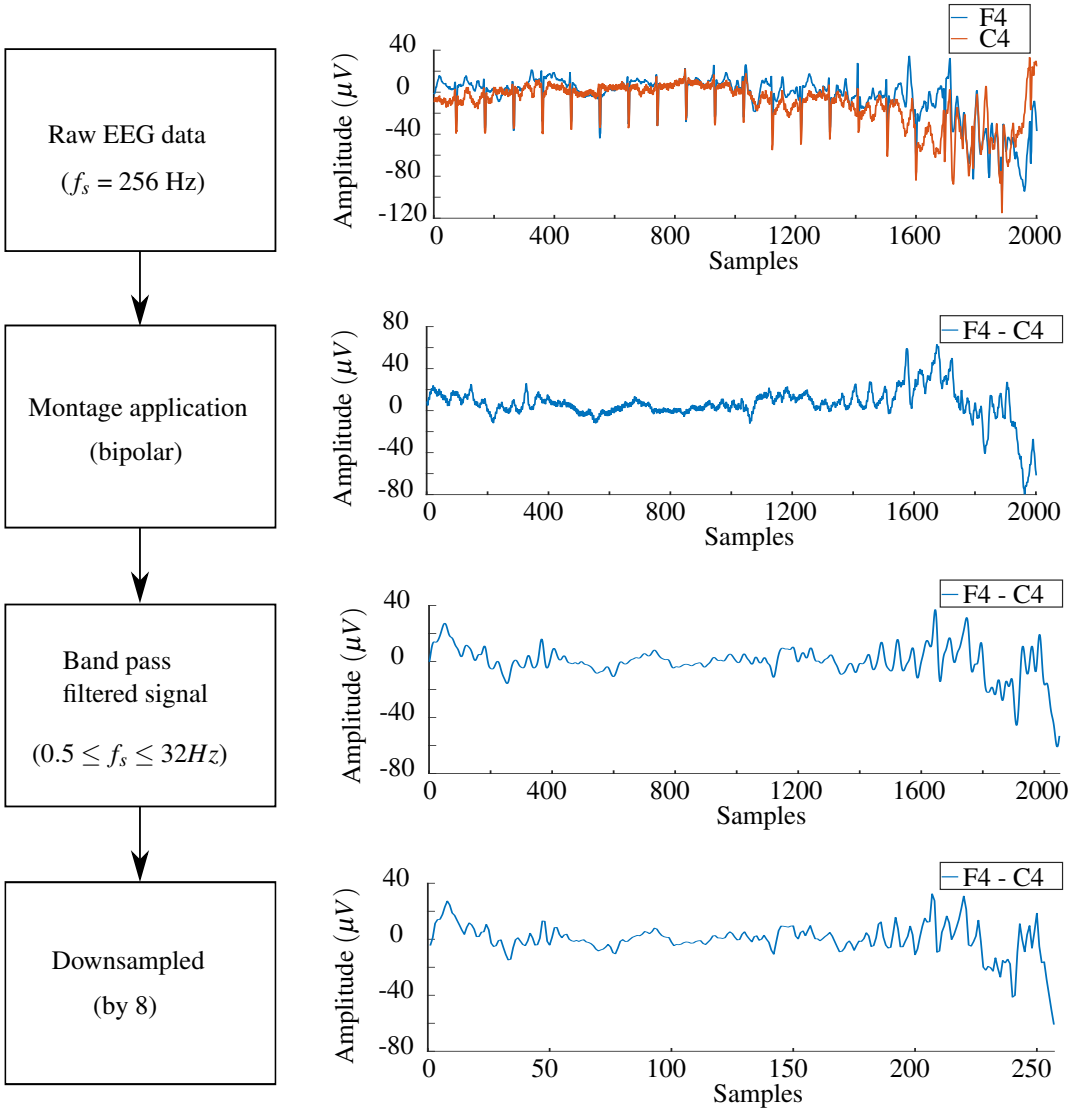


Figure 3.1: The pre-processing stages. Initially the bipolar montage is applied to the individual EEG channels. Following this the bipolar montage channels are band-pass filtered between $0.5Hz$ and $32Hz$. Finally the signals are downsampled by 8 to make the new sampling frequency $32Hz$.

more meaningful representation. The feature set used was previously used in other studies for seizure detection [27, 29, 33, 60], artefact detection [80] and HIE grading [46].

The feature set is drawn from three domains which are the frequency, time and information theory domains. Table 3.1 shows a list of 55 features that are used.

Table 3.1: Extracted features

Domain	Features
Time	• Non-linear line length • Number of maxima and minima • Root mean squared amplitude • Skewness • Kurtosis • Hjorth parameters • Zero crossings (raw epoch, Δ, $\Delta\Delta$) • Non-linear energy • Autoregressive modelling error (model order (1-9)) • Variance (Δ and $\Delta\Delta$)
Frequency	• Total power • Power in 2 Hz sub-bands((0-2Hz), (1-3Hz),. . . ,(10-12Hz)) • Normalised power in subbands • Peak frequency • Spectral edge frequency
Information Theory	• Shannon entropy • Singular value decomposition entropy • Fisher information • Spectral entropy

The feature distributions are examined by plotting the probability density functions (PDF) of the extracted features. The work in this thesis primarily focuses on the six different neonatal EEG types which were discussed in Chapter 2. In order to carry out feature extraction on each of these event types an 8-second sliding window with a 50% overlap was used. This window can capture elements of the short seizure events as the minimum duration is defined as being ten seconds [81]. Each event recording consists of 13 epochs and results in a compression from 1920 samples (60 second EEG at $32Hz$) to 55×13 or 715 feature values in total.

In the coming sections, each PDF plot will contain the PDF for each of the six EEG events. The aim of this is to highlight the discrimination between the different

events types. The PDFs shown in this chapter were generated using 150 randomly sampled feature epochs from each of the six events. The low number of samples used was because there is only a small database available especially for the short seizure events.

3.3.1 Time domain

The time domain features described in this section include;

- Line length
- Root mean squared amplitude
- Skewness
- Kurtosis
- Hjorth parameters
- Number of zero crossings
- Nonlinear energy
- Autoregressive modelling error

Line length

The line length (L) as given by Equation 3.1 is a feature used to measure the complexity of a signal,

$$L(X(j)) = \sum_{k=1}^{n_s-1} \left| x_j(k+1) - x_j(k) \right| \tag{3.1}$$

where $X(j)$ is the j^{th} epoch for the feature, x_j is the time series data contained in the j^{th} sliding window and n_s is the size of the sliding window. This equation is a sum of the absolute distances between consecutive data points within the epoch. This feature was initially used when working with seizures to detect seizure onset [82]. Figure 3.2 shows the PDF for the line length feature for each of the event types. From this figure, it can be appreciated that there is a lot of overlap with the TA, pulsatile artifact and long seizure data having a larger line length. An interesting observation here is that the mode for the short seizure is closer the mode of background data as

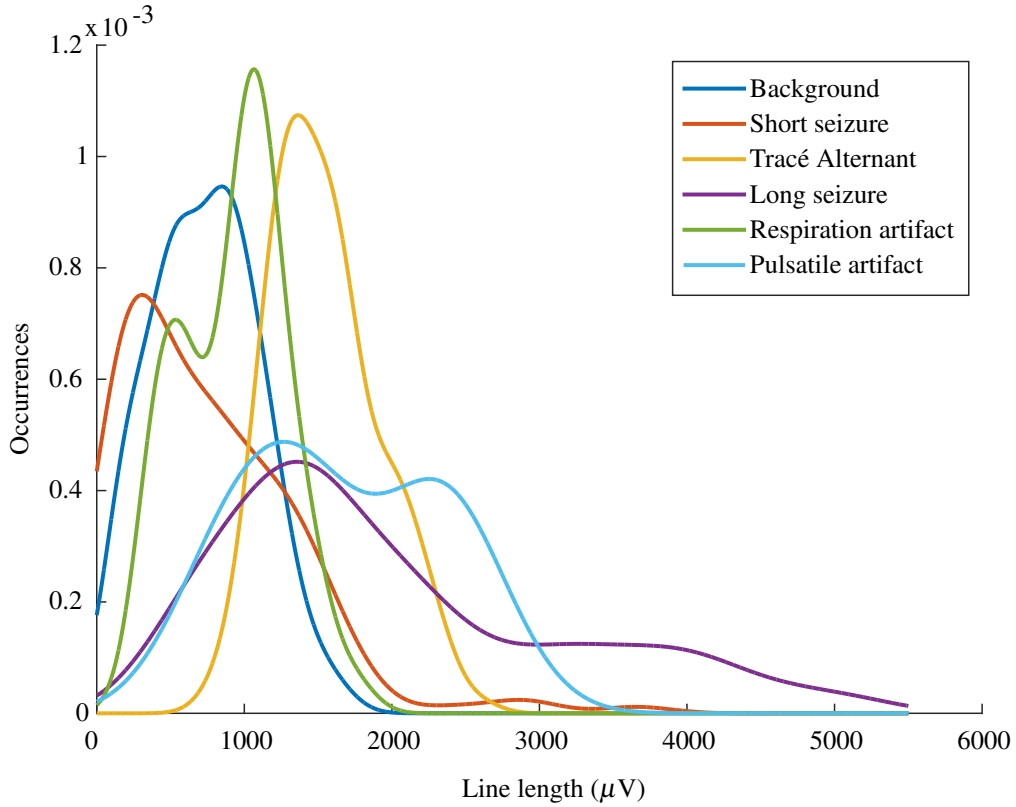


Figure 3.2: Line length PDF plot for the six neonatal EEG event types

opposed to the long seizure mode. This because the short seizure events also have background data which would typically make up approximately 66% of the overall event.

Root mean squared amplitude

The root mean square (RMS) amplitude is given by Equation 3.2.

$$RMS(X(j)) = \sqrt{\frac{1}{n_s}\sum_{k=1}^{n_s} x_j^2(k)} \tag{3.2}$$

The RMS value can be used as a signal strength estimator [83, 84]. It is also known as the quadratic mean, and it is good at detecting amplitude differences between different events. Figure 3.3 shows the different event class distributions. There is considerable overlap between the event classes and the TA pattern, pulsatile artifact and long seizure events have the highest RMS amplitudes. These high RMS amplitude values were expected as these types of events have higher amplitudes in general as compared to background data and the slowly changing respiration artifact. The

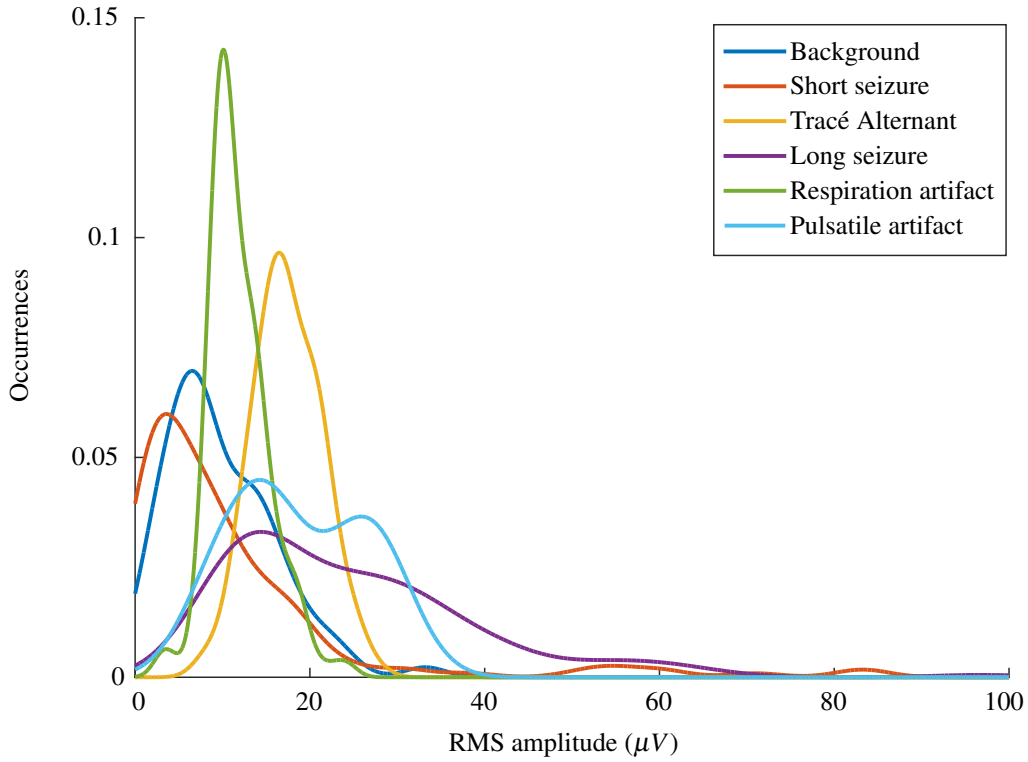


Figure 3.3: RMS amplitude PDF plot for the six neonatal EEG event types

long seizure events also have the most spread.

Skewness

Skewness as calculated from Equation 3.3 is a measure of the amount of asymmetry that exists in the probability distribution of the data about the mean [85]. If the data is not symmetric, it will either be positively or negatively skewed. Typically data that has a mean of zero would be symmetric as the data would be evenly distributed at either side of the mean.

$$Skewness(X(j)) = \frac{\frac{1}{n_s}\sum_{k=1}^{n_s}(x_j(k)-\mu_j)^3}{\left(\frac{1}{n_s}\sum_{k=1}^{n_s}(x_j(k)-\mu_j)^2\right)^{3/2}} \tag{3.3}$$

Where μ_j is the mean of the j^{th} epoch x_j given by:

$$\mu_j = \frac{1}{n_s}\sum_{k=1}^{n_s} x_j(k) \tag{3.4}$$

Figure 3.4 shows that the distributions are overlapping and suggests that the data appears to be symmetric. The data appears to be symmetric as the mode for the

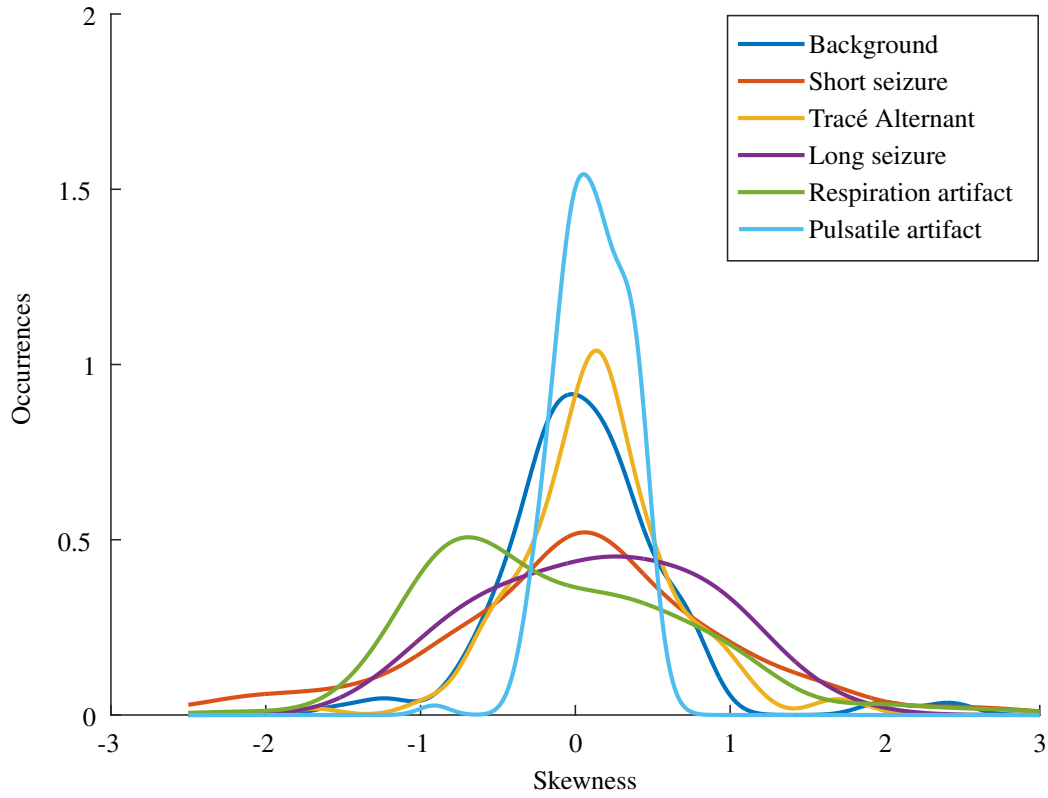


Figure 3.4: Skewness PDF plot for the six neonatal EEG event types

majority of the event types is in the region of zero and a skewness value of zero means the data is symmetric and not skewed. The pulsatile artifact, TA pattern and background data all have narrow distributions around their respected modes. The PDF of the respiration artifact has a large distribution range, and a mode located just greater than minus one which is interesting as it differs from all other events.

Kurtosis

Kurtosis is a measure of the peakedness of the distribution of the data [86]. Kurtosis focuses on the tails of the probability distribution from the data [87]. The kurtosis for a normal distribution is three. Distributions with few outliers are in the region of three or lower while distributions with many outliers would have a larger kurtosis value. Kurtosis is calculated using Equation 3.5:

$$Kurtosis(X(j)) = \frac{\frac{1}{n_s}\sum_{k=1}^{n_s}(x_j(k)-\mu_j)^4}{\left(\frac{1}{n_s}\sum_{k=1}^{n_s}(x_j(k)-\mu_j)^2\right)^2} \tag{3.5}$$

Figure 3.5 shows the kurtosis values plotted for each of the event classes. The

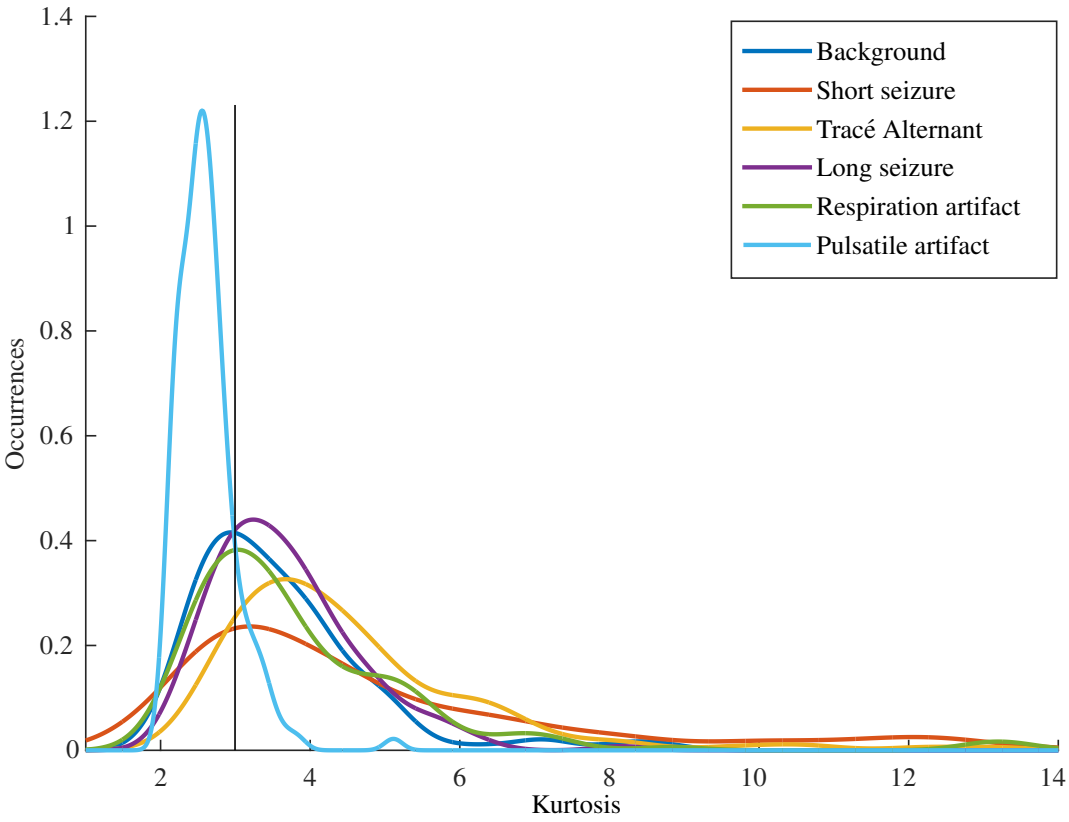


Figure 3.5: Kurtosis PDF plot for the six neonatal EEG event types

repetitive pattern of the pulsatile artifact has the narrowest distribution. The modes of the background, short seizure and respiration artifact event types are in the region of three which indicates they a normal distribution. The vertical black line in Figure 3.5 corresponds the kurtosis value of three.

Hjorth parameters

The Hjorth parameters were introduced in [88] as a method to produce a quantitative description of the EEG trace. The first Hjorth parameter is the activity also known as the variance, and this measures the spread of the data as given by Equation 3.6.

$$Activity(X(j)) = \sigma_{x_j}^2 = \frac{1}{n_s}\sum_{k=1}^{n_s}(x_j(k) - \mu_j)^2 \tag{3.6}$$

The second Hjorth parameter is mobility, calculated using Equation 3.7. This gives a measure of the standard deviation of the slope with reference to the standard deviation of the amplitude [88], therefore it is based on the first derivative of the

EEG signal ($\Delta x_j(k) = x_j(k+1) - x_j(k)$).

$$Mobility(X(j)) = \sigma_{\Delta x_j} / \sigma_{x_j} \tag{3.7}$$

Here $\sigma_{\Delta x_j}$ is the standard deviation of the first derivative of the EEG epoch x_j. The third Hjorth parameter is complexity, calculated using Equation 3.8 and it compares the signal to a pure sine wave. The higher the dissimilarity to the pure sine wave the larger the complexity value. A pure sine wave would return a complexity value of one.

$$Complexity(X(j)) = \frac{\sigma_{\Delta^2 x_j} / \sigma_{\Delta x_j}}{\sigma_{\Delta x_j} / \sigma_{x_j}} \tag{3.8}$$

Where $\sigma_{\Delta^2 x_j}$ is the standard deviation of the second derivative of the EEG epoch x_j ($\Delta^2 x_j(k) = \Delta x_j(k+1) - \Delta x_j(k)$) and denominator is the mobility.

In Figure 3.6 (a) the PDF for the activity is plotted. The activity for background and short seizure events have modes in similar areas and are highly overlapping. This may have resulted from the short seizures being partly comprised of background data. The modes of the other event types are more dispersed. In Figure 3.6 (b) the PDF for the mobility feature is plotted. The most unique event in this plot is the pulsatile artifact as it has a narrow distribution range about its mode. The other events have a more variable distribution. In Figure 3.6 (c) the PDF for the complexity is plotted. The events are overlapping with the long seizure event having the largest distribution as it is spread across the full range of values. The pulsatile artifact has a lower mode than the other events and the background event has the largest mode value.

Number of zero crossings

The number of zero crossings given by Equation 3.9, is the number of occasions when the signal amplitude has a change of sign within the EEG signal. It is related to changes in frequency [89].

$$zero(X(j)) = \sum_{k=1}^{n_s-1} \begin{cases} 0, & \text{if } x_j(k+1) \cdot x_j(k) \geq 0 \\ 1, & \text{if } x_j(k+1) \cdot x_j(k) < 0 \end{cases} \tag{3.9}$$

The number of zero crossings were also computed for the first and second derivative of the epoch. The first derivative represents the number of local EEG maxima and minima. Figures 3.7 (a), (b) and (c) shows the zero crossing plots for the epoch,

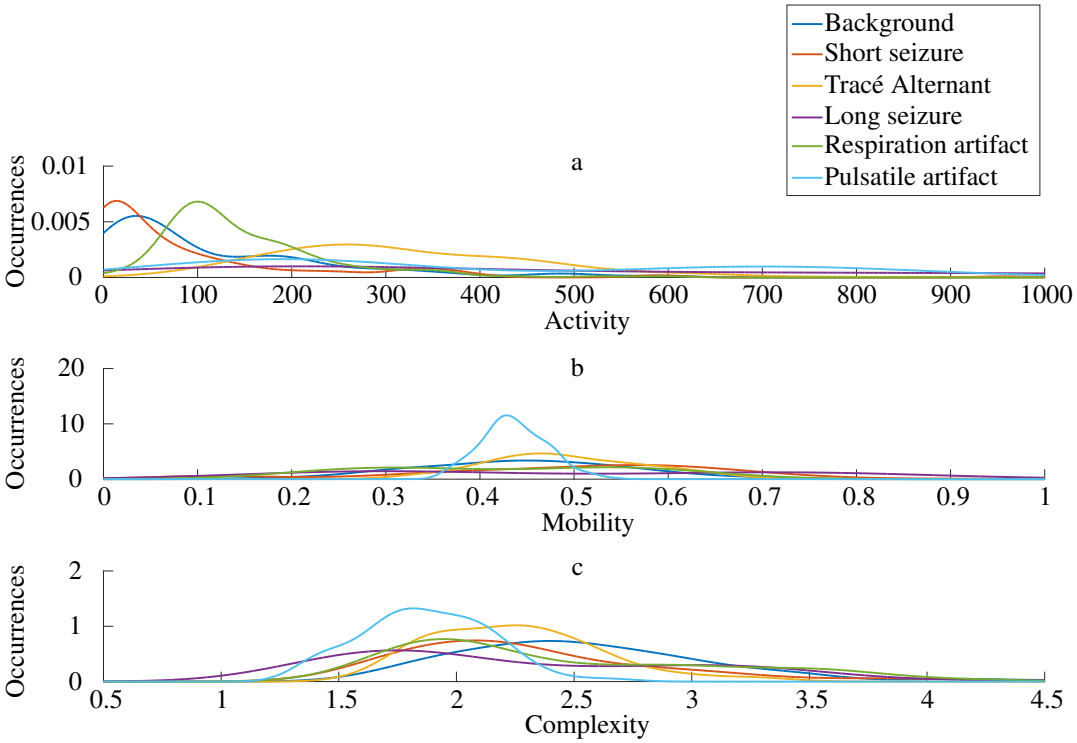


Figure 3.6: Hjorth parameters PDF plots for the six neonatal EEG event types. (a) is activity. (b) is mobility. (c) is complexity

first derivative and second derivative of the epochs respectively. The six event types are highly overlapping for Figure 3.7 (a), with the pulsatile artifact having the narrowest distribution. Figure 3.7 (b) shows more separation between the events with the short seizure event having the largest mode value. The TA event is now the most focused event though there still exists overlap with other events. Figure 3.7 (b) shows that the TA event group is still the most focused. There appears to be more separation between the short seizure event and the TA event.

Nonlinear energy

This feature looks at the amplitude as well as the changing amplitude in the signal as seen in Equation 3.10.

$$NLE(X(j)) = \frac{1}{n_s - 2} \sum_{k=2}^{n_s-1} x_j(k)^2 - x_j(k-1)x_j(k+1) \tag{3.10}$$

This feature was published by [90] who described it as Teager's algorithm which was later used for seizure detection in adult patients with suspected epilepsy [91]. Figure 3.8 shows the nonlinear energy PDF plot. The background and short seizure

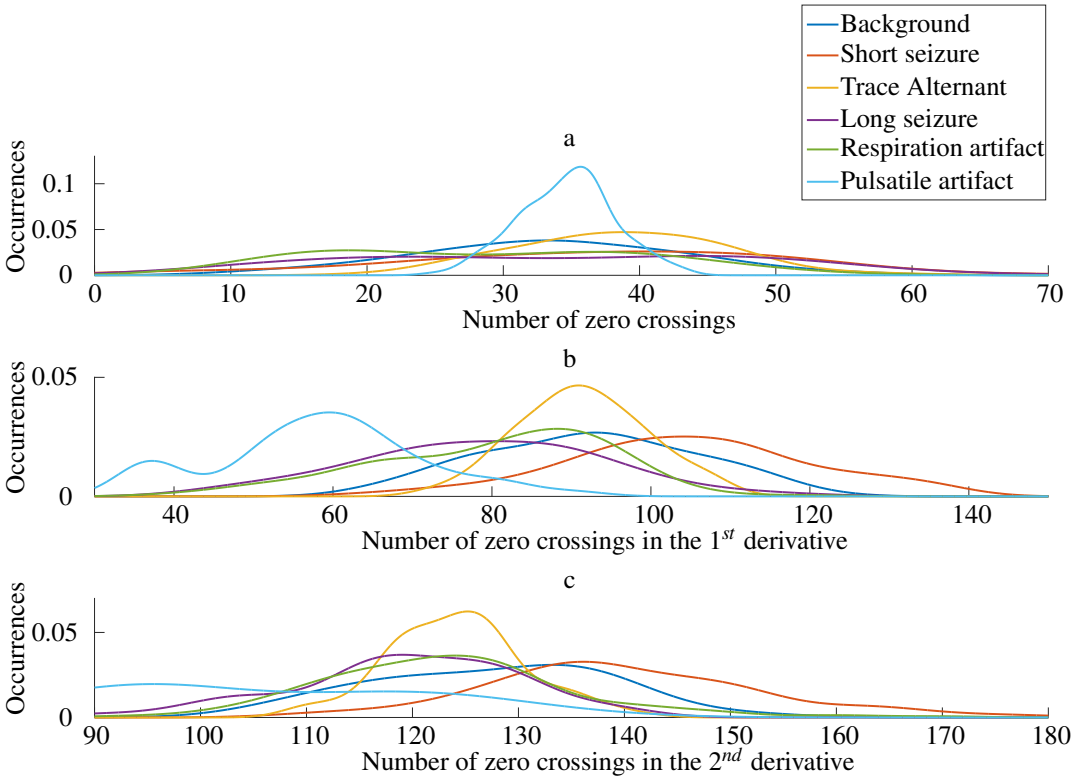


Figure 3.7: Zero crossing PDF plots for the six neonatal EEG event types. (a) Time series. (b) First derivative of time series. (c) second derivative of time series

events are closely aligned and have the shortest distribution range and the long seizure event class is more separated as it has a larger mode. The pulsatile artifact event has a multimodal distribution.

Autoregressive modelling error

An autoregressive (AR) model is used to predict future time series values ($\hat{x}(k)$) from a linear combination of past observations $x(k-i)$ and is given by Equation 3.11,

$$\hat{x}(k) = \sum_{i=1}^{n_p} \varphi_i x(k-i) + \varepsilon_k \tag{3.11}$$

where φ_i is the i^{th} AR coefficient and n_p is the AR order. White noise with a zero mean (ε_k) is added to account for error that may arise in the prediction step. When performing AR modelling, the data is split into two sections. The first $\frac{n_s}{2}$ samples are used to train the φ_i parameters using the Yule-walker method [92]. The second half of the data are used to perform one step ahead prediction. The percentage fit is

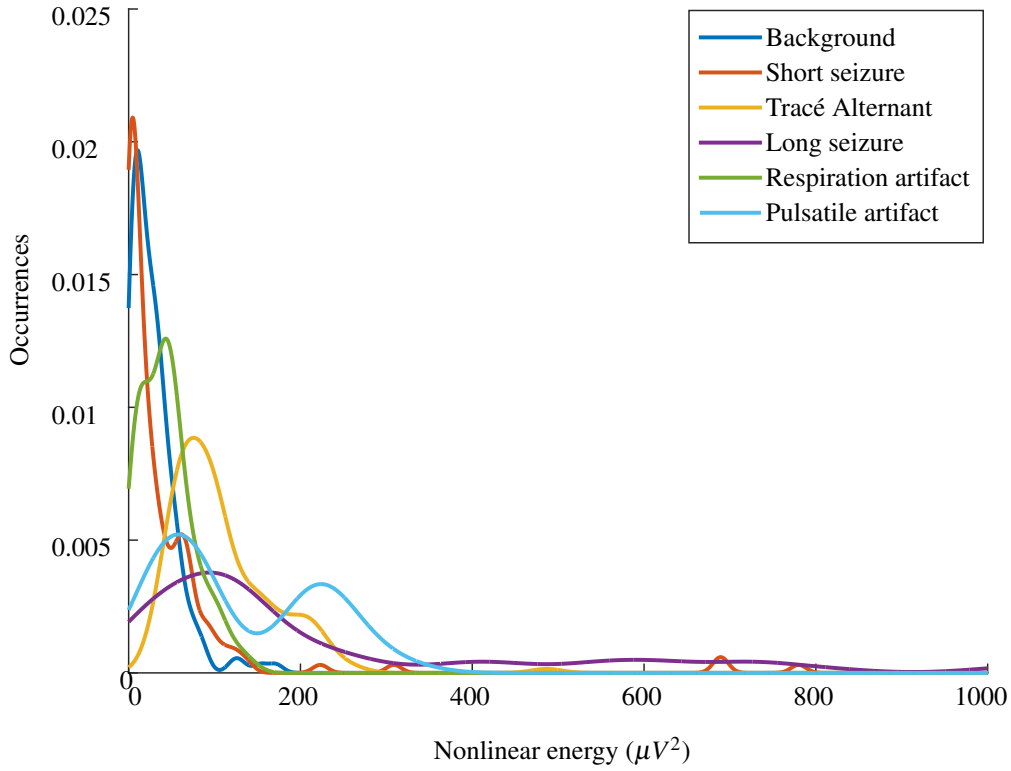


Figure 3.8: Nonlinear energy PDF plot for the six neonatal EEG event types

then calculated using Equation 3.12,

$$AR_{fit}(X(j)) = 100\left(1 - \frac{\sum_{k=\frac{n_s}{2}+1}^{n_s} \left|x_j(k) - \hat{x}_j(k)\right|}{\sum_{k=\frac{n_s}{2}+1}^{n_s} \left|x_j(k) - \bar{x}_j(k)\right|}\right) \tag{3.12}$$

where

$$\bar{x}_j = \frac{1}{n_s/2} \sum_{k=\frac{n_s}{2}+1}^{n_s} x_j(k) \tag{3.13}$$

AR modelling has previously been used in the analysis of EEG for the task of person identification [93]. The approach used in this thesis generated nine features corresponding to the AR error when using models of order (n_p) from one to nine. The error is the difference between the predicted and actual value of the data. Figure 3.9 shows the PDF plot for the second-order model fit. The graph for the second-order model was chosen as it had the most distinctive distributions for the different event types. There is a lot of overlap for the features. The mode of the short seizure distribution is in the region of 50%. The pulsatile artifact event has the narrowest probability density band with a mode in the region of 70%. The respiration artifact and long seizure events appear to be multimodal distributions.

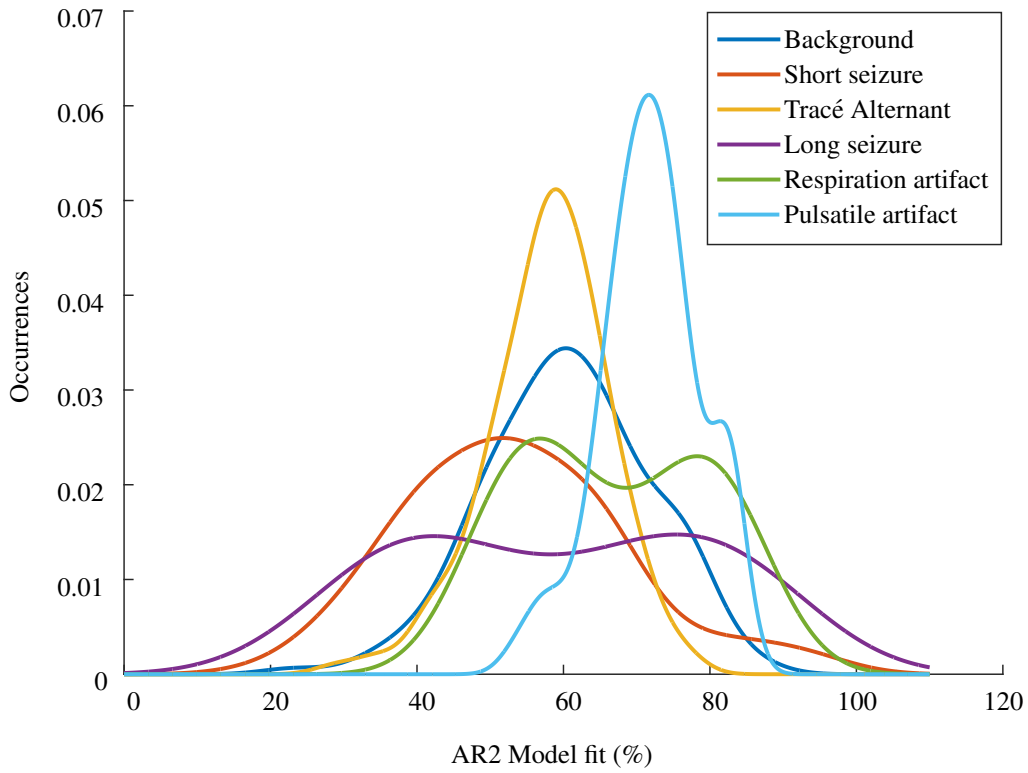


Figure 3.9: 2^{nd} order autoregressive model fit error PDF plot for the six neonatal EEG event types

3.3.2 Frequency domain

The frequency content of the neonatal EEG can be viewed by computing a Fast Fourier Transform (FFT) followed by computing the power spectral density (PSD) of the FFT. The FFT created a frequency representation that contained 256 frequency bins (n_f), which are complex coefficients and the PSD is then computed. This PSD is equivalent to squaring the absolute values of the FFT and dividing the resulting values by n_f. The first half of the PSD was used giving $\frac{n_f}{2}$ frequency coefficients for the feature extraction. The PSD values can be represented in the form a_j = [$a_j(0)$,$a_j(1)$,...$a_j(i)$,...$a_j(\frac{n_f}{2})$] where at the frequency $i\frac{f_s}{n_f}$ the amplitude is $a_j(i)$ when the sampling frequency is f_s. This process enables the frequency components of the EEG to be visualised. An example of neonatal EEG is seen in Figure 3.10, with Figure 3.10 (a) showing the time domain representation of 8 seconds of EEG. Figure 3.10 (b) shows the PSD of the frequency spectrum which was computed using a 256 point FFT. Most of the energy is contained in the lower frequency components of the signal as seen in Figure 3.10 (b).

The frequency domain features described in this section include;

- Total power
- Power in $2Hz$ sub-bands ((0-2Hz), (1-3Hz), ..., (10-12Hz))
- Normalised power in $2Hz$ sub-bands ((0-2Hz), (1-3Hz), ..., (10-12Hz))
- Peak frequency
- Spectral edge frequency (frequency under which a certain percentage of the power in the PSD lies)
- Spectral entropy

Total power

The total power is the sum of the frequency bins up to the $12Hz$ bin and is given by Equation 3.14.

$$P_{tot}(X(j)) = \sum_{k=0}^{12\left(\frac{n_f}{f_s}\right)} a_j(k) \tag{3.14}$$

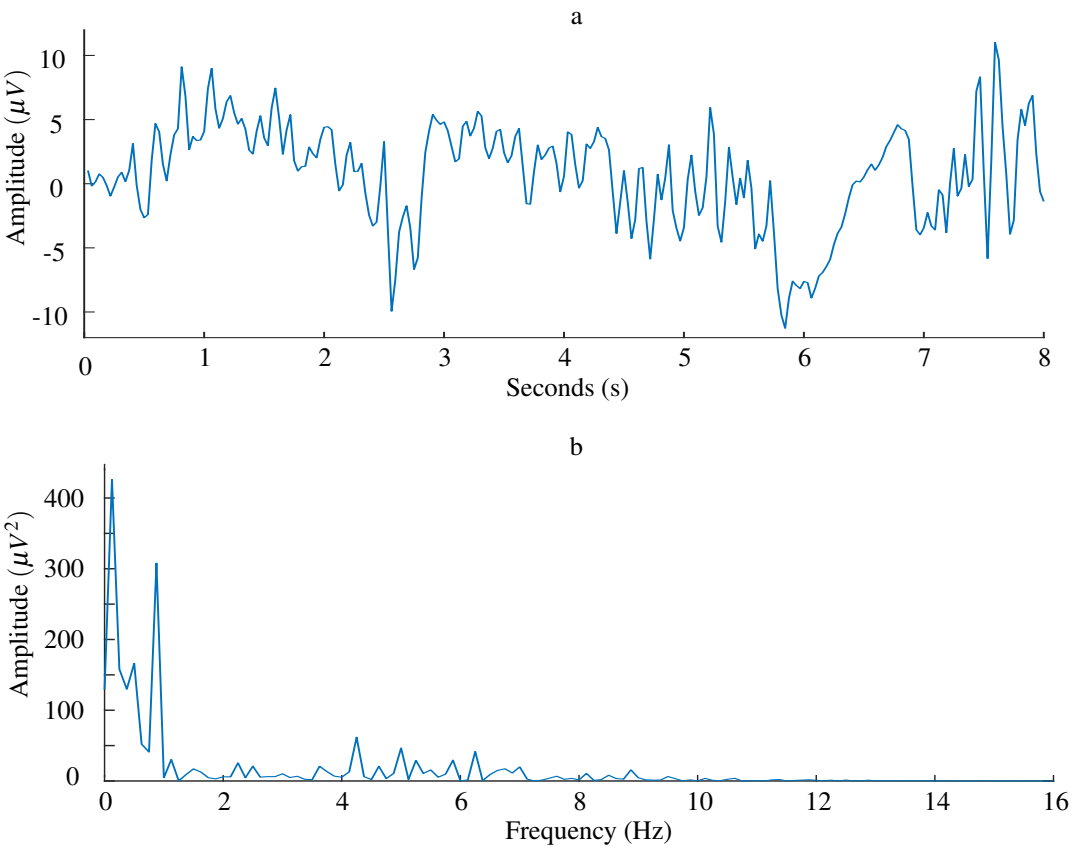


Figure 3.10: An 8 second epoch from healthy normal EEG (a). The PSD of this epoch is displayed in (b), which was computed using a 256 point FFT.

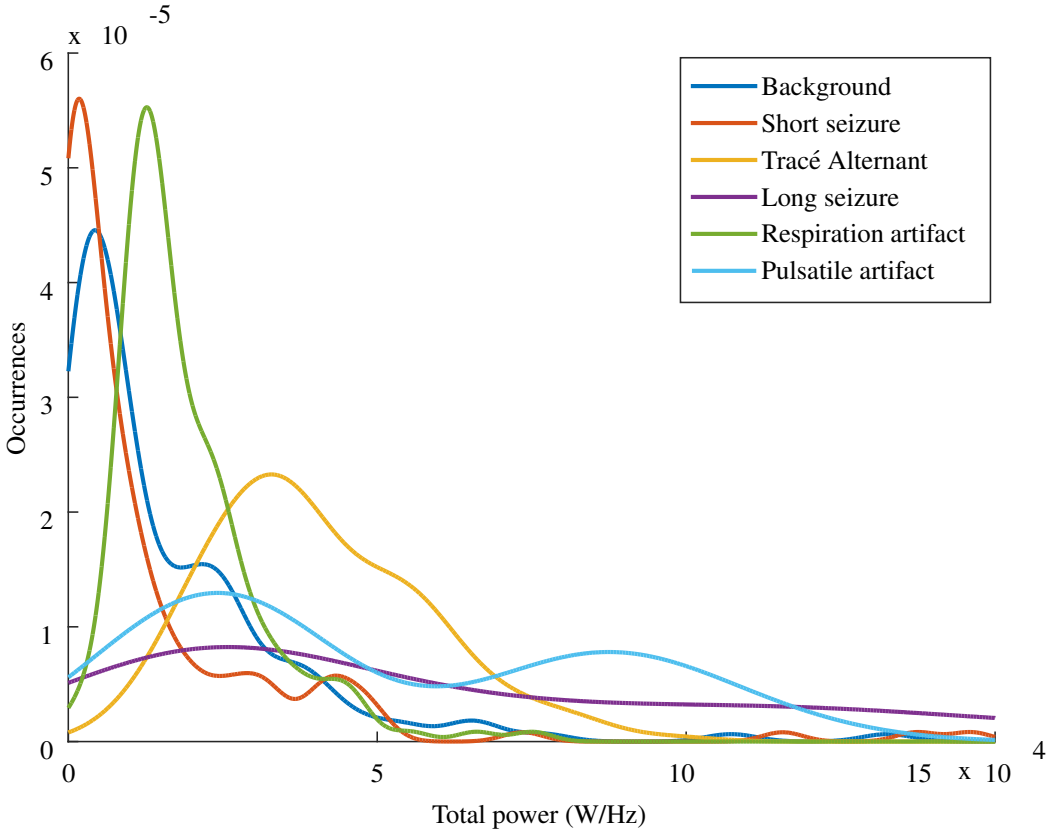


Figure 3.11: PDF plot for the total power contained within the region of $0-12Hz$ for the six neonatal EEG event types

Where $a_j(k)$ is the power in bin k of epoch $X(j)$ and $a_j(\frac{12n_f}{f_s})$ is the power in the $12Hz$ bin. Figure 3.11 shows the total power PDF plot for the six different event types. The short seizure and background events have a similar amount of power. The mode of the respiration artifact is higher than the modes of the short seizure and background events although it is lower than the TA event. The background, short seizure and respiration artifact events have narrower distribution bands than the other events. The total power is a common feature in neonatal EEG analysis and was used in various studies including [33, 39, 94].

Power in sub-bands

The previous feature looked at the total power over the frequency bins ranging from $0-12Hz$. This subset of features looks at the power in frequency bands of width $2Hz$ starting at $0-2Hz$ with increments of $1Hz$ and is calculated using Equation

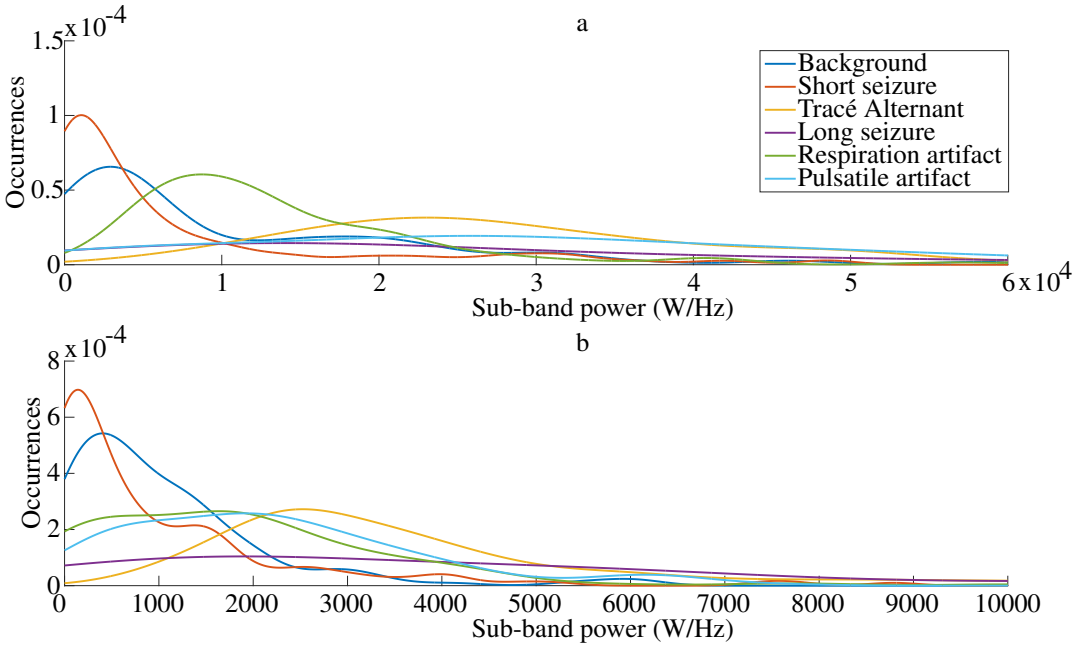


Figure 3.12: PDF plots for the power in sub-bands (a) $0-2Hz$ and (b) $3-5Hz$ for the six neonatal EEG event types

3.15.

$$P_{0-2Hz}(X(j)) = \sum_{k=0\left(\frac{n_f}{f_s}\right)}^{2\left(\frac{n_f}{f_s}\right)} a_j(k) \tag{3.15}$$

This feature has previously been used for the detection of seizure onset [95] and epilepsy diagnosis [96]. The events will have different PDFs for different energy bands. This is demonstrated in Figure 3.12 (a) and (b). Figure 3.12 (a) shows the PDF for the $0-2Hz$ frequency band in which the shortest seizure distribution is the most focused out of all the events. It also has the lowest mode sub-band power. The TA, long seizure and pulsatile events have the largest distribution range and are the most overlapping. Figure 3.12 (b) shows the PDF for the $3-5Hz$ frequency band. They are different from the $0-2Hz$ frequency band. The short seizure distribution still has the lowest mode and is the smallest distribution range out of all the events. The respiration event has a larger range and is more overlapping with other events. As the signal frequency increases, the power typically decreases. This decrease in power is typical for neonatal EEG and is backed up by Figure 3.10 (b) which shows the lower frequency components having a higher amplitude.

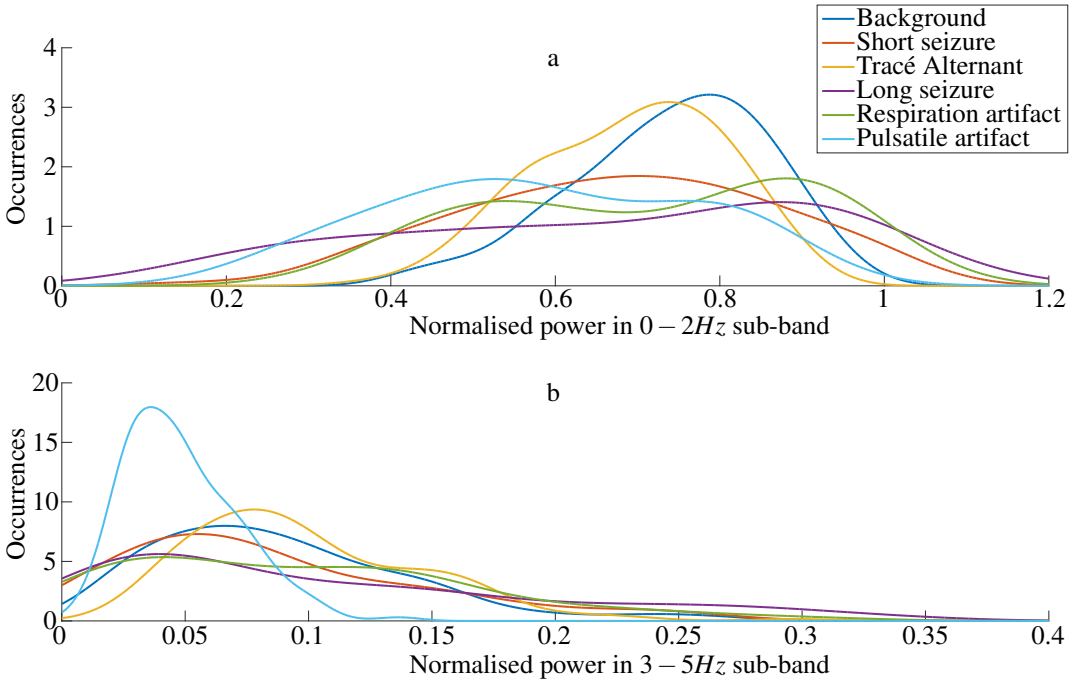


Figure 3.13: PDF plots for the normalised power in sub-bands (a) $0-2Hz$ and (b) $3-5Hz$ for the six neonatal EEG event types

Normalised power in sub-bands

The next set of features that were extracted were the normalised power in the sub-bands which is given by Equation 3.16.

$$P_{norm_{0-2Hz}}(X(j)) = \frac{P_{0-2Hz}(X(j))}{P_{tot}(X(j))} \tag{3.16}$$

This equation is merely the power in a specific sub-band divided by the total power for the epoch $P_{tot}(X(j))$. This enables the examination of how much of the total power is captured in the various sub-bands. This feature has previously been used for seizure detection [29]. Figure 3.13 (a) and (b) shows the normalised sub-band power in the bands $0-2Hz$ and $3-5Hz$ respectively. There is a lot of overlap in the PSD images for the normalised sub-band power. In Figure 3.13 (a) the pulsatile and respiration artifacts appear to have muiltimodal distributions. The background and TA events have the narrowest distribution bands with modes within a small range of each other. In Figure 3.13 (b) the modes of the normalised power are all lower than the modes in Figure 3.13 (a). In Figure 3.13 (b) the pulsatile artifact has a single mode and it has a narrow distribution band.

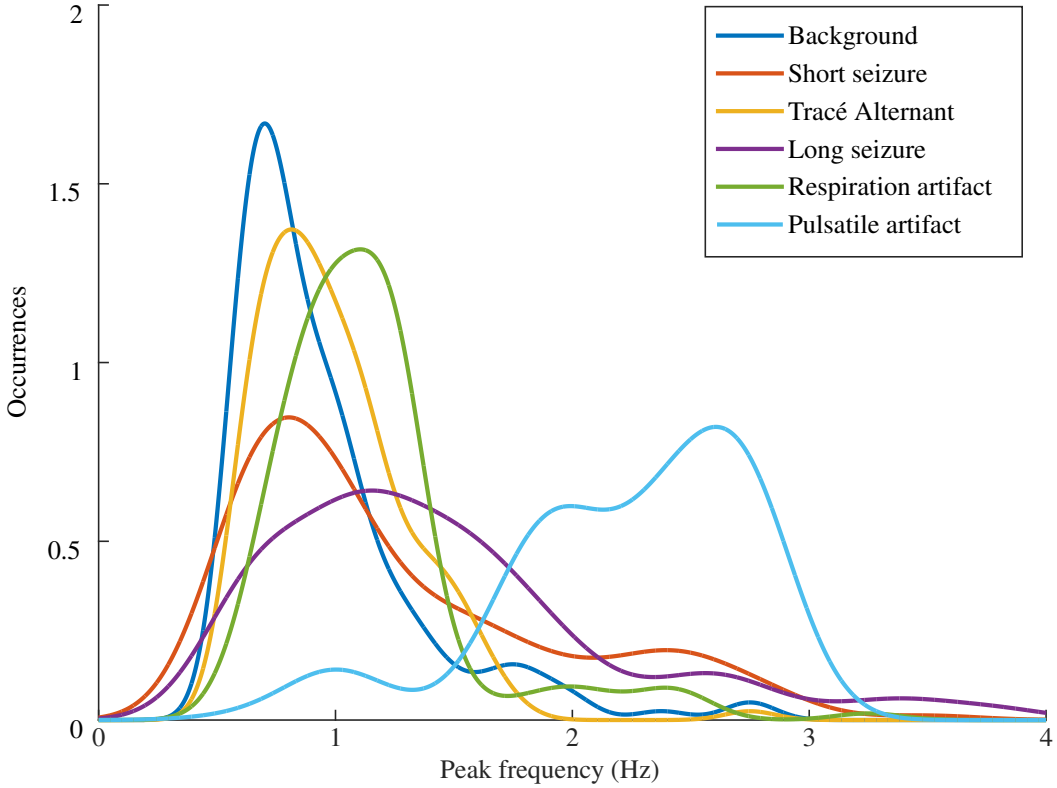


Figure 3.14: Peak frequency PDF plot for the six neonatal EEG event types

Peak frequency

The peak frequency is the frequency component in the FFT that had the highest amplitude. This feature was initially defined by [97] as the dominant frequency and is found using Equation 3.17.

$$f_{peak} = k_{peak}\frac{f_s}{n_f}, \qquad \text{where} \quad k_{peak} = \underset{k}{\text{argmax}} \quad a_j(k) \tag{3.17}$$

As the peak frequency is the component with the highest amplitude in the FFT, it would be expected that this frequency would have a significant contribution to the underlying structure of the EEG signal. Figure 3.14 shows the PDF plots for the different events which are overlapping although they have different distributions. The distribution of the pulsatile artifact is the most separated distribution and it has the largest peak frequency mode value. An explanation for this is that the pulse of a neonate can be in the region of 170 beats per minute [98] which is approximately 2.8 Hz.

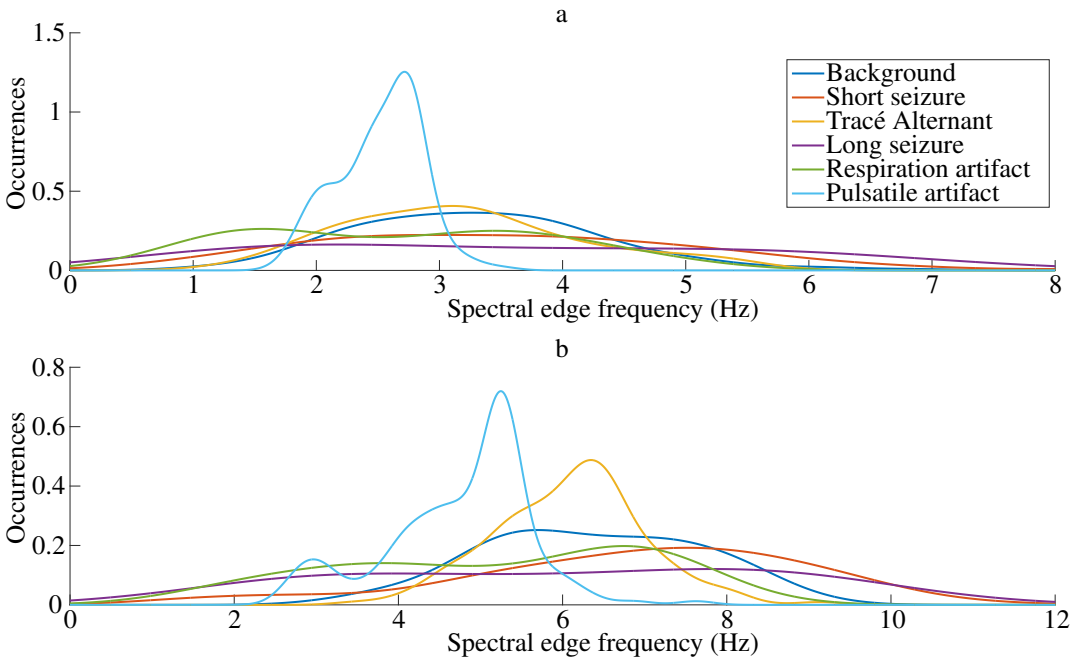


Figure 3.15: PDF plots for the spectral edge frequency at (a) 80% and (b) 95% for the six neonatal EEG event types

Spectral edge frequency

The frequency under which a certain percentage of the power in the FFT lies is defined as the spectral edge frequency (SEF). This feature was initially used to assess the depth of anesthesia [99]. The SEF feature was extracted for 80%, 90% and 95% of the power in the FFT. The PDF plots for a SEF of 80%, 90% and 95% are similar. For this reason, only two were plotted which are shown in Figure 3.15 (a) and (b) which shows the SEF PDF plots for the 80% and 95% SEF respectively.

What is apparent from Figure 3.15 (a) is that there is a high probability that 80% of the total power for the pulsatile artifact is contained between $0-3.5Hz$. Each of the other events have wider ranges and the modes of all the events are contained below $3.5Hz$. Figure 3.15 (b) shows that there is a high probability that 95% of the power is contained between approximately $0-8Hz$, showing that the majority of the power is contained within the lower frequencies of the EEG signal.

3.3.3 Information theory domain

The idea of information theory stemmed from the work published by Claude Shannon [100]. The concept of entropy as a measure of uncertainty was described in the works of Shannon. In this section, it is used for three measures in the EEG domain

space. The information theory domain features described in this section include;

- Shannon entropy
- Singular Value Decomposition entropy
- Fisher entropy
- Spectral entropy

Shannon entropy

Shannon entropy [100] is a measure of the degree of randomness in a set of data. The Shannon entropy for a time series signal is calculated from the probability mass function (PMF) of the signal. The PMF gives the probability that a discrete random variable drawn from a set of variables is equal to some value within the set. To generate the PMF the time series signal is initially converted into a set of discrete values. To convert the signal into a set of discrete values a histogram with $\sqrt{n_s}$ or 16 bins is computed and Figure 3.16 provides an example of the PMF of an epoch from a background event. Background EEG activity typically has an average amplitude of $0\mu V$ and this is reflected in Figure 3.16 as the bin around $0\mu V$ is the largest. The Shannon entropy is then calculated using Equation 3.18.

$$H_{sh}(X(j)) = -\sum_{i=1}^{16} P_i(x_j)\log P_i(x_j). \tag{3.18}$$

The equation was calculated for the j^{th} epoch where P is the PMF and P_i is the probability corresponding to the i^{th} histogram bin. Figure 3.17 shows that the majority of the event types have similar and overlapping distributions with the exception being the background and short seizure events. The background and short seizure events have larger distribution bands for the Shannon entropy feature. Low values of Shannon entropy indicate low levels of uncertainty and as the uncertainty increases so does the Shannon entropy. As background EEG is random containing no set patterns or frequencies, it is expected that there would be a higher level of entropy. The wider distribution explains this for the background and short seizure events in Figure 3.17.

Singular value decomposition entropy

The SVD is a form of matrix decomposition or matrix factorisation. It was initially proposed by [101] as a complexity measure for EEG. SVD reduces a matrix A into

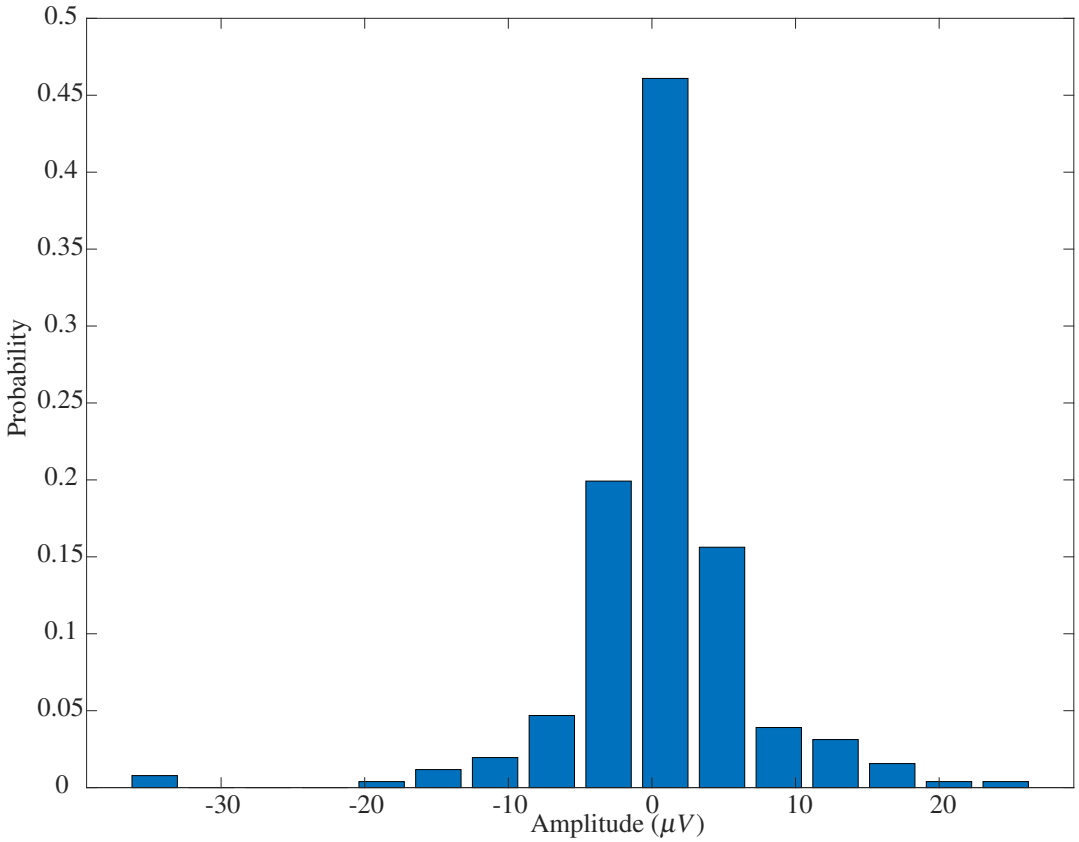


Figure 3.16: Example PMF plot for a random epoch from a background event

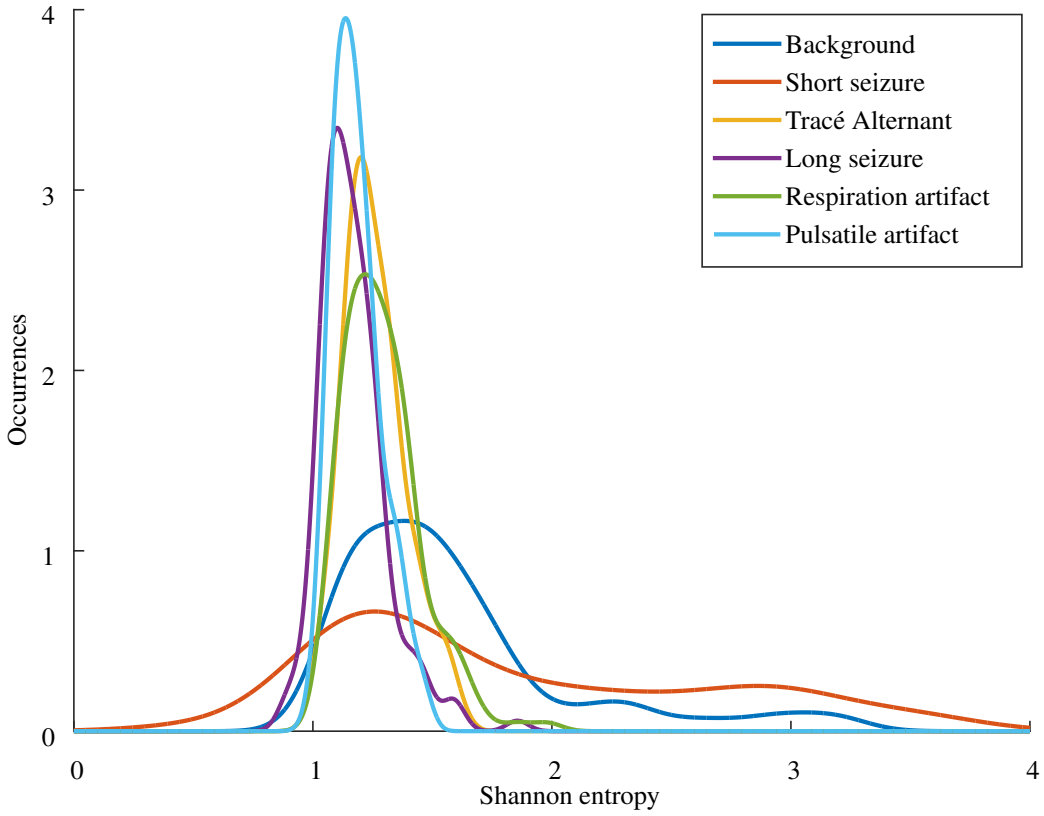


Figure 3.17: Shannon entropy PDF plot for the six neonatal EEG event types

its constituent parts as given by Equation 3.19.

$$A = U\Theta V^T \tag{3.19}$$

Where U is the left singular vectors of matrix A, Θ is a diagonal vector of the singular values and V is the right singular vectors. U and V are both orthonormal. The number of non-zero ($>$ low threshold) singular values is a measure of the amount of information contained the signal. The SVD entropy is then calculated using Equation 3.20 from the singular spectrum $\{\eta_1 ... \eta_{d_E}\}$ where d_E corresponds to the number of singular values in Θ,

$$H_{SVD}(X(j)) = -\sum_{i=1}^{d_E} \hat{\eta}_i \log \hat{\eta}_i. \tag{3.20}$$

where

$$\hat{\eta}_i = \frac{\eta_i}{\sum_{j=1}^{d_E} \eta_j} \tag{3.21}$$

Figure 3.18 shows the PDF plot for the SVD entropy for the six different event classes being tested. There is overlap although the distributions are different. The respiration artifact, pulsatile artifact and long seizures events appear to be multimodal. The distribution for the TA event is highly concentrated about its mode.

Fisher entropy

Fisher information [102] was published before the works of Shannon. Shannon entropy is heavily influenced by the power of the signals being analysed [103]. Fisher's information measure uses the normalised SVD singular values which highlight changes in the singular spectrum. The idea of using fisher entropy in the analysis of EEG was introduced in [103]. The normalised SVD singular values as were defined in Equation 3.21 are used to calculate the Fisher entropy as shown in Equation 3.22.

$$I_{Fisher}(X(j)) = \sum_{i=1}^{d_E-1} \frac{(\hat{\eta}_{i+1} - \hat{\eta}_i)^2}{\hat{\eta}_i} \tag{3.22}$$

In signals that are more deterministic such as seizure, pulsatile artifact and respiration artifact, there will be a small number of dominant singular values. More deterministic signals usually lead to a higher fisher entropy as there is usually a larger difference between the singular values. Figure 3.19 shows the PDF plot for the Fisher entropy for the six different event classes being tested. There is much

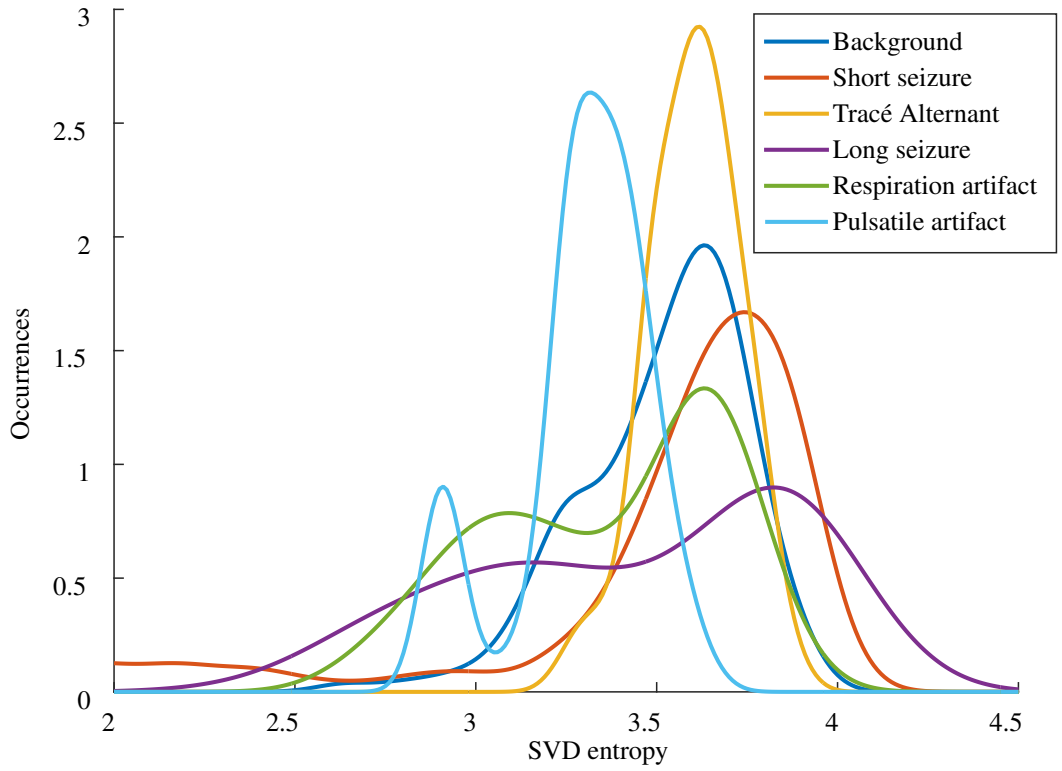


Figure 3.18: SVD entropy PDF plot for the six neonatal EEG event types

overlap between all the events and events such as long and short seizures are more drawn out.

Spectral entropy

Spectral entropy is based on the Shannon entropy. This feature could also be described as a frequency based feature as it uses the power spectral components a_j as were described in Section 3.3.2. It is a measure of the spectral power distribution of the signal and calculated from Equation 3.23,

$$H_{spec}(U_j) = -\sum_{k=0}^{\frac{n_f}{2}} \tilde{a}_j(k) \log_2 \tilde{a}_j(k) \tag{3.23}$$

where

$$\tilde{a}_j(i) = \frac{a_j(i)}{\sum_{l=0}^{\frac{n_f}{2}} a_j(l)} \tag{3.24}$$

Figure 3.20 shows the PDF plot for the spectral entropy of each of the six classes. The distributions are overlapping in Figure 3.20. The TA event has the narrowest distribution and the long seizure event has the widest distribution. This feature was

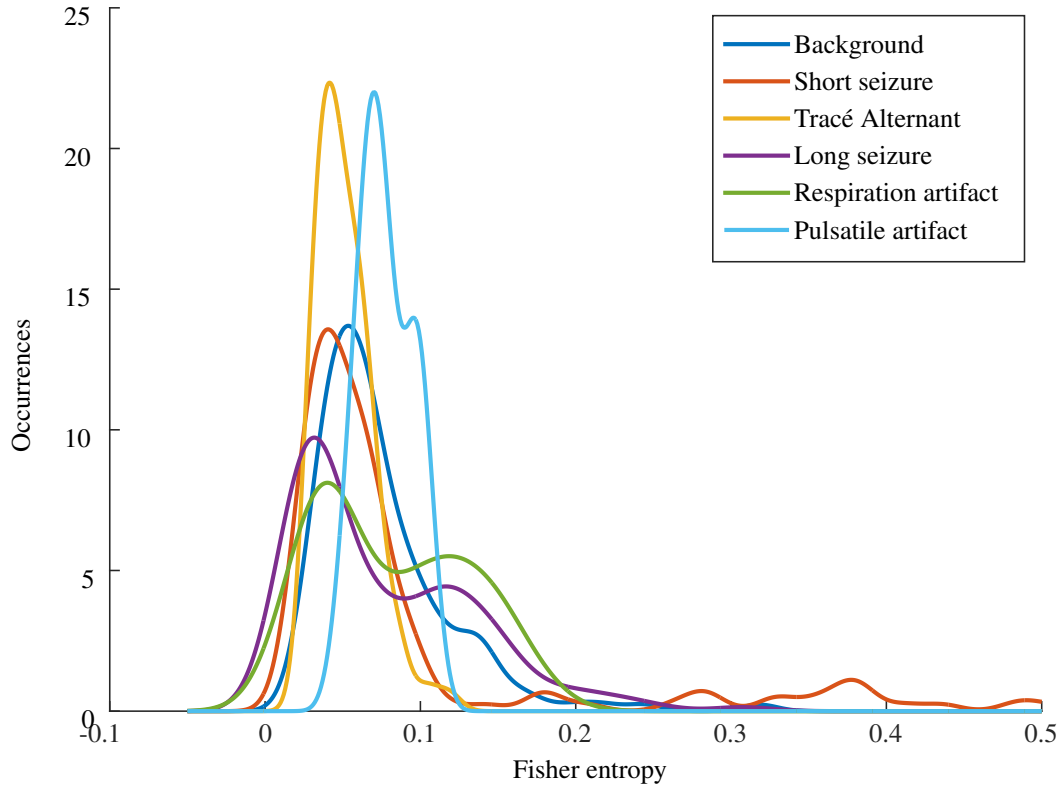


Figure 3.19: Fisher entropy PDF plot for the six neonatal EEG event types

used in EEG applications for seizure detection [33] and sleep stage detection [36] while also having uses in other biomedical signal processing applications [104].

3.4 Principal component analysis

The set of features that were described were originally designed for seizure detection and were used in the studies [27,29,33]. The features did prove useful at capturing the general characteristics of EEG and were used in studies other than seizure detection such as artifact detection [105] and HIE grading [46].

This thesis is focused on the nearest neighbour search of neonatal EEG events. The performance of each system is important, but the memory requirements are also important. The work in this thesis was carried out on a small database, but the assumption that the database will grow is considered when designing systems. With this in mind after the feature extraction step, there are 55 features for each epoch that need to be stored. In the ideal scenario, the lowest amount of storage possible would be used. Feature selection is not considered in this thesis as it may bias the features to the classes currently present in the event database. For this reason, the

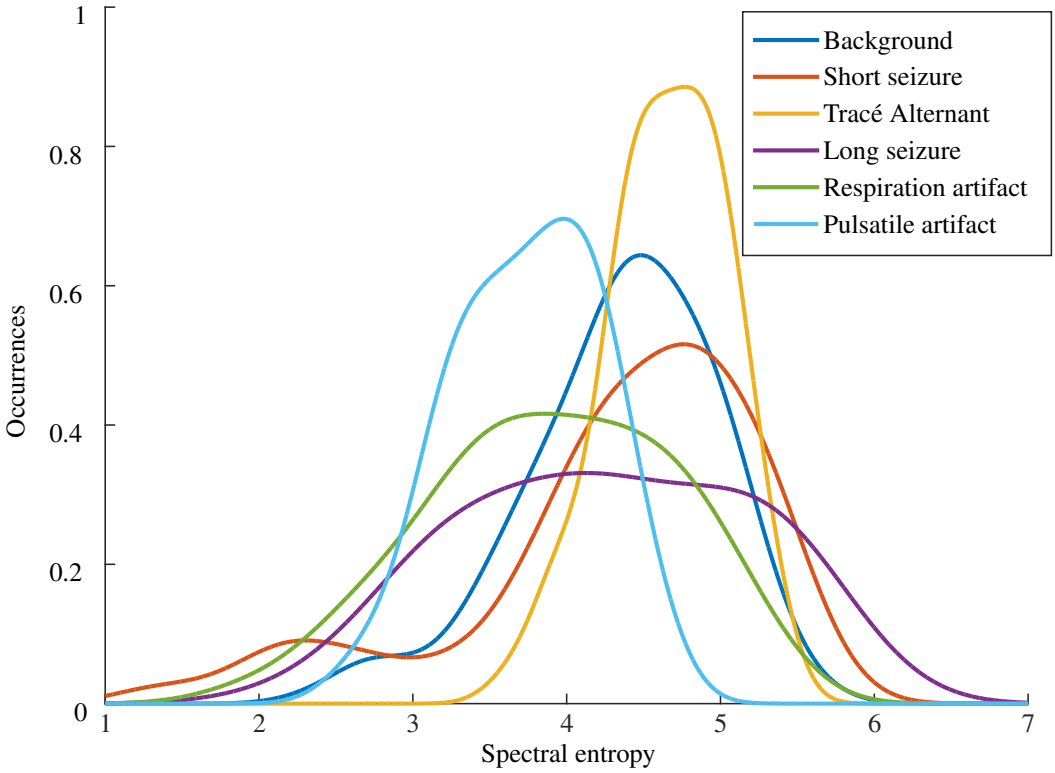


Figure 3.20: Spectral entropy PDF plot for the six neonatal EEG event types

dimensionality reduction technique of principal component analysis (PCA) [106] was used to reduce the memory requirements. As seen from the PDF plots in this chapter, some features are highly overlapping and have narrow distributions and other distributions are wider with more separation between the event classes. As not all the features may not be providing valuable information, PCA helps reduce the dimensionality by removing features that have little variance.

The dimensionality reduction technique PCA was initially proposed by [107]. The PCA algorithm transforms the features into an orthogonally separated uncorrelated space with each new feature known as a principal component (PC) retaining information that was originally within the features. The PCs are ordered by the variance they retain, with the first PC having the most significant variance [106]. This PC order acts as a dimensionality reduction technique where typically the last few components have minimal variance and therefore do not positively contribute to the performance. Using this approach the dimensionality is reduced by removing the PCs that have little variance. With any dimensionality reduction technique, data will be lost. However, the PCA algorithm is a useful compression tool to reduce the data while keeping as much information as possible.

The PCA transformation process is computed in the following manner. Initially, the data is normalised to have a zero mean and unit standard deviation using Z-score normalisation [86] as given by Equation 3.25.

$$\overline{X_f}(i) = \frac{X_f(i) - \mu_f}{\sigma_f} \tag{3.25}$$

Where $X_f(i)$ is the i^{th} feature value for feature f, μ_f and σ_f are the template mean and standard deviation respectively which are calculated from all the data and $\overline{X_f(i)}$ is the i^{th} normalised feature value for feature f. The normalisation process is repeated for each of the F_n feature epochs and all of the F feature. Normalising each event individually without the templates loses important amplitude information maintained within the features.

The next stage is to compute the covariance matrix. The covariance is a measure of the variability of two variables or features in this case. The covariance between feature vectors $\overline{X_j}$ and $\overline{X_k}$ is computed using Equation 6.10.

$$\sigma_{\overline{X_j},\overline{X_k}} = \sum_{i=1}^{F_n} \frac{\left(\overline{X_j}(i) - \mu_j\right) - \left(\overline{X_k}(i) - \mu_k\right)}{F_n - 1} \tag{3.26}$$

The F dimensional covariance matrix will be of the form

$$\Sigma = \begin{bmatrix} \sigma_{\overline{X_1},\overline{X_1}} & \cdots & \sigma_{\overline{X_1},\overline{X_F}} \\ \vdots & \ddots & \vdots \\ \sigma_{\overline{X_F},\overline{X_1}} & \cdots & \sigma_{\overline{X_F},\overline{X_F}} \end{bmatrix}$$

From the matrix Σ, the eigenvectors and eigenvalues are computed. They are of the form

$$\Sigma\vec{v} = \lambda\vec{v} \tag{3.27}$$

Where $\vec{v}$ is the eigenvector and λ is the corresponding eigenvalue. The eigenvectors are orthogonal to one another. When they are plotted, they cut through the centre of the data as seen in Figure 3.21. In Figure 3.21 (a) the original data can be seen, it is clear that the data points are correlated. In Figure 3.21 (b) the eigenvectors are plotted as the red and black lines. These eigenvectors represent the primary directions that the data is moving. The eigenvector with the highest eigenvalue is the first principal component. The eigenvalues sort the eigenvectors from highest to lowest. This sorting is essential for the dimensionality reduction stage as the

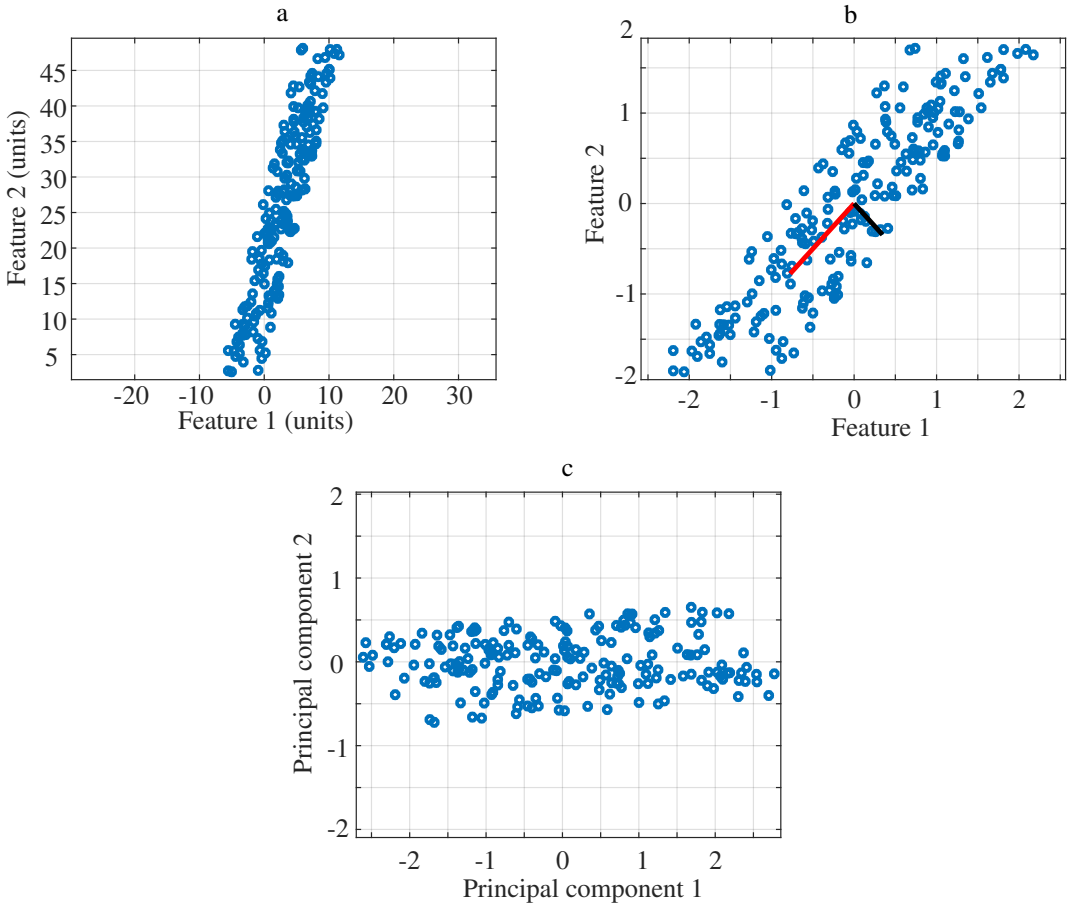


Figure 3.21: Example of the PCA process. (a) correlated raw feature data. (b) normalised feature data and eigenvectors. (c) orthogonally separated data

components will be in the order of importance or the amount of variance maintained [108]. When a lower dimension is required the last or lowest eigenvector will be removed. In Figure 3.21 (b) the red eigenvector corresponds to the axis for the first principal component. The black line corresponds to the second principal component. The next stage in the PCA algorithm is to transform the data into a new orthogonally separated space based on the eigenvectors. This transformation is carried out using Equation 3.28.

$$X' = \vec{v}^T X \tag{3.28}$$

This equation rotates the feature matrix X around the new axis based on the eigenvectors as seen in Figure 3.21 (c) where the data from Figure 3.21 (b) is simply rotated. Now the x-axis represents the first principal component and the y-axis represents the second principal component. The data is no longer correlated.

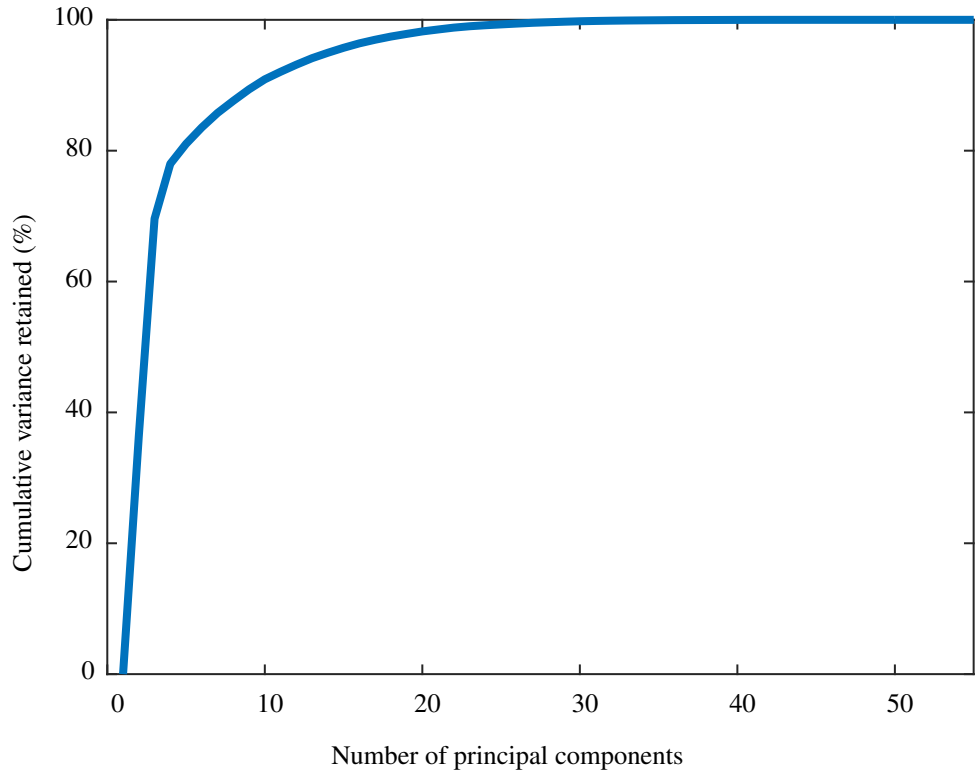


Figure 3.22: Cumulative variance retained as the number of principal components is increased

When performing dimensionality reduction, the eigenvalues are important. These contain the amount of variance each principal component maintains and the percentage contained is given Equation 3.29.

$$Var(i) = \frac{\lambda(i)}{\sum_{j=1}^{F} \lambda(j)} 100 \tag{3.29}$$

Where $Var(i)$ is the percentage variance maintained when keeping principal components 1 to i and F is the number of principal components. An example of this for the feature set given in Table 3.1 is shown in Figure 3.22 where it can be seen that the variance maintained is monotonically increasing. Figure 3.22 was generated using all the feature data from each of the 430 events. The first 25 components provide 99.3% of the total variance within the features, meaning that there are a lot of principal components that contain very little variance.

A point worth noting here is that PCA is a dimensionality reduction technique and removes the components that contain the lowest variance. Components that contain little variance still have the potential to positively contribute to the task.

When performing dimensionality reduction using PCA the user can decide the

percentage of variance they would like to maintain and then the number of associated components can be found by using Equation 3.29. The work carried out in this thesis used a relatively small database and as such the experiments were quick to compute.

As the experiments were quick to compute, the performance was able to be evaluated when using a single component, then the top two components up until using the full 55 components. This approach was used to pick the number of components to retain. The number of components retained was the set that gave the best performance as will be described in the following chapters.

3.5 Summary

In this chapter, the idea of converting the raw EEG signal into quantitative features was discussed. Quantitative features were created using help from experts with domain knowledge in the area of neonatal EEG. The features that were used in this work were chosen due to their proven success in the area of neonatal EEG [27,29,46]. These features helped describe the signals in the time, frequency and information theory domains. Each feature was described, and the equations used were provided.

There are also a wide variety of other potential features that could have been investigated for the work carried out in this thesis. In [60] the authors used features by [33] and an additional 19 features were used for neonatal seizure detection. These features were spike correlation and time-frequency correlation features. Features have also been designed for the detection of temporal lobe seizures [89]. Complexity and spectral features have been used to detect early stages of the Alzheimer's disease [109]. Nonlinear features have been developed for use in the examination of sleep in children [110].

Following a description of the features, the PCA dimensionality reduction process was described. The important normalisation step before the application of the PCA process was also described. This normalisation prevents high amplitude features from biasing the data. This process is important as it helps in the reduction of data that carries little variance.

The following chapters use the feature set described in this chapter, along with the PCA technique to achieve the maximum performance while using as little data as possible.

Chapter 4

Brute force nearest neighbour EEG recall

4.1 Introduction

The work presented in this chapter is the development of brute force tools to extract information from neonatal EEG data using distance metrics. A brute force approach is a way in which all combinations of a particular system can be tested. Examples of this include cracking databases by trying all password keys [111], finding vulnerabilities in systems [112], time series motif discovery [113] and finding shapelets for human gait recognition [114] to name a few. The work in this chapter employs brute force search techniques to locate the nearest neighbour (NN) neonatal EEG event from a database of EEG events.

This chapter is laid out as follows: Section 4.2 gives a brief introduction into the area; Section 4.3 discusses the fixed point Euclidean distance approach for both the time series and feature-based representations. Following this, in Section 4.4 the elastic dynamic time warping (DTW) approach is also detailed for both time series and feature-based representations. Next, in Section 4.5 a data transformation approach called the bag of patterns (BOP) approach is discussed. The results for each approach are presented and discussed in Section 4.6.

4.2 Background

In this work, the goal is to return a similar event from an existing EEG database in response to a query event; effectively a similar pattern is mined from a database of

patterns. The concept of data mining will first be introduced along with the techniques that have been used in various areas of research. The pros and cons of each system will be discussed along with the reasons for further investigating each implemented system in this chapter.

4.2.1 Data mining

Knowledge discovery in datasets (KDD) is used to discover useful information from data [115]. KDD refers to the whole discovery process, with data mining forming a particular step in the KDD process, in which algorithms extract patterns from the data [115]. Data mining has attracted much attention in recent years as large amounts of data have started to be collected and there existed a need to extract useful information from this data [116].

Data mining has uses in many different areas such as stock trading [117], text mining [118], fraud detection [119], music data mining [120] and for mining time series data [121, 122]. There are a plethora of data mining techniques that exist for each of the data mining areas mentioned previously. As the work in this thesis focuses solely on time series data, it is this area that will be explored further.

4.2.2 Time series data

The following are examples where time series data mining is used; prediction [123, 124], subsequence searching [125, 126], motif discovery [127, 128], classification [61, 129], clustering [130, 131] and anomaly detection [132, 133]. These approaches span different areas within the time series domain. The work in this thesis is carried out on physiological time series data and the EEG in particular with examples in the area including [46, 61, 134]. Examples of data mining techniques that have been applied to physiological signals include [127] in which the authors use motif discovery on ECG data. Classification can be carried out on EEG signals for the detection of seizures [61]. Clustering has previously been used to cluster six different ECG beat types [135].

The work presented in this thesis is based on the idea that there is a query event that is searched against a database of events to find the most similar event. The methods such as motif discovery and anomaly detection typically do not deal with labelled data, as these are employed to find useful information in the data. Motif discovery looks for unique repeating patterns in the data, while anomaly detection

looks at detecting differences from the normal operating conditions. Prediction approaches are used to predict future values from current trends. Clustering approaches are designed to cluster events based on their similarity and there are many useful ideas present in this area [131].

Fixed point distance approaches

For physiological time series data, the problem is one of identifying sequence similarity. This approach effectively compares the query sequence with every other sequence in the database by computing a distance metric in a similar way as was carried out by [136] on time series data. An example of a distance measure is the fixed point Lp distance measures, which are commonly used in data mining. The $L1$ ($p = 1$) or Manhattan distance is effectively the sum of the absolute difference between points in a set [137]. It is suggested that the Manhattan distance (L1) may be preferable to the Euclidean distance for high dimensional data as the distance between the points is not squared [138]. The Euclidean distance, also known as the $L2$ ($p = 2$) distance is a fixed point or point to point distance measure. Meaning it is the straight line distance from one point to another point, therefore it directly compares time series points that have the same time index. The Euclidean distance (L2) measure is widely used in the data mining field as it is one of the most straightforward measures to implement [48, 139].

The Euclidean distance has been previously used on physiological signals either directly on the time series data or on a feature representation of the data [134, 140, 141]. In [134] the authors investigate the use of the Euclidean distance and the DTW distance as pattern matching approaches for use in biometric applications. The minimal distance value between the comparison subjects is used to either accept or reject the individual identification. A distance threshold value can be used as a method to vary the strictness of the approaches. The paper by [140] investigates similarity measures for clustering ECG complexes. The paper compared the performance of the Manhattan distance (L1), Euclidean distance (L2), normalised correlation coefficient and the simplified gray rational grade for use as similarity measures. The authors stated that the Euclidean distance had the fastest execution time and each comparison approach was able to achieve an accuracy of over 99% for clustering correctly. In [141] the authors classified epileptic seizures using EEG, the Euclidean distance between EEG samples was used as a feature for the neural network classifier.

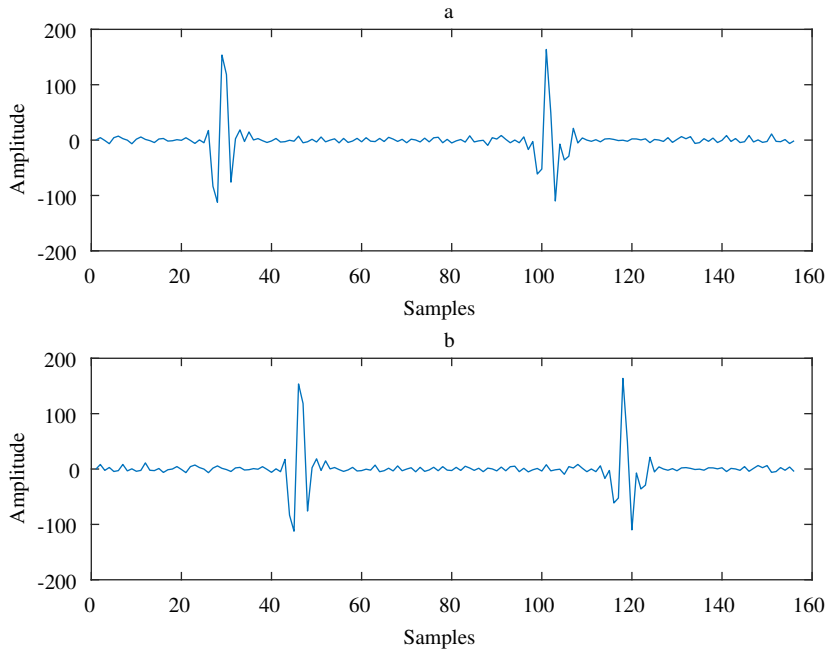


Figure 4.1: Example demonstrating the need for elastic approaches (a) two beats in an ECG, (b) same beats with 16 sample delay.

As the Euclidean distance is widely used, simple to compute and was previously used with EEG data it was decided to explore this distance metric further for the task of NN neonatal EEG recall.

Elastic distance approaches

A problem that exists with fixed point approaches such as the Euclidean distance approach is that alignment steps may need to be carried out prior to computing the Euclidean distance. Figure 4.1 shows an example of this where there are two ECG QRS complexes that are identical but have a delay of 16 samples. The Euclidean distance between these two signals is 480.29 as it cannot account for the misalignment. Elastic approaches exist that overcome the alignment problem of the Euclidean distance approach such as DTW [142], edit distance approaches [143] and longest common subsequences [144]. Using the DTW elastic approach to compare the signals in Figure 4.1 results in a distance of 25.03. The reason for this lower distance is that each elastic approach is dynamic and not a fixed point to point process. DTW can warp signals which reduces a misalignment problem and can work with signals of different sizes [145]. The amount of warping is the distance measure used for DTW, with a lower warping distance considered as a closer match. The disadvantage of DTW is that it is more computationally expensive than the Euclidean

distance search. Much work has been carried out to reduce the computation time of DTW [52, 146, 147]. The edit distance was initially developed for string similarity measures and it is the number of operations needed to transform one sequence into another sequence [148]. The authors of [143] expanded the edit distance idea from the string similarity area to time series applications. This edit distance approach shares some similarities with the DTW algorithm in the warping ability. The longest common subsequence is another elastic approach that can be used for time series similarity. This approach does not require the events to be aligned. The similarity between two time-series events is determined by the longest common subsequence between the events [144]. It has been shown that the longest common subsequence performs well for noisy signals and outperforms the Euclidean distance approach in similarity problems [149].

What is common for all the elastic approaches is that they can be computationally expensive. Constraints can be implemented to decrease the computational cost of elastic measures although these can change the behaviour of each method. DTW has been demonstrated to have a slight advantage over other elastic approaches depending on the constraint system used [150]. There has been a significant amount of work carried out in the area of DTW in terms of increasing the performance and computational speed of the algorithm [52]. In this work both Euclidean distance and DTW methods will be used for NN EEG recall.

4.2.3 Transforming time series data

Computing distance measures on time series data can be computationally expensive and can also return poor results [78, 151]. It has been shown that calculating the Euclidean distance between feature representations as opposed to using the direct time series representation yields improved performance [151]. The poor performance of using the raw time series is also backed up in [139] where it is said that data mining approaches require a higher level representation of the data.

Feature representation

The idea of features was introduced in Chapter 3. In a summarised version, features are used to represent the data in a compressed and meaningful manner. The features then become the new representation of the data and the distance measurements would be carried out directly on the sequence of features [151].

There exists a broad range of possible features that can be extracted from physiological time series signals. The extracted features are generally engineered with input from a domain expert, examples include; seizure detection in neonatal EEG [33], ECG arrhythmia detection [152], EMG pattern recognition [153] and EEG emotion recognition [154] to name a few. These features may share come commonalities although in most cases they are designed and fine-tuned for each particular application. The neonatal EEG features from Chapter 3 [33] were used for this work, as they were specifically designed to help capture meaningful information from the neonatal EEG.

Compressed representation

Apart from feature extraction other techniques have previously been employed to transform physiological signals into a more compressed representation. These techniques involve reducing the length of the time series signal or using the structural representation to index the signals [155]. These methods include different sampling techniques to reduce the amount of original time series data points [121], compressing the signal by averaging subsections [156], transforming the time series signals into symbolic representations [157], transforming the time series into a compressed binary representation [158] and more techniques can be found in [131].

Each method has its strengths and weaknesses, with each type of transformation or compression resulting in a loss of data. The work in this chapter focuses on using a combination of averaging subsections and transforming the signal into symbolic representations. A reason for this is that both the averaging and symbolic representations can be fine-tuned to decide what level of compression the application requires.

4.3 Euclidean distance

The first method presented is the Euclidean distance approach, which is a popular and straightforward point to point distance measurement [48]. The Euclidean distance has many uses spanning several different fields including image recognition [159], DNA sequencing [160], speaker identification [161] and EEG biometric identification [134] to name a few.

In this work, the Euclidean distance is used as a distance metric to locate the NN

neonatal EEG event. It is also used to generate an ordered list of the most similar events based upon their distance.

The distance between two points of time series signals x and q where x is a database event and q is the query event are of the form $x = (x(1), x(2), ..., x(n))$ and $q = (q(1), q(2), ..., q(m))$ for time $t = (1, 2, ..., m)$ is computed using Equation 4.1.

$$d(x(i), q(i)) = \sqrt{(x(i) - q(i))^2} \tag{4.1}$$

This is the Euclidean distance between points $x(i)$ and $q(i)$. Equation 4.2 is used to compute the Euclidean distance for a whole time series event, this computation is computed for each point and summed together to give a single distance value over the m points in the sequence.

$$d(x, q) = \sum_{i=1}^{m} \sqrt{(x(i) - q(i))^2} \tag{4.2}$$

This single value $d(x, q)$ can be used as a measure of how similar the signals x and q are, and for the fixed point distance metrics m must equal n. Figure 4.2 shows two example calculations of $d(x, q)$, with $m = n = 50$. The distance for corresponding points $d(x(i), q(i))$ is computed as indicated by the vertical lines. Summing the distance for each of the point pairs between the two signals gives a distance of 7.07 units for example Figure 4.2 (a) and 1.86 for Figure 4.2 (b). It was expected that Figure 4.2 (b) would have the lowest distance as the signals are very similar.

The distance between a query event q and every event in the neonatal EEG event database is computed based on Equation 4.2. This will produce a distance metric for every event in the database against the query q. The event with the lowest distance will then be returned as the NN. There exists two main problems with this approach;

1. Signals have to have the same number of samples ($n = m$).

2. Additional alignment steps would need to be carried out.

The problem with alignment is that if the signals are slightly misaligned a cost will develop and grow as the signal length increases. To demonstrate this idea, Figure 4.3 shows two signals with the same frequency of $0.5Hz$ but with a phase delay of $\pi/2$. The Euclidean distance between these signals thus grows as the signal increases in duration despite being very similar.

The Euclidean distance can work well with smooth and slowly changing time series signals. Signals with higher and mixed frequencies such as EEG can lead to

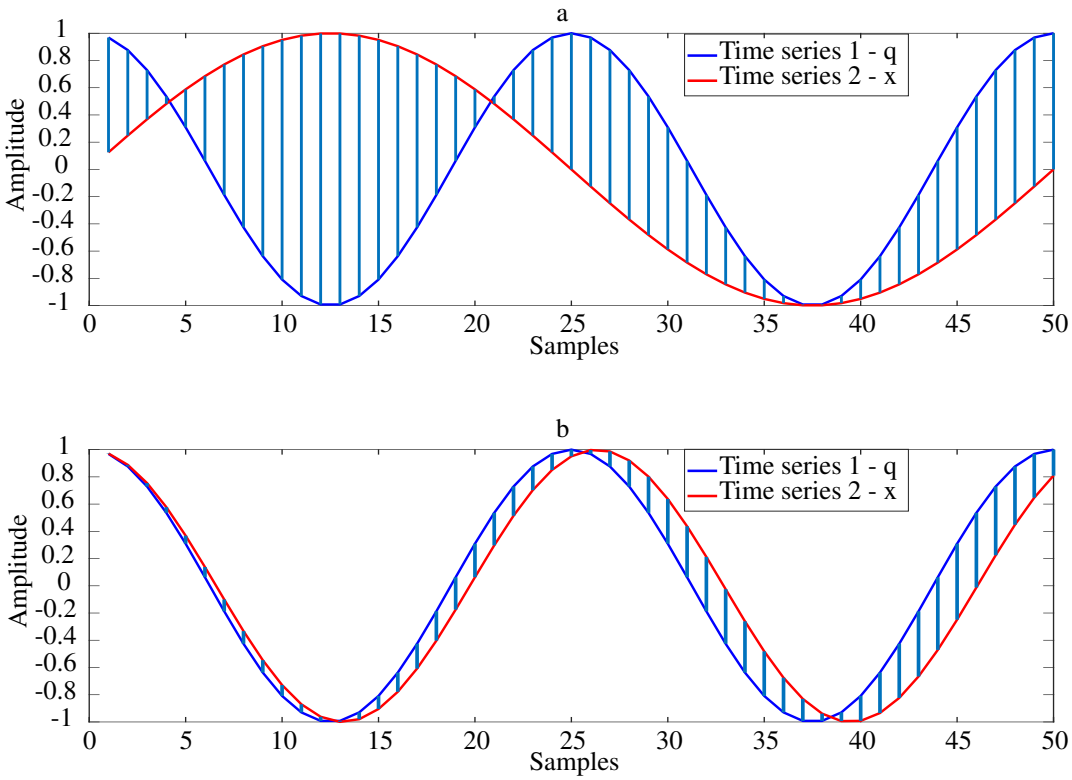


Figure 4.2: Example demonstrating how the Euclidean distance is computed for (a) signals with different frequencies, (b) signals with similar frequencies. The query is represented by q and the database signal is represented by x. The length of the vertical lines between the points indicate the associated distance.

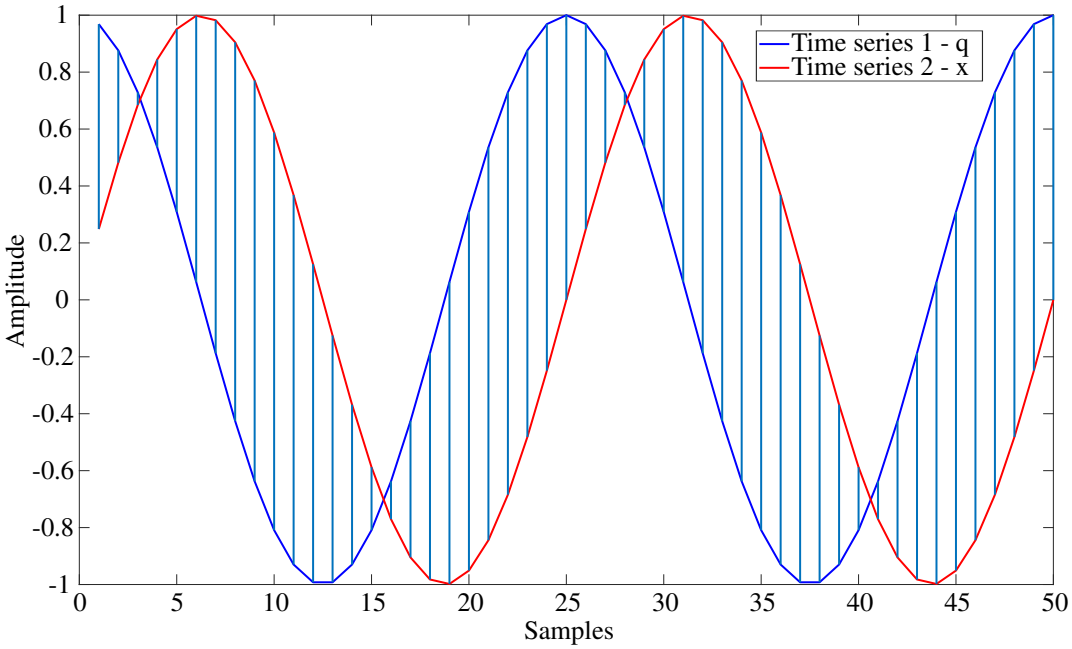


Figure 4.3: Euclidean distance with constant phase delay.

relatively poor performance in the time domain, notably for more extended time series [78].

4.3.1 Extending to feature domain

When extending an EEG time series distance-based approach to a feature-based approach, it is important to use meaningful features. The features offer a more informed and compact representation of the neonatal EEG signals. The features that were extracted were discussed in Chapter 3. These are extracted for signals x and q to give feature sequences $X = [X_1, ..., X_F]^\intercal \in \mathbb{R}^{F \times F_n}$ and $Q = [Q_1, ..., Q_F]^\intercal \in \mathbb{R}^{F \times F_n}$ respectively where F is the number of features, F_n is the number of feature epochs in the feature domain representation and $X_f \in \mathbb{R}^{F_n}$ is the vector sequence corresponding to feature f. As mentioned in Chapter 3 the features are extracted from the time series signals with 8-second epochs and a 50% overlap.

The next task was how to incorporate the features into a NN recall system. Each feature has different units of differing sizes. To overcome this problem each feature from each event is normalised using the Z-score normalisation [86]. This normalisation is performed using Equation 4.3

$$\overline{X}(f,i) = \frac{X(f,i) - \mu_f}{\sigma_f} \tag{4.3}$$

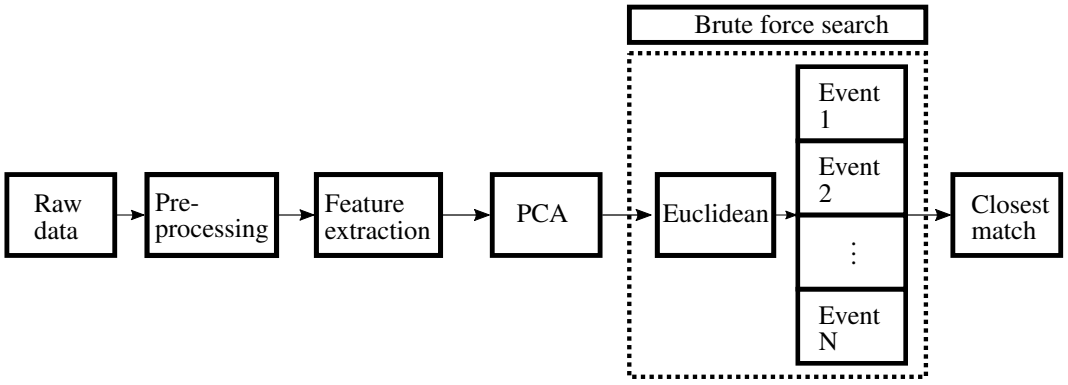


Figure 4.4: Euclidean flowchart for when a query is being carried out. The area where the query is brute force searched through the database of events is highlighted.

where $\overline{X}(f,i)$ is the normalised feature value of $X(f,i)$ for the i^{th} epoch of feature f, μ_f and σ_f are the template mean and standard deviation respectively for feature f and were computed using all the data in the database. This normalises the features and using template means and standard deviations ensures that the relative amplitudes for each event is maintained. Normalising each event individually without the templates loses important amplitude information maintained within the features. Equation 4.3 thus eliminates bias from larger amplitude features. To compute the Euclidean distance between the feature sets of event x and query q represented in the feature space as X and Q respectively, Equation 4.2 is extended to sum together the distance from each of the F_n feature epochs for each of the F features as shown in Equation 4.4.

$$D_Euclidean(X,Q) = \sum_{f=1}^{F} \sqrt{\sum_{i=1}^{F_n} \left(\overline{X}(f,i) - \overline{Q}(f,i)\right)^2} \tag{4.4}$$

This equation, similar to Equation 4.2 provides a single distance measure value between events X and Q. The query process is the same as was carried in the time domain. The query q has its features extracted followed by the query Q being compared to every event in the database using Equation 4.4.

As was discussed in Chapter 3, some features may not contain valuable information. It was for this reason that the system was also evaluated using a lower dimensional feature set. The dimensionality of the feature set was reduced using principal component analysis (PCA) as it is essential that the amount of data retained is kept as small as possible for when the database grows. The results obtained and dimensionality reduction are discussed Section 4.6.

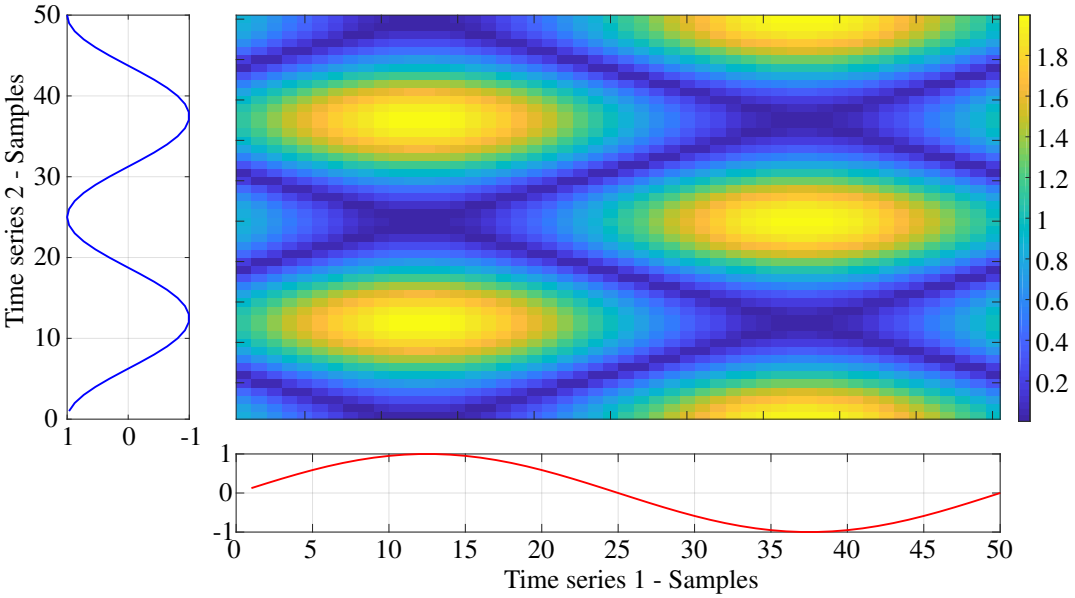


Figure 4.5: Cost matrix for 2 example signals. The colour at each cell in the matrix indicates the difference between the points, with the colour bar providing a guide to the differences.

Figure 4.4 shows the system flowchart when a query is being carried out. This brute force search approach is linear and in this flowchart, a single closest match or NN is presented. The performance accuracy of this system is based on the NN match alone.

4.4 Dynamic time warping

The next technique to be developed is an elastic distance method [162]. DTW was chosen as it is the most widely used elastic distance method for comparing time series signals [142].

In this work, DTW was used to locate a NN neonatal EEG event from a database of EEG events when a query is presented. DTW can be divided into multiple stages [163]. A $n \times m$ cost matrix c is first generated by comparing the n elements in x $(x(1),...,x(i),...,x(n))$ to the m elements in q $(q(1),...,q(j),...,q(m))$. Note, m does not need to be equal to n for elastic methods.

$$c = \begin{bmatrix} d(x(1),q(1)) & \dots & d(x(1),q(m)) \\ \vdots & \ddots & \vdots \\ d(x(n),q(1)) & \dots & d(x(n),q(m)) \end{bmatrix}$$

Where $d(x(n), q(m))$ is calculated using Equation 4.1. Figure 4.5 shows an example of a cost matrix for the signals depicted in Figure 4.2 (a). Figure 4.5 shows that when a peak is aligned with a trough, it results in a considerable distance, which is apparent while looking at the cost intensity at approximately $(12, 12)$ in Figure 4.5 indicated by the bright yellow intensity. Similarly aligned peaks or troughs have a low associated cost which can be seen at approximately $(12, 25)$ in Figure 4.5 where the intensity is dark blue.

From the cost matrix c, the goal is to find the minimum alignment cost. This is achieved by creating a warping path $\{w_1, ..., w_L\}$, where $w_k = (i, j)_k$, for $k \in \{1, ..., L\}$ under the following conditions:

1. Boundary condition: $w_1 = (1, 1)$ and $w_L = (n, m)$
2. Monotonicity: $w_k = (a, b)$, $w_{k-1} = (a', b')$ where $a - a' \geq 0$ and $b - b' \geq 0$
3. The set of possible steps: (w_{l+1} - w_l) $\in \{(1,0),(0,1),(1,1)\}$ for $l \in$ [1:L-1]

It is worth noting that the length of the warping path (L) is dependent on the warping path that is chosen. There are many possible warping paths available with the warping distance defined as D_w is shown in Equation 4.5.

$$D_w(x, q) = \frac{1}{L} \sum_{k=1}^{L} c_{w_k} \tag{4.5}$$

The value c_{w_k} is the k^{th} warping path position $(i, j)_k$ from variables x and q in the cost matrix c. With a large number of possible warping paths available, an accumulated cost matrix is used as a faster and less computationally intense method of computing every possible warping path. The accumulated cost matrix Ac is of the form:

$$Ac = \begin{bmatrix} c(1,1) & \dots & \sum_{j=1}^{m} c(1,j) \\ \vdots & \ddots & \vdots \\ \sum_{i=1}^{n} c(i,1) & \dots & D(n,m) \end{bmatrix}$$

When $1 < i \leq n$ with $j = 1$ and $1 < j \leq m$ with $i = 1$ the accumulated cost is calculated as the sum of the costs along the axis. For all the other positions Equation 4.6 is used.

$$\begin{aligned} D(n,m) = min(D(n-1,m-1), D(n-1,m), \\ D(n,m-1)) + c(n,m) \end{aligned} \tag{4.6}$$

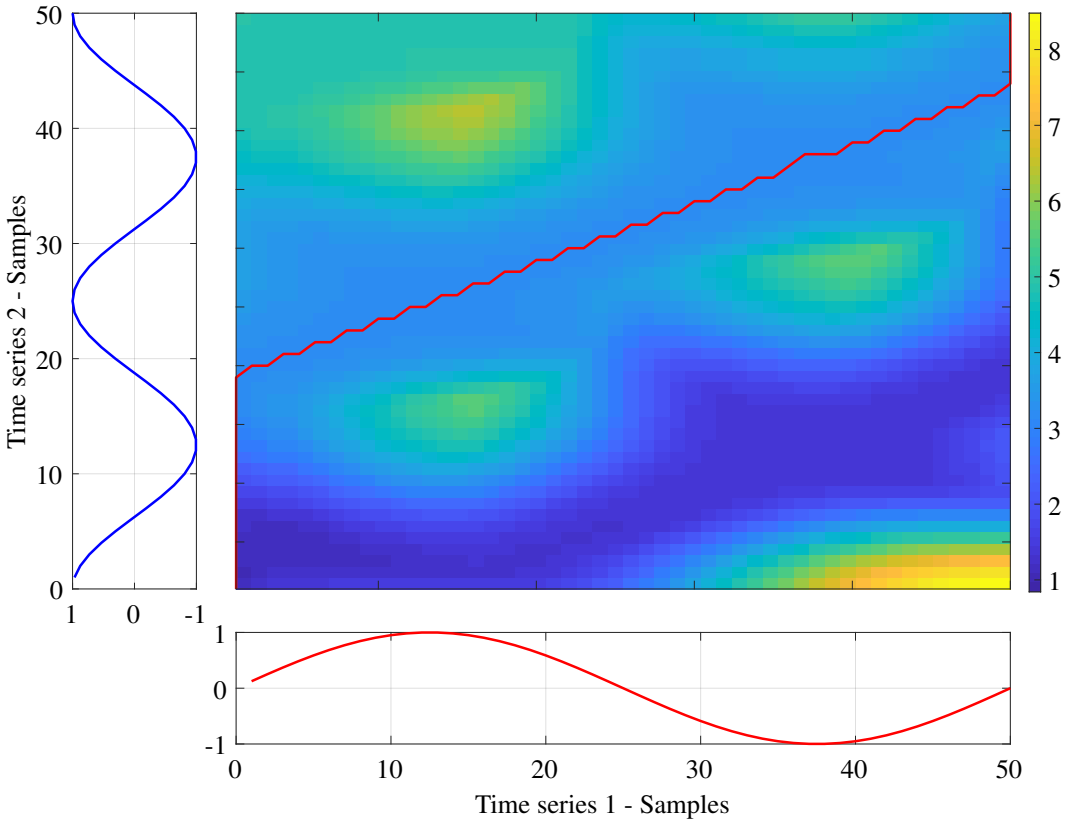


Figure 4.6: Accumulated cost matrix with optimal warp path example. As both signals move from the origin the associated cost increases. The optimal warp path is shown in red. This is the path that has the least distance when warping the signals to be as similar as possible.

This accumulated cost matrix Ac is of the size (n,m) and shows the evolving cost over the sequence. The optimal warping path is then found by working back from the end to the start of this matrix while keeping the lowest cost [163]. Figure 4.6 shows the accumulated cost matrix for the signals depicted in Figure 4.2 (a). It is apparent in the image that the cost increases with distance from the start point. The optimal warp path is the red line. In the ideal case when the signals being compared are very similar the optimal warp path would traverse the diagonal of the accumulated cost matrix. It would be expected that the optimal warp path for Figure 4.2 (b) would nearly traverse the diagonal as the signals are very similar.

The optimal warp path alignment of signal x to signal q is shown in Figure 4.7, and this has a distance of 3.74 units for this particular example. The mapping lines are the lines connecting time series signals x and q. The angle and magnitude of the mapping lines indicate the associated cost. There are vertical and near vertical lines at the start and end indicating a higher cost in these regions. The more horizontal lines

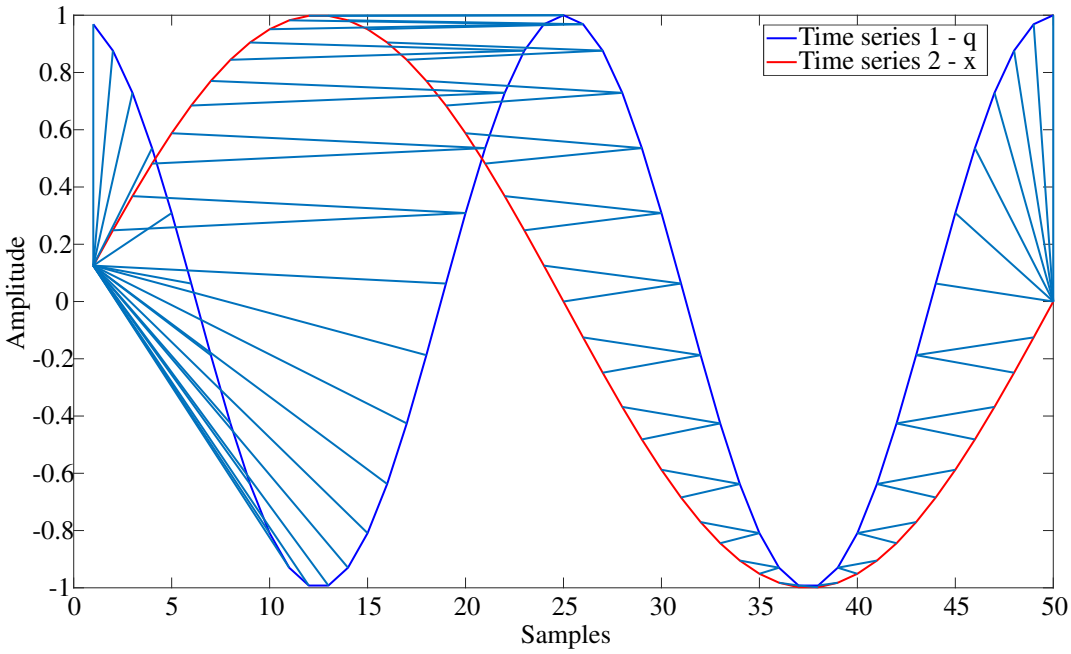


Figure 4.7: Example demonstrating DTW mapping between time series signals x and q.

with a slight angle indicate a small cost. It can be noticed that often two mapping lines from signal x are going to the same location on time series q. This mapping is occurring as it appears time series x is being compressed onto time series q. The mapping is clearly different to that of Euclidean approach in Figure 4.2 (a) which are all vertical.

Figure 4.8 (a) shows the original signals before warping. Following the application of DTW, the optimal warp path was attained which is shown in Figure 4.8 (b). It is clear that a large portion of the warping cost occurred at the start and end of the lower frequency signal. This warping was because the lower frequency component was compressed to fit the central section of the higher frequency signal, which is consistent with the vertical lines seen in Figure 4.7.

What is interesting between Figure 4.8 (a) and (b) is that there is a different number of samples between the two images. This difference occurs as a result of computing the optimal warp path. If the warp path or red line in Figure 4.6 traversed the diagonal, it would indicate the signals are the same and would have the same number of samples. As the warp path drifts from the diagonal, it introduces more samples into the warping path. The warp path in Figure 4.6 does not travel along the diagonal during the start and end of the warping process, and this is where the extra samples were introduced in Figure 4.8 (b). Table 4.1 (a) shows a simple example

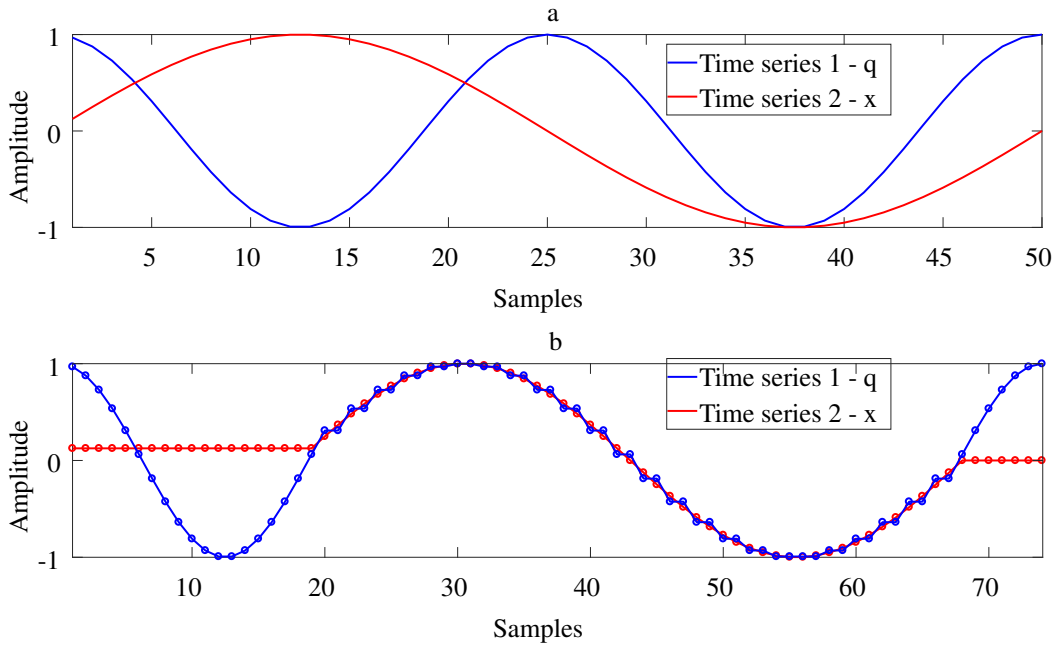


Figure 4.8: (a) Signals before warping, (b) signals warped according to the optimal warping path.

of the optimal warp path traversing the diagonal. Table 4.1 (b) shows the pessimal path while following the boundary conditions and monotonically increasing. The numbers in Table 4.1 correspond to the samples required to complete the warping.

One observation is that traditional DTW is an expensive process to compute and this becomes more expensive with more extended sequences. There has been much work in this area to speed up the process by using bounding and early abandoning techniques. The work presented in [52] has utilised these techniques and developed them into the publicly available UCR suite. The UCR suite was used for the DTW computation in this work.

Table 4.1: Tables showing optimum (a) and pessimal (b) warping path.

(a) Optimal path

				5
			4	
		3		
	2			
1				

(b) Pessimal path

5	6	7	8	9
4				
3				
2				
1				

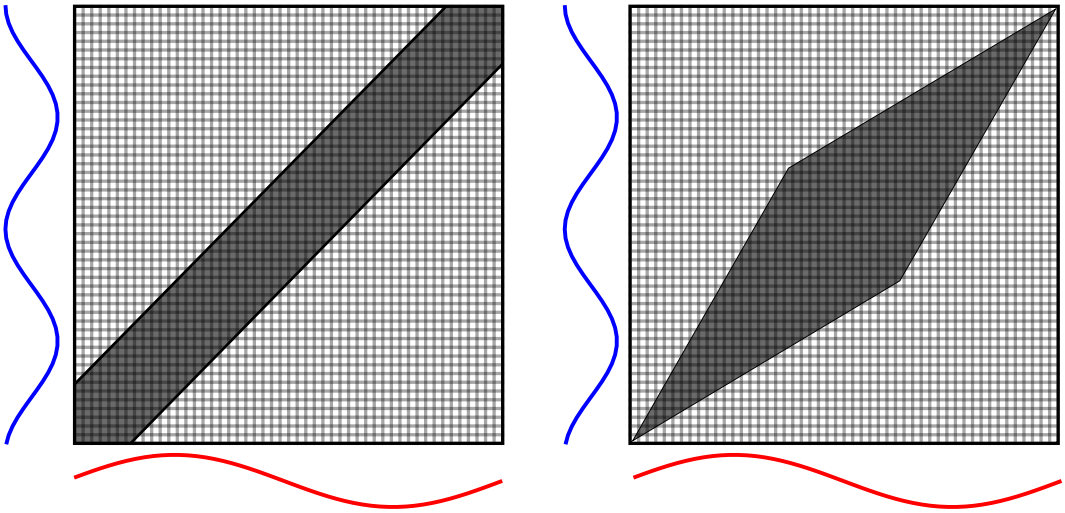


Figure 4.9: Sakoe-Chiba band and Itakura parallelogram bounding.

4.4.1 Bounding

There are several types of DTW bounding approaches that exist. The traditional and well know methods include the Sakoe-Chiba Band [142] and the Itakura parallelogram [164]. These are global constraints on the warping path, that constrain the warping of the query event q to be kept within a specific region of the database event x as depicted by the shaded regions in Figure 4.9.

The warping region allowed (the reach), is denoted by e, and the indices of warping path position $w_k = (i, j)_k$ have a reach as follows: $j - e \leq i \leq j + e$. This reach allows warping of e positions before and after the current point in the warping window. The warping reach e for the Sakoe-Chiba band is fixed whereas the warping reach e for the Itakura parallelogram varies, depending on the position i. These methods allow the signals to be warped by a certain margin and must be kept within the boundary as depicted by the shaded region. The bounding creates an envelope around the signal that the query must lie within. This approach prevents poor alignments as seen in Table 4.1 (b) where minimal warping takes place. A more explicit example of this is presented in [165] where they show the envelope created by the Sakoe-Chiba band around the query. The bounding region can be expanded or compressed depending on what the user wants. As the bounding region is compressed the distance measure approaches the Euclidean distance.

Two bounding techniques used in the UCR suite are known as LB_{Kim} [166] and LB_{Keogh} [47]. The LB_{Kim} bounding technique computes a four point vector

from each sequence being compared. The first point in the bounding vector is the difference between the start points of the two signals being compared. The next point in the bounding vector is the difference between the end points of the two signals being compared. The next point is the difference between largest amplitudes of both signals. The last point in the bounding vector is the difference between the lowest amplitude points of the signals being compared. This bounding vector now consists of four values and the largest absolute value of these four points is taken as the lower bound. When the signals are normalised the maximum and minimum values tend to be similar, it is for this reason they are not used in the UCR suite [52]. The LB_{Kim} bound is quick to compute and is used as an initial bounding measure before the application of the LB_{Keogh}.

The LB_{Keogh} method uses the reach e of the global constraints such as the Sakoe-Chiba band or Itakura parallelogram to create two sequences u and l standing for the upper and lower respectively represented by Equations 4.7 and 4.8.

$$u(i) = max(q(i-e) : q(i+e)) \tag{4.7}$$

$$l(i) = min(q(i-e) : q(i+e)) \tag{4.8}$$

When these sequences are available, they are then used in the LB_{Keogh} method as a part of the distance measure. This measure takes the aggregate distance of the event and either the upper u or lower l sequences when the event is outside these sequences. This measure is represented by Equation 4.9:

$$LB_{Keogh}(x,q) = \sqrt{\sum_{i=1}^{n} \begin{cases} (x(i)-u(i))^2 & \text{if } q(i) > u(i) \\ (x(i)-l(i))^2 & \text{if } q(i) < l(i) \\ 0 & \text{otherwise} \end{cases}} \tag{4.9}$$

Here x and q represent two time series signals of length n and m respectively, $q(i)$ is the i^{th} time series point from the q time series and values $u(i)$ and $l(i)$ represent the upper and lower envelope sequence positions for the i^{th} time series point. More information and graphical demonstrations on these bounding techniques are presented in [47, 52, 166]. In the UCR suite, the envelope sequences are computed more efficiently using the method provided in [167].

The LB_{Kim} and LB_{Keogh} bounding techniques can then be used as a form of

early abandoning to the DTW computation to save computational effort as will be discussed next.

4.4.2 Early abandoning

When searching a database of neonatal EEG events, it is essential to be as efficient as possible. It is for this reason that early abandoning is important. If the user of the system decides only to return the most likely match, then early abandoning can be used to reduce the computational time of the brute force search if specific criteria are met.

In general, a Best So Far (BSF) value is initially set to infinity. After the distance is computed between the query and the first event in the database the BSF value is updated to be this new calculated distance. The NN has the lowest BSF value, so the aim is therefore to find events that have a lower DTW distance than the BSF.

When working through the accumulated cost matrix to find the optimal warping path between a query and an event in the database, a cost or distance is developing. If this distance surpasses the BSF distance, the DTW distance computation can be abandoned for this event as it is no longer a suitable candidate for the NN. It can be abandoned because the optimal warp path can only monotonically increase, meaning the distance will never drop below the BSF distance. In essence early abandoning halts the computation of the DTW distance if the distance is higher than a specified value (BSF) [168, 169].

The lower bound measures LB_{Kim} and LB_{Keogh} are used for early abandoning. If the values of LB_{Kim} or LB_{Keogh} are higher than the BSF value, the DTW computation can stop. In [166] they state $DTW(x,q) \geq LB_{Kim}(x,q)$ always holds. In [47] they showed that $LB_{Keogh}(x,q) \leq DTW(x,q)$ and that is why LB_{Keogh} can be used for early abandoning when the value of $LB_{Keogh}(x,q) > BSF$. If $LB_{Keogh}(x,q) < BSF$ the full DTW distance calculation will be calculated and the BSF value will be updated to the new value. If five nearest neighbours were required the BSF value is set to the lowest distance in the top five results. When testing if a new event performs better than the BSF, the event with the highest distance is removed from the list.

4.4.3 Extending to feature domain

DTW can work well for short physiological time series data as discussed in [52, 170]. As the time series grow in length, it becomes unrealistic to use DTW on the direct

time series data [52, 171]. One reason for this is that if a physiological time series signal such as neonatal EEG is warped too much the signal loses its originality, and it may effectively be creating a new random signal. The other problem faced by DTW is that as the signal lengths increase the computational cost increases as stated in [172].

Neonatal EEG time series signals can be transformed into a feature space representation. This transformation was achieved by extracting the features that were discussed in Chapter 3. Since the features are not randomly chosen the feature set thus represents a compressed representation of the original signal. The DTW is then applied to the feature domain representation as was similarly carried out in [171]. Figure 4.10 gives an illustration of the process for two signals x and q. Initially, the time series signals x and q have the 55 features extracted. Before the DTW algorithm is applied to the feature vectors these need to be normalised as different features have different units and higher amplitude features dominate. This normalisation is the same method as previously discussed for the Euclidean approach in Section 4.3.1. The template normalisation step is omitted in Figure 4.10 for simplicity. Then the DTW distance is calculated for each feature, where "DTW-1" is the distance when computing the DTW distance for feature 1 between both signals. As the features would have been normalised it enables the distances from each feature to be summed together without being biased from higher amplitude features and a single distance metric between the two signals can be returned.

To compute the DTW distance between the feature sets of event X and query Q, the optimal warp path Equation 4.5 is extended to sum together the distance from each of the F features as shown in Equation 4.10.

$$D_DTW(X,Q) = \sum_{f=1}^{F} D_w(X_f, Q_f) \tag{4.10}$$

It is worth noting that each of the F features have an optimal warp path. Thus L_f is the warping path length for feature f. Using Equation 4.10 will thus provide a single distance value between the feature events X and Q.

The PCA dimensionality reduction technique is needed for the same reasons as described in Section 4.3.1. To be consistent with the Euclidean distance method the same dimensionality reduction approach of PCA was carried out and the results are presented in Section 4.6.

Figure 4.11 shows a flowchart of the DTW system when a query is present. This

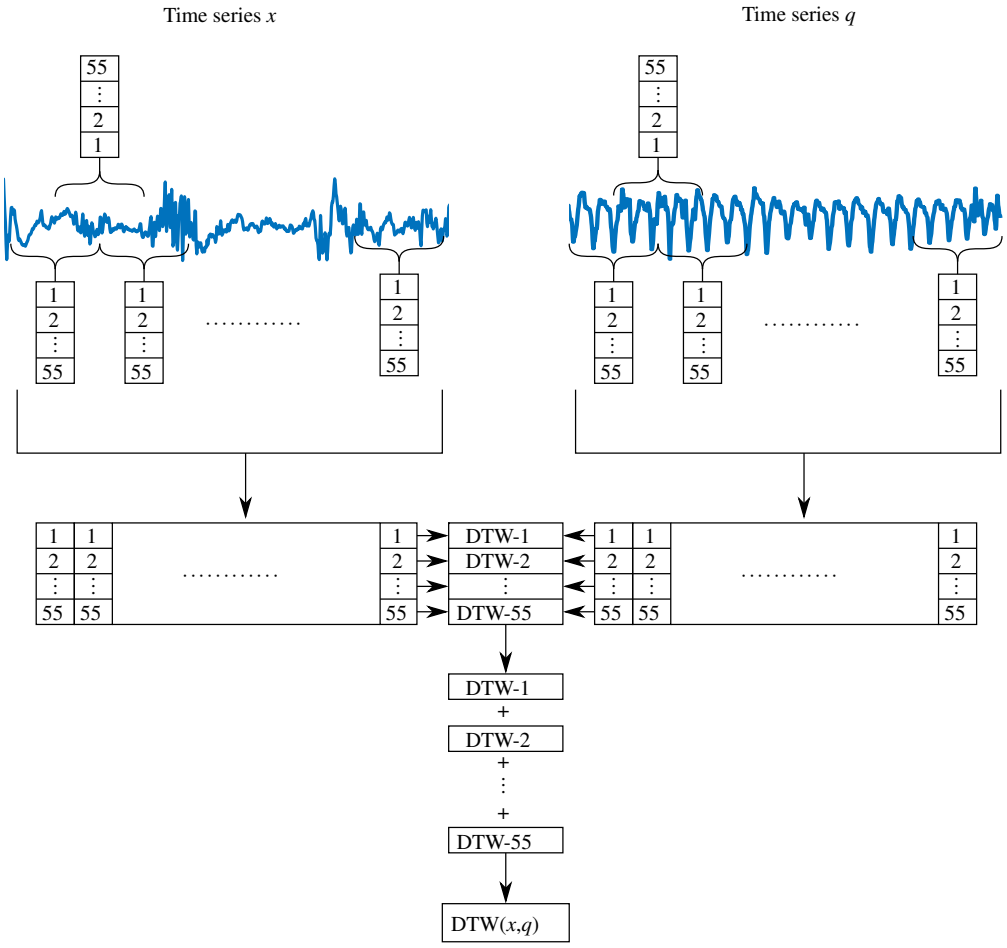


Figure 4.10: Example of the DTW process using a set of features extracted from the EEG signals.

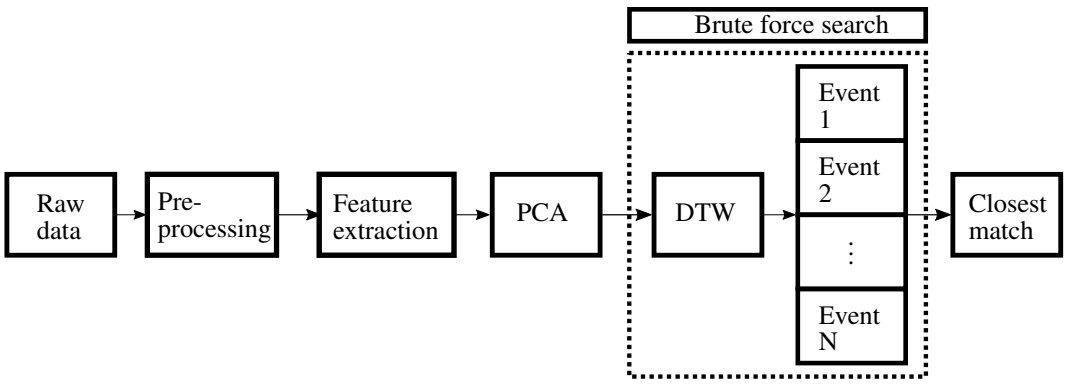


Figure 4.11: DTW flowchart showing the steps involved in the DTW search algorithm. The area where the query is brute force searched through the database of events is highlighted.

flowchart is very similar to the Euclidean distance flowchart in Figure 4.4 with the difference being the DTW block in the brute force section.

4.5 Bag of patterns

Brute force distance metric approaches can be expensive in terms of both time and memory to compute on time series signals and this increases as the size of the database increases [172].

Sections 4.3 and 4.4 discussed brute force distance metric based searches based in the time and feature domains. These sections included the idea that applying the approaches directly to the time series data would yield poor results. It was for this reason that the time series events were transformed into a compressed feature-based representation.

An alternative transformation approach known as Bag of Patterns (BOP) [78] was applied to the neonatal EEG with the aim of reducing computational effort and memory burden while maintaining a similar level of performance.

The premise for the BOP came from the Bag of Words (BOW) concept, which has been used to represent the structure of documents for text classification [173]. Essentially all the words in the document are thrown into a "Bag" and then sorted by the occurrences of each word, completely disregarding the order of the words. The output array is a fixed size, and this is the vocabulary size or the range of possible values. The output array is then the frequency of occurrence of all the potential words. This output array may be sparsely filled if a large vocabulary exists. Table 4.2 shows an example of BOP where the first column contains the set of possible words. The second column then contains the frequency of occurrence for event 1. If a word is not present in the event, that table entry is zero.

The paper by [78] stated two problems that exist when trying to apply the Bag of Words approach to time series data;

1. What pattern vocabulary will be used?

2. Breakpoint (delimiters) between patterns?

Time series data are data points that are indexed by time. These can consist of random or definite patterns. Therefore, an infinite vocabulary could be required.

The second problem is how to divide a pattern into words. In the text space, these are known delimiters, as follows:

Table 4.2: BOP example showing how the N events of a fixed size are stored.

	Event 1	Event 2	...	Event N
1111	0	15	...	8
1112	9	2	...	11
⋮	⋮	⋮	⋮	⋮
3124	6	3	...	12
3131	4	5	...	0
⋮	⋮	⋮	⋮	⋮
4444	1	0	...	0

- After each letter
- The end of a word
- A paragraph break
- Special characters (full stop, comma, brackets)
- HTML tags

The delimiter used depends on the applications such as word segmentation which could use any of the first four shown above [174] and website text extraction that could use HTML tags [175]. Delimiters are unfortunately not present in time series signals.

It is hard to know how long or how many samples of the EEG should be used to form a word. It is for this reason that the EEG is transformed into a compressed representation, from which words are generated. The symbolic aggregate approximation (SAX) algorithm is used to generate words [157].

4.5.1 SAX

The SAX algorithm is applied to the time series event x using a sliding window p of length n_s to transform the signal into a collection of unique words. This technique is a form of lossy compression where the original time series data undergoes an irreversible encoding process. The user can define the level of compression, and different applications may require different amounts of encoding. Before the SAX algorithm transforms the subsection of data p, it undergoes a normalisation to ensure

the data has a mean of zero and a standard deviation of one. This normalisation is carried out using Equation 4.11

$$y(i) = \frac{p(i) - \mu}{\sigma} \tag{4.11}$$

where μ and σ are the mean and standard deviation of the subsection data p and y is the normalised signal p. Following the normalisation step, the piecewise aggregate approximation (PAA) [156] as a dimensionality reduction step is applied using Equation 4.12.

$$\bar{p}(i) = \frac{\omega}{n_s} \sum_{j=\frac{n_s}{\omega}(i-1)+1}^{\frac{n_s}{\omega}i} y(j) \tag{4.12}$$

Where $\bar{p}$ is the new compressed representation of the normalised time series signal y and $\bar{p}$ has ω sections. The PAA algorithm takes the average of the samples contained in each of the ω sections. The size of ω is user-specified and can be changed depending on the signal used. The PAA algorithm compresses p of length n_s samples to $\frac{n_s}{\omega}$ samples.

The next task is to transform the ω samples from PAA into ω symbols. At this point in the algorithm, there exists a real-valued array $\bar{p}$, which is a compressed representation of the original data as shown in Figure 4.12. The task is now to transform this real-valued array into a fixed sized vocabulary. The transformation assumes a Gaussian distribution to determine the ω SAX symbols by dividing the Gaussian distribution into α equally sized areas (and hence have equal probabilities), where α is the alphabet size.

Figure 4.13 shows the distributions for each of the subsections for the six different EEG events being examined. This justifies the usage of the Gassian distribution to divide the signal into α possible symbols. Transforming $\bar{p}$ into an alphabet will give a vocabulary size of α^{ω}. The symbols used in Figure 4.12 are the integer numbers from 1 to 4 and in this particular example symbol 4 never occurs.

4.5.2 Transformation

If the SAX transformation was carried out on the whole time series data x it would either lead to an unrealistically large data reduction (low ω) or an unrealistically large vocabulary size (large ω). Instead, the SAX transformation is performed on a subsequence of the data p using a sliding window of length n_s with a user-defined

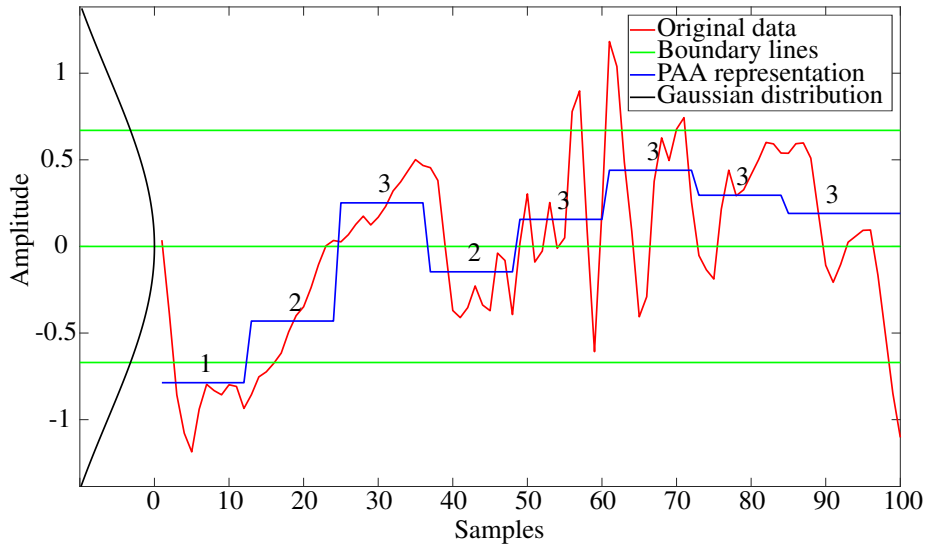


Figure 4.12: SAX computation example using a subsequence p with a length n_s of 100 samples, word length ω of 8 and alphabet size α of 4. The original signal is represented by the red line and the PAA reduced signal is represented by the blue line. The black Gaussian line on the vertical axis is used to assign symbols of equal probability, as given by the green boundary lines.

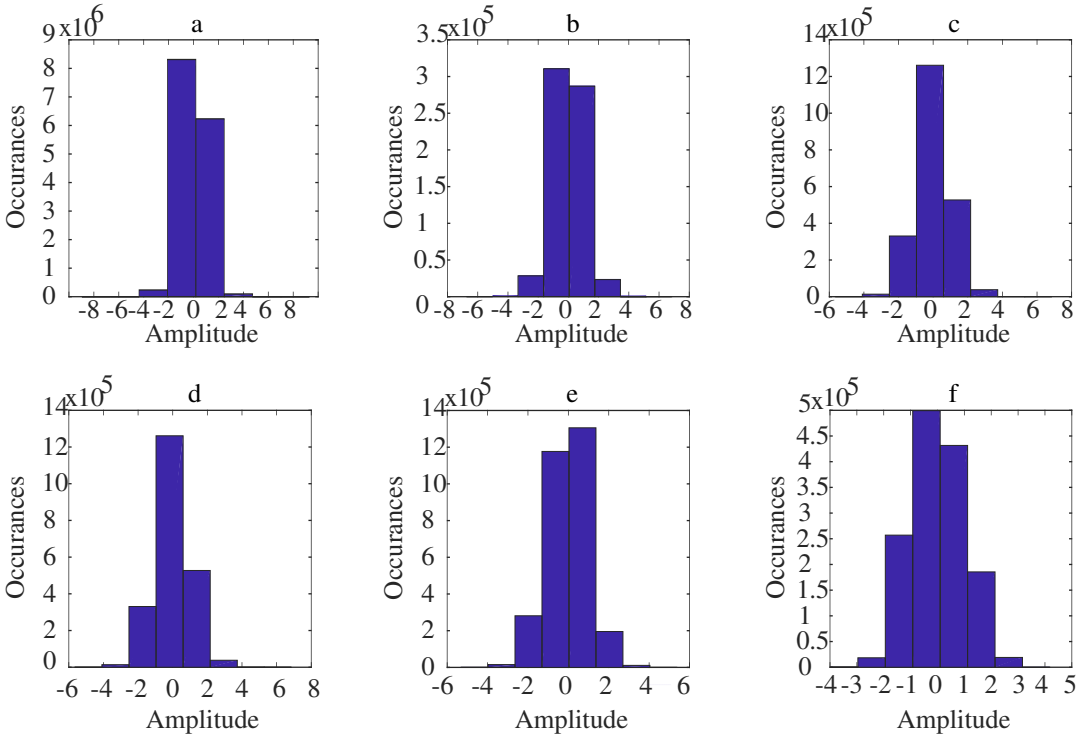


Figure 4.13: Distribution of data for all subsections of length n_s for each event group. (a) Background, (b) Short seizure, (c) Tracé Alternant, (d) Long seizure, (e) Respiration artifact, (f) Pulsatile artifact.

shift ρ. Figure 4.14 shows an example of how this is carried out on real data. Here, the SAX computation is carried out using windows of length n_s with a shift of ρ between SAX computations. The SAX operation is carried out a total of $\frac{n-n_s}{\rho}$ for each event.

The size of the sliding window is dependent on the data type. As highlighted in [172], smooth slowly changing data has the potential to have many occurrences of the same SAX pattern. It was suggested in [78] that it is feasible for smooth patterns to be described with a small ω and higher frequency patterns would use a large ω to capture the critical changes. When carrying out the SAX step there may be several consecutive sequences that have the same SAX sequences, the first sequence is kept, and the others are ignored. This step is known as numerosity reduction and is carried out to reduce the amount of "trivial matches" as stated in [78].

Once the original time series has undergone dimensionality reduction using SAX with a sliding window, the resulting set of sequences is used to create the BOP matrix entry for that particular time series event. Initially, an array of zeros which is the size of 1 x α^ω is generated. Each entry in the array corresponds to one of the α^ω sequences. Next, the frequency of each sequence present in the generated SAX sequences list is counted. The frequencies are then entered into their corresponding locations in the array. This 1 x α^ω array is the new higher level structural representation of the original data.

The above approach is repeated for all the N files that are in the database. Once this is complete a full BOP matrix of size N x α^ω is developed as depicted in Table 4.2. This is now the "database" which is to be searched. It is possible to add new events to this table without needing to regenerate the BOP matrix. If either the ω or α parameters are to be changed the whole matrix needs to be regenerated.

Figure 4.15 (a) shows background EEG data and Figure 4.15 (b) shows pulsatile artifact EEG data. The pulsatile waveform is more rhythmic and appears less stochastic than Figure 4.15 (a). Figure 4.16 (a) and (b) show the BOP transformation for the signals in Figure 4.15 (a) and (b) respectively. Figure 4.16 (a) is more densely populated than Figure 4.16 (b). The different distributions in the words is what assists the system to distinguish different event types, as different events will have different SAX sequence distributions. The pulsatile event has a higher density in some areas and is not as dispersed as the background event. This example used an alphabet size of 3 and a sequence length of 7. In the implementation that was used to generate the results that are presented in Section 4.6, an alphabet size of 3, a

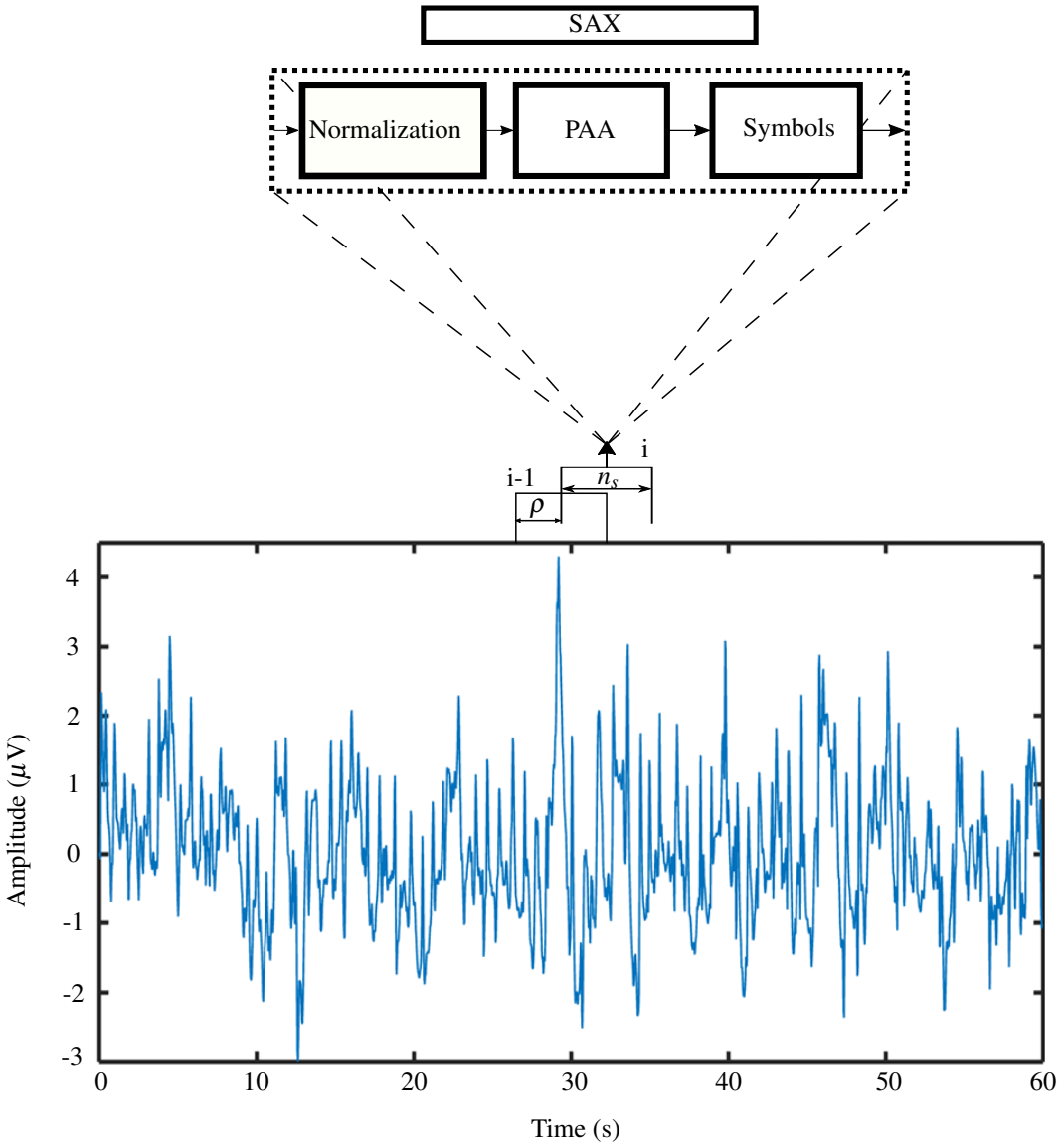


Figure 4.14: An example of how the SAX process is carried out on a 60 second signal, the process is carried out using a sliding window of size n_s and a shift of ρ between each window.

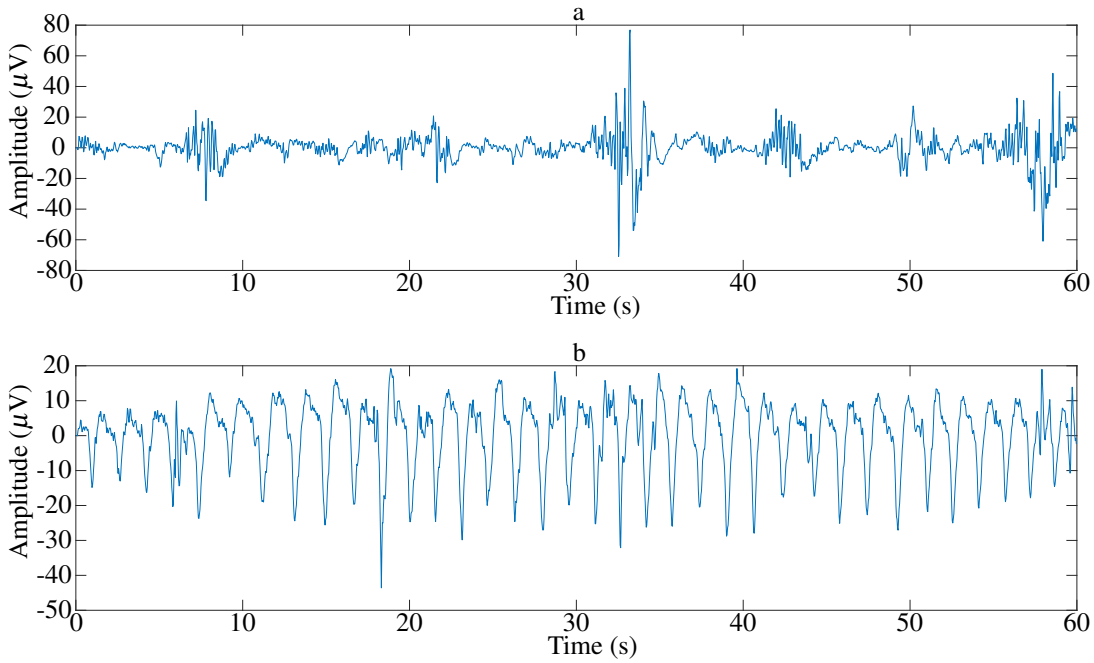


Figure 4.15: Example time series signals. (a) Background event. (b) Pulsatile artifact event

sequence length of 7, a window length of 89 samples and the window is shifted by 3 samples for each SAX computation. As there was a small number of parameters, these parameters were found via a brute force search of the parameter space.

When a query is carried out, the data q is converted into a BOP representation (1 x α^{ω} array). This BOP representation is brute force searched against the N x α^{ω} matrix to find the best match. The Euclidean distance is the analysis metric that is used. The event that has the lowest Euclidean distance is then seen as the closest candidate or the NN. Figure 4.17 shows the flow graph for the BOP approach.

4.6 Results

This section will describe and detail the results obtained for each system discussed in this chapter. Each of the systems were evaluated using the neonatal EEG event database. The performance of each system will be evaluated looking at three key areas;

- Accuracy
- Query running time
- Memory requirements

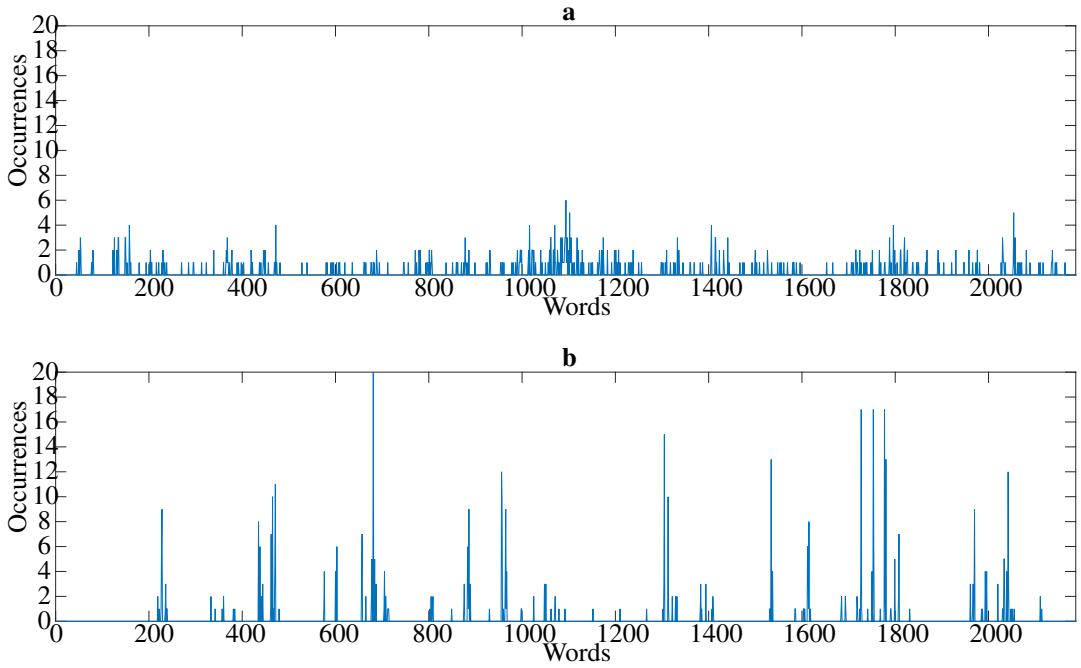


Figure 4.16: BOP transformation examples for (a) background event and (b) Pulsatile artifact event

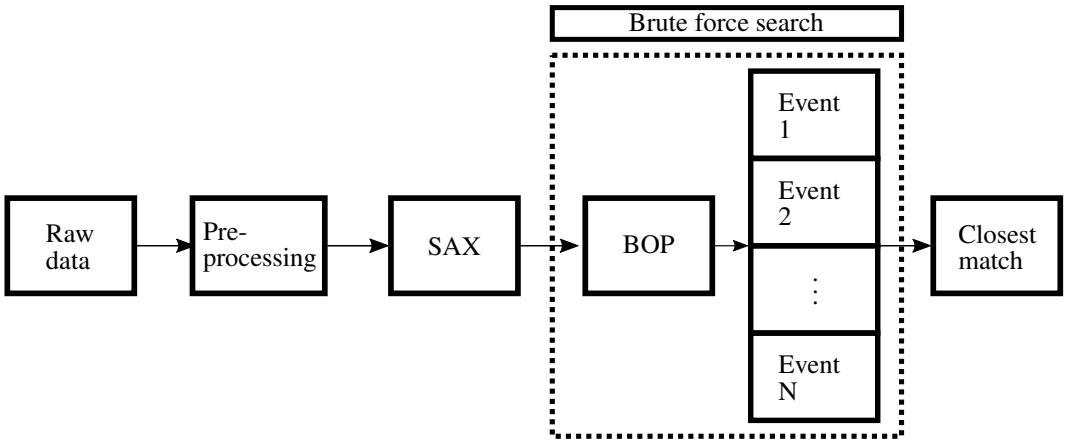


Figure 4.17: BOP flowchart for when a query is being carried out. The area where the query is brute force searched through the database of events is highlighted.

When evaluating the accuracy two measures were examined. The first measure was the group majority. This measured the event type that was recalled the most when evaluating the event types 2 to 6. For example, when evaluating all respiration artifact events (event 6), the class that was returned the most for this event group is seen as the event majority. It enables the evaluation into which event is recalled the most. Over half of the database was background data (event 1), therefore it is of interest to evaluate if this had much of an impact or bias on the performance of the different systems. The background events (event 1) were not tested in the query search. The reason being is that in a real system the background data or meaningless data would not be queried and with a large number of background events in the database it could skew the results. The second accuracy measure evaluated was the system recall accuracy and this accuracy is defined as the ability to recall the correct event type when testing events 2 to 6. This accuracy was calculated for each event group individually using Equation 4.13.

$$\text{Accuracy}(\%) = \frac{\text{Number of correct NN in group}}{\text{Number of events in group}} * 100 \tag{4.13}$$

For example, when testing the short seizure event group (event 2) the numerator is the number of correct NN matches where the event recalled came from group 2. The denominator is the total number of group 2 events in the database. The average accuracy was taken for event groups 2 to 6, and this was the metric used for the evaluation.

The query running time is the time it takes each system to carry out queries on the data. This measure is important as it enables estimations to be made about how each system will scale with an increasing database size. The memory requirements refers to the amount of memory required by each system. Again, this measure is important as it enables estimations to be made about how each system will scale when there is a larger database. If the systems take too long to compute queries or require too much memory alternate approaches may need to be implemented.

4.6.1 Accuracy

The results for the group majority and recall accuracy are presented in Tables 4.3 and 4.4 respectively. This subsection will detail the performance of the three systems. Firstly, the Euclidean and DTW time series and feature representation results are discussed. Secondly, the PCA results for the Euclidean and DTW approaches are

discussed. Thirdly, the BOP results are discussed.

Time series and feature accuracy

Previous work has shown that using time domain signals yields a poor performance [52,78] and this poor performance was observed in the group majority results shown in Table 4.3. The Euclidean time series approach failed to identify the correct group majority for four groups and the DTW time series approach failed to detect the short seizure group. The Euclidean and DTW time series approaches failed to identify the correct group majority for the short seizure data as they both identified the short seizure data as background data. Since the short seizure data is a mixture of background and seizure data it would have been expected that the background data could have influenced the group majority. The feature approaches had perfect group recall for all of the event groups.

Performance accuracy provides a more informative insight into the event group performance and is presented in Table 4.4. In this table for example for the DTW TS test, if the query event is event 2 (short seizure) then the performance accuracy is 25%, meaning that when all the short seizure events were tested, only 25% of the recalled events were from the same event type. However, when testing event 3, it was able to correctly recall an event from the same event type 90% of the time. The value reported in the bottom row is the average over all the five groups that were tested.

The results in Table 4.4 are consistent with the results in Table 4.3. The time series Euclidean distance test performed the poorest in each table and has an accuracy of 30.25%. There was a surprising result for the DTW time series system as it achieved a recall accuracy of 71.11%. It was expected that the time series results would be poor [151].

There was a large increase in performance when going from the Euclidean time series representation to the feature representation. The accuracy increased by $\approx 50\%$ to 80.24%. There was a reduction in performance of 0.32% for the DTW system when going from the time series domain to the feature domain. It was expected that DTW would have performed better in the feature space as opposed to applying DTW directly to time series signals. A reason for the drop in performance is because the UCR suite normalises each of the signals being compared and important amplitude information contained in the template normalised features is lost.

Table 4.3: Group majority results for each system evaluated in this chapter

Event	Euclidean (TS)	DTW (TS)	Euclidean (55 F)	DTW (55 F)	Euclidean (19 PC)	DTW (54 PC)	BOP
2	1	1	2	2	1	2	2
3	1	3	3	3	3	3	3
4	1	4	4	4	4	4	4
5	5	5	5	5	5	5	5
6	1	6	6	6	6	6	6
Average (%)	20	80	100	100	80	100	100

Table 4.4: Performance accuracy for each system evaluated in this chapter

Event	Euclidean (TS)	DTW (TS)	Euclidean (55 F)	DTW (55 F)	Euclidean (19 PC)	DTW (54 PC)	BOP
2	33.33	25	50	75	41.67	75	58.33
3	20	90	90	70	95	85	90
4	30.43	47.83	73.91	82.61	78.26	86.96	52.17
5	63.64	92.73	87.27	76.36	89.09	87.27	94.54
6	3.85	100	100	50	100	57.69	96.15
Average (%)	30.25	71.11	80.24	70.79	80.80	78.38	78.24

PCA accuracy

A standard method of choosing how many dimensions or components to retain in PCA is decided by the amount of variance the user would like to maintain. With the small database size used in this work it was possible to evaluate the recall accuracy while focusing on one component, then two components and evaluating the recall accuracy in this way until a recall accuracy evaluation is carried out using all 55 components. The number of PCs that resulted in the best recall accuracy was chosen as the number of components to retain.

Figure 4.18 shows the recall accuracy of both the Euclidean and the DTW systems as the number of PCs is increased from 1 PC to all 55 PCs. The two individual markers indicate the recall accuracy when testing with the full 55 features without the application of PCA. The y-axis represents the accuracy of the recall system. The first thing that is clear in Figure 4.18 is that the Euclidean approach performs better than DTW approach when working in the feature domain either with or without using PCA. As stated previously, a contributing factor to this is that the UCR suite normalises the data before computing the warping distance. As the UCR suite is one

of the fastest DTW search algorithms, it was decided to proceed with the algorithm without modification. The short number of samples in the PCs of only 13 samples could have also attributed to the weaker recall accuracy as little warping would take place, thus nullifying the main advantage of DTW over the Euclidean distance.

After the feature data is transformed using PCA, it is expected that the components with low variance would not contribute much to the recall accuracy. This is seen for the Euclidean approach which achieved the best recall accuracy when using 19 PCs with the recall accuracy only slightly fluctuating from that point on. The highest accuracy for the DTW approach occurred when maintaining 54 components. The UCR suite normalisation step is a contributing factor as to why there is a low dimensionality reduction. The effect of the renormalised PCs is seen in Figure 4.18 as the recall accuracy continues to rise gradually.

For the full 55 features without PCA, the accuracy for the Euclidean and DTW approaches was 80.24% and 70.79% respectively as seen by the single markers in Figure 4.18. The highest accuracy occurs using 19 PCs for the Euclidean distance metric, and this was 80.8%. There was an increase in recall accuracy of approximately 0.56%. Therefore, the recall accuracy increase was negligible, however there was a substantial dimensionality reduction. The DTW approach had the best result when using 54 PCs which was 78.38%. The dimensionality was reduced from 55 to 54, this is small but the recall accuracy increased by approximately 7.59% from that of when using all 55 features.

It is clear that the application of PCA had a positive impact with only a slight increase in the Euclidean distance case. The Euclidean distance method had a significantly higher dimensionality reduction compared to the DTW method. It was expected that using a much lower number of components would yield better results in terms of accuracy due to the curse of dimensionality [176].

Since the UCR suite normalises the data, it only considers the shape and not the amplitude. Figure 4.19 demonstrates this as it shows the first PCs of the Euclidean and DTW methods while carrying out a query on an event. There is a correlation between the shapes in the match presented in Figure 4.19 (b). However, there is a significant difference between the original principal component and the match found in terms of amplitude and it is not a correct match. Figure 4.19 (a) is the Euclidean distance match, the shapes are not as similar, but it is a correct match.

As regards to the group majority results in Table 4.3, when using PCA and 19 components for the Euclidean distance approach the short seizure data had an incor-

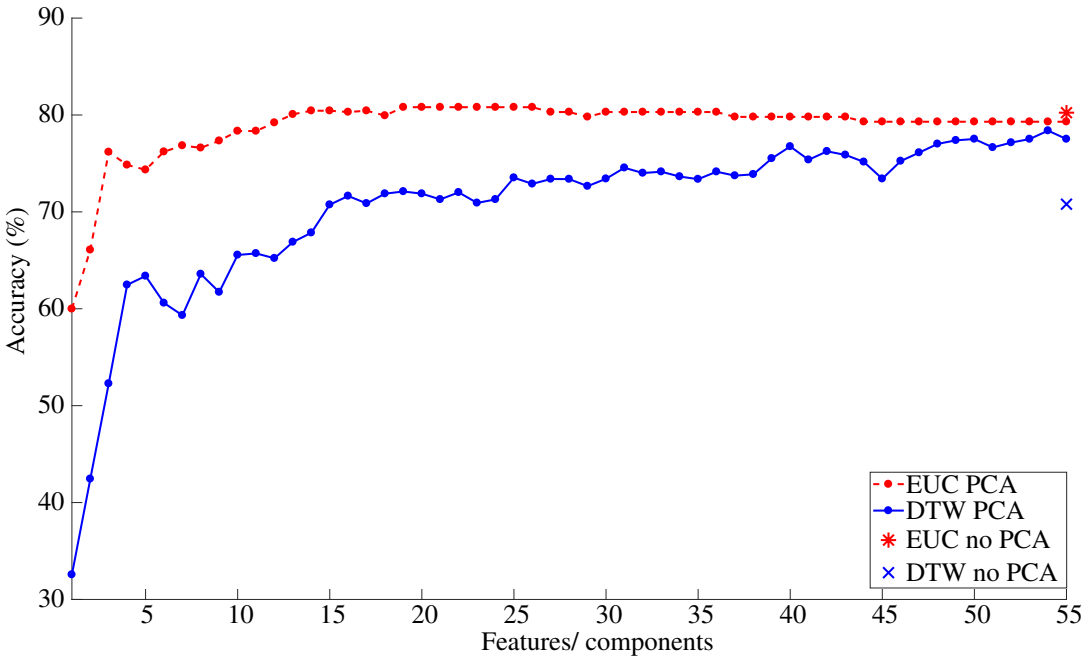


Figure 4.18: Performance accuracy as the number of PCs used is increased and when the full 55 features are used

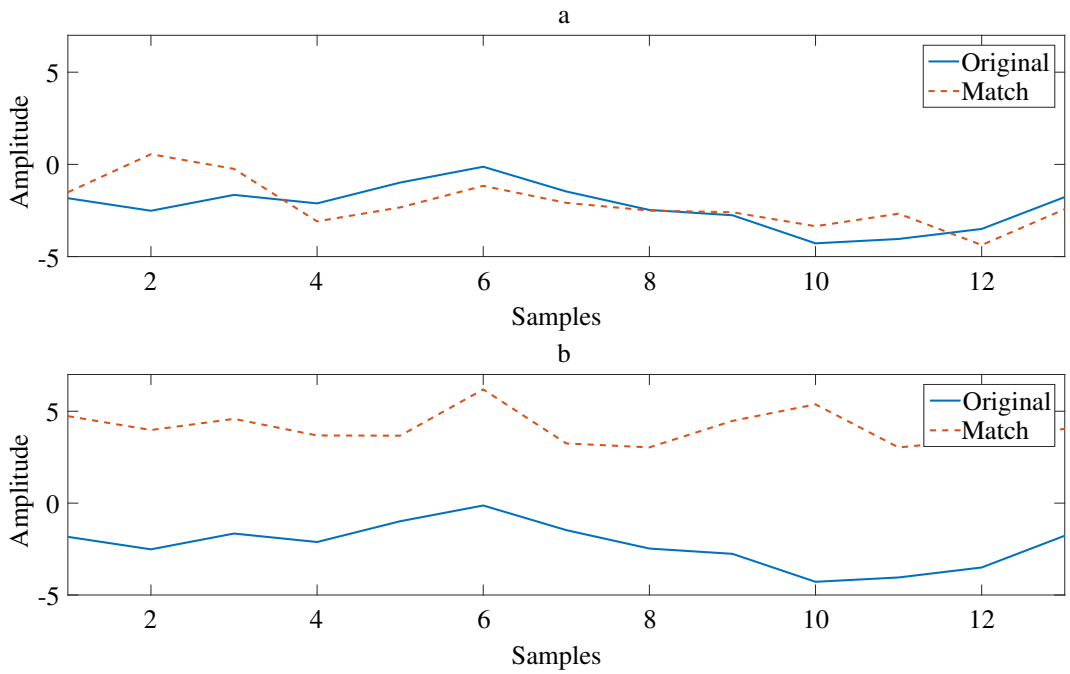


Figure 4.19: Matches obtained when testing both the (a) Euclidean distance (correct match) and (b) DTW distance (incorrect match) on the same query

rect group majority. The background event type was chosen as the group majority. The group majority for the DTW approach with 54 components resulted in perfect group majority results.

In Table 4.4 it is seen that the Euclidean PCA approach has the highest overall performance. One surprising result is the fact that there is only a 0.56% increase in performance from using 55 features to using just 19 PCs. It was expected that a higher performance would have been attained when switching from using the full 55 features to 19 PCs [176]. The PCA DTW approach had an accuracy increases of approximately 7.6% to 78.38% when using 54 PCs.

BOP accuracy

In Tables 4.3 and 4.4 the last results columns are for the BOP approach. Perfect group majority results were achieved and the average recall accuracy was 78.24%, which is 2.56% below the best performing Euclidean distance PCA approach. The BOP ranks third in performance with the Euclidean and DTW PCA distance tests performing better.

4.6.2 Query run time

The performance accuracy of a system is essential but, if the time taken to compute the query is too long the answer may become irrelevant. For this reason, the speed of the search algorithms was investigated. The speed being referred to is the time taken to compute a query. To get an accurate representation of the speed, each system has the time measured to compute 25 queries which was then repeated 25 times to get an average. All the query run time tests carried out in this thesis were run on a windows 7 machine that had a 3.6 GHz Intel i7 processor and 12 GB of RAM.

As the database used for testing is small, speed tests were carried out while increasing the database size by 1 event each time, starting with an initial two events. This test will give an idea of how the speed of the system changes as the database grows and enables an evaluation of how the system will scale. Figure 4.20 shows these results. The Euclidean time series test and BOP test have very similar performance speeds, with the BOP being slightly slower due to the BOP transformation process. The Euclidean distance tests with features or PCA are very similar in performance and have nearly flat slopes. These slope values were expected as the point to point operations are fast to compute. These lines are highly overlapping in Figure

Table 4.5: Slopes of the query timing plots depicted in Figure 4.20

Method	Euclidean (TS)	DTW (TS)	Euclidean (55 F)	DTW (55 F)	Euclidean (19 PC)	DTW (54 PC)	BOP
Slope s/event	5.66E-04	7.74E-02	4.25E-03	1.40E-02	3.73E-03	1.40E-02	6.02E-04
Time DB = 10000 (s)	5.67	774.00	109.44	214.52	104.40	215.93	6.84

4.20.

Interestingly enough the DTW tests with features or PCA start with a similar time to the Euclidean approach; however as the database increases in size so too does the time required by the DTW approach. This difference in time was expected as the DTW algorithm is more computationally intense than the Euclidean distance computation. This same trend of the DTW speed decreasing as the database in increased in size is echoed in the DTW time series test. The time taken for the DTW time series test increases rapidly and if the size of the database were either doubled or tripled the time to compute 25 queries for the DTW time series approach would have surpassed the DTW features approach.

The feature and PCA tests appear to have an initial offset. This offset is caused by the feature extraction process and will always be present. This offset is a once off computational cost that does not impact the systems as the database grows in size as these features have already been extracted for the database before the queries being performed.

Table 4.5 shows the slopes for the results depicted in Figure 4.20. These provide a more concise form of the results. Table 4.5 shows how the systems will scale with database size. As expected the time series DTW approach has the largest slope as it is performing the DTW operation on the longest signals. Following this is the DTW computation on the features and PCs. The slope for the DTW features and PCA approach is very similar due to the poor dimensionality reduction of 1 dimension and the added cost of the application of the PCA transformation. These similar slopes are evident in Figure 4.20. The next best performing slope is the features Euclidean distance test followed by the PCA Euclidean distance test. The features test is slightly slower as the PCA Euclidean approach has a dimensionality reduction of 36 dimensions. The fastest approaches are the Euclidean distance applied directly to the time series and the BOP test. The time series Euclidean distance approach is only marginally faster than the BOP approach. It is worth noting that this system has

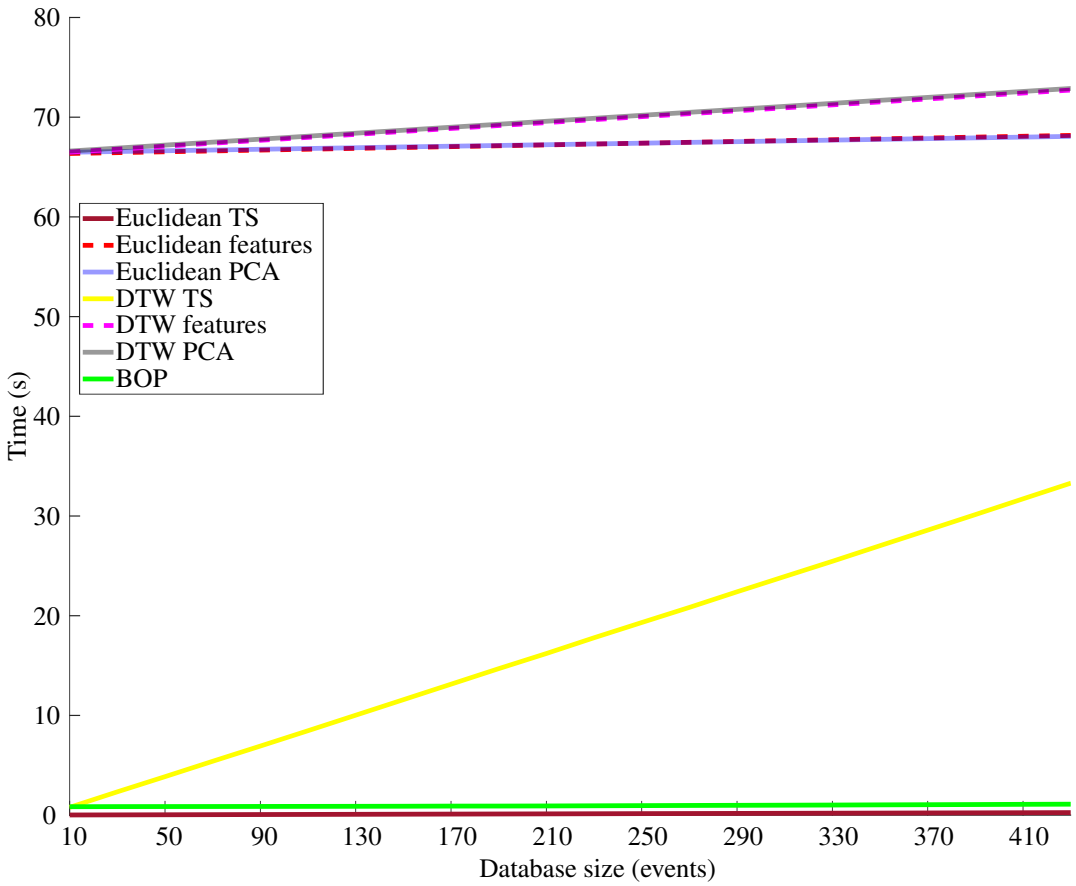


Figure 4.20: Database timing graphs for increasing database size for each system evaluated in this chapter

only 430 events and therefore the slopes will help examine how each method scales.

In Table 4.5 under the slopes, there is the projected time to compute 25 queries on a database of 10000 events. This projected time is based on the assumption that the lines in Figure 4.20 are linear. The time series DTW approach is the worst with it taking 774 seconds or nearly 13 minutes per query. The Euclidean PCA approach which has the best performance accuracy takes 104.4 seconds to compute the queries whereas the BOP approach takes 6.84 seconds. This BOP approach is approximately 15 times faster than the Euclidean PCA approach.

Note, the BSF early abandoning technique described for the DTW approach in Section 4.4.2 could be applied to the Euclidean approach to potentially achieve a slight query run time improvement.

4.6.3 Memory requirements

Another parameter that was investigated was the memory required by each system to store the database files. The metric used was bytes. The memory value was computed by saving the event database as a MATLAB *.mat* format file. The size of the file was then read into the computer and stored.

This method of getting the database size was repeated for a database of size one and incremented by one until the full database size of 430 events and was carried out for each of the systems. The results are plotted in Figure 4.21. Since the time series Euclidean and DTW approaches use the same data only one set of memory values was computed and plotted in Figure 4.21, this is noted as “Brute force TS”. The Euclidean and DTW feature-based approaches use the same features, so they also have a single set of values called “Brute force 55 features”.

The results show that the time series method required the most substantial amount of memory and the memory requirements increase at the fastest rate. The DTW PCA and brute force feature approaches have very similar memory requirements with the DTW PCA method requiring marginally less memory, as the DTW PCA method only requires 54 components as compared to 55 features.

The Euclidean PCA approach saves 19 components and thus requires much less memory to store each event. The BOP approach requires N x α^{ω} memory where N is the number of events in the database, and α^{ω} is the vocabulary size. In this case, the size is $Nx2187$. This size is in the region of the size of the time series signals ($Nx1920$), but the memory is much lower. The reason for this is that the BOP

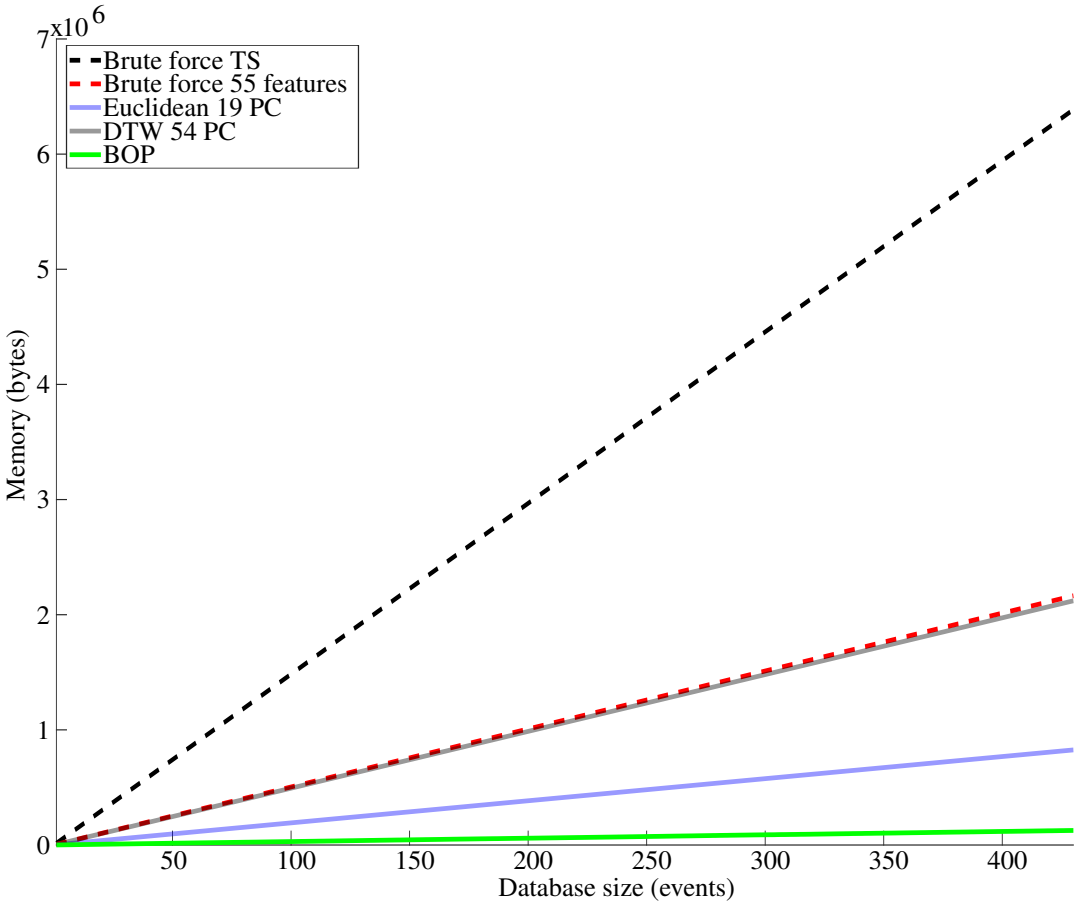


Figure 4.21: Database memory graphs for increasing database size for each system evaluated in this chapter

Table 4.6: Slopes of the database memory plots depicted in Figure 4.21

Method	Brute force (TS)	Brute force (55 F)	Euclidean (19 PC)	DTW (54 PC)	BOP
Slope bytes/event	14850.81	5033.55	1919.92	4931.68	293.29
Size DB = 10000 (MB)	141.63	48.00	18.31	47.03	2.80

approach works with integers and each of the other systems work with 64 bit floats. Working with integers enables a significant reduction in the memory needed to save the database.

The slopes of the lines in Figure 4.21 are presented in Table 4.6. With the slopes, it is possible to get an idea of the scalability of the algorithms. The slopes show that the BOP approach is the best performing system in terms of memory usage. Using the direct time series data requires the most memory.

The projected memory requirement for a database of 10000 events, is provided in Table 4.6, based on the assumption that the lines in Figure 4.21 are linear. This projection shows that storing the original time series events would require the most memory. This result was expected as the other techniques are focused on the compression of the data. As there was not a substantial dimensionality reduction for the DTW feature space the size of the DTW features and DTW PCA are similar and in the region of 48 MB. The Euclidean PCA uses only 19 components and would require approximately 18 MB for storage. The BOP approach requires the least amount of storage at a projected value of approximately 3 MB. The Euclidean PCA approach requires approximately 7 times the amount of storage than the BOP approach.

4.7 Summary

In this chapter, fixed point and elastic brute force NN neonatal EEG recall systems were developed and evaluated. The two distance metric approaches were extended to the feature domain which showed a significant performance increase over the time series domain. The goal of this work was for NN neonatal EEG recall, although an event classifier had to be used as a proxy as too much time and money would be required to get an experienced neurophysiologist to evaluate the nearest neighbours

from each query.

This chapter focused on the performance accuracy of each system but also looked at other essential aspects of the systems such as query running time and memory requirements. The query running time and memory are essential and need to be considered if it is decided to scale systems to work with large databases.

The primary findings of this work are that the Euclidean approach applied to the principal components gives the best results while not requiring much memory with only a little overhead in time for a moderately sized database. The state of the art UCR suite that was used in the DTW experiments worked better than expected on the time series tests. The DTW tests did not outperform the Euclidean features and PCA tests as it was designed for shape similarity and disregards scale. Longer feature vectors may have increased the DTW systems recall accuracy. DTW was the slowest algorithm tested and would not be suitable while working with larger databases when a brute force and event ranking test is being carried out. The BOP approach came third in terms of accuracy by a narrow margin, but it is both the fastest and least dependant on memory. For these reasons, it would be recommended as a strong contender for use when working with large databases.

Brute force tests may well be unrealistic for large databases. They do provide a solid base as to what needs to be reached in terms of accuracy, speed and memory for non-brute force NN searches.

Chapter 5

EEG self identification using hashing

5.1 Introduction

In Chapter 4 the concept of brute force methods for nearest neighbour (NN) recall based on distance metrics was introduced. These approaches were evaluated in terms of performance accuracy, query time and memory requirements. The latter two elements need to be considered when an extensive database exists. This chapter introduces, describes and evaluates NN neonatal EEG recall methods that are not based on brute force approaches.

The approach used to combat the growing query time and memory requirements of brute force systems, while maintaining recall accuracy, was the use of hashing for the task of NN neonatal EEG recall. The chapter is laid out as follows; Section 5.2 introduces the concept of hashing for NN recall systems, Section 5.3 provides an introduction into the area, Section 5.4 discusses the spectrogram based hashing approach, Section 5.5 presents the feature quantisation based approach, Section 5.6 discusses the locality sensitive hashing (LSH) method and data transformation approaches tested. The performance of each system is discussed in Section 5.7.

5.2 What is hashing?

In brute force methods, the concept is to compare the query event with every other event in the database. In the evaluation of brute force systems, the performance, query time and memory required were examined. A problem exists when there is a database that grows in size, where the query time and memory requirements then become a concern. If the database is large, it may become unrealistic to store each

event in memory, or it could take too long to carry out a query. This chapter aims to address these potential problems as brute force applications are not always practical when there is a large database [177].

To overcome the limitations of brute force search techniques, the application of hashing approaches to neonatal EEG were investigated. In general, hashing is the process of converting an input, such as an EEG event into a fixed size representation known as the hash value. This conversion process is achieved using a set of mathematical operations which are known as a hash function. This basic idea is represented in Figure 5.1 (a) where the "EEG event 1" is the input block, the "Hash function" block represents the set of mathematical operations and the "h1" block represents the generated hash value. This hash function can be seen as $h = H(M)$ where h is the generated fixed length hash value, H is the hash function and M is the input data that is to be hashed. Hash functions are used to map data to fixed-size values.

When the term hashing is mentioned, the initial thought is of some form of secure hashing technique such as SHA-192 [178]. Secure hashing schemes have a property called strong collision resistance, which means that it is computationally infeasible for two different sets of data to generate the same hash values. However, this property is not required in the use of hashing in this thesis. The goal of the thesis is to find the "nearest" match and not the "exact" match. There will not be an exact match to a real-time query as the EEG varies from person to person even if they exhibit the same EEG event type, thus making exact matching impossible. To find the nearest match, it is necessary to lower the collision resistance. By doing this, it enables similar hashes to be generated for similar data such as the less secure or non-exact song identification hashing algorithm originally used by Shazam [179]. The authors found that using this non-exact hashing scheme enabled the correct identification of data even after the query suffered from distortion.

As discussed in Chapter 4 applying brute force techniques directly to longer time series results can result in poorer NN accuracy results versus when applied to a compressed representation such as that of a feature representation [151]. Chapter 4 introduced the idea of compressing the EEG data into both feature representation and the bag of patterns (BOP) representation. Both approaches resulted in increased performance accuracy for the NN problem. This concept is investigated here for the hashing space. Due to the stochastic nature of EEG, applying a hashing technique directly to the EEG has the potential to return poor results when performing a NN

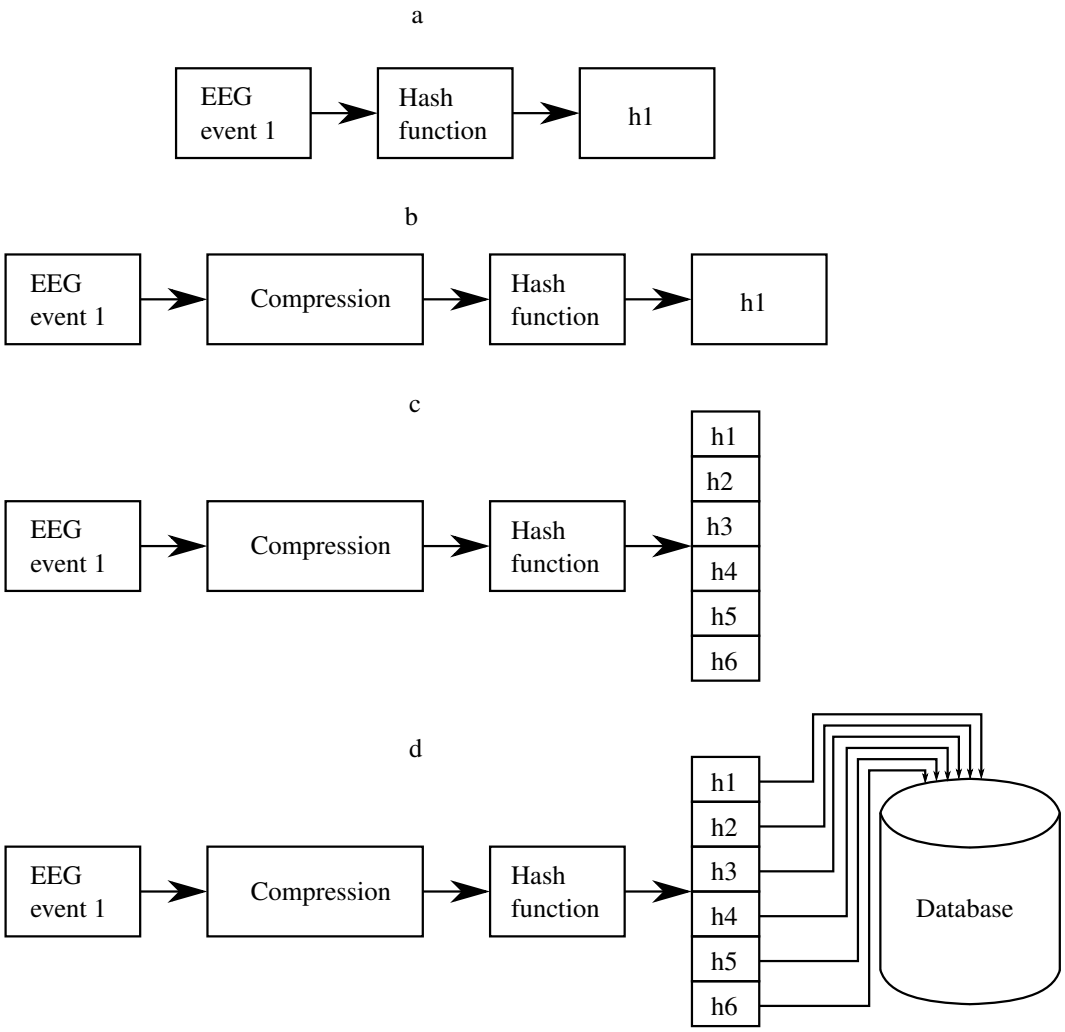


Figure 5.1: (a) Basic hashing idea where the raw EEG data is passed directly through a hash function and a single hash value is returned. (b) Hashing system is expanded with a compression step to reduce the amount of data being hashed into a single value. (c) The hashing system now generates several hash values from the same event. (d) The final stage in the basic hashing system is the storage of the hashes, this is done via the database.

search. In Figure 5.1 (b), the idea of compression is introduced into the hashing framework. The type of compression used is dependant on the application and user requirements. If features are to be used to compress the data, it is beneficial that the features used are generated with domain knowledge. Arbitrary features not relating to the underlying characteristics of the signal have a higher probability of returning poor matches. There exist several data compression techniques not based on feature extraction in the literature with many examples listed in [131] such as singular value decomposition (SVD), discrete cosine transformation, piecewise aggregate approximation (PAA) and symbolic aggregate approximation (SAX) to name a few.

In Figure 5.1 (a) and (b) there is a single hash value generated for the entire event. This approach works for a single secure hash. As the work in this chapter is interested in nearest neighbours and not exact matches, a less secure non-exact hashing scheme is implemented. If using a non-exact hashing scheme and only a single hash to represent each database event, there will be a large number of matches to a query event and no way to assess the strength of the match. This problem is addressed in Figure 5.1 (c) where many simpler hashes are generated from the event. For this particular example, there are six hashes. The hashing system can now be more robust for the NN search problem by using multiple non-exact hashes.

Before NN EEG recall using hashing can be discussed the method in which the hashes are stored needs to be described. Each hash that is generated from an EEG event is stored in a database with their event identifier which makes it easy to distinguish which hash was identified. These form a pair known as a key-value pair and these are used to store events in a hash table [180]. For example purposes, assume the hash generated is an integer number. This integer number is the key to the hash table. The event identifier is then stored in the location indexed by that key. If multiple different events have the same hash number, the event identifier is appended to the row in the matrix indexed by the hash number. Figure 5.1 (d) demonstrates the idea of storing hashes from each event in a database. Each of the generated hashes from the event are added to the database individually. The same process is then repeated for all the events that are to be added to the database. Figure 5.2 demonstrates how a key-value pair is added to a hash table. In this example, the hash key or number generated is 2, and it is already in the database. There are three event identifiers associated with this hash number. The new event identifier will then be appended to the three existing event identifiers.

The idea of using hashing for the process of NN recall is straightforward. First

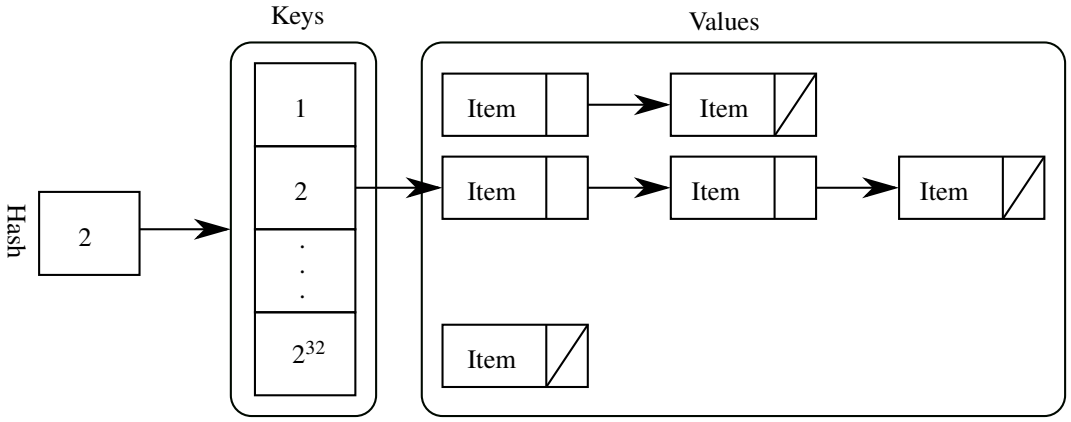


Figure 5.2: Example showing how a key-value pair is added to a hash table. In this example the key is 2 and it already has values associated with it. The new value is appended to the existing list of values.

there is a database development stage. Next, when there is a query, the hash is not added to the database, the database matrix row corresponding to the hash number generated is checked. If there are event identifiers at that hash indexed row, they are returned as potential matches. This idea is portrayed in Figure 5.3 (a). In Figure 5.3 (a), the hash function generated a single hash $h1$. When this hash searched the database, it found that the EEG event 7 has generated the same hash. The EEG event 7 was the only other event to have the same corresponding hash, meaning that EEG event 7 is the NN to the query event.

The purpose of this basic single hash query stage was to familiarise the reader with the query process. Next, the idea of multiple hashes per event is introduced. This idea is represented in Figure 5.3 (b), where there are six hashes generated for the query event. Each of these hashes is individually searched through the databases. Database events that have the same hash number as the query are added to the list of potential matches. If a database event has multiple hashes in common with the query event, the number of common hashes is stored. In the example in Figure 5.3 (b) the EEG event 4 had five hashes in common with the query hashes out of a total possible number of six hashes. Whereas events 1 and 15 only had a single hash in common with the query event.

The final stage of the query process is to return the NN to the query event. A straight forward approach is to count the number of matching hashes to the query and choose the event with the highest number of hashes in common. In the example in Figure 5.3 (b), this would return EEG event 4 as the NN. The number of hashes in common with the query event can thus be treated as a confidence measure. The

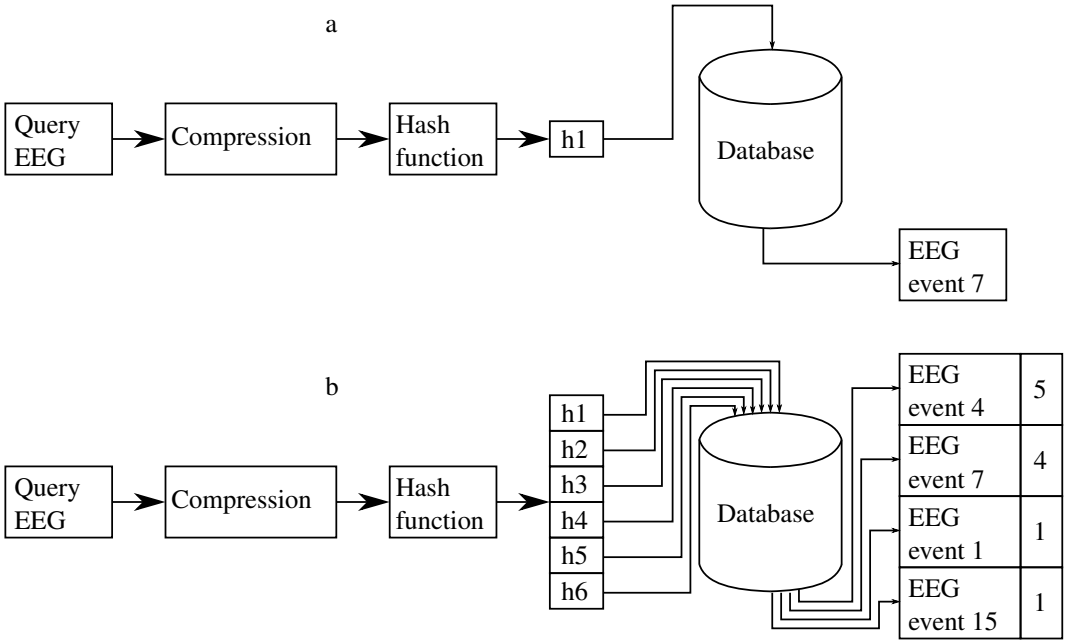


Figure 5.3: (a) Basic hashing based query idea. (b) Hashing when events create multiple hashes.

more hashes in common, the more confident the NN match.

There are examples where counting the number of matching hashes may not return the actual NN match. An example of this is when the hash function used has little variability in the hash values generated. When cases like this occur, other techniques of choosing the correct NN need to implemented. This chapter covers cases where further processing steps are needed to generate more likely NN matches.

This chapter details multiple hashing approaches at a hashing system level. Inside each system, every stage of the hashing process will be discussed in the following order;

1. Compression technique
2. Hashing method
3. Database development
4. Search algorithm

5.3 Review of hashing systems

Hashing techniques have been used in many different domains such as audio [179, 181, 182], text [183–186], physiological signals [50, 158, 187–190] and image [191–193] to name a few.

A commonality between each method is that they check the similarity between the query hash or hashes and the hashes in the database. The uniqueness of the hashes depends on the application. For example, hashing for biometric authentication [188] systems needs to be secure and so may use secure hashing algorithm such as SHA-192 [178] in order to create highly unique hashes, so there is high confidence in having a correct match when a similar hash is found. Due to the variability in neonatal EEG from patient to patient, this approach would not be suited for use as a NN neonatal recall system as the hashes are too exact. An example that uses non-exact hashes is [191], which uses binary hashes to map similar images. As this approach uses non-exact hashes, the system can find similar hashes that do not come from the same source. This style of hashing is suited to the problem of NN EEG recall.

In the majority of hashing systems, the primary purpose is based around similarity. For example, the proposed hashing in [179] is to find the most similar song within an extensive database of songs. In [186], the primary purpose of the algorithm is to hash sentences so they can be used as an efficient plagiarism checker. In [188], they extract EEG features and then generate hash keys to be used as a biometric-based cryptographic key.

In this chapter, popular hashing approaches from the audio domain and text similarity domain are focused on, as these represent time series data which is similar to EEG. There is a large amount of work published in the areas of audio and text similarity. Biometric authentication approaches [187, 188] were not investigated as it was determined that the generated hashes are too unique due to the nature of the application.

5.3.1 Audio hashing

Audio hashing is used to create a compact representation of audio recordings [181]. Audio hashing has been used for areas such as speech and speaker identification [194–196] as well as the more commonly known area of song identification [179, 182, 197, 198].

The goal of speaker identification is to identify a speaker from a recording. In the speaker identification method described in [194], the authors investigated the use of hashing to identify telephone spam calls. The methods of hashing used stems from the hashing concept which was initially developed for audio identification [197]. The authors created a binary hash from the extracted Mel Frequency Cepstral Coefficients (MFCC) which are used commonly in speaker recognition [199]. The MFCCs are features that are used in the audio domain to represent the frequency content of the signal as humans perceive it. These MFCC features represent the compression step described in Section 5.2 as these features now represent the raw data. The speaker identification method described in [195] was used as a speech authentication algorithm. The hashing technique implemented is based on linear prediction analysis as it is known to be effective for speech signal analysis [195]. In [196], a speaker identification system for use on a large scale is presented. This methods couples a known speaker identification approach with a method of hashing known as LSH that is used for approximate NN searches with the aim of improving the search speed, memory and processing requirements of traditional NN search algorithms [200]. LSH preserves the relationships that exist in the data through the hashing scheme, which makes it more scalable than true NN search algorithms. It uses multiple random projections or random hash functions from a family of hash functions to generate hashes from the data. These hashes are then stored in a database for later recall. When creating hashes from the data, the idea is that similar sets of data have similar hashes so in essence $h(q) = h(x)$ if similar and $h(q) \neq h(x)$ if not similar, where $h(x)$ stands for the hash generated from event x [201]. There is a possibility that different event types may have the same hash and so, multiple hashes from the hash family are computed. The authors showed that when using their framework, they were able to achieve faster speaker identification with the sacrifice of a small identification accuracy. This paper has shown that using an LSH scheme that uses random projections to hash time series signals can yield fast and accurate results.

In song identification, hashing has been used to identify a song name which had uses in music information retrieval (MIR) and digital rights management (DRM) [202]. MIR systems typically extract and store hashes for each song in an extensive database ranging from thousands to millions of songs. When a song has to be identified from a snippet of audio, the MIR system generates hashes for the snippet of audio and matches this against the database to return the best match, according to

specific metrics. The aim is to reduce the size of the original song while retaining robustness in the face of potential signal distortions [203]. The diversity of song styles, beats and tempos, for example, make the creation of a unique hash for identification a difficult task. The compression methods used for hash generation are numerous, including, for example, zero crossing rate, energy, loudness, or perceptually significant features [204]. The choice and number of features have an impact on the success of the system.

The most robust hashing technique described in [182] computes the factorisation and summation technique of SVD on the MFCC features. The MFCC is the compression step, as seen in Section 5.2 and SVD is the hash function as it is a summarised version of the MFCC features. A hashing technique which is based on the quantisation of energy bands from the audio source was used in [197]. This technique is robust as the hash generation via quantisation can be used for many different types of features, provided there are no dominant features. This method works well for audio identification, but there are concerns about the number of hashes generated due to the large amount of overlap between the small sub-sequences of data. Two problems can arise if there are many hashes. The first problem is that the memory required can approach that of loading the whole event in memory. The second problem is that if the hashes generated are not very unique, there could be too much overlap between the hashes in the database. This overlap could cause problems when recalling events as the query event could have many matches to the events in the database. However, this quantisation idea has been implemented in other audio identification techniques [205] and speaker identification [194] systems. The approach has shown overall that quantisation can be used for the hash generation following a time-series compression step.

The well-known song identification system Shazam uses a method of hashing based on the spectrogram of the audio data [179]. The generated hashes are made from a combination of time and frequency components from the spectrogram. An audio hashing method that is similar to the Shazam spectrogram hashing approach was presented in [198], which used spectrograms calculated with constant Q transforms rather than the standard spectrogram. This type of transform is more suited to audio signals as the constant Q transform will have a constant number of bins per note. The hashes made from this spectrogram are then made in a similar manner to [179]. As spectrograms can represent time-series data, this opens up the idea of using spectrograms as a means to generate hashes for time series signals from

different domains. The authors of [206] proposed the idea of using a spectrogram to represent the audio. This spectrogram is then compressed by taking the top 50 to 800 wavelet coefficients. The compression step is needed to summarise the most important sections of the spectrogram. These wavelets are then compressed down further into sequences using min-hashing [207]. An example using binary data is that the data is initially randomly permutated, the first location of a 1 is the min-hash value. This process can then be repeated several times to generate a signature for the event consisting of several min-hash values. For the system in [206], this min-hash signature is the output of the compression block described in Section 5.2. The LSH hashing technique is then used to store and find similar matches efficiently. Overall this paper showed that hashes do not need to be generated directly from the spectrogram data. The use of LSH on min-hash signatures allows for approximate matches as opposed to exact matches. This approximate match is of interest as it accounts for noise that can be present in the compressed representation of the signal.

The audio signals have a more structured signal when compared to EEG signals. Applications of audio hashing for song identification include broadcast monitoring, audience measurement, finding song name, duplicate detection and metadata collection [181, 208]. The work presented in this chapter stems from three different audio hashing techniques. These techniques were chosen due to their success in the audio domain and their adaptability for use on neonatal EEG data. These approaches used different techniques to generate hashes from a compressed data representation.

The first approach is based on the popular Shazam song identification algorithm, which creates hashes from spectrograms of the songs [179]. The second approach came from Philips, and it is also a song identification algorithm that creates binary hashes using feature quantisation [197, 209]. The third approach is based around the idea of applying LSH to a compressed representation of the time-series data. All of these methods follow the steps described in Section 5.2. They each compress the data into a different representation, this is followed by hashing and storing events in a database and lastly, each method has a query scheme developed for each application. Unlike the previous two hashing techniques, LSH has previously been used with physiological signals. LSH was applied to mean arterial blood pressure signals to retrieve similar signals and the results showed that a good retrieval accuracy was achieved [50]. Likewise LSH was used as an approximate-NN system for ECG data and GAIT (the pattern of how a person walks) data for subsequence matching [210]. LSH was evaluated with adult EEG in [51], which looked at the correlation between

EEG patterns and alcoholism; a problem with this dataset is that it only has two EEG classes (alcoholic and non-alcoholic). The methods used and implemented are discussed in more detail in sections 5.4, 5.5 and 5.6 respectively.

5.3.2 Text similarity

The concept of text similarity is to measure the similarity between pieces of text, be it words, sentences, paragraphs or full documents [211]. As more digital data is created, there is a need to efficiently arrange and file the data for fast recall and similarity. There are a wide variety of distance measures available for computing text similarity [211]. In text similarity, the goal is to find similar sections of text or similar documents. The text or document is typically compressed, and then a distance measure can be computed. This approach is similar to the idea of brute force NN searching as was discussed in Chapter 4. An example compression technique is to create n-grams, which are sub-sequences from either characters or words [212]. The similarity is measured by counting the number of similar n-grams divided by the total number of n-grams [211]. Alternative similarity searches on text can be based on the semantic similarity of the content [213]. Semantic similarity is for example when there are text files that use different words but have the same overall meaning [214]. When text or documents are compressed into a binary hash representation, the Jaccard distance can be used to compare similarity [215]. This is a measure of the similarity between two sets and it provides a measure to compare compressed time-series signals. The problem with this similarity measure is that to find the most similar events, the Jaccard distance needs to be computed for the whole database.

There are numerous examples in the literature where hashing is used for faster text similarity. Comparing text or documents directly with other documents will lead to poor similarity results due to variations such as words used and the sentence structure. This is the same with EEG as seen in Chapter 4, when compression techniques are applied, the NN results improved. The idea is that compressed data is more robust to the noise and distortions present in the raw data. The method of compression and search techniques will impact the results obtained. The remainder of this section looks at common compression and search strategies that are used in the area of text and document similarity and assesses their applicability to neonatal EEG for NN recall.

In [216], a privacy-preserving hashing system for document similarity compar-

isons is presented. This paper expanded an approach called simhash. Simhash can be seen as a dimensionality reduction technique as it compresses the document into a fixed size bit-string using document terms and frequencies [214, 217]. The authors [216] expanded the simhash idea and used it as a secure way to check for similar documents in different databases. The scalability for a large database is unknown from this paper although the compression into a fixed sized bit string could be adapted and potentially used in an EEG compression scheme. SimseerX [218] was developed to find similar documents from large databases. It has a web interface where the user uploads a document and the system then uses CiteSeerExtractor [219] to extract text and metadata from a document corpus and returns similar documents. The web interface gives users four different similarity measures. Two measures that stand out are the simhash and key phrase search. These perform the quickest search in practice through the website and the author presented results of how the key phrase approach scales. The key phrase search extracts high-level descriptions of the documents after being trained on a document database. The simhash approach generates a $64-$bit binary code to represent documents, and the Hamming distance is then used to find the most similar documents. This paper showed that using compression systems and brute force searches can return results fast for large databases. The negative aspect is that this approach is still brute force as it requires every binary code to be compared. Different compression techniques will yield different performances in terms of speed, accuracy and data compression.

A novel document summarisation algorithm [183] was developed which used a compression strategy comprising of n-gram extraction and min-hashing. An n-gram is a sub-string of a sentence and multiple n-grams would be extracted from the sentence. LSH is then applied to the min-hash string to find similar sentences; it could be computationally expensive to find all the matching min-hash signatures. The LSH algorithm would hash min-hash signatures into the same bin if they are similar, reducing the need for a brute force search. When they have the sentence similarity, a ranking system was used before summarising the results. This paper has an interesting use of LSH here in order to avoid a brute force search. If LSH were applied to neonatal EEG, different compression techniques would need to be investigated. Text similarity hashing is also used in document similarity. A system for sub-linear time NN search in a very large collection of documents was presented in [185]. Initially, the documents are represented by tokens. Following this, the LSH approach is used to reduce the search pool. Once the search pool is reduced, there

is still a chance that the LSH matches have low precision and therefore, a method called iterative quantisation is used to re-rank the search pool.

An alternate approach for finding similar documents, which stems from the idea of using semantic hashing to design binary codes to represent the documents was presented in [220]. Here, hashing both the documents and terms simultaneously according to their semantic similarity was proposed. This approach offers an intelligent document compression scheme although the similar document retrieval stage may still require brute force searching on the extracted hashes from the query document.

A semi-supervised hashing method for use on huge databases that does not rely on random projections like that of an LSH hashing scheme was proposed in [193]. This system uses hash functions that maximise the separation between classes on the training dataset. The training dataset is a collection of data that is labelled to belong to a specific class and it is used to train the system to have the best separation between the classes. The hashes aim to map the data into a Hamming space using weighted projections. The weights on the projections are optimised to ensure documents from the same classes have similar hashes. This hashing approach provides an interesting solution to the similarity problem. As the system is semi-supervised, it does require labelled documents to achieve the best results. If new event classes were to be added, the training phase would need to be rerun to find the best class separation. Adding new event classes would lead to problems as there would need to a sufficient amount of data for the training phase. This particular approach was evaluated on an image database, although the underlying ideas can be applied in different domains.

When searching the literature and as seen from the examples above the main hashing types are simhash, LSH, semantic hashing and semi-supervised hashing. For the particular application to neonatal EEG, the concept of LSH is most appealing as it is an approximate search algorithm as opposed to brute force. LSH can be directly applied to neonatal EEG and also to different compressed representations of the data. The simhash approach has shown promising results in the text domain although it is a brute force method and neonatal EEG is stochastic and less structured; compressing it down into a Hamming representation has the potential to lose variable information. If the data is compressed too much, the similarity search results would be trivial and if there is not enough compression, then it becomes no better than a brute force search problem as was described in Chapter 4. As simhash is a data compression step, this technique or other binary compression techniques could be used to compress the neonatal EEG data before the application of LSH. Semantic hashing cannot be

applied to EEG and semi-supervised hashing could possibly provide more accurate results, but training data is needed and requires a brute force search to be carried out. For these reasons, the LSH approach will be investigated for use as a NN neonatal EEG recall system. The application of LSH to neonatal EEG for event recall will be discussed in Section 5.6.

5.3.3 Review summary

There has been much research in the audio and text domains that focused on similarity using hash-based techniques. These hash-based similarity techniques were researched as they yield faster results than carrying out a query directly on the data.

A main observation from the literature is that the hashing techniques apply compression steps before hashes are generated. The method in which the compression is carried out can be varied depending on the hashing approach chosen. In both the audio and text domains, compression techniques specific to each particular domain was carried out, such as MFCCs [221] for audio and n-gram [212] generation for text. The methods of compression for neonatal EEG will need to be investigated when implementing the different hashing schemes.

The hashing techniques that are explored further in this chapter were chosen as they are not brute force methods and they have achieved good performance in their respected areas. These approaches are all adaptable for use within the neonatal EEG domain. The first approach is a spectrogram hashing technique [179]. The second approach uses quantisation to generate hashes from an extracted feature [197]. The third approach investigated is the application of LSH to neonatal EEG to generate approximate matches [201]. It is worth noting here that no previous work has been found in the literature that applies any of these three hashing techniques to neonatal EEG. The work in this chapter will show how the techniques were adapted to work with neonatal EEG and will also show the strengths and weaknesses of the three hashing systems.

5.4 Spectrogram hashing of neonatal EEG

This method was initially developed as a commercial product for the identification of songs which was used by the music identification company Shazam which was founded in 1999. A general overview of the system was published in 2003 [179].

Due to its success in the audio domain for NN, it was decided to investigate the performance of the spectrogram hashing system when used with neonatal EEG.

As the audio and neonatal EEG domains are very different and as the paper [179] provides a high-level overview of the algorithm, several steps needed adapting for EEG and optimisation of the parameters to be selected was required. This section of the chapter will present in detail the steps involved in the algorithm. Following this, the optimisation technique will be presented.

The spectrogram hashing algorithm is a multi-stage system that uses both the time and frequency content of the signal to represent it and store it in a database. This method has three stages;

1. Hash creation
2. Database development
3. Search algorithm

5.4.1 Data compression and hash creation

The EEG signals are preprocessed using the preprocessing steps discussed in Chapter 3. Once the signal is ready, the hashes can be generated. The following is a list of the steps involved:

1. Spectrogram generation
2. Spectrogram cleaning
3. Peak location
4. Hash generation

Steps $1-3$ resemble the compression box from Figure 5.1.

Spectrogram generation

Spectrograms show the time-frequency relationship of signals using the Short-Time Fourier Transform (STFT). The signal was initially split into overlapping epochs, with a Hamming window applied to reduce edge effects. The overlap, epoch length and the number of points in the FFT will affect the resolution of the spectrogram. Figure 5.5 shows an example spectrogram representation for the 8 second EEG

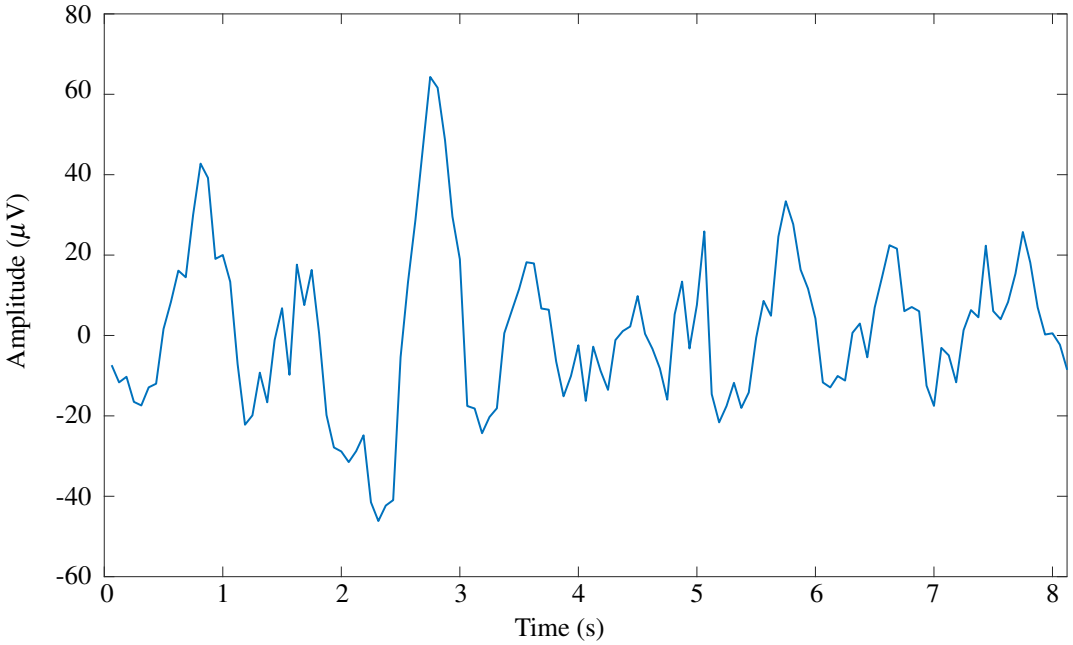


Figure 5.4: An example 8 second window of an EEG time series signal at sampling rate 32 Hz

time series signal shown in Figure 5.4, where the sampling frequency is $32Hz$. The colour intensity indicates the energy at the corresponding time and frequency energy point. As seen in Figure 5.5, the lower frequency region of the spectrogram contains the highest concentrations of energy. The spectrogram in Figure 5.5 focuses on the frequency range $0Hz - 8Hz$. The energy peaks with higher intensities contribute to the structure of the EEG event.

Hashes are generated from the energy peaks contained within the spectrogram. At this point in the algorithm, there exists an extensive collection of spectrogram energy peaks, and thus every point in the spectrogram could potentially be used to make hashes. Using every spectrogram energy point to make hashes leads to no data compression and long query times.

To achieve data compression, some of the energy peaks need to be removed from the spectrogram. Many approaches could be implemented to remove energy peaks. However, it is important to keep the energy peaks that contain information about the EEG event and remove the energy peaks that contain little information.

Spectrogram cleaning

In the original paper [179] the authors do not describe the methods they used for peak location following the computation of the spectrogram. They state the following “A

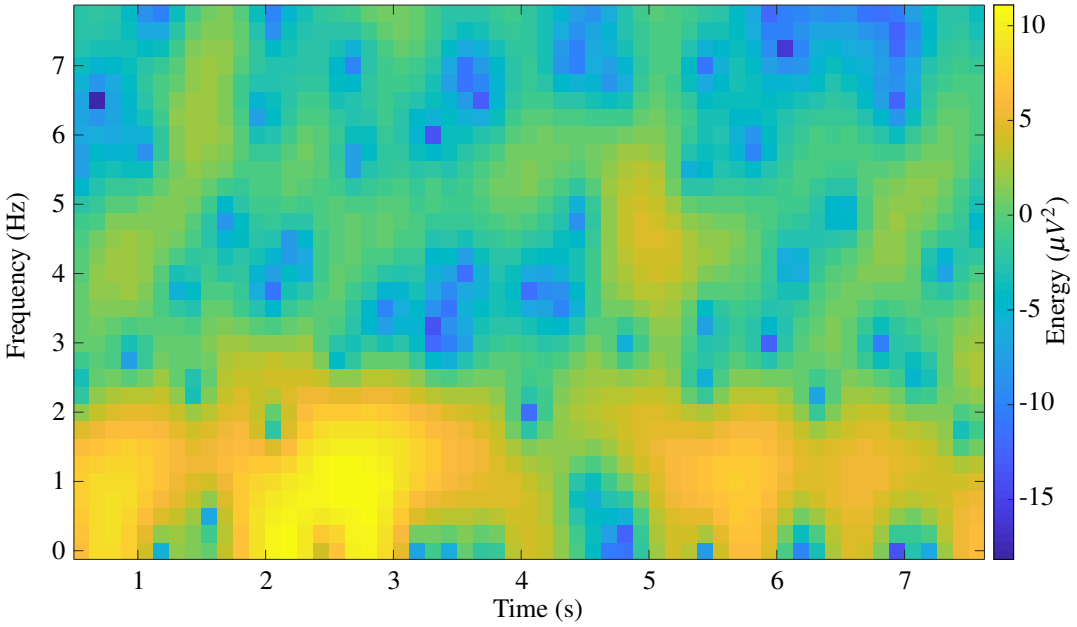


Figure 5.5: The spectrogram of the 8 second EEG time series signal seen in Figure 5.4

time-frequency point is a candidate peak if it has a higher energy content than all its neighbors in a region centered around the point".

The spectrogram is 2-D, it can be seen as an image. For this reason, image processing techniques were implemented to reduce the number of energy peaks from the spectrogram. A log transform is applied to reflect relative change and thus prevent very large energy components from dominating and skewing the hashing process. The spectrogram is then normalised between zero and one using Equation 5.1.

$$\hat{s}(t,f) = \frac{s(t,f) - min(s)}{max(s) - min(s)}. \tag{5.1}$$

Where s is the log transformed spectrogram. The lower energy components of the spectrogram are then removed as they contain little information. Removing the lower energy components saves resources rather than storing and processing useless information. Energy peaks with an energy of zero are not considered as peaks, and this is why the spectrogram was normalised between zero and one as it ensures the lowest possible energy peak is at zero. The following energy peak removal step was

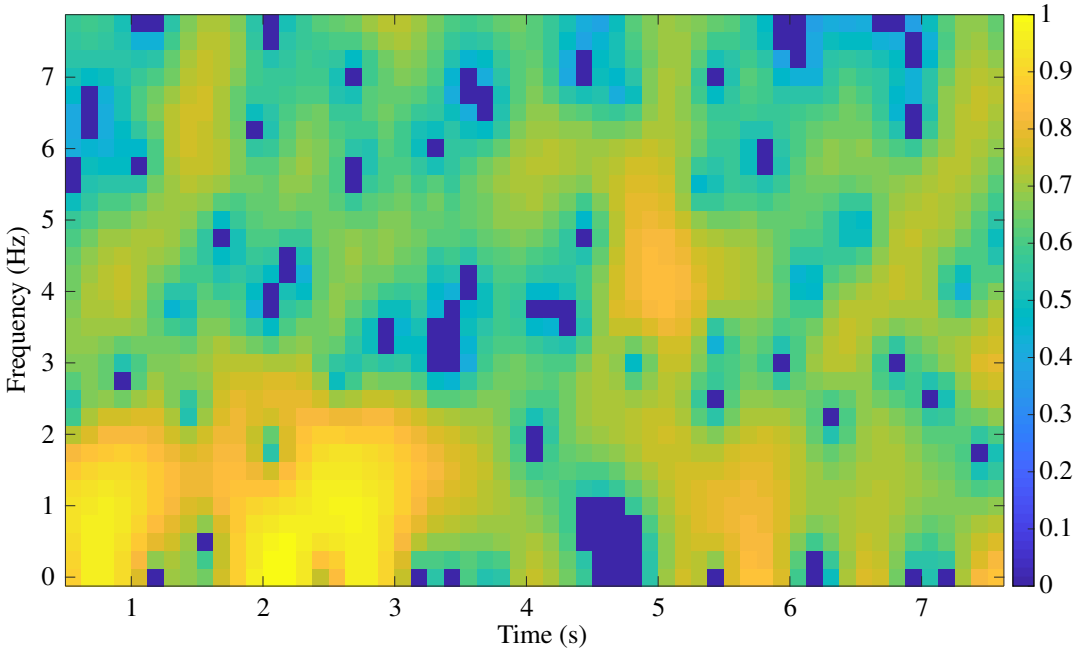


Figure 5.6: Spectrogram after application of a peak removal threshold of 71.4%

applied:

$$\hat{s}(t,f) = \begin{cases} \hat{s}(t,f) \times 1, & (\hat{s}(t,f) - ((\frac{1}{T}\sum_{t=1}^{T}\hat{s}(t,f)) \times Threshold)) \geq 0 \\ \hat{s}(t,f) \times 0, & (\hat{s}(t,f) - ((\frac{1}{T}\sum_{t=1}^{T}\hat{s}(t,f)) \times Threshold)) < 0 \end{cases} \tag{5.2}$$

where T is the number of spectrogram time windows and $\hat{s}(t,f)$ is the energy peak at time t and frequency f. This equation has taken into account the fact that the lower frequency components of neonatal EEG have higher amplitudes than the higher frequency components. This is done by taking the mean of the frequency band corresponding to $\hat{s}(t,f)$ as opposed to the mean of the whole spectrogram. Taking the mean of the whole spectrogram could remove some potentially important higher frequency energy peaks.

The dark sections in Figure 5.6, which have an intensity of zero, show the resulting graph after the application of the spectrogram threshold. The lowest intensity pixels are showing (dark blue) as the blocks that did not satisfy Equation 5.2 when using a threshold of 71.4%. This threshold parameter and all the other parameters for the system were found using an optimisation strategy that will be discussed in Section 5.4.4. The parameter values used for this system are also detailed in Section 5.4.4.

Figure 5.7 shows an example of the system after a threshold of 100%. The

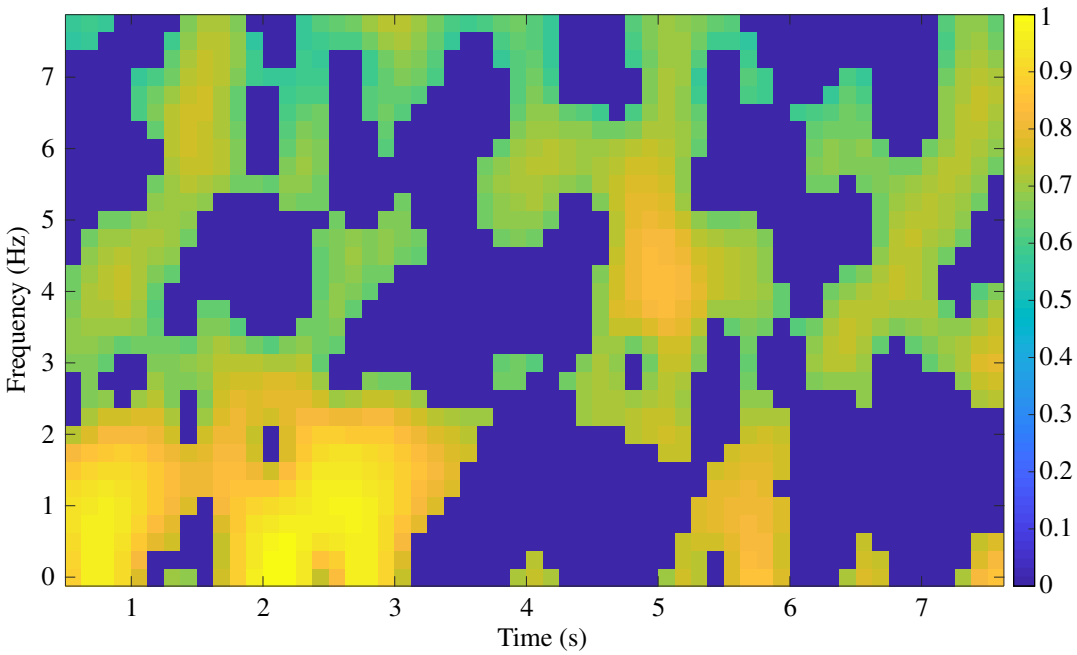


Figure 5.7: Spectrogram after application of a peak removal threshold of 100%

number of energy peaks has drastically reduced. This example shows that as the threshold is increased, more data is discarded. Compressing the data is important for speed and storage requirements and varying this threshold is an example of how the data compression can be controlled.

Peak location

The next step in the hash generation is finding energy peaks that have the highest significance. To do this, the image processing technique of dilation was applied [222]. Dilation is a morphological operation and it was applied across the spectrogram. A morphological operation is a technique used to alter images or in this case spectrograms. The dilation operation applies a quadrilateral structuring element to each pixel in the spectrogram image and sets the pixel value to the maximum value contained in the structuring element. This approach places more emphasis on the higher energy peaks in the spectrogram. Figure 5.8 shows an example of the dilation process. The original image is a cross, and the structuring element is a rectangle with the anchor pixel being the rightmost pixel in this example. As the structuring element is moved across the image, if it comes into contact with at least 1 pixel of the same colour, the anchor pixel will be activated. When this was applied over the original image, the result can be seen in the dilated image in Figure 5.8. The green

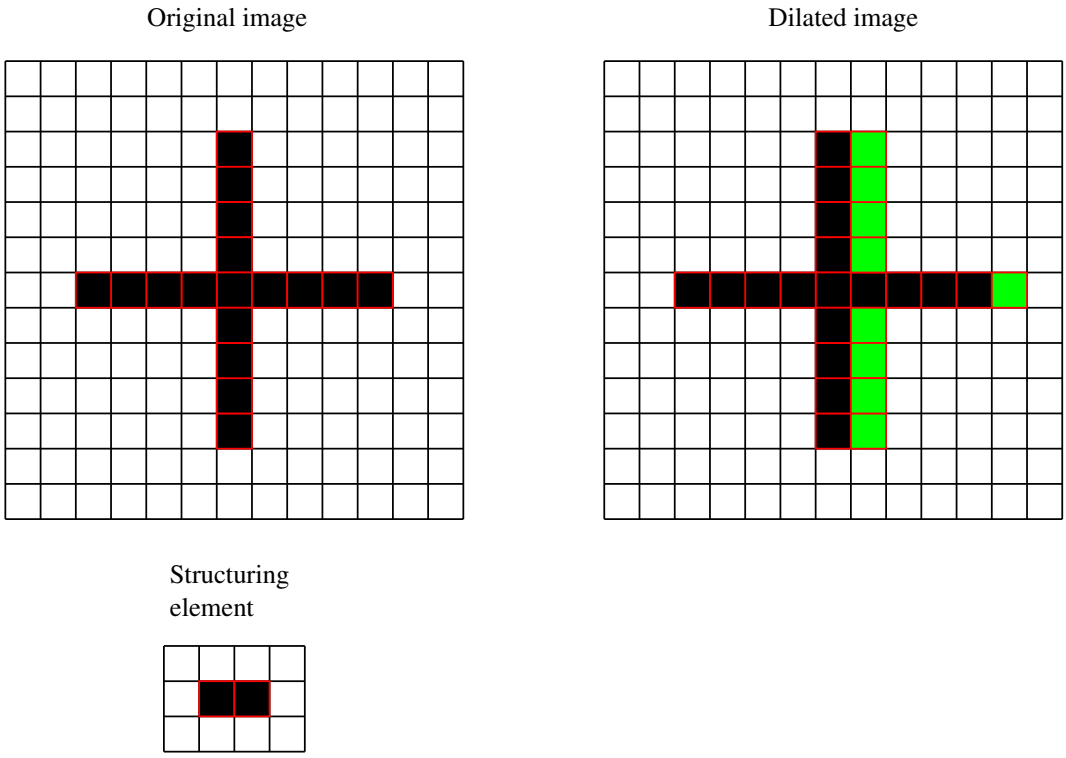


Figure 5.8: Binary dilation example showing the original image, structuring element used and the resulting dilated image

pixels are the new pixels that were added.

The dilation process is applied to the spectrogram with Figure 5.9 showing the resulting image. The higher amplitude regions in the spectrograms are now more dominant.

This new representation is then compared to the spectrogram before dilation was applied. Matching energy peaks between the original and dilated spectrograms are seen as the most significant energy peaks with the other peaks being discarded. This process reduces the spectrogram data to a collection of energy points. The size of the dilation window can be varied independently for both the time and frequency components of the spectrogram. The black dots in Figure 5.10 are now the energy peaks that will be used for hash generation.

Hash generation

Each point contains a time, frequency and energy component. Using the time and frequency components of each individual energy peak would make the matching process difficult and may require a vast database. A significant problem here is that

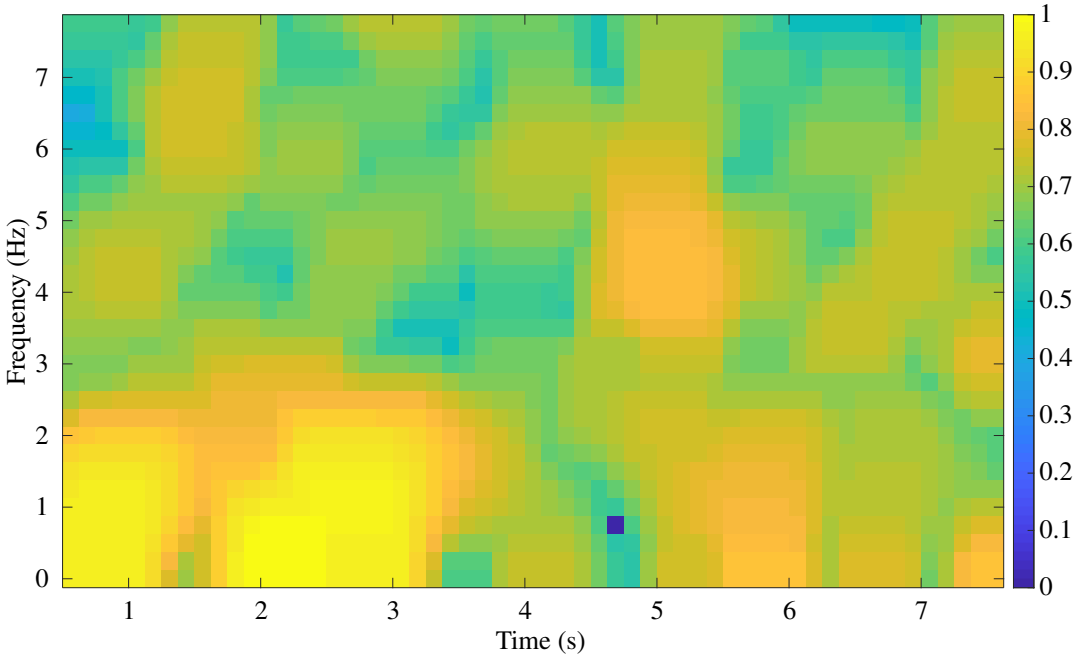


Figure 5.9: Spectrogram after the dilation was applied showing the dominating higher amplitude components

exact time alignment is required, which is not feasible. In [179], the authors solved this problem using combinatorial hashing which combines information from two energy peaks to make a hash.

The system loops through each of the energy peaks and attempts to find point pairs. For example, on iteration i, energy peak i known as $point(i)$ needs at least one other energy peak to make a hash. The energy peak it needs, has to be contained within a specific region of the spectrogram as opposed to anywhere in the spectrogram. By doing this, it makes the hashes define the particular area they reside in and ensure they are not spread across the whole spectrogram or drawn to particular energy peaks throughout the event. This region is known as the target area. The energy peaks (f_p) contained within the target area must meet the following conditions in order to be used to create a hash, with the energy point, $point(i)$ considered to be located at (t_0, f_0);

1. The energy peak has to be bounded within t_{0+1} and t_{o+tlim}.
2. The energy peak has to be bounded within either $f_{0+1} \leq f_p \leq f_{o+flim}$ or $f_{0-1} \geq f_p \geq f_{o-flim}$
3. The energy peaks with the highest energy are chosen.

The parameters $tlim$ and $flim$ are tunable and application specific, and these represent the target area. They should be tuned so on average $point(i)$ has a set number of points contained within the target area. Using a target area also reduces the possibility of the database exploding in size. The target area may contain many other energy peaks and to limit the number of hashes generated, $point(i)$ was limited to generate a set number of hashes which is discussed in Section 5.7.1.

When there are more than the set number of energy peaks within the target area, a method was needed to choose which energy peaks to use. The most straightforward approach is to choose the closest energy peaks. In this work however an alternate approach that utilised the energy component of the energy peaks was used, which picked the energy peaks that had the largest energy components. This puts extra emphasis on the higher amplitude energy peaks as they are seen to contain more information. The reason $point(i)$ and another energy peak cannot be located on either the same time or frequency values is to prevent an influx of similar points when there may be noise or other disturbances in the time series. If a section of the time series was corrupted or if there were a constant frequency interference in the system, the hashes would be attracted to them.

Each point pair is used to make a hash using Equation 5.3

$$Hash = (Fanc - 1) \times 2^9 + \Delta F \times 2^4 + \Delta T \tag{5.3}$$

with $Fanc$ being the anchor point frequency, ΔF is the frequency difference between the points which can be positive or negative and ΔT is the positive time difference. This hash equation is used to create the hashes for the events. The idea of using this type of hash came from [179]. By applying Equation 5.3 the relative time between the point pair is reserved alleviating the need for the query to be initiated at the same time as when the initial time series event was added to the database. The landmarks are seen in Figure 5.10 which are the red lines between the peak points. These landmarks form the hash.

The spacing of the hash in Equation 5.3 was designed to prevent overlap from different combinations of $Fanc$, ΔF and ΔT. The values displayed in Equation 5.3 are application specific as ΔT and ΔF may vary for other applications. The time and frequency resolution of the spectrogram, along with the size of the target area when generating the point pairs were taken into consideration when designing the hash.

Figure 5.11 shows the final product of the full spectrogram hashing algorithm

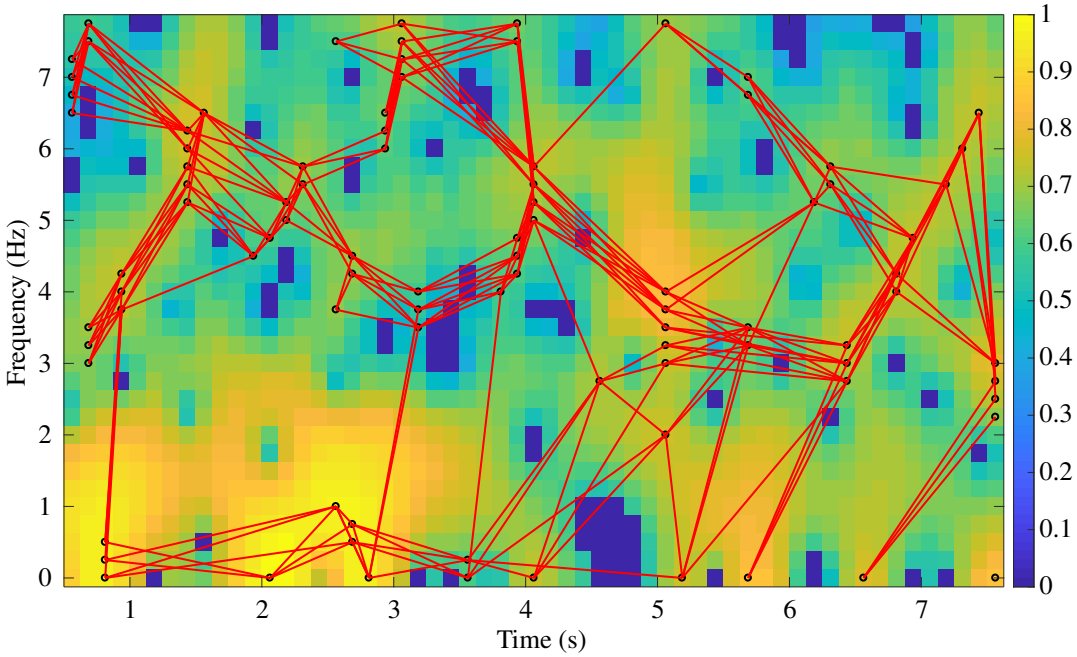


Figure 5.10: Spectrogram with the energy peak landmarks overlaid. The energy peaks are the black circles and the red lines connecting the two energy peaks show the peak pairs that were used to create the hashes.

that was applied to a chirp signal ranging between $0Hz - 16Hz$, that is repeated three times. This spectrogram is 60 seconds in duration and the frequency ranges from $0Hz - 16Hz$. The landmark hashes are overlaid. The three higher intensity diagonal lines correspond to the three cascaded chirp signals. What is clear from Figure 5.11 is that the hashes or landmarks are drawn towards the higher energy sections of the spectrogram.

5.4.2 Database development

The hash value generated using Equation 5.3 is used as a key for the hash table [180]. This key, along with the time series event identifier, coupled with $point(i)$ start time, are stored in a hash table as the value for quick recall in the query stage. The hash table is an ideal storage type for systems where memory is limited. Section 5.2 and Figure 5.2 described the process of adding hashes to a hash table.

The process described above in Section 5.4.1 is run iteratively over all the time series events that are to be stored in the database. This extraction process is the most time-consuming step in the system, albeit it has only to be carried out once for the database development. New events can easily be added to the database without the need to recreate a whole new database. Once the hash table is generated, and the

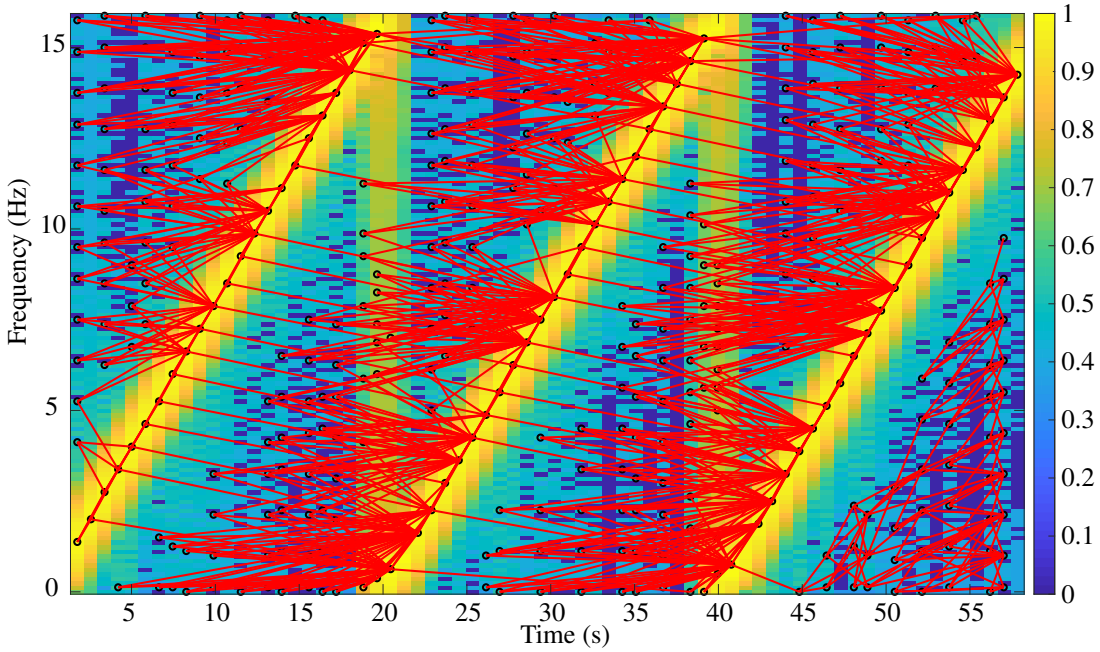


Figure 5.11: Spectrogram with landmarks overlaid for a longer signal consisting of three cascaded chirp signals from $0Hz$ to $16Hz$

events are indexed, querying is then possible.

It is worth noting that with this method, there is a likelihood that the same hash can occur multiple times for multiple different events. When they occur, they are merely appended to the value list in the hash table for that particular hash key.

5.4.3 Search algorithm

The original purpose of the Shazam algorithm described in [179] was to recall the closest matching song from a database of songs, it was, therefore, a NN of one search algorithm. This work has a similar aim with the exception that it is being carried out in the neonatal EEG domain. Starting with an unprocessed query time series, it undergoes the same processing that has occurred for the time series events that were used to build the database. A group of hashes are then generated in the same manner as described in Section 5.4.1. This collection of hashes now represents the query time series event. Each hash in the query collection is compared against the database; if the hash key exists in the database, the values associated with it are returned as potential matches. This hash query is repeated for all the hashes, and a list of candidate matches is generated.

The hashes are initially sorted by the candidate event. The times from the query hashes are compared to the times embedded in the values retrieved from the hash

table. Then the candidate event with the most frequently occurring time differences between the query and stored times is considered as the most likely match. This simple and effective method is quickly able to return the first nearest neighbour. Using this method to sort the events eliminates potential matches that do not have an associated time correlation. These potential matches can be frequent and occur by chance, as the original hashing (Equation 5.3) only takes into account the time difference and not the occurrence time.

A flow graph of when a query is being carried out is given in Figure 5.12. When there is a query, this process is carried out with the same parameters as were used when developing the database. The hash generation block can be divided into four stages described in Section 5.4.1. The lookup block is responsible for checking if the query hashes exist in the database. The best-connected matches block arranges the returned matches in order of most likely to least likely. The most likely match is then returned as the final match and this is the NN event.

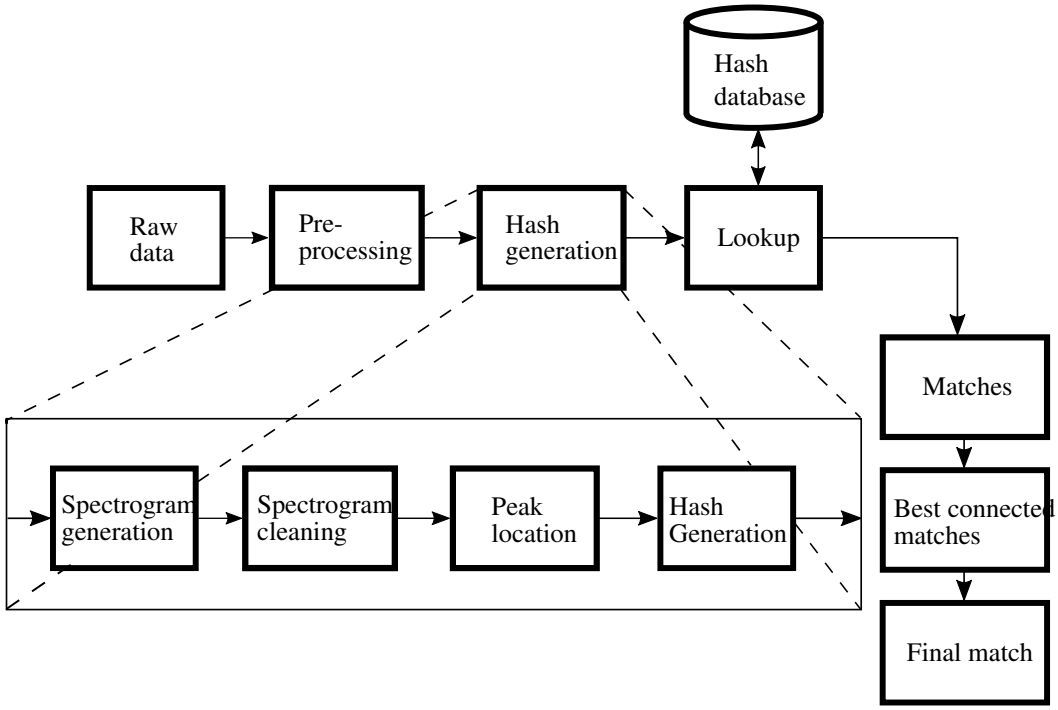


Figure 5.12: Spectrogram based method query flow graph showing the steps involved in the hashing process

5.4.4 Optimisation

The method developed in [179] has several tunable parameters which are optimised for the audio application. In this section an optimisation technique is proposed to optimally tune these parameters for use on neonatal EEG.

A brute force parameter optimisation technique is not a feasible option, although a coarse search may provide insight into the optimisation space. Another option is to optimise individual parameters by fixing the other parameters with the advantage that multiple tests could be run in parallel. This method may lead each parameter being stuck in a local minimum, mainly when there are interdependencies among the parameters. Different search approaches, like the Monte-Carlo sampling approach [223] could be used. There are two circumstances where the Monte-Carlo method produces good results; the first by coincidence and the second is by taking a large number of samples. As the Monte-Carlo method relies on chance or a large number of samples, there is little chance that the optimum parameters found are the global optimum parameters.

A metaheuristic optimisation algorithm which locates the optimal or near optimal parameters was investigated [224]. Metaheuristic algorithms are gradient free problem independent optimisation techniques and so are suited to this general parameter optimisation problem. There are various types of metaheuristic methods such as simulated annealing [225], evolutionary algorithms [226] and swarm intelligence methods [227].

In this work, a form of swarm intelligence called particle swarm optimisation (PSO) [228] was used to locate an optimal or near optimal solution. PSO was chosen as it is a multi-objective optimisation technique that has a lower chance of getting stuck in a local optimal parameter position due to the use of multiple particles searching the parameter space. This method is based on the flocking patterns of birds, where the particles or "birds" would work through the parameter space or cost function while in search of the optimal parameter set to minimise the function cost. Each particle has a personal memory of their best position P_{b_p} and a social memory component which is the global best parameter set within the whole swarm G_b. These memory components and the previous particles' velocities guide the particle to its next position. Each particle is initially assigned random values.

An inertia weight (I_w) [229] or a constriction factor (K) [230] is used to help prevent the explosion and encourage the convergence of particles. These are two

well-known methods for controlling the velocity of particles. If the velocity of the particles is not controlled, an optimal solution where the particles converge may not be found; the particles could keep exploring in different directions on the parameter plane.

The PSO particle velocity (v_p) is given by Equation 5.4

$$v_p(i) = K \times [I_w \times v_p(i-1) + \phi_p \times r_p \times (P_{b_p} - \aleph_p(i-1)) + \phi_g \times r_g \times (G_b - \aleph_p(i-1))] \tag{5.4}$$

for particle p on iteration i and it combines both inertia weights and constriction. The parameters ϕ_p and ϕ_g are the personal and social acceleration constants respectively. These constants will determine whether to perform a more personal search by having each particle put further emphasis on their own best P_{b_p} as opposed to the global best G_b which would not lead to a convergence of the particles. Alternatively, more emphasis can be placed on the social aspect of the global best. This social aspect leads to fast convergence, albeit to a potential local optimum. In this work $\phi_p = \phi_g$ as this is typically done [231, 232] to balance the exploration and convergence. The variables r_p and r_g are random numbers with a uniform distribution between [0, 1] which are used to introduce randomness into the system. The particle position ($\aleph_p$) is given by Equation 5.5.

$$\aleph_p(i) = \aleph_p(i-1) + v_p(i) \tag{5.5}$$

The constriction factor K is calculated using Equation 5.6,

$$K = \frac{2}{\left|2 - \phi - \sqrt{\phi^2 - 4\phi}\right|}, \text{where } \phi = \phi_p + \phi_g, \phi > 4 \tag{5.6}$$

and the inertia weigh I_w is user-defined. The inertia weight or constriction approach can be disabled by fixing the inertia weight $I_w = 1$ for all iterations, similarly fixing constriction factor $K = 1$ will disengage this damping. Algorithm 1 shows the pseudo code for PSO with D dimensional particles. It is worth noting that the D dimensional particles are initially set to random positions within a specified range as can be seen in the pseudo code in Algorithm 1 from lines $1-8$. The cost function that is to

be minimised is given by $f(\aleph_i)$. As seen in Algorithm 1 after each iteration the particles personal best (P_{b_p}) (line 16) and global best (G_b) (line 19) are updated if the cost function produces $f(\aleph_i)$ has a new minimum value.

Figure 5.13 shows an example of the PSO algorithm using 10 particles for 15 iterations on the parameter plane shown. The dark regions in Figure 5.13 indicate a low cost and as the colour moves towards a yellow the cost increases. The x-axis and y-axis correspond to particular parameters that are being optimised. In Figure 5.13 (a), it is clear that the particles are spread out around the parameter plane. The personal, global and net acceleration from each particle is shown via the arrows protruding from each particle. The arrows are pointing in the direction where the acceleration is taking them and the size of the arrow represents the level of acceleration. In 5.13 (b), it is clear that the majority of particles converge to the global minimum. Due to the constriction, the acceleration has decreased and this is evident from the shorter arrows. The final particle is trying to reach the global best as seen from the large personal and global arrows. Due to constriction, it cannot make it as the net acceleration is vastly reduced. For more detail on PSO, please refer to [228].

The optimised parameter set for the spectrogram hashing algorithm found using PSO are presented in Section 5.7.1.

5.5 Feature hashing

The algorithm chosen for this work was initially presented in [197]. It has shown promising results for the audio domain and was one of the audio hashing techniques discussed in the audio hashing review in Section 5.3.1. It is a hash-based system and therefore it is expected to be faster than the brute force systems presented in Chapter 4. The overall algorithm consists of four main steps;

1. Data compression
2. Hash generation
3. Database development
4. Search algorithm

These four steps are the same as mentioned in Section 5.2.

Algorithm 1 Particle Swarm Optimization

for all i in 1..Ω **do** // Initialize population position and velocity
  for all j in 1..D **do**
    $\aleph_{i,j} \sim U(\aleph_{j_{lo}}, \aleph_{j_{hi}})$;
    $v_{i,j} \sim U(-|\aleph_{j_{hi}} - \aleph_{j_{lo}}|, |\aleph_{j_{hi}} - \aleph_{j_{lo}}|)$;
    $P_{b_{i,j}} = \aleph_{i,j}$;
  end for
  $G_b = \arg\min f(P_{b_i})$;
end for
for all ii in 1..T **do** // Number of Iterations
  for all i in 1..Ω **do** // Loop through of particles
    for all j in 1..D **do** // Loop through each dimension
      $r_p \sim U(0,1)$; $r_g \sim U(0,1)$;
      $v_{i,j} = K[I_w v_{i,j} + \phi_p r_p(P_{b_{i,j}} - \aleph_{i,j}) + \phi_g r_g(G_{b_j} - \aleph_{i,j})]$;
      $\aleph_{i,j} = x_{i,j} + x_{i,j}$';
    end for
    if $f(\aleph_i) < f(P_{b_i})$ **then**
      $P_{b_i} \leftarrow \aleph_i$
    end if
    if $f(P_{b_i}) < f(G_b)$ **then**
      $G_b \leftarrow P_{b_i}$
    end if
  end for
  $I_w = I_w * w_{damp}$ // Inertia weight damping
end for

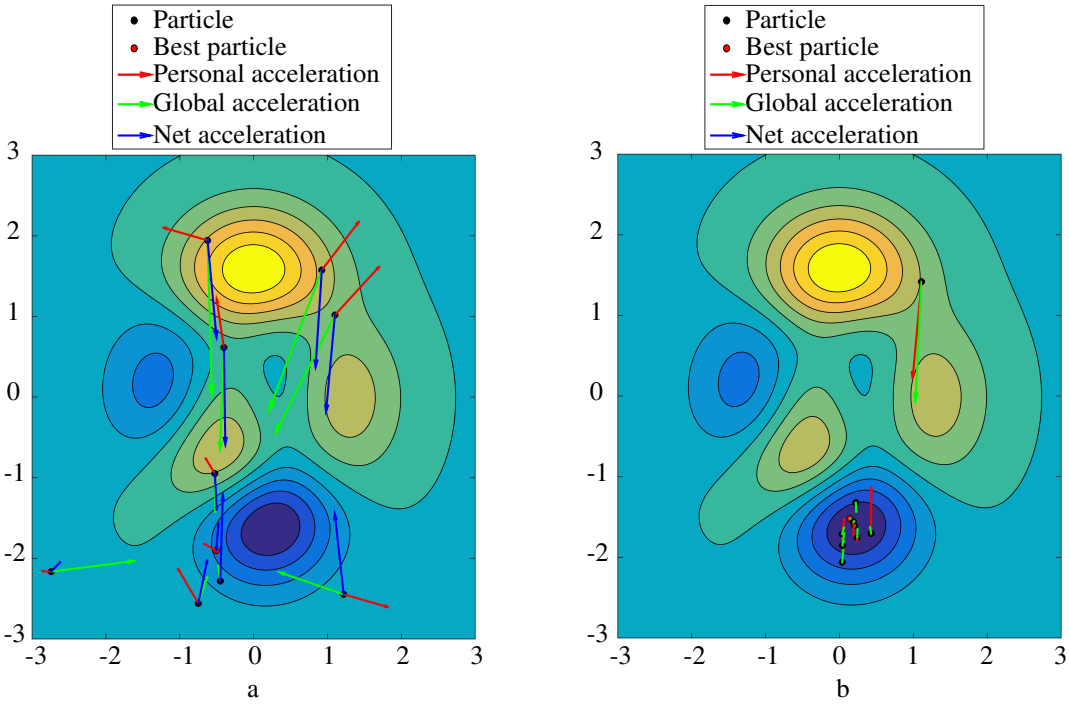


Figure 5.13: PSO on a two parameter mathematical function using constriction with 10 particles and 15 iterations. (a) showing after the first iteration. (b) Showing after the 15^{th} iteration.

5.5.1 Data compression and hash creation

This section investigates two different hash techniques. The first technique is based on quantising the signal using the energy in different frequency bands [197]. The second approach used the same principal of quantising a feature but extended this from one feature to a full set of features that were designed for use on neonatal EEG [33].

Frequency based hash

Hash extraction is used to make the events identifiable within large databases. The method used by [197] uses the energies in different sub-bands to quantise the signal and is depicted in Figure 5.14.

The signal is initially divided into fixed size overlapping segments. A Hanning window was used to smooth the epochs by reducing edge effects. The overlap helps capture the transition of time series over time and helps rectify any misalignment between the query and original event.

As this method focuses on the frequency content, the Fourier Transform was

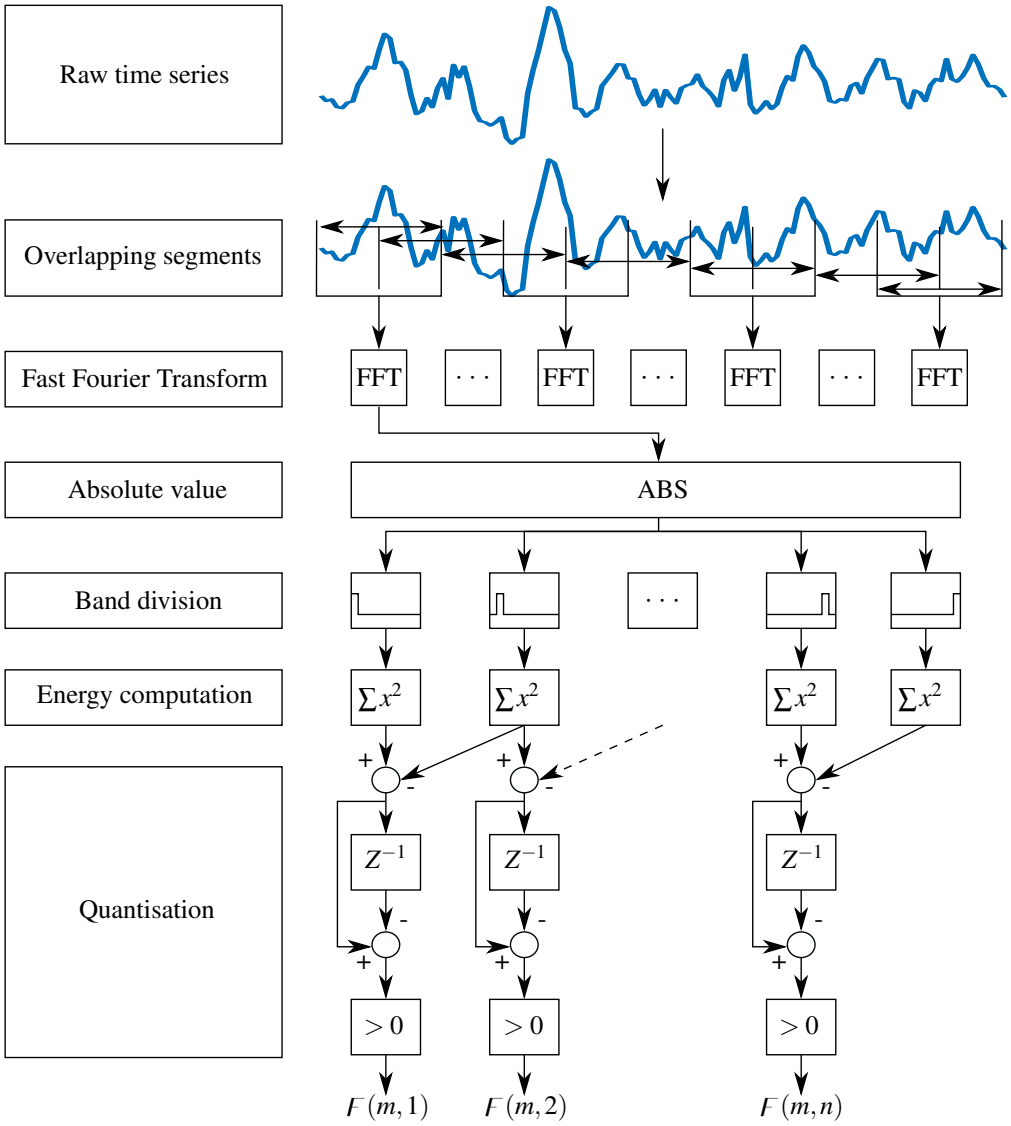


Figure 5.14: Frequency hashing algorithm shows the multiple stages involved

computed for each of the overlapping epochs to capture the frequency spectrum. The spectrum is then sub-divided and this is where differences exist between the audio and neonatal EEG domains. In the audio domain, the bark scale [233] or the Mel scale [234] could be used to divide the spectrum. Then the MFCC are typically used as audio identification features [221]. As the frequency range for neonatal EEG is low, the bark and Mel scales cannot be applied. In place of this linearly spaced bands across the neonatal EEG spectral range were used. As seen in Figure 5.14 following the band division, the energy inside each frequency band is computed. There are in total $m+1$ energy bands.

The quantisation block represents the final stage of this hashing algorithm in

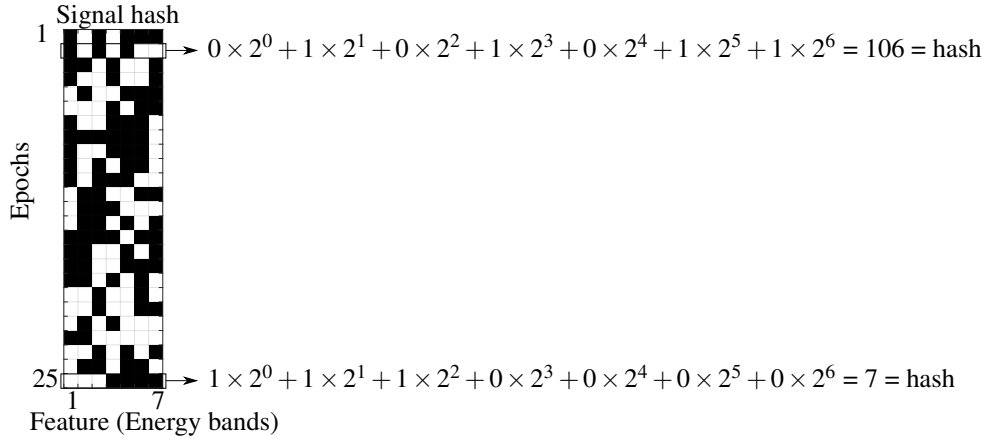


Figure 5.15: Example of the hashes generated from the EEG signal and how they are converted into integer numbers

Figure 5.14. This quantisation is represented mathematically by Equation 5.7.

$$F(j,k) = \begin{cases} 1 \text{ if } E(j,k) - E(j,k+1) - (E(j-1,k) - E(j-1,k+1)) > 0 \\ 0 \text{ if } E(j,k) - E(j,k+1) - (E(j-1,k) - E(j-1,k+1)) \leq 0 \end{cases} \tag{5.7}$$

The output bit $F(j,k)$ is generated where $E(j,k)$ is the summed energy in the k^{th} frequency band for epoch j. In Figure 5.14, the block containing Z^{-1} stands for a delay of one epoch. A hash is created for epoch j by computing $F(j,i)$ for the bands $i = 1$ to m. A hash block is constructed by repeating the process for each of the epochs which then represents the event. The parameters for this system were optimised using PSO and are presented in Section 5.7.1.

Figure 5.15 shows an example of the hash block that was generated for an EEG event. Each row in Figure 5.15 corresponds to a single hash, which is a seven bit binary number. The black pixels correspond to zeros and the white pixels correspond to ones. The leftmost bit corresponds to the least significant bit. Each hash is then converted into an integer using the conversion shown for the second and 25^{th} hashes. It is these integers that are later used as keys to the hash tables where they are stored. There are in total 374 hashes per event. With 7 bits there are 128 possible hash values. This number can be increased or decreased by changing the system parameters. Increasing the number of possible hash values could make the hashing too unique, leading to incorrect NN or no matches if the hashes are too unique.

Feature based hash

The approach described in Section 5.5.1 primarily focused on using the energy in frequency bands as a method to quantise events. This can be extended by using a neonatal feature set to quantise an event. A feature set that was previously developed and optimised to describe neonatal EEG would compress the neonatal EEG in a more meaningful manner. It was the assumption that quantising these features would result in more information being captured. Quantising energy features across multiple energy bands, might cause noise to be introduced into the system mainly for the higher frequencies. The feature set introduced in Chapter 3 is used. It was the assumption that using these features would encode more information into the binary quantised hashes.

Features were extracted from 8-second epochs with a 50% overlap giving thirteen feature epochs per one minute event. PCA was applied to the features in an attempt to reduce the dimensionality. The principal components could be quantised using the basic quantisation algorithm of Equation 5.8

$$F(n,m) = \begin{cases} 1 \text{ if } X(n,m) - X(n,m+1) > 0 \\ 0 \text{ if } X(n,m) - X(n,m+1) \leq 0 \end{cases} \tag{5.8}$$

where $F(n,m)$ is the m^{th} hash bit for principal component n from the principal components vector X. The problem with this algorithm is that the hash size is not fixed as it is determined by the number of epochs in the event. If an event exists with a large number of epochs, then the hash will be large and this may make it very unique, which in turn will make it very hard to find a NN match. When using Equation 5.8 for the database described in Chapter 2, all the hashes for this system will be twelve bits in length and there will be up to a maximum of 55 hashes per event, one hash for each principal component. There is a trade-off between hash length and events recalled.

Another quantisation approach was implemented as the hash approach using Equation 5.8 does not have a fixed size that can be controlled. The ability to vary the number of bits in a system enables the user to tune the system to have either more or less events returned in the query stage. This control of the number of bits in a hash is crucial as it impacts the memory and query speed as will be discussed in further detail in Section 5.5.3.

The approach used to resolve this is to divide the signal into S sections to generate

S bits. Within a section, the first half of the section is subtracted from the second half of the section. If the value was greater than zero, the binary bit was set to one or else it was set to zero. This approach is then repeated for the remaining sections to generate a hash corresponding to an individual principal component. The whole process is then repeated for each of the principal components. This process is represented by Equation 5.9.

$$F(n,m) = \begin{cases} 1 \text{ if } \frac{1}{B-A}\sum_{i=A}^{B} X(n,i) - \frac{1}{D-C}\sum_{i=C}^{D} X(n,i) > 0 & S \leq \frac{F_n}{2} \quad m \leq S \\ 0 \text{ if } \frac{1}{B-A}\sum_{i=A}^{B} X(n,i) - \frac{1}{D-C}\sum_{i=C}^{D} X(n,i) \leq 0 & S \leq \frac{F_n}{2} \quad m \leq S \\ \\ 1 \text{ if } X(n,m) > 0 & S > \frac{F_n}{2} \quad m \leq (2S - F_n) \\ 0 \text{ if } X(n,m) \leq 0 & S > \frac{F_n}{2} \quad m \leq (2S - F_n) \\ \\ 1 \text{ if } X(n,M-1) - X(n,M) > 0 & S > \frac{F_n}{2} \quad m > (2S - F_n) \\ 0 \text{ if } X(n,M-1) - X(n,M) \leq 0 & S > \frac{F_n}{2} \quad m > (2S - F_n) \end{cases} \tag{5.9}$$

where the following were used to make the equation more presentable $A = \left\lfloor \frac{(m-1)F_n}{S} \right\rfloor + 1$, $B = \left\lfloor \frac{F_n + 2F_n(m-1)}{2S} \right\rfloor$, $C = \left\lfloor \frac{F_n + 2F_n(m-1)}{2S} \right\rfloor + 1$, $D = \left\lfloor \frac{mF_n}{S} \right\rfloor$ and $M = 2m - (2S - F_n)$. The m^{th} hash bit for principal component n is $F(n,m)$, F_n is the number of principal component epochs, and there is a total of S hash bits. It is essential that each principal component for each event is normalised to have a zero mean before the application of Equation 5.9. Failure to normalise could result in some hashes being entirely ones or entirely zeros. This problem is mainly the case when $S > \frac{F_n}{2}$ and $m \leq (2S - F_n)$ as the bit decision is being made from a single value as opposed to the difference between two or more values. The hash generated is similar to that of Figure 5.15 with the exception that the hash for this approach has a fixed size of $S\times$ the number of principal components, where S is the number of bits per hash and the number principal components is also the number of hashes in the hash block. Unlike the spectrogram hashing algorithm from Section 5.4, the number of hashes generated can be controlled and set to a fixed number of hashes by varying S. As the value of S increases so too does the variability that can exists in the hashes. The optimised parameters for this system are presented in Section 5.7.1.

5.5.2 Database development

Following the extraction of the hashes from the training EEG events, the hashes need to be stored in a manner that enables quick recall. The approach for the database storage and search is similar to that of [197] where they use the idea of candidate matches. A candidate match occurs when there is a match between at least one hash of the event in the database and the query event. This hash comparison can be achieved by using a hash table [180] where the key is the unique hash generated and the value associated with the key is a file index number. By using a hash table as described in Section 5.2, new events can be quickly added with ease and a fast search where only the relevant candidate matches are returned can be carried out. There is a wide variety of unique hashes possible. Using a hash table can be more memory efficient than traditional storage methods where the hash space can be sparse. Traditional storage methods require all the memory to be pre-allocated for every possible hash value. If the hash space is sparsely filled, the memory will still be allocated. Hash tables do not require the pre-allocation of memory to every possible hash before they exist. The hash table is of the same type as described in Section 5.2, where Figure 5.2 was used to display it visually.

5.5.3 Search algorithm

When a neonatal EEG event has to be identified, the hashes are extracted. Each hash from the query event is searched against the developed database. First, the system checks if the hash key exists; if it does, the indices associated with that hash are then returned. This approach is then repeated for each of the hashes for the query event.

Every event that has at least one hash match is seen as a candidate match. The hash event block for the query is compared to each of the candidate hash event blocks. This comparison enables a ranking of the matches. The comparison that is computed is the Bit Error Rate (BER), which captures the differences between the hash blocks, as illustrated in the Figure 5.16 (e). This illustration shows the difference between the hash blocks for the first and second seizure examples. These seizure hash blocks were generated from the data examples for the first and second patients respectively, which are shown in Figure 5.16 (a) and (b). Figure 5.16 (c) and (d) are the hash blocks for patients (a) and (b) respectively. These hash blocks share some similarities, although looking at the bit errors in Figure 5.16 (e) it is clear that there are many differences. Figure 5.16 (e) shows that there are only two exact hash

matches. The red boxes highlight this in Figure 5.16. In the ideal case of an exact match, there would be a BER of zero; the worst match should technically have a BER of one. The BER for example, in Figure 5.16 is 0.452. If the user required a certain degree of confidence in the matches, a BER threshold could be set.

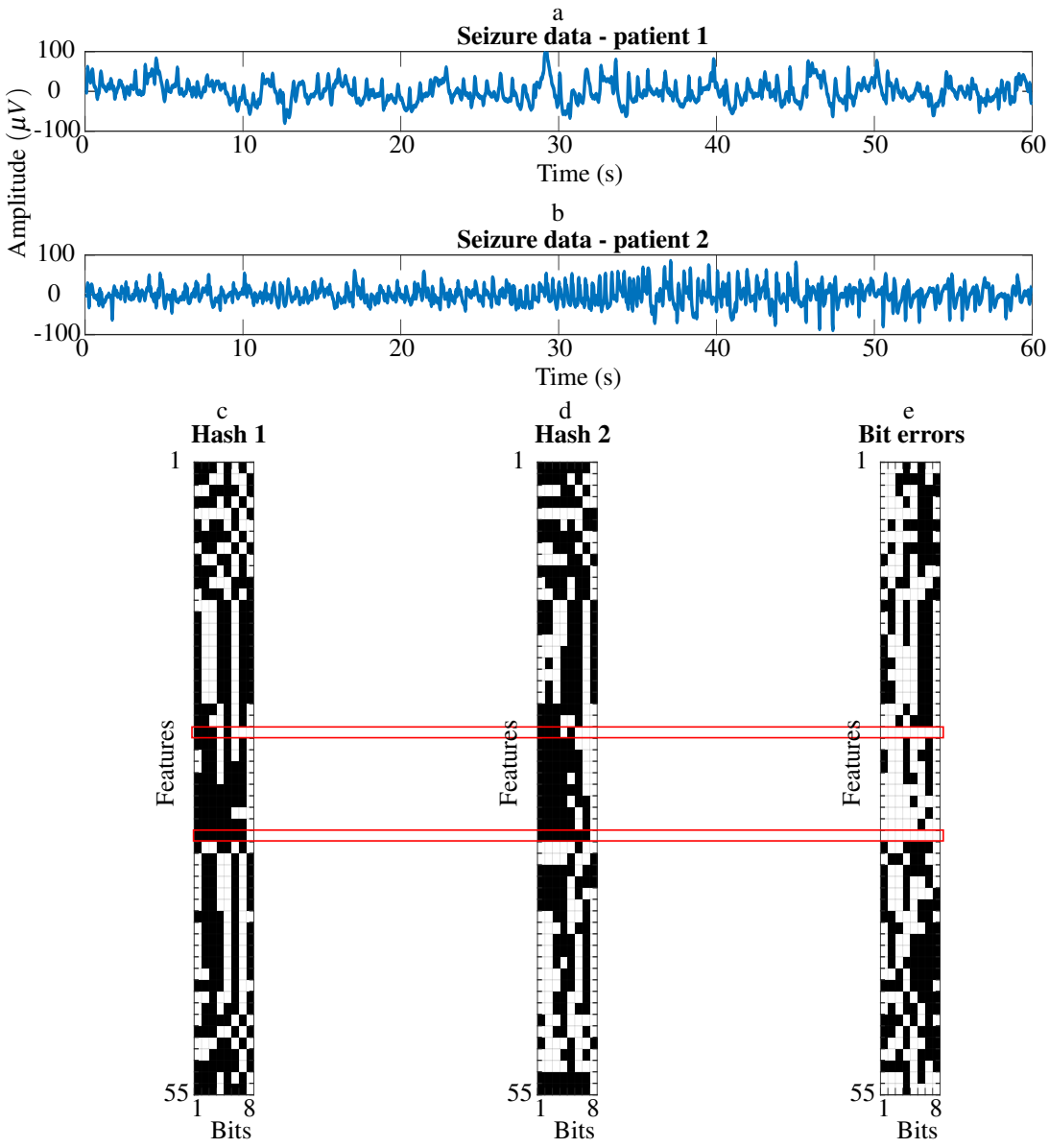


Figure 5.16: Example showing the difference between two seizure events (a) and (b) which were converted into hash blocks (c) and (d) with the differences between the events shown by (e).

Figure 5.17 shows a flow graph of the algorithm when a query is carried out. This flow graph is similar to the spectrogram hashing flow chart from Section 5.4.3. The differences are the hash generation process and the search approach. In Figure

5.17, the BER block is used to return the NN. This query process is not brute force as events are only queried if there is a single matching hash. Since the system uses the BER, it has the potential to turn into a brute force search. This is dependent on the uniqueness of the hashes generated. The results for this hashing method are discussed in Section 5.7, where the NN accuracy, query time and memory requirements are evaluated.

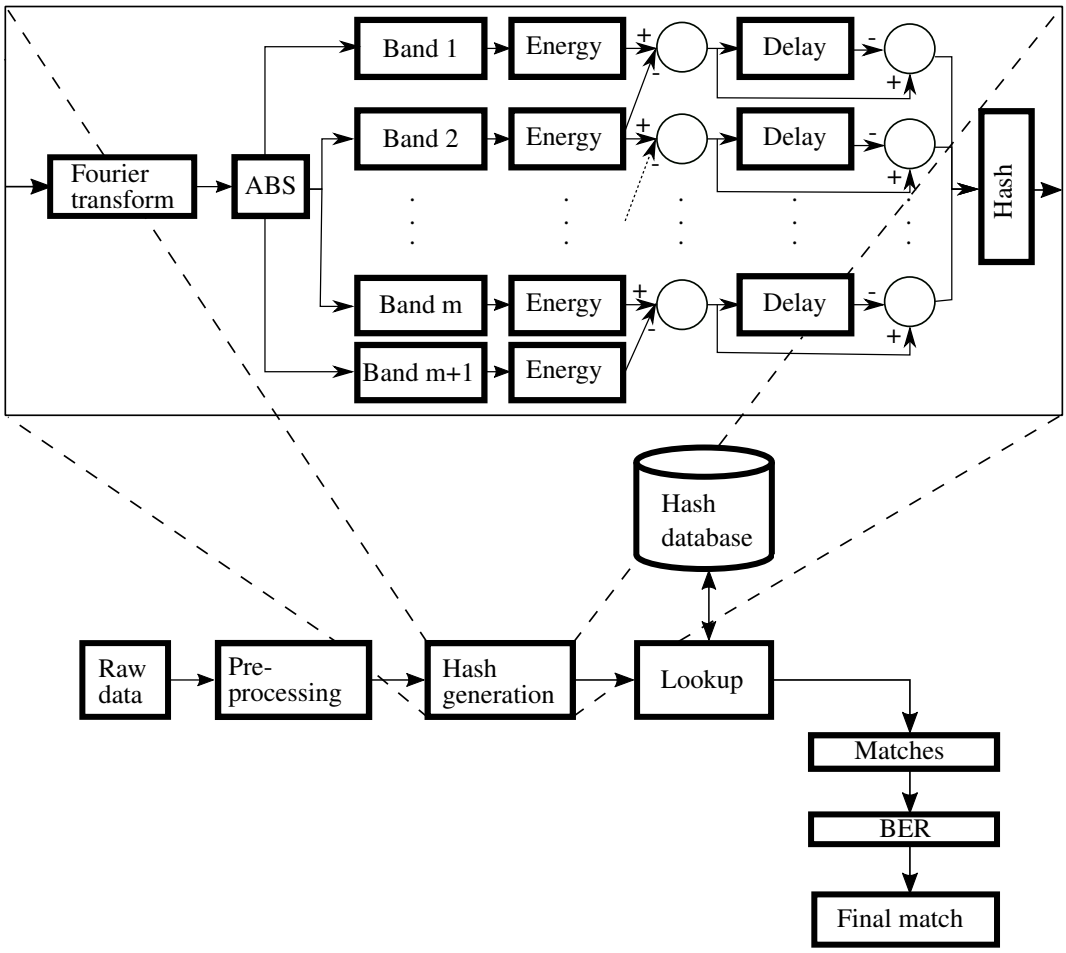


Figure 5.17: Quantisation based query flow graph for the frequency band quantisation hash method

5.6 Locality sensitive hashing

The last hashing scheme that was investigated is LSH. LSH creates a compact representation of the data, that can be stored in memory for quick access when performing a similarity search [235]. This type of hashing has performed well in both the audio

and text domains, as was described in Section 5.3. This algorithm follows the four steps described in Section 5.2, which are;

1. Compression technique
2. Hashing method
3. Database development
4. Search algorithm

5.6.1 Data compression

Data compression is required before hash generation as was described in Section 5.2. Due to the stochastic nature of neonatal EEG, the application of LSH directly on neonatal EEG would result in poor NN recall performance. Three compression steps were considered namely the sketch and shingle method [49], BOP representation [78] and neonatal EEG features as described in Chapter 3.

Sketch and shingle

The sketch and shingle method from [49] was computed for the neonatal EEG. This system will be referred to as the LSH SS approach. This two-stage process transforms the data from a time series representation into a compressed binary representation. Sketching converts the signal into a binary representation, and shingling is then used to divide this binary representation into small binary sequences.

Sketching of the time series signal $x = (x_1, x_2, ..., x_n)$, commences by sliding a normally distributed random filter (τ) across the time series. The random filter has a length W with a step size δ [236, 237]. Equation 5.10 demonstrates how a bit is generated for $x_{i,...,i+W} \subset x$ using the dot product of τ and $x_{i,...,i+W}$.

$$B_x = \begin{cases} +1 : \tau \cdot x_{i,...,i+W} \geq 0 \\ -1 : \tau \cdot x_{i,...,i+W} < 0 \end{cases} \tag{5.10}$$

The sketching process is applied across the entire time series creating the bit string $B_x = (B_x^{(1)}, B_x^{(2)}, ...B_x^{\left(\frac{n-W}{\delta}\right)})$. The step δ is crucial as it helps capture misalignment between time series and it determines the level of compression. Figure 5.18 shows an example of the sketching process. The blocks on the left represent the time series signal. In this example, it uses four samples to create a single bit of the bit

string, and it has a step size δ of two. The dot product with the random filter (τ) is shown on the right of Figure 5.18, and bit string B_x shows the style of the output bits.

With the sketching process, it is inevitable that information will be lost, but the amount of information lost is dependent on the window length W and the choice of step δ. Choosing W and δ to be small will result in more data being retained, although there will be little compression. On the other hand, large values will lead to too much compression, where the compressed sketches become meaningless and indistinguishable from other sketches.

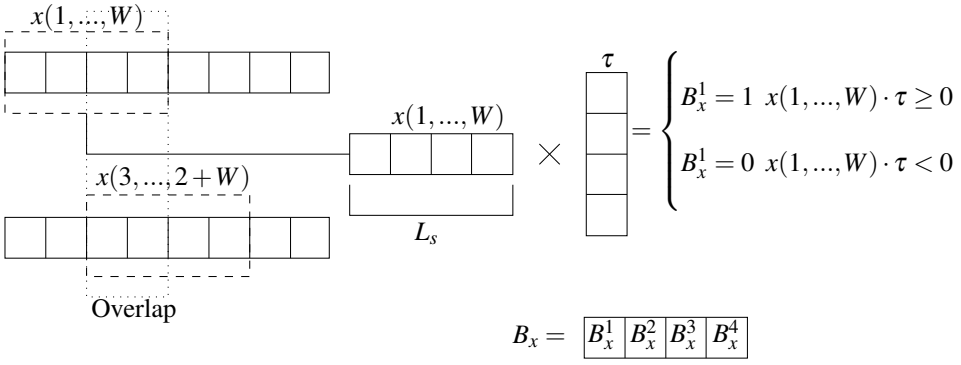


Figure 5.18: Example of the LSH sketching process

If time series events are similar, it is expected that the bit strings would be similar. If bit differences occur between the two full binary sketches, its hashing could prevent a match occurring. Shingling is used to prevent this from happening. Shingling divides the binary strings into subsequences known as shingles or n-grams of length L_s. Instead of comparing two long bit strings, shingling compares multiple shorter sub-sequences, which are more likely to match if events are similar. Shingling has previously been used in text categorisation, which helped account for some spelling errors in documents [212]. The shift between shingles used in this work is one bit.

The length of the shingle L_s will determine the number of possible unique shingles (2^{L_s}). Once the list of all the shingles is generated, a weighted set is generated where each shingle and its frequency of occurrence is stored in a set such as $S_x = ((S_x^1, S\omega_1),(S_x^2, S\omega_2),...,(S_x^z, S\omega_{2^{L_s}}))$, where $S_x^i = (B_x^{(i)}, B_x^{(i+1)}, ...B_x^{(i+L-1)})$ and $S\omega_i$ is the frequency of occurrence of shingle S_x^i.

A low L_s results in little variety in the shingles; therefore there is a higher chance of having a large number of results generated. Having a substantial results pool could make it difficult and time-consuming to find correct matches. A large L_s will reduce the likelihood of having a similar match because there will be a broad range

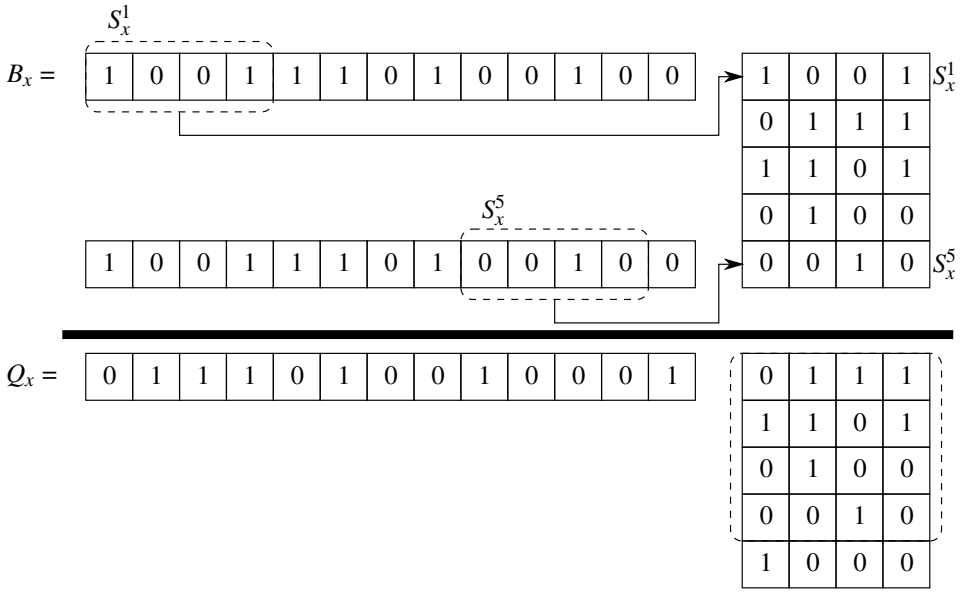


Figure 5.19: Example of the LSH shingling process with query that was misaligned

of possible shingle values, which in turn makes the hashes more unique to an extent where only an exact match gets a hit.

Figure 5.19 shows an example of the shingling process. The bit string B_x generated from the sketch step is divided into shingles of four bits; therefore the shingle length L_s is four. There is a shingle shift of two bits used in this example. In the implemented version a shingle bit shift of one used. From B_x, there were five shingles created as seen from the set of shingles on the right-hand side of Figure 5.19. The bottom half of Figure 5.19 shows an example of the query bitstring Q_x. This bit string is nearly identical to B_x except it is misaligned by two bits. Comparing the bit strings directly will result in a similarity of 38%. Once the shingles are extracted the similarity is increased to 80% as the majority of shingles are identical.

In the ideal scenario, similar events will have similar hashes. The key term here is "similar" and not "exact" thus a balance needs to be found so similar matches can be generated without being inundated with useless results. The parameters for this system were optimised using PSO and are presented in Section 5.7.1.

BOP representation

The application of the BOP approach was discussed in Chapter 4. This compression approach followed by hashing, will be referred to as the LSH BOP approach. This process transforms the time series data of length n (n is the number of samples) into an array of length α^{ω} (α is the alphabet size, and ω is the word length). The

hashing is then carried out on this new representation, as will be discussed in Section 5.6.2. The parameters for this system were optimised using PSO and are presented in Section 5.7.1.

Feature representation

Each feature vector will be used to make a single hash value. Therefore if there are 55 features, this results in 55 hashes being generated. As discussed in Section 5.6, $h(q) = h(x)$ has the potential to occur when the events q and x are different. Therefore multiple hashes need to be taken. If there were multiple hashes taken for each of the 55 features, the hash database would grow at a fast rate.

To account for the growing database, the number of features used was reduced. A crude method of reducing the features was used. The LSH recall accuracy when testing each feature individually was computed using 20 hashes. The features were then organised by the performance they achieved. This process ranked the features by the performance they contributed to the recall system. A brute force optimisation strategy was then implemented that evaluated the performance of the system using one feature while increasing the number of times the feature was repeated from 10 repetitions to 20 repetitions. Then the same approach was tested using the best two performing features and then the best three features and so on until all the features were investigated. A minimum of 10 repetitions was used in order to reduce chances of false hash collisions as it is more like $h(q) = h(x)$ if event q is similar to event x. The results from this optimisation step are presented in Section 5.7.1.

This approach of finding the optimal features and amount of repetitions was only possible due to the limited size of the database. If there were a larger database, an alternative strategy of finding the best features and amount of repetitions would need to be carried out such as PSO which is not a brute force approach.

5.6.2 Hashing

Following the application of the compression steps, the hashing was computed for each of the methods. A hash family H is a collection of hash functions h. The hashing is used to generate the time series event keys for the hash tables. The number of hash functions h used relates to the number of hashes generated for each event. As described in Section 5.3, the goal of LSH is to maximise similar events having similar hashes such as $h(q) = h(x)$ where x and q come from the same event class.

The methods developed in this chapter use weighted min-hashing [238] to generate the hashes. This hashing technique was chosen as it was designed to work with real values. The hashing technique is based on the Jaccard similarity measure $J(q,x)$, which is the ratio of the intersection between two sets q and x, to the union of those two sets as can be seen in Equation 5.11.

$$J(q,x) = \frac{q \cap x}{q \cup x} = \frac{\sum_k \min(q_k, x_k)}{\sum_k \min(q_k, x_k)} = Pr[h(q) = h(x)] \tag{5.11}$$

The Jaccard similarity is not a distance measure although $1 - J(q,x)$ is a distance measure bounded between 0 and 1.

Algorithm 2 shows the pseudo code for a weighted min-hashing Algorithm [238]. The input to this algorithm is a vector M which is the output of the compression step. The algorithm uses two gamma-distributed variables (Γ,b) which are seen in lines 2 and 3 in Algorithm 2. These have shape and scale parameters of 2 and 1 respectively, and then a uniform variable β is seen on line 4. The uniform variable β may prevent line 7 in Algorithm 2 from being floored to zero which has a higher chance of occurring when an element in the vector is close to zero (< 0.5). The output from Algorithm 2 is the hash that is then used to represent the input M.

Algorithm 2 Weighted Minhash

Input: Vector M
Output: Weighted Minhash (k^*, t_{k^*})
1: **for all** k such that $M_k > 0$ **do** // Sampling of random variables
2: $\Gamma_k = Gamma(2,1)$
3: $b_k = Gamma(2,1)$
4: $\beta_k = Uniform(0,1)$
5: **end for**
6: **for all** k such that $M_k > 0$ **do** // Minhash generation
7: $t_k = \left\lfloor \frac{lnM_k}{\Gamma_k} + \beta_k \right\rfloor$
8: $y_k = e^{\Gamma_k (t_k - \beta_k)}$
9: $a_k = \frac{c_k}{y_k * e^{\Gamma_k}}$
10: **end for**
11: $k^* =$arg $min_k\ a_k$

In the general case, Algorithm 2 is repeated n_T times where n_T is the number of hash tables required. The random variables used are pseudo-random as the same variables need to be used for each event. The number of tables n_T was optimised and is presented in Section 5.7.1.

A graphical representation of each of the three processes compression and hashing is seen in Figure 5.20. Figure 5.20 (a) and (b) compress the original event data down into a single array (Shingle weight and word count). The hash functions are then applied to these to create the hash values. Algorithm 2 was repeated n_T times to create n_T individual hashes to represent the event. In Figure 5.20 (c) there are $n_T \times feats$ hashes created. The first hash in Figure 5.20 (c) is created from the first row of feature values from the feature matrix.

5.6.3 Database development

The hashes are used to identify events, and are stored in a hash table for quick recall. Different events types have the potential to have the same hashes, although it is more likely that similar events will have similar hashes. In sections 5.4.2 and 5.5.2, all the hashes generated were added to a single database. This approach is not the same for the LSH systems. If a single hash table was used to store all the n_T hashes from each event, there would be many cases where hashes from different event groups would collide. This is because the random parameters from Algorithm 2 create random hashes for each of the n_T iterations. For this reason, it is important that only the hashes generated using the same parameters are added to the same databases. Therefore there needs to be in total n_T hash tables, one for each iteration through Algorithm 2.

When an event is being added, each of the n_T hashes generated are added to the corresponding n_T hash tables with the hash being the key and the event ID being the associated value. Each event has the appropriate compression and hashing carried out before being added to the database. This process is executed once per event and does not need to be executed in the query stage. This compression reduces a time series event down to n_T hash values.

5.6.4 Search algorithm

In the query or search stage of the algorithm, an unseen event is compressed in the same manner as the events in the database. Following this compression, the hashes are generated in the same manner as the events in the database. These hashes are not added to the existing hash tables but used to search the hash tables instead.

Each of the n_T hashes will search the associated n_T databases and return candidate matches that have the same keys as the query hashes. After each of the n_T

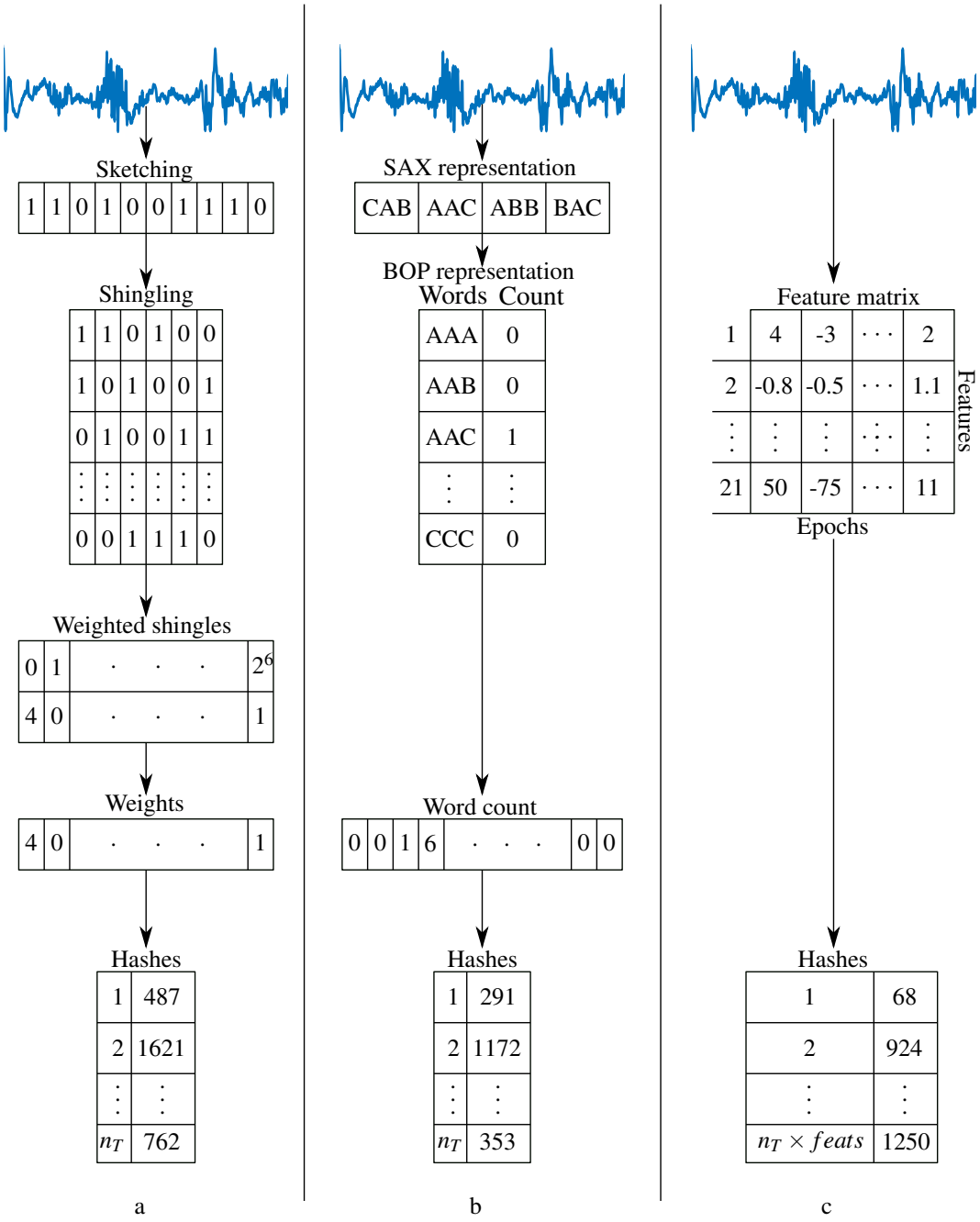


Figure 5.20: Example of the LSH data transformations. (a) is the sketching and shingling approach. (b) is the BOP approach. (c) is the features set approach

databases are searched, the full candidate event list is generated. The full list of candidate events is then sorted by the number of hits each candidate event has received. The candidate with the highest number of hashes in common is seen as the most likely candidate since $h(q) = h(x)$ if similar.

Figure 5.21 shows the flow graph for the system. The idea of using an individual

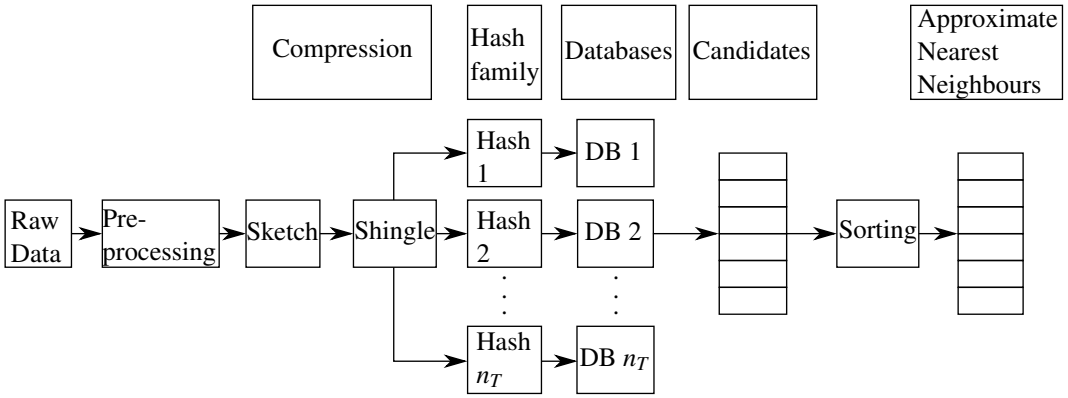


Figure 5.21: LSH based method query flow graph

database for each of the n_T hashes is presented here. There is no connection between the databases. When performing a query, the candidate matches from all the databases are stored in the same place with no further separation, as seen in Figure 5.21. These are then sorted to return the NN which has the highest number of hashes in common with the query.

5.7 Results

In this section of the chapter, the performance evaluation of the six systems is carried out. These systems include spectrogram hashing, quantisation of extracted energy features, quantisation of the feature set principal components (PCA quantisation), LSH SS, LSH BOP and LSH feature set. The same tests that were carried out in Chapter 4 are carried out in this chapter with the database of neonatal events that were discussed in Chapter 2. An additional test called the "hits returned" was also added. As the hashing tests move away from the computational requirements of brute force systems, it was of interest to see how many database events were recalled with each query. The tests that were carried out on each system were;

1. Group majority result
2. Performance accuracy
3. Query running time
4. Memory requirements
5. Hits returned

By evaluating these areas, the performance, efficiency and scalability can be evaluated for each of the six systems. Before the performance results for each of the systems are discussed, the optimised system parameters are presented.

5.7.1 Optimisation

The purpose of this section is to detail the optimised parameters for each of the systems discussed in this chapter. The PSO parameter optimisation strategy was discussed in Section 5.4.4 and was used to optimise the parameters of several different systems. The goal of PSO is to find the optimal parameters of a cost function. The cost function used for finding the optimal parameter set was the accuracy of the system at the NN recall problem. The goal was that the particles would converge to an optimum parameter set that achieved the highest NN accuracy regardless of speed or memory requirements. The memory requirements or speed were not considered as the recall accuracy was the focus in an attempt to achieve comparable accuracy results with the methods discussed in Chapter 4.

If both the memory and NN accuracy require optimisation, the value returned by the cost function would be modified to take into account the memory aspects of the system. This would be done by examining the amount of hashes generated for each event, more hashes equals more memory. Weights can be added to place extra emphasis on the aspects that are most important. As the NN accuracy is more important than the memory, the weights applied would reflect this. The optimisation was carried out using the entire database that was described in Chapter 2.

Spectrogram

The parameters that were optimised for the spectrogram hashing system are shown in Table 5.1. These parameters were optimised to achieve a maximum NN accuracy. The epoch size, FFT size and FFT overlap were all optimised to produce the spectrogram that gave the highest accuracy for the EEG events. The peak removal threshold parameter was discussed in Section 5.4.1. This threshold was optimised to decide roughly how many energy peaks should be removed before the peak location section. The time and frequency dilation parameters were optimised to decide the size of the rectangular dilation structuring element that was used in Section 5.4.1 to find the dominant energy peaks from the spectrogram.

The final parameter that was optimised was the number of points that were able

Table 5.1: Optimised spectrogram parameters

Epoch Size (s)	3.525
FFT size	256
FFT overlap (%)	77.4
Peak removal threshold (%)	71.4
Time dilation	2
Frequency dilation	7

Table 5.2: Optimised frequency parameters

	Frequency
Epoch Size (s)	1.345
FFT size	32
FFT overlap (%)	79.515
# Frequency bands	8

to be generated from each anchor point. A test was carried out that evaluated the system performance while varying the amount of hashes that each anchor point can generate. This test found that allowing three hashes for each anchor point returns the best NN accuracy.

Feature quantisation

The PSO algorithm described in Section 5.4.4 was used to find the optimal or near optimal parameters for the algorithm to work with neonatal EEG; these are presented in Table 5.2. These are the parameters that return the highest NN accuracy in recalling an event from the same class as the query.

For the feature set quantisation process, PCA was applied to the features and the NN recall performance was evaluated using one principal component, then two and this increased until all 55 principal components were evaluated. Using 54 principal components resulted in the highest NN recall performance. The number of sections S, which corresponds to hash bits, that yielded the optimum results was found to be 9. This value was found be evaluating the performance for all possible values of S $(1 - F_n)$.

LSH

The parameters that were optimised using PSO for the LSH sketch and shingle compression step (W, δ, L_s) are presented in Table 5.3. Using these parameters

Table 5.3: Optimised LSH sketch and shingle parameters

Sketch window length W	55
Sketch step size δ	11
Shingle length L_s	8

Table 5.4: Optimised LSH BOP parameters

Alphabet size α	3
Word length ω	6
SAX window length	40

resulted in 169 shingles generated for each event with 256 possible shingle values.

The parameters that were optimised using PSO for the LSH BOP compression step (α, ω, SAX window length) are presented in Table 5.4. With these parameters, there are α^{ω} or 729 possible words and there are 1879 words generated for each event.

The feature set LSH approach approach had only one parameters to optimise. This parameter corresponded to the number of features used. As described in Section 5.6.1 a brute force technique looking at the different combinations was carried out using the entire database. The result of feature set LSH optimisation found that the use of 21 features gave the best NN recall accuracy.

The final parameter found for the LSH systems was the number of hash tables. The number of tables was varied for each system and it was found through testing that there was a minimal performance increase when using more than 150 hash tables for the LSH SS and LSH BOP systems. This approach was altered for the feature set LSH approach, as each feature generates a single hash. Therefore the number of hash tables had to increase in increments of 21 giving $21 \times n_T$ hash tables where it was found that $n_T = 17$ gave the best results.

5.7.2 Accuracy

The accuracy is the systems ability to correctly recall an event with the same event class as the query event. Two evaluations carried out in this section;

1. Group majority
2. Performance accuracy

When a query is carried out, a single event is returned as its NN. If this NN has

the same event class as the query, this is seen as a correct match, and a binary one is returned along with the event class ID. If the returned NN is of a different event class, a binary zero is returned along with the NN event class ID. Five different event types are being tested with background data added to test the performance. These signals were initially discussed in Chapter 2 and are;

1. Background data
2. Short seizures
3. Tracé Alternant
4. Long seizure
5. Pulsatile artifact
6. Respiration artifact

The first aspect investigated was the group majority, and this the event type that was returned the most while evaluating each of the five event groups. Over half of the database came from the background data event type; therefore it is of interest to evaluate if this had much of an impact or bias on the different classes. The group majority results are presented in Table 5.5.

The first column in Table 5.5 corresponds to the events being tested. The background event 1 is not in the table as if these events were evaluated, they would skew the results due to the large number of events. The bottom row of Table 5.5 corresponds to the percentage of all groups that had the correct group majority. The values in the rows $2-6$ represent the event types that were recalled the majority of the time when the event type corresponding to the row was queried. For example, the LSH SS approach recalled the background event type (1) the most when querying both short and long seizure events (2 and 4).

The overall averages reported in Table 5.5 are high except for the LSH approaches not based on the use of features. These approaches misidentified two groups when carrying out this test. What is common amongst the groups that have been misclassified is that the background class (event type 1) was the class that was identified as being correct. A reason for this could be the significant quantity of background data used to evaluate the system. The short seizure class (event type 2) was misidentified in the majority of the tests. A potential reason for this is that the

Table 5.5: Group majority results

Event	Spectrogram	Feature energy	Feature set	LSH SS	LSH BOP	LSH feature set
2	1	2	1	1	1	2
3	3	3	3	3	1	3
4	4	4	4	1	4	4
5	5	5	5	5	5	5
6	6	6	6	6	6	6
Average (%)	80	100	80	60	60	100

short seizure files consist of approximately 25 seconds of annotated seizure activity and 35 seconds of what was deemed to be normal data. This normal data does share a resemblance to the background data that is in the database. These group majority results determine which event type is most dominant when testing each of the different event classes.

A more detailed look at the results is presented in Table 5.6. This table looks at the performance accuracy of each system for each of the different events. The performance accuracy is the systems percentage accuracy at recalling an event of the same class type when carrying out a query. Similar to Table 5.5, the first column is the event type numbers and the bottom row is the average performance for all the event groups. Columns $2-7$ contain the performance accuracy for each of the hash types that were evaluated.

The LSH approaches that compressed the data without input from neurophysiologists have the lowest performance out of all of the systems. This poor performance indicates that the compression into a feature-based approach is the better approach. The performance of the PCA quantisation is weak when compared to the energy quantisation system. Results for speed and memory will show that the higher performance for the energy quantisation system will come with a cost.

An interesting result here is that the performance of the spectrogram and energy quantisation approaches are within 0.54% of one another. This similar performance is interesting as these systems were initially used in the audio domain for NN song recall [179, 197]. The analysis of the memory and query time will show differences between the systems.

Both quantisation approaches perform undesirably when evaluating the respiration artifact class (event type 6). The performance is lower than 62% although the other systems have a perfect recall for this event. Another surprising result is that the

Table 5.6: Performance accuracy

Event	Spectrogram	Feature energy	Feature set	LSH SS	LSH BOP	LSH feature set
2	33.33	58.33	41.67	8.33	41.67	66.67
3	67.5	95	67.5	62.5	35	97.5
4	91.30	91.30	78.26	21.74	52.17	82.61
5	92.73	89.09	85.45	98.18	98.18	90.91
6	100.00	53.85	61.54	100	100	100.00
Average (%)	76.97	77.51	66.88	58.15	65.40	87.54

LSH approaches which are not based on features, work well for the more clear and repetitive artifact patterns of the respiration and pulsatile artifacts. These systems appear to capture this information better than other less structured events.

The system that attained the highest performance accuracy is the LSH system that uses the feature set. The average performance is 87.54%, which is approximately 10% greater than the next closest performing system. The following analysis will show how each system deals with both the query time and memory requirements as the database increases in size.

5.7.3 Query run time

Speed is vital in NN searches and research is frequently carried out to improve the NN search speed [239]. The same query running time experiments that were carried out in Chapter 4 are carried out here.

Figure 5.22 shows the time taken to carry out 25 queries while the database size is increased from 2 events initially, in increments of 1 event up to the full database size of 430 events. The energy quantisation approach does not scale well as the database is increased in size. A contributing factor to this is the fact that this system relies on computing the BER between the query event and every candidate match. When carrying out a query, if this query has at least one matching hash to every other event in the database, then this approach becomes a brute force problem. An interesting aspect is that the PCA quantisation approach does not face the same problem. It is clear that there is an initial offset for the feature set quantisation approach which is caused by the feature extraction process. Following this, the system increases slowly in time. In Figure 5.22, both the PCA quantisation and feature set LSH approaches take longer to compute than the energy quantisation. This longer time is due to the

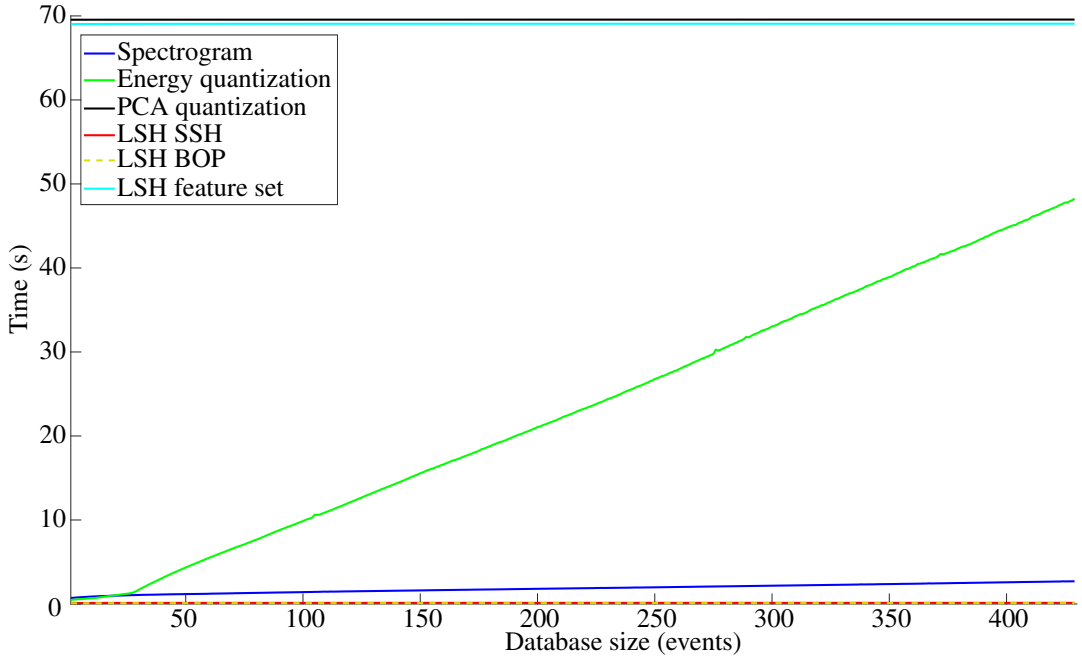


Figure 5.22: Database timing graphs for increasing database size

feature extraction step. However, if the database had more events, it is clear that the energy quantisation approach would surpass all of the other methods in terms of query time requirements. The LSH SSH and LSH BOP hashing approaches are the quickest in Figure 5.22. Figure 5.23 focuses on the LSH SSH and LSH BOP approaches. The initial offset for these approaches is small as the data compression step is quick to compute. Both of these approaches have a similar shape, although the LSH BOP approach is faster to compute, and this is because it is quicker to perform the BOP transformation than the sketch and shingle transformation.

To approximate how each system scales to an increased database size, a power curve equation was fitted to each curve in Figure 5.22. The R-squared values, bounded between zero and one, were then used as a goodness of fit measure [240]. The R-squared value measures the variance of the model to the original data over the total variance in the original signal. From a visual inspection and by computing the R-squared values, it was decided that the equations used to fit the curves in Figure 5.22 were adequate models of each systems' performance. Figure 5.24 shows an example of a model that was fitted to the LSH feature set approach. This model fits the data well as it captures the general structure and it has an R-squared value of 0.9925, which indicated it captures the majority of the variance in the system.

Table 5.7 contains the estimated query time when there are 10000 events in the database and the time difference between when there are 2 and 10000 events in

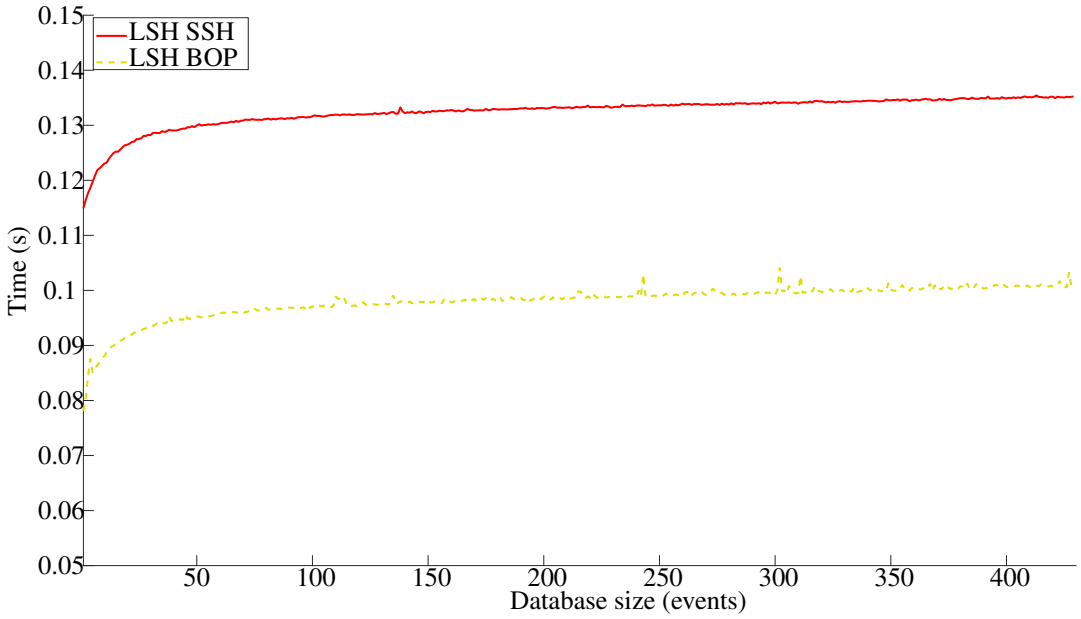


Figure 5.23: Database timing graphs for LSH SS and LSH BOP systems

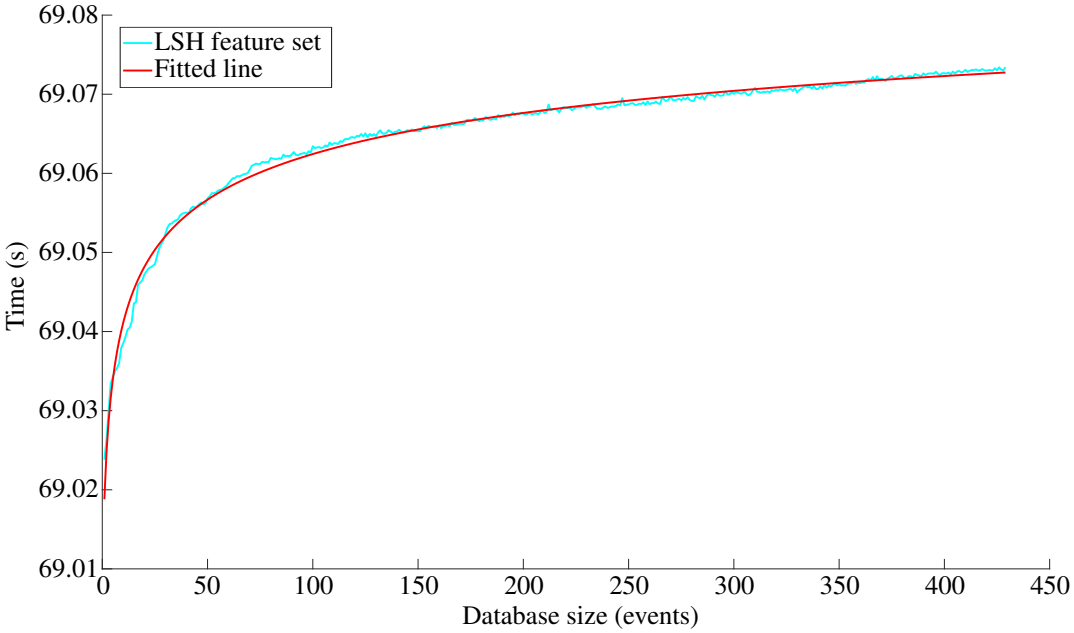


Figure 5.24: Database timing graph for the LSH feature set approach

Table 5.7: Query time for a database of 10000 events and time difference between a database of 2 and 10000 events

Method	Time (s) DB = 10000	Time difference (s)
Spectrogram	24.33	23.61
Feature energy	1413.49	1413
Feature set	69.69	0.21
LSH SS	0.14	0.03
LSH BOP	0.11	0.03
LSH feature set	69.09	0.07

the database. The energy feature quantisation approach takes the longest to compute queries on a database of 10000 events. A reason for this is that as there are more queries, there are more BER computations to make to find the NN. The PCA quantisation approach also needs to compute the BER to find the NN, although this approach had only a slight increase in time when there were 10000 events in the database. The LSH approaches all have a very little time increase as the database increases in size. This little increase is seen in Table 5.7 to be 0.03, 0.03 and 0.07 for the LSH SS, LSH BOP and LSH feature set respectively. The contributing factor to the LSH feature set increasing twice as fast is that this approach generates over double the amount of hashes than the other LSH approaches as will be discussed in the next section.

5.7.4 Memory

The memory requirements are essential, especially for large databases. Systems that require a vast amount of storage/ memory are not suited to larger database deployments. By evaluating the requirements on the current database, it is possible to estimate how each system scales to bigger databases. This test is the memory required by each system to store the database files. The metric used was bytes. The memory value was computed by saving the MATLAB event database as a *.mat* format. The size of the file was then read into the computer and stored. This method was repeated for a database of one event, then two events, up until the full database size 430 events. Figure 5.25 shows the plots for each system. The x-axis is the number of events that were in the database and the y-axis corresponds to the bytes of memory needed for storage.

Looking at Figure 5.25, the spectrogram method has the most significant memory

Table 5.8: Average number of hashes generated per event

Method	Spectrogram	Feature energy	Feature set	LSH SS	LSH BOP	LSH feature set
Hashes per event	1443	374	54	150	150	357

requirements. Then two groups have similar performances, the first is the quantisation of the energy feature and the LSH feature set approaches. These appear to rise at a similar rate, although it appears that the energy feature quantisation approach rises faster. The second group includes the PCA quantisation, LSH BOP and LSH SSH approaches. The LSH BOP and LSH SSH approaches are highly overlapping. This overlap is because both approaches have 150 hash tables each. The PCA quantisation approach has the lowest memory requirements overall. This low memory requirement is because it only generates 54 hashes per event. Table 5.8 shows the average number of hashes that are generated per each event. These values align with Figure 5.25. In general, systems that generate more hashes require more memory.

To estimate how each approach will scale when there is an increased amount of data, a power curve equation was fitted to each line in Figure 5.25. The estimated database memory size when there are 10000 events is shown in Table 5.9. Non-linear power curve models were used because as the database hash table increases in size it does not have a linear increase. Hashes that already existed in the database are not re-added, the file identifier is appended to the other file identifiers that have the same hash. A visual representation of this was seen in Figure 5.2. When a hash is appended, it requires less memory than creating a new hash (key) and index (value) pair. This appending of hashes is a reason for the lines in Figure 5.25 not being linear. Figure 5.26 shows an example of the model that was fitted to the LSH feature set approach. This model captures a lot of the variance contained in the original data and this model has an R-squared value of 0.9997.

The spectrogram approach rises the quickest and requires 26 MB of memory to store a database of 10000 events as can be seen from Table 5.9. This approach produces the most hashes per event as seen in Table 5.8. It produces a significantly higher portion of hashes than all of the other hash approaches. The PCA quantisation approach has the lowest memory requirements as the database grows to 10000 events at just 2 MB. It is also worth noting that the collective hash blocks for each event need to be stored for the quantisation systems. They need to be stored because the

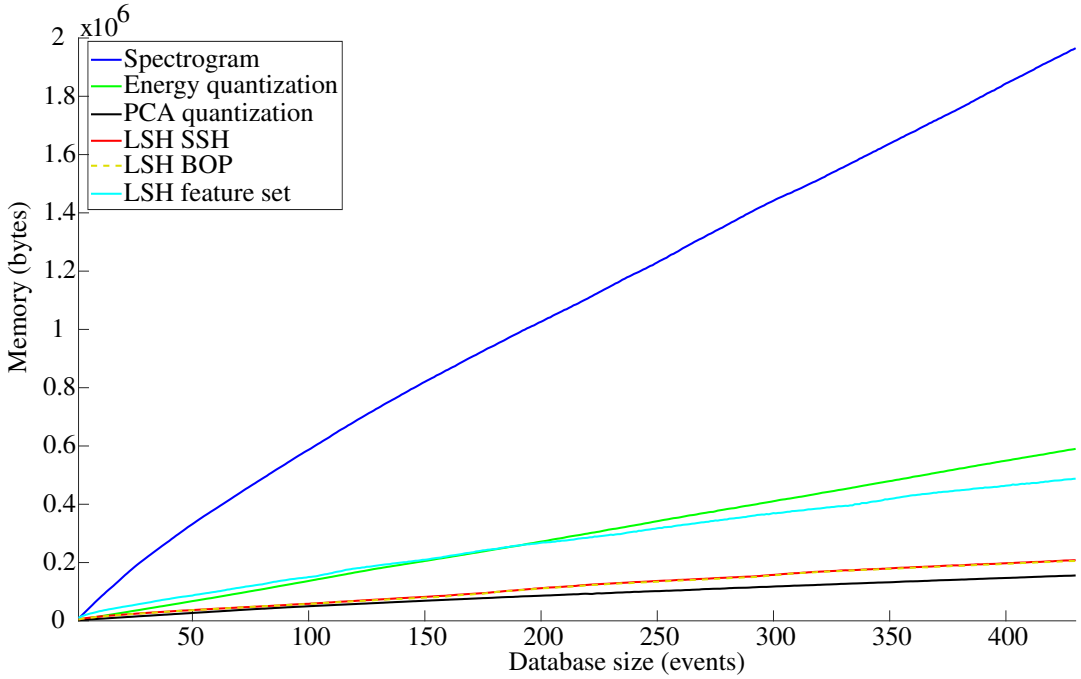


Figure 5.25: Database memory graphs for increasing database size

BER needs to be computed for the candidate matches. As the LSH BOP and LSH SS approaches produce 150 per event and so it was expected they have a similar memory requirement, which is supported by Table 5.9 where both require approximately 3 MB of memory for a database of 10000 events. The LSH feature set approach, on the other hand, created just over twice the amount of hashes than the other LSH approaches as is seen from Table 5.8. This increase in hashes was reflected in the memory needed for a database of 10000 events with it being approximately 6 MB, which is double the other approaches. There is a clear relationship between the number of hashes generated and the memory required.

It is worth noting here that optimisations were carried out for accuracy and did not take into account speed and memory requirements. The spectrogram approach has the highest quantity of hashes. It is also the only approach where the number of hashes generated cannot be strictly controlled.

5.7.5 Database hits

As Chapter 4 is based on brute force applications, all the events are examined when there is a query. This chapter moved away from brute force searches where all the data from all the events need to be compared. It was decided to examine on average how many of the database events get at least one hash match per query. When

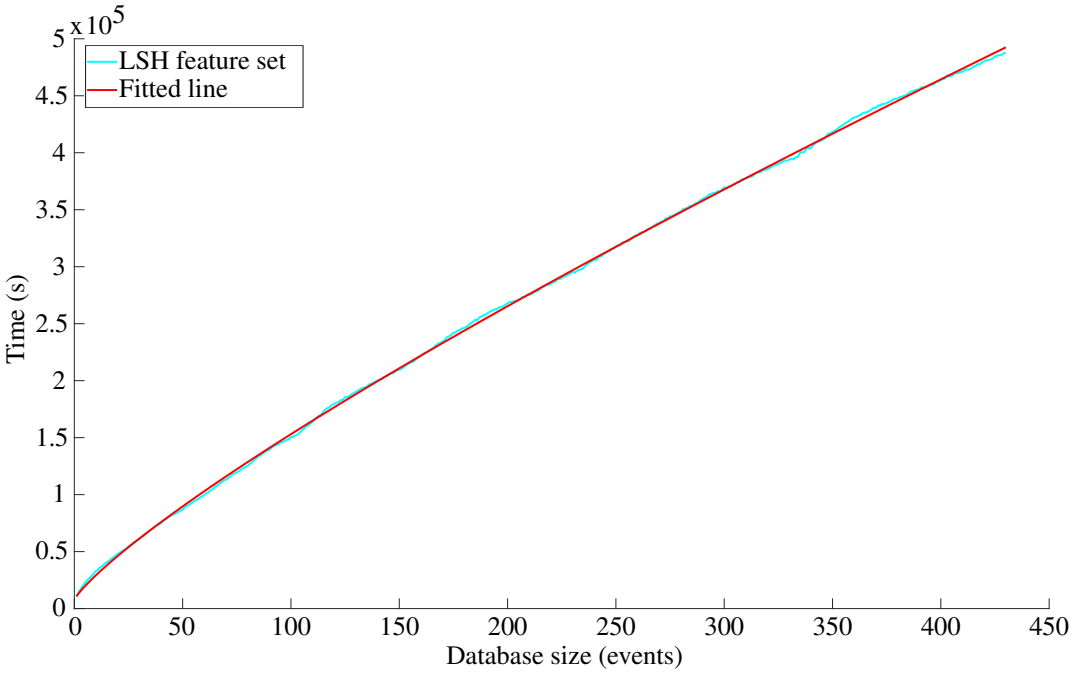


Figure 5.26: Database size graph for the LSH feature set approach to show fit

Table 5.9: Predicted database size for when there are 10000 events

Method	Size (MB) DB = 10000
Spectrogram	26
Feature energy	14
Feature set	2
LSH SS	3
LSH BOP	3
LSH feature set	6

carrying out a query, if a similar match in the database exists, this creates a candidate match. The number of these candidate matches is referred to as the database hits. The number of database hits is significant as if these systems are to be seen as better approaches than brute force systems, they should have a lower number of hits than the database size. This statement is more accurate for the quantisation approaches as they require the BER calculation to be computed for every candidate match.

The number of database hits depends on several factors, of which the most significant factor is the uniqueness of the original hash. If the hash is not too unique, the system will return many hashes. If the hashes are too unique, there is a chance that there will not be many matches. In machine learning terms, the uniqueness can be seen as training. Hashes that are too unique lead to overtrained systems and the opposite for poorly trained systems.

In the ideal case, a query generates a small number of matches with the correct event type being within the matches. From work carried out in this thesis, this does not seem to be the case. Table 5.10 shows the average number of database hits when a query is being carried out.

The majority of the approaches recall the entire database except for the PCA quantisation approach, which recalls approximately 58 events from the database. As the quantisation approaches require the BER to be computed, the quantisation of the energy feature is seen as the worst. This approach has become a brute force search problem, which explains the poor speed performance seen in Figure 5.22.

The other hashing approaches do not require a measure to be computed after getting a list of candidate matches, which is a positive sign. They only need to have the number of occurrences of hashes for the candidate events stored. This problem of large database hits has an impact on the quantisation of the energy feature as the database grows in size, as was predicted in Table 5.7. The PCA quantisation is not a big problem as it only needs to perform the BER calculation on approximately 13% of the database for each query.

Table 5.10: Average number of database event hits when carrying out a query

Method	Spectrogram	Feature energy	Feature set	LSH SS	LSH BOP	LSH feature set
Hits	429	429	58	429	428	429

5.8 Summary

This chapter detailed three different types of hashing approaches; a spectrogram hashing approach, quantisation approaches using EEG energy and the principal components of an extracted feature set and the final approach was an LSH approach which used three different data compression methods.

The main findings of this chapter are that the feature set LSH approach has the highest accuracy while having performance in the average range for both the memory and speed tests. The amount of memory required is proportional to the number of hashes generated, which is as expected. Having the ability to control the number of hashes can help design a system to meet specific memory requirements.

The majority of the hashing approaches recalled at least one hash from every event stored in the database. This is not a problem for the majority of hashing approaches as they do not require further processing on these events. The PCA quantisation and energy band quantisation requires a distance measure called the BER to be computed for every possible match, leading to poor performance for the energy band approach in terms of speed. The performance for the PCA quantisation is not as bad in terms of speed due to the fact it only has approximately 13% of the database to search on average and has nearly seven times fewer hashes than the energy band quantisation approach. The speed is also quicker due to the hash block size being only 14% of the size of the energy band quantisation approach, meaning fewer distance computations for each query.

In summary, this work was the first of its kind where hashing algorithms were developed for neonatal EEG. This work has shown that hashing approaches can be used for the task for NN neonatal EEG recall. Larger quantities of data and an expert are required to carry out further tests without the need for using a proxy multi-class classification problem for evaluation.

The best overall hashing approach from this chapter is the LSH feature set approach. It has the highest accuracy, slowly increasing query time and acceptable memory requirements.

Chapter 6

Multi-class classification of neonatal EEG

6.1 Introduction

As discussed in Chapter 2, the amount of data available is limited and as a proxy, the systems in this thesis are evaluated in a way that resembles a multi-class classifier. For this reason, this chapter examines the performance of machine learning classifiers for the purpose of classifying the neonatal EEG events discussed in Chapter 2.

The work in this chapter involves developing models from each of the different pattern types and assigning a class label to a query, depending on the class it most likely belongs in. This chapter is laid out as follows. Section 6.2 provides background information on classifiers. Section 6.3 introduces the K-nearest neighbour (KNN) classifier. Section 6.4 introduces the Gaussian mixture model (GMM) classifier. Section 6.5 discusses the validation and parameter optimisation, along with the performance of both classifiers for the task of classifying the different neonatal EEG events.

6.2 Classifier background

Machine learning has been successfully used in many areas such as speech recognition [241], handwriting recognition [242], financial modelling [243], physiological signal classification [244] and music information retrieval [245]. There are many examples where machine learning techniques have been applied to physiological

signals. In particular machine learning has been applied to EEG signals in applications such as brain computer interfaces (BCI) [246, 247], biometric authentication [248] and healthcare applications [28, 46]. Likewise ECG signals can be classified for healthcare applications, for example machine learning classifiers have been used to classify ECG arrhythmia [249] and for ECG beat classification [250]. Machine learning has been used for the classification of movement from electromyography (EMG) and electrooculogram (EOG) signals [251, 252].

The purpose of machine learning for classification is to classify unseen data into one group out of two (binary) or more (multi-class) possible groups or classes. An example of a binary classification problem is seizure detection using EEG, where the EEG can be classified as either containing seizure activity or not [27]. An example of multi-class classification, is the grading of neonatal EEG into one of four possible Hypoxic-Ischemic Encephalopathy grades [46]. The classifier is developed from a training dataset that contains EEG events from all the classes. An expert (in this case a neurophysiologist) labels all of these event. When developing a training dataset, every data sample is assigned a label to the class it belongs. This process of assigning class labels to the data is known as annotating.

Classifiers developed from patients data, can be patient specific or patient independent. A patient specific classifier is a model developed on data that contains data from the patient being classified. These classifiers can be used for specialised treatment, where the models need to be fine tuned to an individual patient. Patient independent classifiers train models on data that are completely separate from the patient being classified. These classifiers are ideal for scenarios where patients are under observation for a potential condition such as an encephalopathy. Patient independent classifiers are ideally suited to the work carried out in this chapter.

Before classifying unseen data from a testing dataset, models are generated from the previously annotated training data. If possible, when developing patient independent machine learning models, data from the same patient should not be used in both the training and testing datasets. Mixing patient data between both training and testing datasets can lead to models that are over optimistic and have poor generalisation. When the training and testing datasets are comprised of data from completely different patients, the models generated are patient independent.

A subset of the data, that is divided into training and validation is used to identify the optimum model parameters. This is done by moving through the parameter space and developing models for each parameter set. The validation set is used to find the

model that has the best generalisation performance on the validation set. Without a sufficient validation dataset, the models may overfit the training data. Overfitting occurs when the models represent the training data too closely and as new data is presented, the model fails to correctly classify it. Once a classifier model is developed and fully trained after finding the optimum hyper-parameters, a fully unseen testing dataset, that was not in the training or validation stages is then used to perform a performance assessment.

There are multiple ways to achieve this performance assessment, for example the data can be divided into t subsets with the patient data not mixed between the subsets. Following this, $t-1$ subsets are used to train the model and the left out subset is used to evaluate the model. This process is repeated until all of the t subsets are used to evaluate the model and the mean performance of the t tests is taken as it would give an indication of the overall performance. Another approach that provides a good indication of model performance is leave-one-out (LOO) testing [253]. The approach assesses the model by training the model on all but one of the training patients and testing on the patient that was left out. This approach is then repeated until all the patients have been left out, the mean performance is then taken. This approach ensures that the model can generalise the data.

Each of the machine learning techniques in the papers cited can be roughly divided into two types - discriminative or generative models [254]. Broadly speaking, a discriminative model uses the data directly to classify unseen data, whereas, the generative models are constructed from the distributions of the training data.

6.2.1 Classifiers

There are a multitude of classifiers that can be used to solve many different problem types. As there are so many potential classifiers that could be described and detailed, this background section will focus on providing a high level description of five classifiers. Two of the classifiers are simplistic classifiers, two of the classifiers are more complex and the last classifier is the current state of the art. The two simplistic classifiers are the KNN [255] and linear discriminant analysis (LDA) [256] classifiers. The two more advanced classifiers are the GMM [257] and support vector machine (SVM) [258] classifiers. The current state of the art classifier is the convolutional neural network (CNN) which has already shown excellent performance in image classification [259].

The high level overview of these approaches will focus on four main areas:

1. Complexity - The term complexity is used to describe how complex the machine learning algorithm is.
2. Capacity - The capacity of a system is essentially the level of information that a system can model.
3. Number of parameters - The number of parameters that a classifier has.
4. Training required - The training required is in reference to the amount of training that is needed to make the classifier models.

By examining these areas an overview of the algorithms will be provided without detailing every aspect of each of the five classifiers.

Complexity

The KNN is the most simplistic classifier of the five. It simply looks for the K closest data points in the training dataset [260]. There are no advanced steps involved and there are only two parameters that can be chosen, which are the K value and the distance metric used. It is a discriminative classifier that relies on all the training data being present when performing a classification.

The LDA classifier is a generative classifier and slightly more advanced than the KNN algorithm. LDA reduces the dimensionality of the data while focusing on maximising the separability of the classes [256]. When used as a classifier LDA is relatively simple. The same dimensionality reduction step that was applied to the training data is applied to the query data, the distance to each class centre in the reduced space is used to classify the query [261].

Moving to a more advanced generative classifier, the GMM uses a weighted set of Gaussian components to model all the training data from each of the classes individually, creating one GMM model per class [257]. When classifying a query, the model that has the highest likelihood of the the data belonging to that model is chosen as the best class. There are more aspects to the GMM algorithm as compared to the KNN and LDA algorithms. The data undergoes an initial dimensionality reduction followed by an approach called expectation-maximisation (EM) to find the optimum number of Gaussian components for the training data. Unseen data

is classified based on the likelihood of it belonging to each of the different class models.

The SVM is a discriminative binary classifier that uses hyperplanes found from the training data to classify unseen data [258]. There are a lot of potential hyperplanes that can exist to separate the two classes, however, the goal is to find the hyperplane that provides the maximum possible separation between the two classes. More often than not the classes being classified are not fully separable and to help account for this a soft margin is used [262]. The soft margin allows some leeway in the classification boundary. Kernel functions can be used when the data is not linearly separable. A kernel function enables the mapping of the data to a higher dimensional space making it easier to separate the classes [263]. In general SVM classifiers are not simple and require appropriate training to ensure the optimum decision surface is found to generalise the training data. The decision surface can be made more or less complex depending on the choice of model parameters.

The current state of the art and most complex classifier is the CNN [259]. In the last few years there has been a renewed interest in the area of neural networks and the CNN in particular. The previous classifiers required hand crafted features to be computed from the raw data. This means that the classifier relies on the features containing discriminatory information between the different classes. This is not the same for the CNN, as it both extracts the features and performs the classification of the raw signal [25], making it a very complex system. The level of complexity of the CNN can be controlled by the choice of architecture that is used. Complex classification problems may achieve better performance when using deeper architectures.

Capacity

The KNN has the lowest capacity, as it contains no underlying information from the data. The classification accuracy decreases as the dimensionality of the data increases [260]. This indicates that the KNN has a low capacity and is more suited to less complex and lower dimensional data.

The LDA classifier initially reduces the dimensionality of data while maximising the class separability and minimising the per class spread [256]. The LDA classifier has a higher capacity than the KNN classifier as the class separability is maximised for the LDA classifier.

The more complex GMM generative classifier is able to perform better than the LDA classifier for more complex data. This is because the GMM uses more

information about the distribution of the data to generate models to represent the data. GMMs are suitable for the modelling of dominant patterns that may exist as they can model the distribution of the data. If the patterns are not dominant and vary a GMM may struggle to provide an accurate reflection of the data in the generative model.

The SVM differs from the KNN discriminative classifier as it only retains the support vectors as opposed to all the training data following the training. The use of non-linear kernels and soft margins in SVMs increases the capacity of learning complex decision boundaries.

The CNNs have the highest capacity, the complexity of the features that a CNN extracts can be varied by using more layers [264]. Generally speaking, the features extracted from shallow networks (low number of layers) relate to small amounts of the initial input data. The features extracted from deeper networks are more complex and encompass more of the input signal as they are generated using a combination of the shorter duration features from the earlier layers. This is what enables the CNN to learn complex features from the raw data.

Number of parameters

The KNN algorithm has two tunable or hyper parameters that can be chosen and these are the choice of K and the distance metric used to compare the query and database samples. The LDA classifier on the other hand has one tunable parameter, which is the dimensionality of the data. The LDA does use further system parameters that are generated from the training data. These are the class means and dimensionality reduction information [256]. The size of these parameters is dependent on the amount of classes and the dimensionality of the data.

Moving onto the more complex GMM classifier, the main tunable parameters for this classifier are the number of Gaussian components used to represent the data and the number of principal component analysis (PCA) principal components (PCs) to retain. When training GMMs, system parameters are developed and like LDA, information in relation to the dimensionality reduction technique is kept along with the distributions of the Gaussian components. The number of total parameters is dependent on the tunable parameters and the dimensionality of the data.

The SVM has several tunable parameters such as the kernel function, the strictness of the soft margin and dimensionality of the data. The amount of system parameters or support vectors generated is dependent on the classification task. For

data that is linearly separable, a hard margin could be used with only a few support vectors to define a strict class boundary. On the other hand, a classification problem that is not linearly separable could need a large amount of support vectors to define the decision boundary [72].

The amount of parameters in a CNN is dependent on the type of architecture that is used. Typically as the number of layers in the model increase, the amount of parameters increases [264]. Without delving too deep into CNNs, the parameters may include the choice of cost function, momentum, learning rate, weights for the convolutions and the size of the convolution window, the type of pooling and the size of the pooling windows and the type of activation function. There are a vast number of parameters needed for a CNN. The depth of the model can depend on the classification task and the amount of data available.

Training required

No training is required for the KNN as no model is generated. As the LDA algorithm is a generative model the training is used to get information about the distributions of the data and for finding the optimal dimensionality reduction.

As the GMM is more complex than the LDA algorithm it requires more training data to get an accurate reflection of the signals being modelled. The training using the EM algorithm helps fine tune the Gaussian components that model the data. The GMM also has more parameters than the LDA algorithm which also indicates it would require more training. The tunable parameters can be found using a validation procedure [27].

The amount of training for an SVM model is dependent on the model parameters and the underlying data. Data that is linearly separable may be trained with a limited amount of data as only a few support vectors would be needed. As the classification problem and models become more complex, more support vectors may be required to define the hyperplane that separates the two classes, such as in [72] where 19% of the training data became support vectors in the seizure detection system. Classification problems that are not linearly separable require more training than linear separable problems. This is because complex decision boundaries need to be found and validated so they can generalise the data. Like the GMM, the tunable parameters can be found using validation.

The CNN performs both the feature extraction and classification steps and so requires a significant amount of training data. Large amounts of training and vali-

dation data are needed to create models that can generalise the training data. The minimum amount of training, training data and validation data is dependent on the depth of the models. If there is not a sufficient amount of training data used to train deep models there is a high chance that the developed model will be overfitted to the training data.

6.2.2 Classifiers summary

Table 6.1: High level classifier summary

	Complexity	Capacity	Number of parameters	Training required
KNN	Low	Low	Low	None
LDA	Low	Medium	Low	Low
GMM	High	High	Medium	Medium/ high
SVM	High	High	High/ Very high	Medium/ high
CNN	Very high	Very high	Very high	Very high

This section provided a high level overview of five different classifiers that ranged from simple classifiers to the complex state of the art classifiers. These classifiers were examined looking at four different aspects, each of which should be considered when choosing a classifier. Table 6.1 provides a summary of the classifier findings.

The KNN and LDA classifiers are quite similar and not complex classifiers as seen from Table 6.1. Therefore the KNN was the initial classifier investigated for the multi-class classification problem. The KNN has also previously been used with EEG for multi-class classification in the area of BCI [265].

Looking at Table 6.1, the GMM and SVM classifiers are quite similar. As the GMM classifier uses a lower number of parameters it was decided to investigate it further; it has been used in previous EEG studies [27,266]. A benefit of using GMMs is if there is not enough data to generate a model for each event group, a background model from all the data can be generated. This model can then be adapted to model the event class that has a limited amount of data [267]. The GMM approach is easily scalable if a new event type is added by making an individual model or adopting a background model.

Table 6.1 shows that the CNN is a very complex classifier and it would not be suitable in this work, with the limited amount of available data.

6.3 K-nearest neighbour

As discussed in Section 6.2.1 the basic KNN classifier is an easy to implement classifier as it has a single parameter to tune which is K, the number of nearest neighbours from which the decision is made. The KNN classifier is a discriminative model that was initially used as a non-parametric method for pattern classification [255]. A discriminative model is a model that attempts to classify based directly on the model inputs (training data) [254]. The KNN method resembles that of the brute force nearest neighbour (NN) methods described in Chapter 4 where a distance metric was used to find the NN event. The KNN method has training samples, and these are all the individual feature epochs from the events that are used to help form the classification decision. These training epochs become the database. Then when a query is carried out, it is tested against each of the individual training epochs in the training database by computing the distance between the points.

The classes of this particular problem consist of six different neonatal EEG patterns, as were discussed in Chapter 2. The raw events were preprocessed and features were extracted. This process was discussed in Chapter 3. This feature set contains $F = 55$ features across the time, frequency and information theory domains. Each feature was computed for an eight-second epoch with a 50% epoch overlap. As the features have different amplitudes and units, they were normalised using Equation 6.1 so higher amplitude features would not bias the results. This normalisation was also carried out for the same purpose in Chapter 4 with the features being normalised to have a mean of zero and a standard deviation of one.

$$\overline{X}(f,i) = \frac{X(f,i) - \mu_f}{\sigma_f} \tag{6.1}$$

Where μ_f is the template mean, and σ_f is the template standard deviation for feature f computed from the training data. This normalisation was repeated for all the F features for both the training and test data. Using template means and standard deviations preserves important amplitude information that exists between different events. The template values are then stored for later use on query events.

Following the normalisation, the distances between different epochs are easily computed for the KNN algorithm. Many different distance metrics can be computed such as the Mahalanobis distance, Manhattan distance and the Jaccard distance to name a few [268]. Although for this work, the Euclidean distance was computed as it attained the best results from the experiments conducted in Chapter 4. The

Euclidean distance for the KNN approach is computed using Equation 6.2.

$$||\overline{X}(i) - \overline{\tilde{X}}(j)||_2 = \sqrt{\sum_{f=1}^{F} \left(\overline{X}(f,i) - \overline{\tilde{X}}(f,j)\right)^2} \tag{6.2}$$

This equation computes the distance between the i^{th} epoch in query $\overline{X}$ and the j^{th} epoch in the database $\tilde{X}$ by summing the distance of each of the F features. When there is a query, the distance is computed between all the training database epochs and the query epoch. Each epoch in the database $\tilde{X}$ has an associated class label C. The classes belonging to the K events with the lowest distances are recorded, where K is the number of nearest neighbours.

To determine the desired output class, the weighted majority vote of the top K classes is taken.

$$WMV_i = \frac{CN_i/K}{DB_{CN_i}/N} \tag{6.3}$$

Where CN_i corresponds to the number of events in class i that were in the top K results, and DB_{CN_i} is the number of events of class i that are in the database. After Equation 6.3 is computed for each of the classes, the class that has the maximum value in WMV is taken as the weighted majority. In Chapter 4 a query event is compared directly with all the events in the database individually and the event with the lowest distance to the query event is seen as the closest match. The KNN approach does not compare events with events, it looks for the closest matching feature epochs from the training database that have no particular ordering. The decision on the closest match is then made using weighted majority voting.

Figure 6.1 shows an example of the KNN problem that contains three classes and ten events in each class. In this particular example, the query belongs to the triangular class. When looking at $K = 3$, the circle class holds the majority with two out of the three events being from the circle class. As K is increased to $K = 10$, the triangle class has the majority with half of the matches. When the value of K is increased to $K = 26$, both the triangle and star classes hold the majority. To prevent situations like this, K is usually set to an odd number. This image demonstrates the importance of choosing the correct value of K. In this example, a small value of K would lead to the query being misclassified as it happened to be near the circular class outliers. Noise has an impact on the value of K. A large value of K includes too much of the data. As this approach has only one parameter (K), the value was

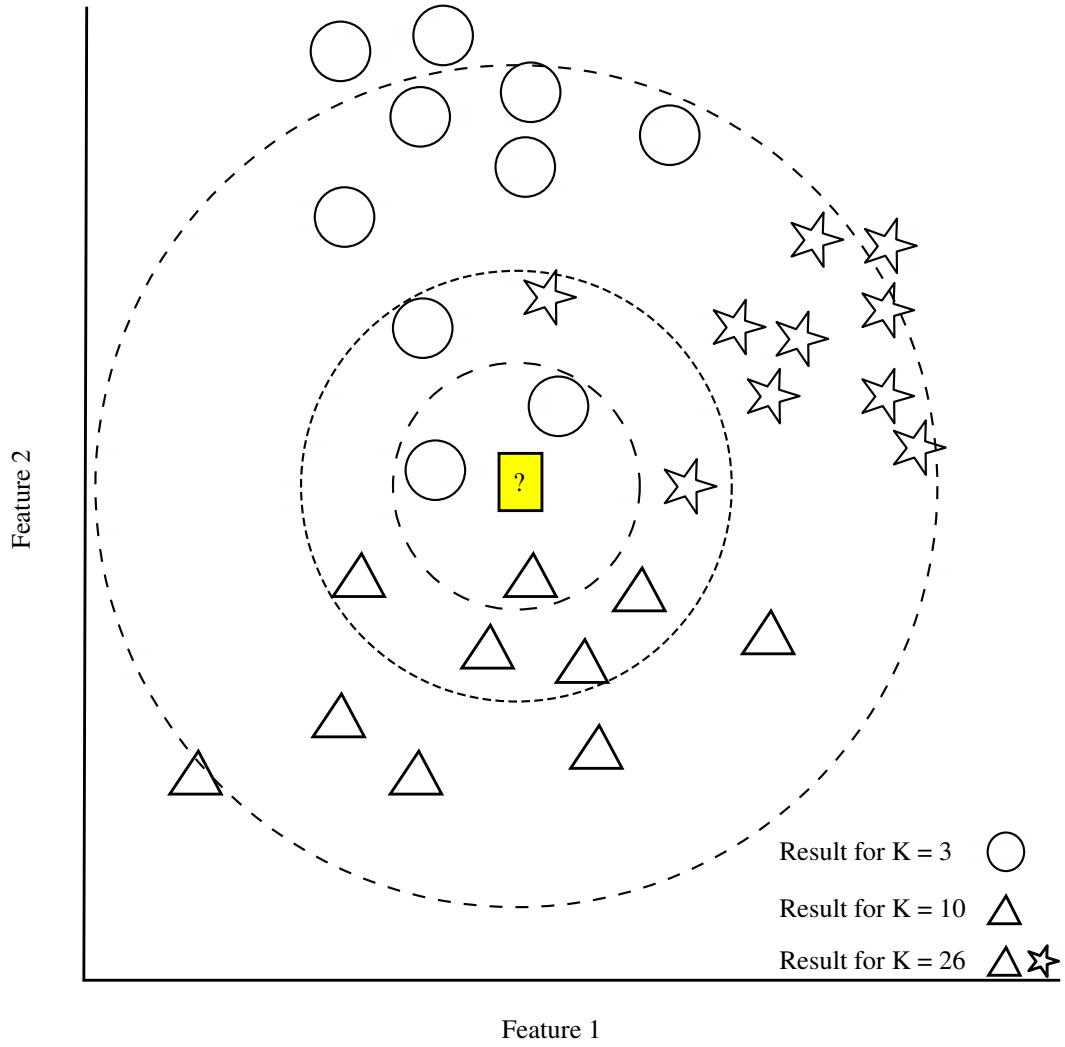


Figure 6.1: KNN example plot showing the importance of choosing the correct K value. There are three different event classes and three different sizes of K.

found by examining the performance while increasing K. The highest performance was achieved when using a K of 31.

6.4 Gaussian mixture model

The GMM is a generative model that uses a finite number of weighted Gaussian components to model the data that is represented in the feature space X [269]. A generative model is a model that represents the training data based on the distribution of the data. The model learns the joint probability distribution when given the observed variables and the target variables [270]. This means that the generative classifiers learn class models from the training feature vectors. These models represent

how these feature vectors are distributed. When there is a test event to be classified, the log likelihood of their feature vectors belonging to each class is computed. The class that has the highest likelihood is chosen to be the most likely class [271].

A note must be made on the data representation. The time-series data is transformed into the feature space and are then normalised in the same manner as was discussed in Section 6.3. The features are transformed into PCs by the application of the PCA algorithm [106]. This both reduces the dimensionality of the data (by keeping the PCs that contain the most information) and diagonalises the covariance matrix by transforming it into an orthogonally separated uncorrelated space. A diagonal covariance matrix requires fewer computations to be carried out when fitting the GMM [272]. The PCA eigenvectors are computed from the training data and then applied to the testing data following normalisation to ensure the same transformation is applied. For the rest of this chapter when a feature vector X is discussed, it is assumed that the data has undergone normalisation and that PCA has been applied.

A GMM is generated for each event class c by using data associated with the class label. For this work, six GMMs are generated, one for each of the event classes. The number of Gaussian components needed to model each class is not related to the number of event classes that exist. The GMM probability density function from feature vector X for class c (with parameter set λ^c) is represented in Equation 6.4 as the weighted sum of Gaussian components.

$$p(X|\lambda^c) = \sum_{i=1}^{M} \omega_i^c g(X, \Sigma_i^c, \mu_i^c) \tag{6.4}$$

Where $\lambda_i^c = \{\omega_i^c, \mu_i^c, \Sigma_i^c\}$ and this is the i^{th} component when there are in total M components. The variables ω, μ and Σ stand for the weights, mean and covariance matrix respectively. The Gaussian weights sum to one. The Gaussian function $g(X, \Sigma_i^c, \mu_i^c)$ is represented by Equation 6.5.

$$g(X, \Sigma_i^c, \mu_i^c) = \frac{1}{(2\pi)^{\frac{F}{2}} \left|\Sigma_i^c\right|^{\frac{1}{2}}} e^{-\frac{1}{2}(X-\mu_i^c)' \Sigma_i^{c-1} (X-\mu_i^c)} \tag{6.5}$$

The values for ω, μ and Σ are optimised using k-means clustering to first find initial parameter values which are then further refined using EM [273] which will be discussed shortly. These values were computed using all the training feature vectors of the form X_T^c, which are a batch containing all feature sequences belonging to class c and are of length F_c. This approach aims to maximise the log-likelihood that the

generated model fits X_T^c as given by Equation 6.6.

$$L(X_T^c|\lambda^c) = log\prod_{j=1}^{F_c} p\,(X_T^c(j)|\lambda^c) = \sum_{j=1}^{F_c} log\sum_{i=1}^{M} \omega_i^c g\,(X_T^c(j), \Sigma_i^c, \mu_i^c) \tag{6.6}$$

It is hard to know how many Gaussian components to use to represent the data. A large number of components could result in the models overfitting the training data and too few components could result in the models not being able to represent the data. Model selection was used to choose the number of PCs to keep after the application of PCA and the number of Gaussian components needed to represent the data. The model selection process and parameter set used are discussed in Section 6.5.1. Once the optimum values are found, a GMM was generated for each of the six event classes. When a query is performed the log-likelihood that the query data belongs to each of the classes is computed using Equation 6.6. The model that gives the best log-likelihood is then chosen as the most likely event class.

Then GMMs for each of the six event classes are generated using the training data belonging to the respective classes. Figure 6.2 shows an example of this GMM process on generated sample data. In the first stage, there is a scatter plot that happened to contain four unique clusters. In order to show that the GMM approach can adapt, three Gaussian components are used to represent the four distinct data clusters. These individual components are illustrated below the scatter plot. The third or right component is modelling two clusters as this was found as the best solution when computing the EM algorithm. The final distribution is then the weighted sum of these components that is seen below the three individual components. This model represents a single event class. The process is then repeated for all the remaining event classes.

The following sections detail the process of making the models in more detail. That is the initial parameter estimation using k-means clustering, followed by the EM algorithm. These two areas are important as they detail the process involved in finding the optimal GMM model parameters that best represent the data.

6.4.1 K-means clustering

The initial Gaussian centres are found using a process known as k-means clustering [274]. The purpose of k-means is to assign the data points to specific clusters.

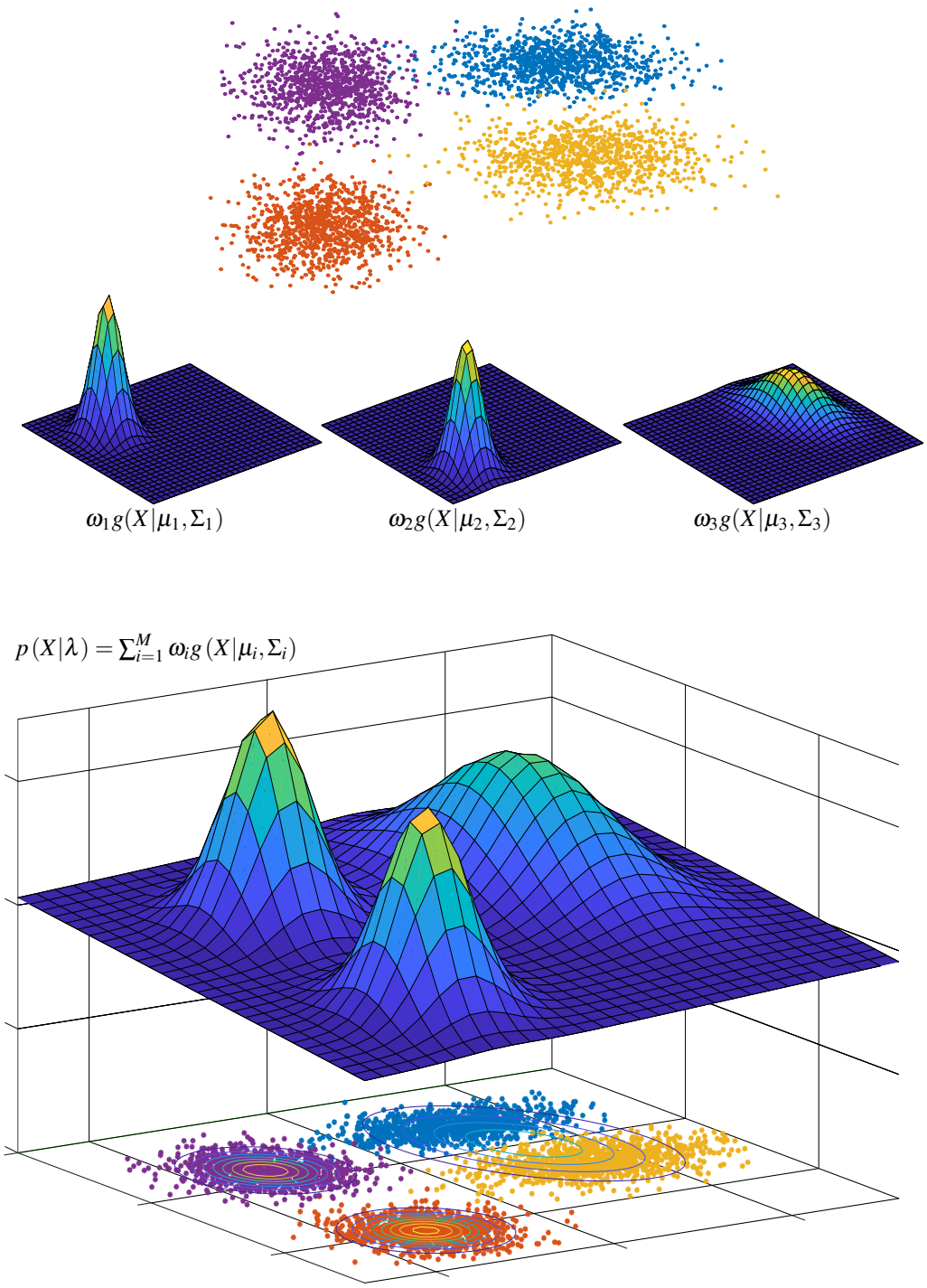


Figure 6.2: Example showing how a GMM model is created. The initial data is seen from the scatter plot, there are four clusters and the model will attempt to fit three Gaussian components. The second stage showing the construction of the three individual Gaussian components. The final stage shows the weighted sum of the Gaussian components from stage two.

This assignment process is seen as an initial estimate to the optimum cluster locations. Following this, the EM process is used to refine the clusters to maximise the probability of belonging to the specific cluster.

The k-means clustering algorithm initially places the k centroids randomly in the data. Then the algorithm loops through all the data points and finds the closest cluster for each data point by computing the distance (such as Euclidean distance) between the data point and each cluster. At this stage in the algorithm, each data point is assigned to a cluster.

The next stage in the algorithm is to loop through each of the k clusters and recompute the cluster centre. This cluster is updated by averaging the points that belong to each particular cluster.

This two-step process is repeated until either a set number of iterations is met, or if after a full run through of the first step, no point has changed class membership. Figure 6.3 shows an example of the k-means algorithm on random data with four obvious clusters. The k-means algorithm was then set to have three clusters to determine how the four groups should be assigned to three clusters. This example took nine iterations until there were no changes in class memberships. The initial cluster positions were chosen at random, as is seen in the first iteration in Figure 6.3 (a). The third iteration shows that the yellow cluster appears to be taking ownership of the two rightmost clusters and the orange cluster is being more focused on the top left group of data points. By the fifth iteration, the k (three) clusters appear more defined. Between the fifth and ninth cluster, only a small number of data samples have changed class membership.

The k-means algorithm used in this work is as described with the exception that the initial cluster centres were not chosen randomly. Choosing the initial centres at random has the potential to produce poor clustering or have a higher number of iterations until they settle at the optimal clusters. The k-means $++$ algorithm [275] was used as an approach to achieve faster and more optimal clustering. This algorithm has a more intelligent approach to choosing the initial cluster locations. The goal of k-means $++$ is to place cluster centres far away from other cluster centres. Initially the first centre chosen at random, the centres following this are iteratively placed to maximise the distance between the centres.

The application of k-means clustering finds cluster centres for each of the Gaussian components. These provide a good starting location for the EM process. Initialising the cluster position using k-means reduces the number of iterations the EM

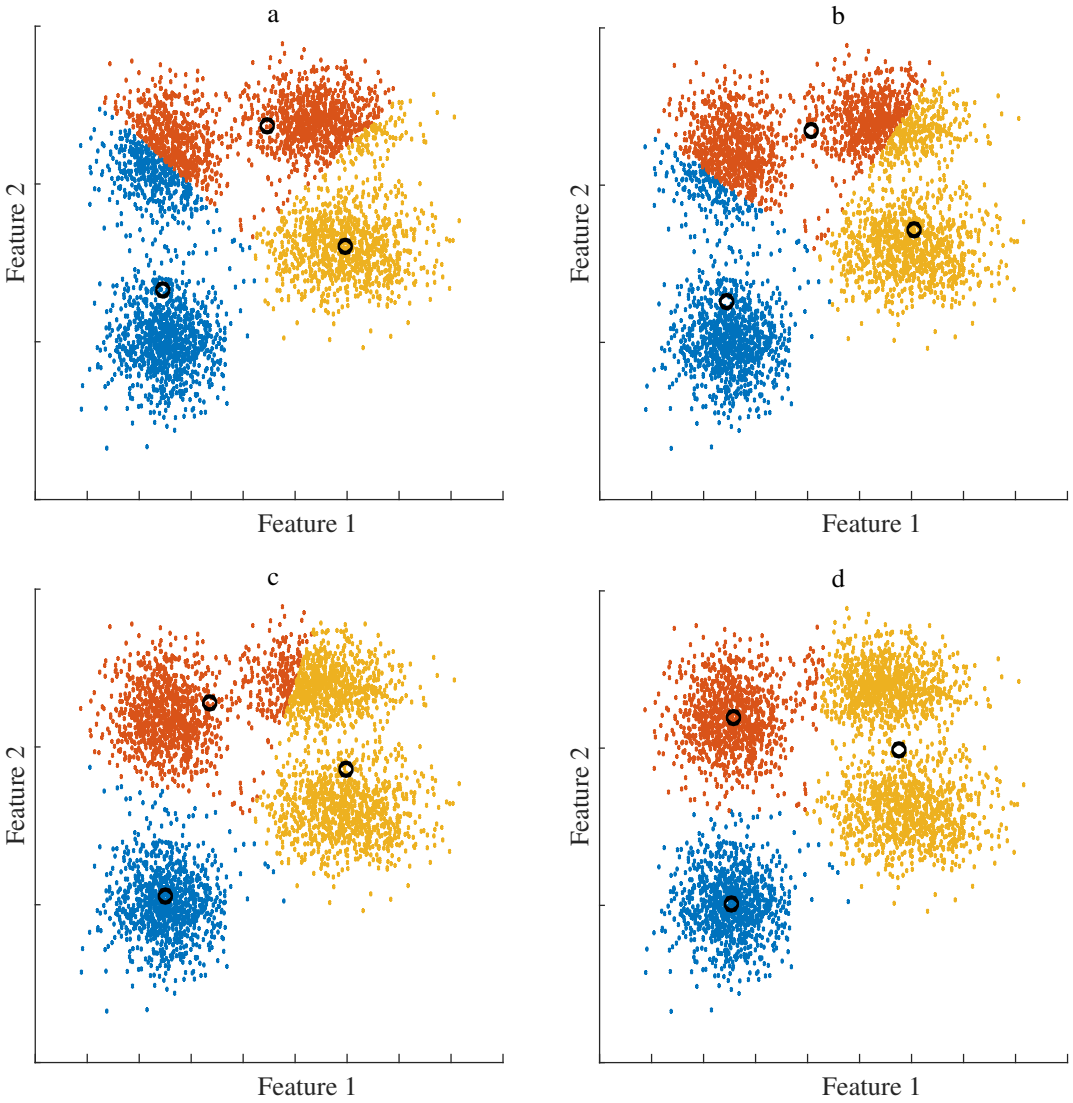


Figure 6.3: kmeans example plot showing the clustering at multiple iterations with the cluster centres highlighted. The data naturally has four clusters but for this example the aim was to locate three cluster centres. (a) Iteration 1, (b) Iteration 3, (c) Iteration 5, (d) Iteration 9.

algorithm needs to tune the GMM parameters to locate the optimal model fit.

6.4.2 Expectation-maximization

Following the k-means clustering, the EM algorithm is used to tune the GMM parameters λ that best fit the training data X_T^c. The EM algorithm is an iterative process in which the parameters are tuned to monotonically increase the fit of the Gaussian components to the training data X_T^c until a convergence threshold, or an iteration limit is reached [269]. This is a two-step process consisting of an expectation step, followed by a maximisation step.

The first step in this process is the expectation step. This step requires the posteriori probability of the data belonging to the components to be computed and is calculated using Equation 6.7.

$$Pr(i|X_T^c(t),\lambda) = \frac{\omega_i g(X_T^c(t)|\mu_i,\Sigma_i)}{\sum_{k=1}^{M}\omega_k g(X_T^c(t)|\mu_k,\Sigma_k)} \tag{6.7}$$

Where this is the posteriori probability for that the feature vector $X_T^c(t)$ belonging to the i^{th} Gaussian component. The next step is the maximisation step, which is used to estimate the updated weights μ_i, means ω_i and covariance σ_i^2 for component i. It estimates these values using the posteriori probability that was found using Equation 6.7. The equations used to calculate these new parameters are given by Equations 6.8, 6.9 and 6.10 for the weights, means and covariances respectively.

$$\omega_i \leftarrow \frac{1}{F_c}\sum_{t=1}^{F_c} Pr(i|X_T^c(t),\lambda) \tag{6.8}$$

$$\mu_i \leftarrow \frac{\sum_{t=1}^{F_c} Pr(i|X_T^c(t),\lambda)X_T^c(t)}{\sum_{t=1}^{F_c} Pr(i|X_T^c(t),\lambda)} \tag{6.9}$$

$$\sigma_i^2 \leftarrow \frac{\sum_{t=1}^{F_c} Pr(i|X_T^c(t),\lambda)X_T^c(t)^2}{\sum_{t=1}^{F_c} Pr(i|X_T^c(t),\lambda)} - \mu_i^2 \tag{6.10}$$

After each iteration, the Gaussian components either match the previous likelihood of the data fitting the Gaussian components or they increase the likelihood of the data fitting the Gaussian components. Figure 6.4 is a simple example of this EM process with sample data. The example in Figure 6.4 did not use the k-means algorithm to find the initial Gaussian parameters. This was to make the EM process that was carried out clearer as using k-means would have found good candidate

cluster means, and the movement from the EM process would not be as obvious in Figure 6.4. The Gaussian components were set to have the same variance, and the means were positioned close to each other as seen from Figure 6.4 (a). All the points to the right of the second Gaussian component will have a higher likelihood of belonging to the second Gaussian. In the first iteration of the EM algorithm, this is noticed, and the means and variances are adjusted. Figure 6.4 (b) shows that there is a larger spread now for the second Gaussian as it tries to model all the points from its original location and the rightmost points. Visually examining the data, it is clear there should be two distinct clusters. At each iteration of the EM algorithm, the means or Gaussian centres will be adjusted to the new best fit location, and this is seen in Figure 6.4 (c) where the Gaussian components have converged to their optimal locations which provide the best representation of the data. Now two distinct Gaussian components capture the data and the cluster centres are highly separated.

The work carried out in this chapter uses the k-means clustering followed by the above described EM process. By applying this technique, models were fit to each of the six different event groups that were described in Chapter 2. When a query is carried out, the log likelihood of the data fitting each of the individual models is evaluated. The model that has the best log likelihood is then chosen as the query event class.

6.5 Performance

This section of the chapter will describe and detail the model selection process and the results obtained from each system. The model selection section will detail the process that was used to find the optimum parameters for each system. Following this, the event class recall accuracy, query time and memory requirements are evaluated in the same manner as the previous chapters. Then a patient independent evaluation is carried out. Each of the classifiers developed in this chapter used the neonatal EEG event database that was described in Chapter 2.

6.5.1 Model selection

The performance evaluation of both the discriminative and generative models was carried out using two versions of LOO cross-validation. The first approach develops the models leaving an event out and then tests the system on the event that was left

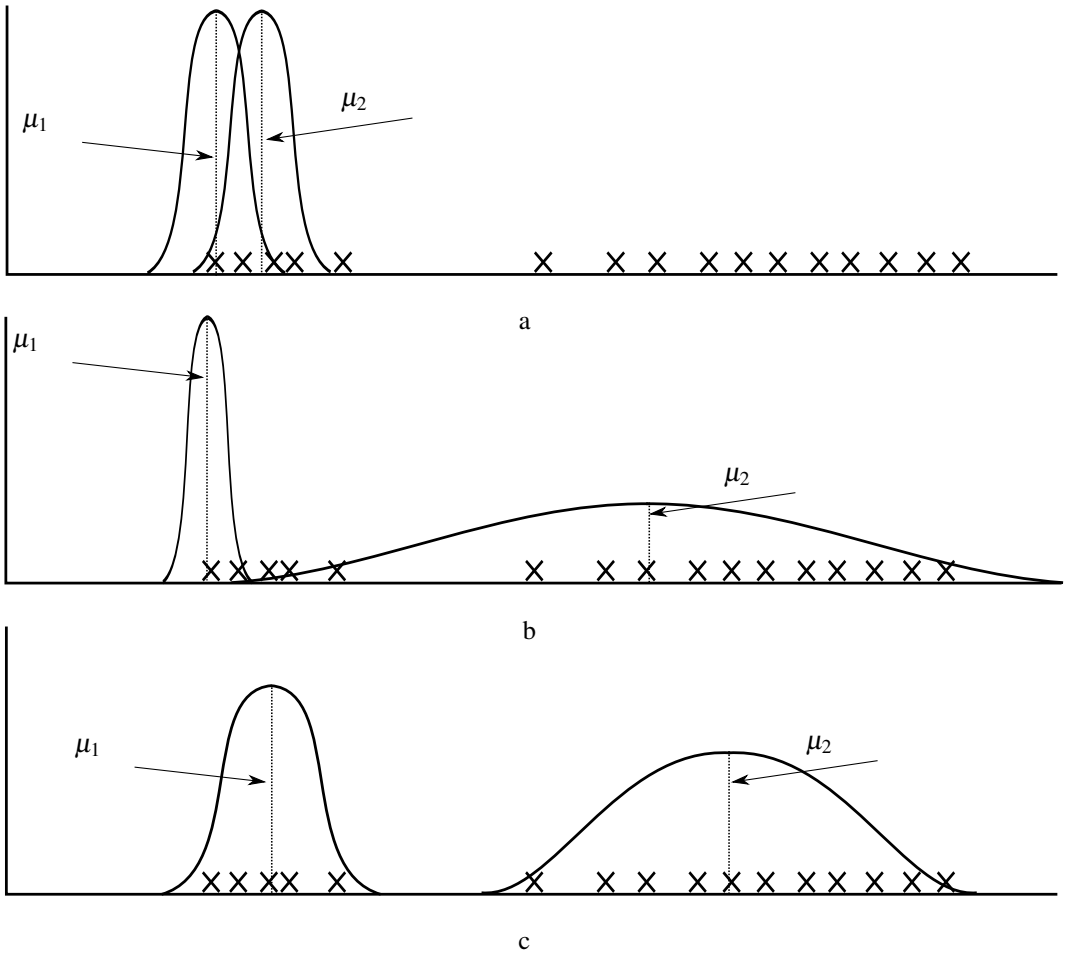


Figure 6.4: EM plot on example data for demonstration. (a) is the initial Gaussian centres positioned close together. (b) is after one iteration of the EM algorithm. (c) is after the algorithm has converged and there now exists a large separation between the Gaussian centres.

out. This approach would be considered as a patient-specific system as the training data still contains events corresponding to the patient being tested. This approach was then repeated for all the events in the database. The event being tested was never included in the database. This evaluation is the same type of evaluation that was carried out on the systems in Chapters 4 and 5. In a real-world application, this validation approach is not practical as it would lead to models that overfit the data and may not be robust to new data. This approach was carried out to have a fair system when comparing the results from the previous two chapters.

The second LOO cross-validation approach leaves all the data from a particular patient out when developing the models, the models are then tested on the patient data that was left out [276]. This approach is considered as a patient independent validation. This process is repeated until each of the patients in the database has been left out from training and used for testing. This is often used when evaluating the performance of classifiers that uses physiological signals from multiple patients as it helps ensure that patient independent results are obtained [60, 80, 171]. The results for the LOO event approach are detailed in Section 6.5.2 and the results for the LOO patient cross-validation are presented in Section 6.5.5.

When used as a multi-class classifier, both the KNN and GMM classifiers produce one output. This output is the event class which has the highest probability. As the classifiers are multi-class classifiers, the results obtained from the systems are converted into a class probability matrix. The event classes from the top K matches from the KNN results are converted into a probability by first computing the weighted top K matches relative to the number of class events in the database using Equation 6.3. The class probability for the KNN is then computed using Equation 6.11,

$$P(C_c|X_q) = \frac{WMV_c}{\sum_{j=1}^{n_c} WMV_j} \tag{6.11}$$

where $P(C_c|X_q)$ is the probability that the event belongs to class c when there are in total n_c classes. It is then repeated for each of the remaining classes to get a probability of the query event belonging to each of the classes. The log-likelihood values for a query event fitting each of the GMMs are converted into class probabilities using Bayes' theorem with the assumption of balanced prior probabilities as given

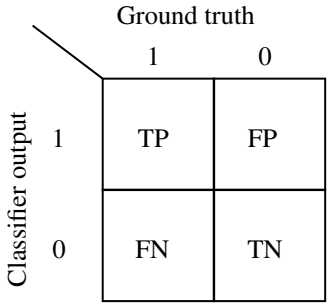


Figure 6.5: Confusion matrix.

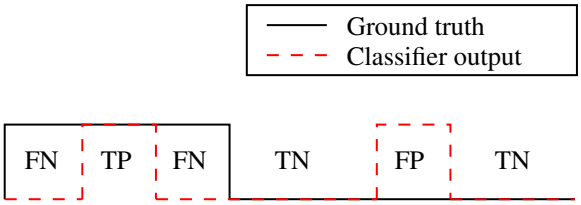


Figure 6.6: Classifier scoring example where the black line represents the ground truth and the red line represents classifier output. The example shows the four possible outcomes which are TP, TN, FP and FN.

by Equation 6.12.

$$P(C_c|X_q) = \frac{L(X_q|\lambda^c)}{\sum_{k=1}^{n_c} L(X_q|\lambda^k)} \tag{6.12}$$

This equation is the posterior probability of the query feature vector X_q belonging to class C_c out of all the n_c possible classes. This is repeated for all remaining classes to get the probability of X_q belonging to each class.

If the class that has the highest probability comes from the correct event class, it is called a True Positive (TP) match. A True Negative (TN) occurs when the classifier output and the label both agree that the event does not belong to the positive class. A False Positive (FP) occurs when the classifier output says the event is positive whereas the label is negative. A False Negative (FN) occurs when the classifier output says the event is false although the label says it is correct. A visual example of this confusion matrix is shown in Figure 6.5. A more graphical representation is shown in Figure 6.6.

As the previous chapters did not contain classifiers, a new metric is introduced that is used to assess how well the classifier models represent the data. The metric used was the area under the curve (AUC) of the receiver operating characteristic curve (ROC). The ROC is used to gauge the performance of a binary classifier

by plotting the sensitivity versus the specificity of the binary classifier [277]. The sensitivity is the proportion of correct class labels that are classified correctly. This measure is represented by Equation 6.13.

$$Sensitivity = \frac{\sum TP}{\sum TP + \sum FN} \tag{6.13}$$

The specificity is a measure of negative class labels that are classified correctly. This measure is represented by Equation 6.14.

$$Specificity = \frac{\sum TN}{\sum TN + \sum FP} \tag{6.14}$$

The ROC graph is generated by varying a decision threshold over the range of sensitivity and specificity values. The AUC is a measure that summarises the ROC into a single value. It is used to determine how well the GMM model suited the data. An AUC value of 1 indicates that the model is well suited to the data as there is a perfect separation between the classes. An AUC of 0.5 indicates that there is no separation between the classes, effectively making the outcome like a random choice [277]

In this work, the ROC and AUC values were computed for the multiple classes by using the "one versus rest" approach to mimic a binary classifier for each of the classes [278]. With this approach, there are six AUC values generated, one AUC value for each class. When creating a ROC for each of the event classes, the event being evaluated is assigned a specific class label such as 1 and all the other five events are assigned the label 0. The ROC threshold is then varied using the probability values from the class probability matrix. Figure 6.7 shows an example plot of ROC curves that each produce different AUC values. In this example, the blue line indicates a perfect AUC of 1, meaning the model is perfectly suited to the data. The red line indicates an AUC value of 0.5, which indicates that the model is poor at deciding the desired output as it is no better than random choice. Both of the green lines indicate an AUC of 0.777. This occurs because when there is a value below 0.5, the inversion of the decision is taken. The goal of using the AUC values is to determine how well the parameters model the data.

From using the leave-one-event-out model selection process described in this section the optimal GMM parameter set was found using 4 Gaussian components, with 94% of the variance retained resulted in the best models. Maintaining 94% variance resulted in a dimensionality reduction from 55 features down to 11 prin-

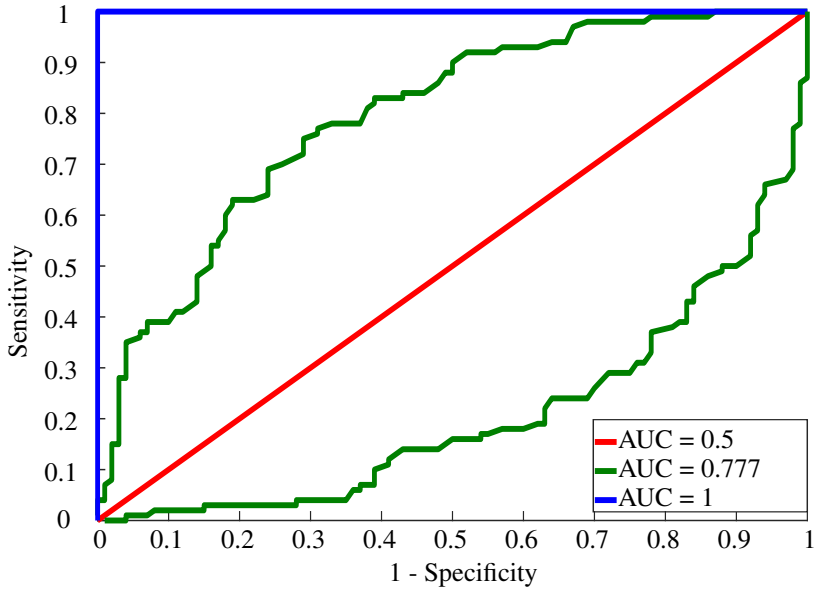


Figure 6.7: ROC and AUC example plot.

cipal components. The values for the GMM were found by running the different variation of the parameters through the model selection process. The parameter set that returned the best AUC values overall were chosen to represent the systems. The procedure of using the AUC was not applied to the KNN as it is not a complex system as it only has a single parameter K.

Table 6.2: AUC results for the GMM classifier

Event type	Event	GMM
Short seizures	2	0.9594
Tracé Alternant	3	0.9974
Long seizure	4	0.9196
Pulsatile artifact	5	0.9818
Respiration artifact	6	≈ 1

The AUC values corresponding to the parameters are presented in Table 6.2. These AUC values can be read in the following way. For example, to evaluate how well the GMM model fits the data for the pulsatile artifact class (class 5), it is seen that the AUC value is 0.9818. This value was generated using the one versus rest technique that was described, where class 5 was assigned the label 1 and all the other classes were assigned the label 0. The AUC values are quite high for each event class, which indicates that the model is suited to the data. The lowest AUC value occurs for the long seizure event, which is 0.9196. The lower AUC value can be explained by the fact that seizures have a lot of variability and can change morphology overtime,

this explains the lower AUC value.

6.5.2 Accuracy

This section will discuss the accuracy of both the KNN and GMM systems. As was presented in Chapters 4 and 5, there were two evaluations carried out. The first is the group majority and the second the performance accuracy. The systems were evaluated on a per event basis as described in their respective sections. The first aspect investigated was the group majority. This test is the event type returned the most while evaluating five event groups. Over half of the database was background data; therefore it is of interest to evaluate if this had much of an impact or bias on the different classes. The group majority results are presented in Table 6.3.

Table 6.3: Classifier group majority results

Event	KNN	GMM
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
Average	100	100

The background data (event 1) is not in the table as discussed previously. The bottom row of Table 6.3 corresponds to the percentage of groups that had the correct group majority. The values in the rows $2-6$ represent the event types that were recalled most frequently when the event type corresponding to the row was queried. The main message taken from Table 6.3 is that both classifiers have the correct event class the majority of the time. This is a positive sign as half of the database consists of background data.

The performance accuracy is the systems ability to correctly recall an event with the same event class as the query event. It is a more detailed look into how well each system and each event class recalls the correct event group. The performance accuracy results for the classifiers are presented in Table 6.4. The bottom row is the average results from each of the event groups. The average performance accuracy for the GMM system is 90.36%.

Overall the results in Table 6.4 show that the GMM approach performs better than the KNN approach. There was only one occasion when the KNN classifier

Table 6.4: Classifier accuracy results

Event	KNN	GMM
2	66.67	83.33
3	100	100
4	60.87	73.91
5	96.36	94.55
6	100	100
Average (%)	84.78	90.36

outperformed the GMM classifier, and this was for the pulsatile artifact (event 5). Both approaches recalled all the TA events (event 3) and respiration events (event 6) correctly. The event that was hardest to recall correctly was the seizure events (event 4), which was an interesting result as the feature set was initially designed for the task of seizure detection. The limited number of seizure events in the database could have been a contributing factor to the accuracy of the seizure events.

These results seem promising, but they were obtained using the LOO event approach. This means they were not patient independent and the database contained other events from the same patient during training. This was carried out to enable a fair evaluation with the results presented in Chapters 4 and 5. A patient independent approach was also examined and these results are presented in Section 6.5.5.

6.5.3 Query run time

The query running time is the time taken to carry out a query on a database. To get an accurate representation of the query time, each system has the time to compute 25 queries taken as opposed to one query. This time to compute 25 queries was repeated 25 times to get an average. The database was increased in size, one event at a time, from an initial three events to a total of 430 events. This process is the same process that was used to evaluate the query time for Chapters 4 and 5. Performing these tests enables an approximate evaluation into how each system will scale as the database is increased in size.

Figure 6.8 shows the query time plots for the KNN and GMM approaches. Two things are clear from this figure. The first is that the KNN query time increases linearly, which was as expected as the KNN approach searches every event in a database and returns the K closest events. This approach is similar to the brute force methods described in Chapter 4, which also increased linearly. The second point is that the GMM query time is approximately constant. The database size should not

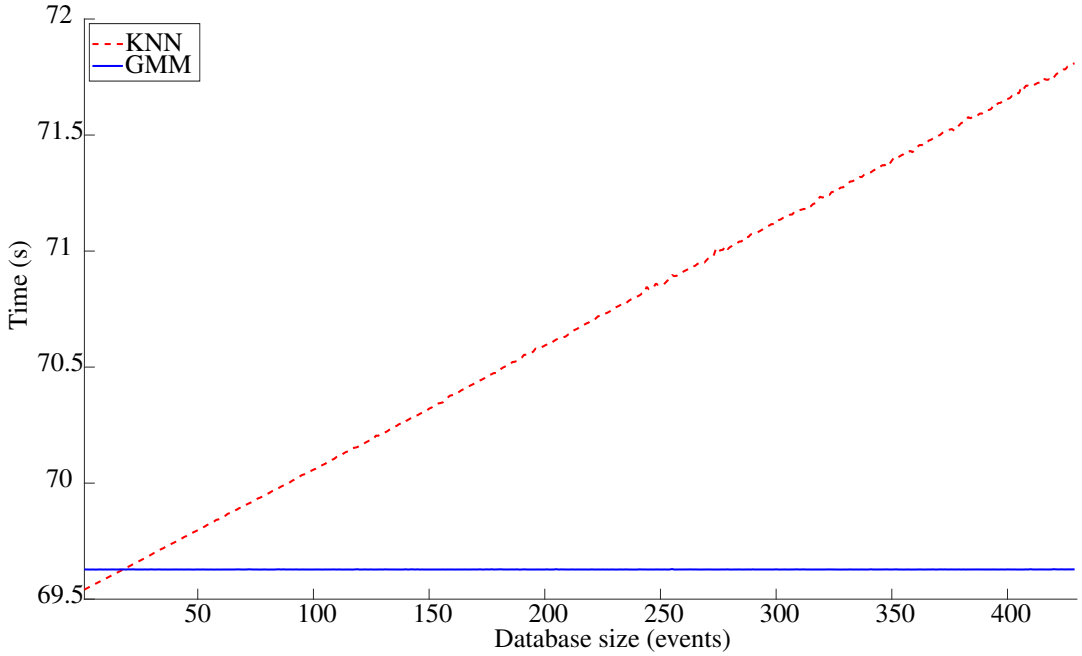


Figure 6.8: Database timing graphs for increasing database size

have an affect on the query time as the models are generated from the training data before carrying out a query.

The next step investigated was the approximate query time when using a database of 10000 events. The KNN and GMM approaches were estimated to take 122.68 and 69.63 seconds respectively and these were approximated by applying linear models to each system. As expected, the GMM approach has a negligible increase, whereas the KNN has a large increase taking 122.68 seconds when there is a database of 10000 events as compared to 69.54 seconds when there were only three events in the database. The 69.54 seconds is due to the extraction of features from the query events.

6.5.4 Memory

When working with large databases on any system, the memory requirements should be investigated to estimate what memory is needed. The memory requirements of the current database were computed by saving the event database as a MATLAB *.mat* format file for the KNN approach. The parameters for the GMM approach were saved as MATLAB *.mat* files. The size of the file was then read into the computer and stored. This method of getting the database size was repeated for a database of one event up until the full database size 430 events and was carried out for each of

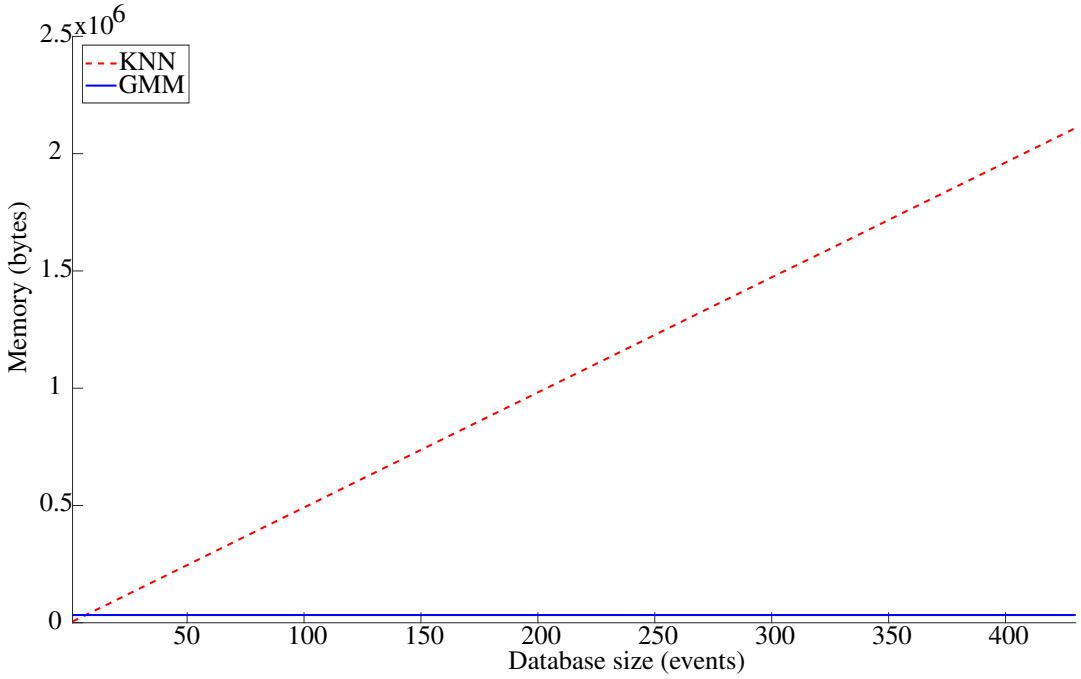


Figure 6.9: Database memory graphs for increasing database size

the systems. The results are plotted in Figure 6.9.

It is clear that the KNN approach has a growing memory requirement, because the KNN uses the direct training database to classify new events. As the training database grows in size, so too does the memory requirements as the training data needs to be stored. This growth in memory is similar to the brute force systems that were evaluated in Chapter 4. Other discriminative methods such as SVMs may require less memory as they only store a small subset of the database (support vectors) which are close to the hyperplane boundaries [279].

The projected memory when there is a database of 10000 events for the KNN is 47 MB and for the GMM it is 32 KB. The projected memory requirement is based on the assumption that the lines in Figure 6.9 are linear. Clearly, from these values, the KNN approach would require significantly more storage than the GMM approach. The KNN approach requires approximately 47 MB, whereas the GMM approach requires only 32 KB. The KNN process needs to store all the training data, whereas the GMM approach only stores the hyperparameters and parameters that describe the distributions of the data.

6.5.5 Patient independent analysis

The amount of available data limits the generalisation performance of the work in this chapter. The respiration and pulsatile artifact events came from 3 and 2 patients respectively. There are 12 short seizure events from 7 patients, with one patient responsible for 4 of the events.

As the amount of data and patients was limited, a patient independent evaluation was not the primary method of model evaluation. The results in Section 6.5.2 were obtained using LOO and leaving a single event out as opposed to leaving one patient out.

Table 6.5: AUC results for KNN and GMM classifiers

Event	GMM
2	0.7366
3	0.9695
4	0.8865
5	0.5109
6	0.9997

As a patient independent system in practice is desired, the parameters for the KNN and GMM systems were tuned and evaluated here using the leave-one-patient-out validation approach which was discussed previously. The K value for the KNN was selected as 128 feature vectors, which is much larger than using the leave-one-event-out approach. This value was found by evaluating the performance as K was increased. The GMM parameters were 2 Gaussian components while keeping a variance of 94%. These parameters were found using the one versus rest and leave-one-patient-out approach. The GMM approaches maintained the same amount of variance while the number of Gaussian components was reduced by two. The AUC values are presented in Table 6.5.

When comparing these AUC values with the AUC values in Table 6.2, it is apparent that the leave-one-patient-out AUC values are lower. A contributing factor to the lower AUC values is because no query patient data was in the training set. These lower AUC values also show that the models trained for the LOO event approaches have overfit the training data. The most significant differences occur for the pulsatile events. The low AUC value was expected as all this data comes from only two patients. This means that the database is halved and the models are developed for a single patient which are overtrained judging by the poor AUC values. The AUC

values for the respiration artifact are the highest even though there are only three patients. The rhythmic nature of the respiration artifact is picked up well by the classifiers. Neonates that are on ventilators would have a more consistent rhythm as it is mechanically controlled.

Once the best classifier parameters were found, the group majority and performance accuracy for each system were evaluated. These results were found using the leave-one-patient-out approach and the results are presented in Tables 6.6 and 6.7.

Table 6.6: Classifier group majority results - leave one patient out

Event	KNN	GMM
2	1	2
3	3	3
4	4	4
5	3	1
6	6	6
Average	60	80

Unlike the results in Table 6.3, the results in Table 6.6 show that both classifiers were not able to get the correct class majority for all the classes that were tested.

The KNN approach got two event classes wrong and the GMM approaches got one class wrong. Both approaches failed to identify the pulsatile artifact. This was expected after observing the AUC results from Table 6.5. The KNN approach also had the incorrect group majority for the short seizure event (event 2). It had picked up the background data as the group majority. A potential reason for this is that over half of each short seizure event was not annotated as seizure. This data could then have been identified as background data when using the KNN classifier as also occurred in the previous two chapters.

Table 6.7: Classifier accuracy results - leave one patient out

Event	KNN	GMM
2	33.33	58.33
3	100	100
4	43.48	56.52
5	29.09	14.55
6	73.08	92.31
Average	55.80	64.34

The recall accuracy for each particular event group along with the overall accuracy was then investigated for the leave-one-patient-out process with the results

presented in Table 6.7. The overall averages are between 55.8% for the KNN and 64.3% for the GMM. The event that had the poorest accuracy across both classifiers was the pulsatile artifact.

It is worth noting that this is not a binary classification problem. Assuming a balanced dataset of six events, with random choice each event has approximately 17% chance of being chosen. Anything better than 17% is considered as better than random. The only test that returned a performance lower than 17% was the pulsatile artifact tests for the GMM classifier. This test had an accuracy of 14.55%. The fact that the pulsatile artifact event group has only two patients reduces the ability of the classifier to generalise over the data, which makes the classifiers overfit to the training data. While overfitting has occurred to this particular classifier the accuracy and AUC values for the other models appear to show they have not overfitted the data as much. There is a strong chance there is still overfitting present in the classifiers, although they do achieve a higher performance accuracy than the pulsatile event.

The main limitation of this work is the amount of available data. If there were more patient event files, leave-one-patient-out training and validation could have been performed to create the generalised model performance. The results presented in Section 6.5.2 are needed for comparison with the results presented in Chapters 4 and 5.

6.6 Summary

In this chapter, the KNN discriminative classifier and the GMM generative classifier were investigated for use as a neonatal EEG multi-class classifier. Each method was discussed with examples given to demonstrate how they operate.

Both of these approaches were evaluated in four different areas, which consisted of the model fit, prediction accuracy, query time and memory requirements. Evaluating these areas enabled a complete picture of the system performance along with the ability to compare the work in this chapter with previous chapters.

The primary finding of this work is that the GMM approach outperforms the KNN approach. Although as seen in Section 6.5.2, there are certain events that the KNN approach classifies better.

One of the main reasons for choosing the KNN and GMM classifiers was because of their ability to add new event classes with a limited amount of data. The KNN can add the data to the training database, whereas the GMM can create a universal

background model (UBM) from the entire database and adapt this UBM to the new data. The amount the UBM is adapted depends on the amount of new event class data available. Further information on the UBM process can be found in [269].

In summary, this work has shown that the GMM classifier performs the best for the task of neonatal multi-class classification. It has the overall highest performance accuracy, lowest query time and required a fixed memory size irrespective of the database size.

Chapter 7

Conclusions and future work

7.1 Introduction

The aim of this thesis was the development of algorithms that assist neurophysiologists in the identification of EEG signal patterns that they have previously encountered. Without the aid of a supporting tools, this task can be very time consuming and involve physically searching through old patient records and books. The primary purpose of a supporting tool is to reduce the search time when locating a particular EEG signal pattern by alleviating the need to physically search through old EEG recording or looking through books such as [4–6].

Chapter 1 introduced and discussed the problem and need for this work. Following this, some prior works carried out on neonatal EEG were discussed. All of these approaches had the primary focus of classifying EEG for different states such as, seizure non-seizure [61], sleep states [40] and HIE grades [46]. None of the prior work focused on finding similar EEG events to a query event. There is a need for this, as some rare conditions may not have enough examples to develop patient independent classifiers.

Chapter 2 discussed the neonatal EEG. The characteristics of normal EEG, most prevalent abnormal EEG and EEG data that was affected by artifacts were discussed with examples given. Finally, the database of neonatal EEG events that were used in this thesis was presented and discussed.

Chapter 3 showed the processing that was applied to the raw signals. Following the processing, the features that were extracted were presented and discussed. These features were initially designed with input from domain experts to describe the neonatal EEG and neonatal seizures in particular [33]. The chapters following these



Figure 7.1: General flowgraph for NN algorithms showing the main stages

were the approaches used to tackle the problem of finding the most similar EEG event from a database of EEG events and will be discussed in more detail in the next section.

7.2 Conclusion

Several steps were needed to tackle the problem described in section 7.1; these steps are illustrated in Figure 7.1. The data is initially preprocessed, as was described in Chapter 3. These preprocessing steps are common to all algorithms. Following this, a compression step is carried out, which varies for each of the approaches, and it transforms the original signal into an alternative representation, such as a feature representation or compressed time-series representation. This compression step was needed as it reduced the amount EEG samples that needed to be stored. As seen in Chapter 4, tests carried out on compressed EEG data had higher recall performance than when carried out directly on the time-series EEG data. The black box in Figure 7.1 represents the algorithm used to find the nearest neighbour (NN). There were a a large number of possible algorithms that could have been explored; however this thesis developed algorithms based on the following three methods:

1. Brute force distance metric search algorithms
2. Hash-based algorithms
3. Scalable multi-class classifiers

These approaches are from three different areas but are generalised by the black box in Figure 7.1. Each of the algorithms in this thesis were analysed looking at three main metrics:

1. Recall accuracy
2. Query time
3. Memory requirements

The recall accuracy was chosen as it is important to evaluate how many correct NN are found for all the event classes being examined. The query time was evaluated as it is important that the NN result is returned as quickly as possible, especially as more events are added to the database. If the latency between the query and returned result is too large then this reduces the practicality of an algorithm. The memory requirements were evaluated, if the algorithms require too much memory, they would not be used as they would be too costly to implement. As the algorithms are intended for use with large databases the query time and memory requirement measures provided insight into how the algorithms would scale as the database increases in size.

The recall accuracy is the most important metric as it is important that the correct event is returned. If an arbitrary event were returned, there would be little confidence in the algorithm to assist in making a correct diagnosis. However, the recall accuracy is irrelevant if there is a substantial increase in either the query time or memory requirements as the database grows in size. The reason for this is that the algorithms developed were intended to be used with large databases.

7.2.1 Brute force distance metric

The brute force distance metric based algorithms were discussed in Chapter 4. These algorithms can be considered as an automated version of a neurophysiologist physically searching through old recordings to find the closest matching pattern. It resembles this as these algorithms search the query event against every other event in the database. Figure 7.2 presents a taxonomy tree of brute force algorithms. Two main brute force algorithms were developed for the fixed point Euclidean distance measure and the elastic dynamic time warping (DTW) measure.

Figure 7.2 shows the two main methods developed which stemmed from the fixed point and elastic approaches. A fixed point approach is where all the corresponding points in the events are directly compared, while an elastic approach has flexibility in the points that are being compared, which helps when signals are misaligned. From Figure 7.2 it is clear that all of the developed methods have linearly increasing query time (QT) and memory requirements (MR) as the database increases in size. It is worth noting that although all the developed algorithms have linearly increasing query time and memory requirements, they increase at different rates for the different systems. For example both the Euclidean and DTW time-series tests increase linearly

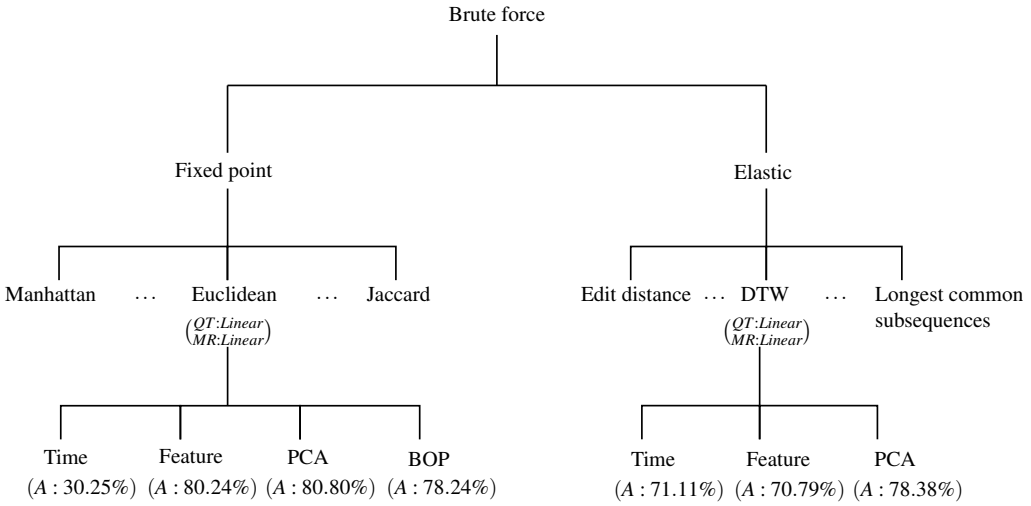


Figure 7.2: Taxonomy tree of the brute force approaches showing the main results for the developed algorithms

in time, but the DTW approach increases at a much faster rate. The accuracy (A) was then reported for each of the developed algorithms.

Applying the Euclidean distance directly to time-series EEG events gave poor accuracy results. These poor results were expected as the EEG is stochastic. Following this, a novel NN recall algorithm was developed that summed the distances between feature vectors that were extracted from windows of the EEG data. As the features were designed with expert domain knowledge; it was thought that this new compact representation would provide a better data representation for event searching. This was reflected in the results. Principal component analysis (PCA) was applied to the features as a dimensionality reduction technique to reduce the amount of data stored and to allow the use of larger databases. The application of PCA provided a minor performance increase but resulted in a large reduction in the amount of memory required. The Euclidean PCA algorithm had the highest accuracy. A further data transformation approach that was applied with the Euclidean distance measure was a technique called bag of patterns (BOP) [78]. This technique used the symbolic aggregate approximation compression technique and then represented the compressed time-series in a BOP representation. The Euclidean distance was used to compute the distance between events. The novel Euclidean BOP algorithm for NN neonatal EEG recall has an accuracy slightly lower than the Euclidean PCA algorithm, although it was both faster and required less memory.

The elastic approach known as DTW was used to develop novel NN neonatal recall algorithms. Three algorithms were developed for the time, feature and PCA

data. DTW was not carried out on the BOP transformed data as the BOP transformation does not maintain the structure or order of the signal. When testing these novel algorithms, it was found that DTW worked better than expected in the time-series domain. However, the Euclidean distance algorithms outperformed the DTW algorithms in the feature and PCA domains as the query time and memory requirements increases slower and had a higher accuracies as seen in Figure 7.2.

The main findings from these algorithms are that the Euclidean BOP algorithm has the lowest increase in query time and memory requirements as the database increases in size. The Euclidean PCA has the highest recall accuracy and the increase in query time and memory requirements as the database grows in not enough to warrant calling the Euclidean BOP algorithm the best algorithm developed in Chapter 4. The Euclidean PCA algorithm is considered as the baseline algorithm as it has the highest accuracy, linear query time and linear memory requirements as seen in Figure 7.2. This algorithm also resembles that of a neurophysiologist physically searching though old records.

7.2.2 Hash based algorithms

The hash-based algorithms were discussed in Chapter 5. This chapter moved past the idea of brute force searching and into a more realistic search space via hashing. Figure 7.3 shows the three main methods developed which were based on spectrogram hashing, feature quantisation and locality sensitive hashing (LSH). From Figure 7.3 it is clear that all of the developed methods had query times and memory requirements that resembles that of power curves as the database increases in size. It is worth noting that although all the developed algorithms have query times and memory requirements that resemble that of power curves, they increase at different rates for the different systems. The accuracy is then reported for each of the developed algorithms.

The first hashing algorithm developed for neonatal EEG recall was the spectrogram hashing algorithm, based on the original algorithm for the popular song recognition method known as Shazam [179]. With the success of Shazam in the audio domain for song recognition, it was decided to adapt this algorithm, to be used for the first time with neonatal EEG. A method of compression was developed to reduce the amount of neonatal EEG energy peaks in the spectrogram. The hashing was then carried out on the compressed spectrogram and the results found that this

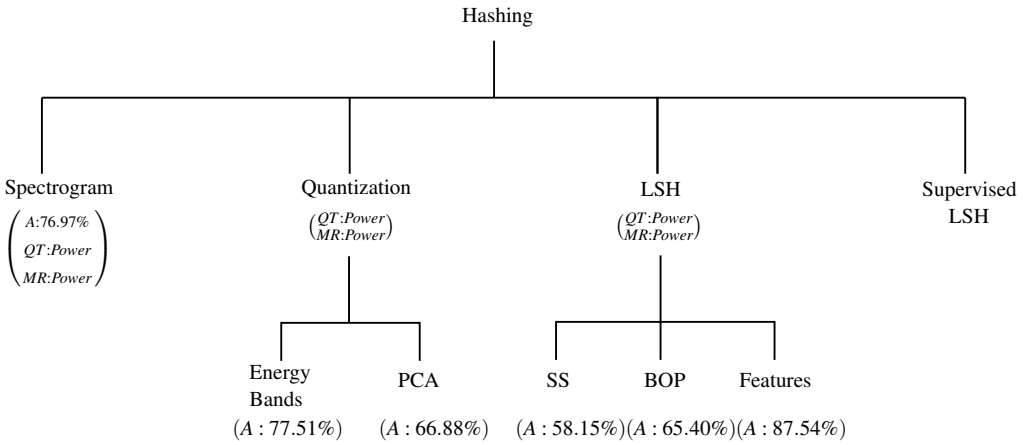


Figure 7.3: Taxonomy tree of hashing approaches showing the different areas where algorithms were developed

algorithm had an accuracy below that of the Euclidean distance PCA algorithm. This algorithm had a slower rate of query time increase as the database was increased in size as compared to the Euclidean PCA algorithm. The memory requirements for this algorithm increase at a faster rate than all of the other hashing algorithms and the Euclidean PCA algorithm.

A second hashing algorithm, again from the audio domain, based on quantisation was investigated [197]. The original audio domain algorithm generated hashes using the energies from different frequency bands across multiple epochs. This approach was adapted for use in the neonatal EEG domain. The novel method of using principal components generated from a neonatal EEG feature set for quantisation was also developed as is seen from the quantisation branch in Figure 7.3. The energy band quantisation algorithm had a higher accuracy than the PCA quantisation algorithm. This performance came at a cost, as the energy band quantisation algorithm approaches that of a brute force algorithm because it needs a bit error rate calculation for every potential match to find the NN. This was reflected in the query time as it increases at a fast rate as the database increases in size. This quantisation algorithm becomes unrealistic and impractical as the Euclidean PCA algorithm is faster and has a greater accuracy. As the quantisation algorithms save binary numbers, the memory requirements are less than that of the Euclidean PCA algorithm. The PCA quantisation algorithm may have a lower accuracy than the Euclidean PCA algorithm, although it also has a much slower rate of query time increase as the database grows and it requires a small amount of memory for the database.

The final hashing algorithms developed and evaluated in Chapter 5 used LSH.

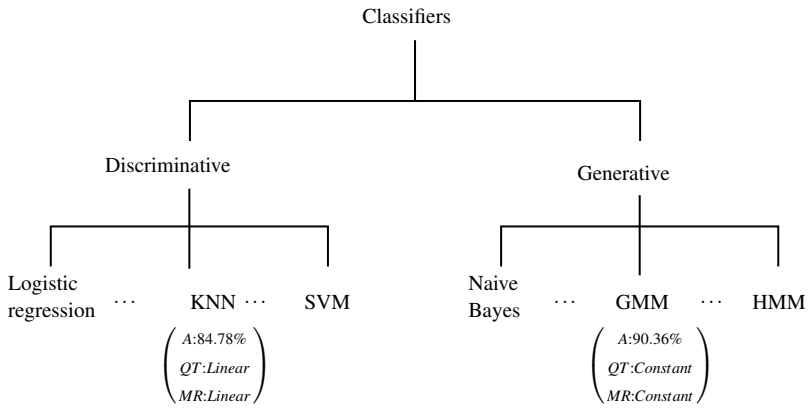


Figure 7.4: Taxonomy tree of classifiers showing the different areas

LSH is an approximate NN search technique and it generates pseudo-random hashes to recall neonatal EEG events. To the best of the author's knowledge, this is the first piece of work that investigates the development of LSH algorithms for neonatal EEG recall. LSH is based on the idea that similar events will have similar hashes [201]. The hashes were generated from a compressed representation of the data. Three compression techniques were examined. The first approach converted the data into a binary representation using sketching and this was followed by computing the weighted set of shingles. The second approach used the BOP compression technique previously used in the brute force NN algorithm. The third compression technique was based on the feature representation of the data. The first two compression techniques resulted in poor accuracy as seen in Figure 7.3, although they scale best in terms of query time. The feature-based LSH algorithm takes slightly longer than the other compression techniques, although it has the best accuracy out of all the hashing algorithms, which is also better than the Euclidean PCA algorithm. The query time and memory curves have shown that that the feature-based LSH algorithm increases at a much slower rate than the Euclidean PCA algorithm. The feature set LSH algorithm is the best performing hashing algorithm for NN neonatal EEG recall. It outperformed the spectrogram and quantisation hashing algorithms in terms of accuracy and query speed.

7.2.3 Scalable multi-class classifiers

Figure 7.4 shows a tree containing the two classifier methods developed in Chapter 6, which were the KNN discriminative classifier [255] and the GMM generative classifier [269]. These methods were developed for the task of neonatal EEG multi-class classification. The KNN approach has linearly increasing query time and memory requirements whereas the GMM approach has a constant query time and a constant memory requirement as seen in Figure 7.4.

The first classifier developed was the KNN classifier. There were two reasons for choosing the KNN classifier. The first was due to the limited database size, which could result in more advanced classifiers having a higher chance of overfitting the training data. As the choice of K is the only parameter that needs to be identified, this approach has a lower chance of overfitting the data. The second reason for choosing the KNN classifier was because if new event classes arise, they can be easily added without the need to fully retrain the classifier. The KNN classifier had an accuracy below that of the LSH hashing algorithm as seen in Figure 7.4. As the KNN algorithm searches all the epochs in the database to find the K nearest neighbours, the query time and memory requirements increase linearly with database size, which is similar to the approaches shown in Figure 7.2. The query time and memory requirements for the KNN algorithm increase at a faster rate than both the Euclidean PCA algorithm and the LSH feature set algorithm as the database increases in size.

The second classifier developed was the GMM classifier. This classifier was chosen as it was previously used successfully with neonatal EEG [27] and there exists the ability to create new event class models with a limited amount of data by adapting a UBM that is trained from all the training data [267]. Leave one out (event/ patient) validation was carried out to find the optimum parameters for the algorithms. This algorithm had the best accuracy out of all the algorithms developed in the thesis. More data is needed to validate and assess the generalisability of each algorithm. The algorithm has negligible query time and memory requirement differences as the database increases in size as seen from the constant value reported in Figure 7.4. Chapter 6 has shown that he GMM algorithm has the lowest memory requirement of all the algorithms developed in this thesis. There is one caveat to using the multi-class classifiers as opposed to any of the other algorithms - the multi-class classifiers can only return an event class whereas the algorithms from Chapters 4 and 5 can

return specific events, as was the primary goal of this thesis. It was understood that the multi-class classifiers would never have the ability to recall specific neonatal EEG events. The multi-class classifiers were implemented as a sanity check to examine if the developed algorithms had a performance comparable to a purpose built multi-class classifier. This sanity check was needed as each of the other algorithms from Chapter 4 and 5 used the multi-class classification problem as a proxy to the NN neonatal EEG event recall problem.

7.2.4 Summary

To summarise, the work in this thesis has found that the feature set LSH based hashing algorithm is the best algorithm for NN neonatal EEG recall. This algorithm has the highest recall accuracy of all the algorithms that return NN events. There was a high level of compression achieved when compressing the time-series representation into a collection of hashes. The LSH hashing algorithms were the quickest and scaled the best in terms of query time. There was an extra query time cost associated with the LSH feature set algorithm as the features needed to be extracted.

In a real world setting, the work developed in this thesis could be used by medical professionals such as neurophysiologists to identify EEG patterns based on previously recorded EEG patterns. The systems can arrange the results in order from the most to least likely match. This would speed up the search time for the medical professionals when trying to identify if and where they have seen a specific pattern previously.

7.3 Future work

The work presented in this thesis, to the author's knowledge was the first work that developed methods for NN neonatal EEG event recall. This thesis served as a basis to evaluate several different potential pathways. All possible avenues could not be covered and this section will discuss some areas where future work could yield performance increases.

The feature set used in this thesis was initially designed with input from domain experts with the main purpose to identify neonatal seizure activity [33]. An investigation into the use of new features could be carried out that would look for more features which describe the characteristics of neonatal EEG. Different sized

feature windows and overlaps could also be investigated. For example, using shorter windows may increase the performance in detecting the pulsatile artifacts as these events are short in duration.

The work carried out in this thesis used PCA as a dimensionality reduction technique. Feature selection techniques could be applied to the original feature as opposed to using PCA. Using feature selection techniques would enable the evaluation of new features and current features and help in the removal of redundant features.

In terms of the NN neonatal EEG recall problem, more research could be carried out in the hashing area. This area has appeared to be the most promising so far with the good results coming from the LSH work. Future research could focus on the development of an amplification strategy to boost the confidence in the results returned, which in turn should reduce the number of candidate events returned [201]. Amplification is a process that uses operators such as AND and OR on the generated hashes to tune the confidence in the matches returned. The features based LSH algorithm yielded the best results out of the hashing schemes tests. The features used for the hashing process could be added to and refined. This thesis investigated three compression methods before the application of LSH, and further compression techniques could be evaluated.

The GMMs were used as they have the potential to model small amounts of data by adapting a background model to the new data [267]. This technique could be investigated for use with new event classes when there are a limited number of events.

The overall GMM performance could be optimised by validating individual parameter sets for each event group as opposed to finding the global best parameter set for all of the classes. Each of the individual event classes may be better represented using a different number of Gaussian components and maintained PCA variance.

As the amount of data available in this thesis was limited, future work would include the further development of the database by increasing the number of patients and types of annotated events in the database. This is particularly applicable for the case of the pulsatile and respiration artifacts. This would enable the creation of training and testing dataset, which would allow for a leave one patient out evaluation to be carried out to show the true generalisability of the algorithms. It would also enable the use of more advanced multi-class classifiers to set a baseline accuracy performance.

All the data in this thesis was annotated using the bipolar montage and therefore restricted the analysis to that montage. A new dataset containing referential montage annotations should be acquired, investigated and compared to the bipolar montage dataset.

The algorithms developed in this thesis were all developed, tested and evaluated using a database that consisted of events that were one minute in duration. To further test the robustness of each algorithm, events that have different durations should be investigated. This would also provide insight into how the algorithms would need to be adapted to handle signals of differing lengths.

Bibliography

[1] CSO, "Vital Statistics Yearly Summary," Tech. Rep. May, 2018.

[2] J. Murphy, A. Nicholson, C. Browne, and G. Turner, "Paediatrics and Neonatology Model of Care for Neonatal Services in Ireland," Tech. Rep. 1, 2015.

[3] E. Sell, F. M. Munoz, A. Soe, M. Wiznitzer, P. T. Heath, E. Clarke, H. Spiegel, D. Sawlwin, M. Šubelj, I. Tikhonov, K. Mohammad, and S. Kochhar, "Neonatal encephalopathy: Case definition & guidelines for data collection, analysis, and presentation of maternal immunisation safety data," *Vaccine*, vol. 35, no. 48, pp. 6501–6505, 2017.

[4] J. Volpe, *Neurology of the Newborn*. Elsevier Health Sciences, 5 ed., 2008.

[5] P. Laoprasert, *Atlas of pediatric EEG*. McGraw Hill Professional, 1 ed., 2010.

[6] J. W. Britton, L. C. Frey, J. L. Hopp, P. Korb, M. Z. Koubeissi, W. E. Lievens, E. M. Pestana-Knight, and E. L. St, *Electroencephalography (EEG): An Introductory Text and Atlas of Normal and Abnormal Findings in Adults, Children, and Infants*. Chicago: American Epilepsy Society, 2016.

[7] R. R. Clancy, "Prolonged Electroencephalogram Monitoring for Seizures and Their Treatment," *Clinics in Perinatology*, vol. 33, no. 3, pp. 649–665, 2006.

[8] D. M. Murray, G. B. Boylan, I. Ali, C. A. Ryan, B. P. Murphy, and S. Connolly, "Defining the gap between electrographic seizure burden, clinical expression and staff recognition of neonatal seizures," *Archives of Disease in Childhood: Fetal and Neonatal Edition*, vol. 93, no. 3, pp. 187–191, 2008.

[9] J. M. Rennie, L. S. De Vries, M. Blennow, A. Foran, D. K. Shah, V. Livingstone, A. C. Van Huffelen, S. R. Mathieson, E. Pavlidis, L. C. Weeke, M. C.

Toet, M. Finder, R. M. Pinnamaneni, D. M. Murray, A. C. Ryan, W. P. Marnane, and G. B. Boylan, "Characterisation of neonatal seizures and their treatment using continuous EEG monitoring: A multicentre experience," *Archives of Disease in Childhood-Fetal and Neonatal Edition*, 2018.

[10] E. Low, S. R. Mathieson, N. J. Stevenson, V. Livingstone, C. A. Ryan, C. O. Bogue, J. M. Rennie, and G. B. Boylan, "Early postnatal EEG features of perinatal arterial ischaemic stroke with seizures," *PLoS ONE*, vol. 9, no. 7, 2014.

[11] J. C. Beal, K. Cherian, and S. L. Moshe, "Early-onset epileptic encephalopathies: Ohtahara syndrome and early myoclonic encephalopathy," *Pediatric Neurology*, vol. 47, no. 5, pp. 317–323, 2012.

[12] J. A. Tibbies, "Dominant Benign Neonatal Seizures," *Developmental Medicine & Child Neurology*, vol. 22, no. 5, pp. 664–667, 1980.

[13] M. I. Shevell, D. B. Sinclair, and K. Metrakos, "Benign familial neonatal seizures: clinical and electroencephalographic characteristics," *Pediatric neurology*, vol. 2, no. 5, pp. 272–275, 1986.

[14] M. A. Mikati, E. Trevathan, K. S. Krishnamoorthy, and C. T. Lombroso, "Pyridoxine-dependent epilepsy: EEG investigations and long-term follow-up," *Electroencephalography and Clinical Neurophysiology*, vol. 78, no. 3, pp. 215–221, 1991.

[15] W. B. Dobyns and C. L. Truwi, "Review article Lissencephaly and Other Malformations of Cortical Development : 1995 Update," *Neuropediatrics*, vol. 26, no. 3, pp. 132–147, 1995.

[16] D. H. Saltzman, C. M. Krauss, J. M. Goldman, and B. R. Benacerraf, "Prenatal diagnosis of lissencephaly," *Prenatal Diagnosis*, vol. 11, no. 3, pp. 139–143, 1991.

[17] S. Menascu, A. Weinstock, O. Farooq, H. Hoffman, and M. A. Cortez, "EEG and neuroimaging correlations in children with lissencephaly," *Seizure*, vol. 22, no. 3, pp. 189–193, 2013.

[18] J. Berger, F. Dorninger, S. Forss-Petter, and M. Kunze, "Peroxisomes in brain development and function," *Biochimica et Biophysica Acta - Molecular Cell Research*, vol. 1863, no. 5, pp. 934–955, 2016.

[19] D. P. Panjan, N. P. Meglič, and D. Neubauer, "A Case of Zellweger Syndrome with Extensive MRI Abnormalities and Unusual EEG Findings," *Clinical Electroencephalography*, vol. 32, no. 1, pp. 28–31, 2001.

[20] S. S. Aziz, S. J. Wallace, J. F. Murphy, C. P. Sainsbury, and O. P. Gray, "Cot-side EEG monitoring using computerised spectral analysis," *Archives of Disease in Childhood*, vol. 61, no. 3, pp. 242–246, 1986.

[21] A. Liu, J. S. Hahn, G. P. Heldt, and R. W. Coen, "Detection of neonatal seizures through computerized EEG analysis," *Electroencephalography and Clinical Neurophysiology*, vol. 82, no. 1, pp. 30–37, 1992.

[22] J. Gotman, D. Flanagan, J. Zhang, and B. Rosenblatt, "Automatic seizure detection in the newborn: Methods and initial evaluation," *Electroencephalography and Clinical Neurophysiology*, vol. 103, no. 3, pp. 356–362, 1997.

[23] P. Celka and P. Colditz, "A computer-aided detection of EEG seizures in infants: A singular-spectrum approach and performance comparison," *IEEE Transactions on Biomedical Engineering*, vol. 49, no. 5, pp. 455–462, 2002.

[24] A. Barron, J. Rissanen, and B. Yu, "The minimum description length principle in coding and modeling," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2743–2760, 1998.

[25] A. O'Shea, G. Lightbody, G. Boylan, and A. Temko, "Neonatal seizure detection using convolutional neural networks," in *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing*, pp. 1–6, 2017.

[26] A. H. Ansari, P. J. Cherian, A. Caicedo, G. Naulaers, M. De Vos, and S. Van Huffel, "Neonatal Seizure Detection Using Deep Convolutional Neural Networks," *International Journal of Neural Systems*, vol. 28, 2018.

[27] E. M. Thomas, A. Temko, G. Lightbody, W. P. Marnane, and G. B. Boylan, "Gaussian mixture models for classification of neonatal seizures using EEG," *Physiological Measurement*, vol. 31, no. 7, pp. 1047–1064, 2010.

[28] M. H. Abdullah, J. M. Abdullah, and M. Z. Abdullah, "Seizure detection by means of hidden Markov model and stationary wavelet transform of electroencephalograph signals," in *International Conference on Biomedical and Health Informatics: Global Grand Challenge of Health Informatics*, pp. 62–65, IEEE, 2012.

[29] A. Temko, E. Thomas, W. Marnane, G. Lightbody, and G. Boylan, "EEG-based neonatal seizure detection with Support Vector Machines," *Clinical Neurophysiology*, vol. 122, no. 3, pp. 464–473, 2011.

[30] C. Donos, D. Matthias, and A. Schulze-Bonhage, "Early Seizure Detection Algorithm Based on Intracranial EEG and Random Forest Classification," *International journal of neural systems*, vol. 25, no. 5, 2015.

[31] S. R. Mathieson, N. J. Stevenson, E. Low, W. P. Marnane, J. M. Rennie, A. Temko, G. Lightbody, and G. B. Boylan, "Validation of an automated seizure detection algorithm for term neonates," *Clinical Neurophysiology*, vol. 127, no. 1, pp. 156–168, 2016.

[32] *ANSeR study. http://anserstudy.com/ Date Accessed: 2019-09-21*.

[33] B. R. Greene, S. Faul, W. P. Marnane, G. Lightbody, I. Korotchikova, and G. B. Boylan, "A comparison of quantitative EEG features for neonatal seizure detection," *Clinical Neurophysiology*, vol. 119, no. 6, pp. 1248–1261, 2008.

[34] M. S. Scher, "Automated EEG-sleep analyses and neonatal neurointensive care," *Sleep Medicine*, vol. 5, no. 6, pp. 533–540, 2004.

[35] V. Gerla, K. Paul, L. Lhotska, and V. Krajca, "Multivariate analysis of full-term neonatal polysomnographic data," *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, no. 1, pp. 104–110, 2009.

[36] A. Piryatinska, G. Terdik, W. A. Woyczynski, K. A. Loparo, M. S. Scher, and A. Zlotnik, "Automated detection of neonate EEG sleep stages," *Computer methods and programs in biomedicine*, vol. 95, no. 1, pp. 31–46, 2009.

[37] I. Korotchikova, N. J. Stevenson, V. Livingstone, C. A. Ryan, and G. B. Boylan, "Sleep-wake cycle of the healthy term newborn infant in the immediate postnatal period," *Clinical Neurophysiology*, vol. 127, no. 4, pp. 2095–2101, 2016.

[38] J. P. Turnbull, K. A. Loparo, M. W. Johnson, and M. S. Scher, "Automated detection of tracé alternant during sleep in healthy full-term neonates using discrete wavelet transform," *Clinical Neurophysiology*, vol. 112, no. 10, pp. 1893–1900, 2001.

[39] M. Carrozzi, A. Accardo, and F. Bouquet, "Analysis of sleep-stage characteristics in full-term newborns by means of spectral and fractal parameters," *Sleep*, vol. 27, no. 7, pp. 1384–1393, 2004.

[40] N. Koolen, L. Oberdorfer, Z. Rona, V. Giordano, T. Werther, K. Klebermass-Schrehof, N. Stevenson, and S. Vanhatalo, "Automated classification of neonatal sleep states using EEG," *Clinical Neurophysiology*, vol. 128, no. 6, pp. 1100–1108, 2017.

[41] K. Pillay, A. Dereymaeker, K. Jansen, G. Naulaers, S. Van Huffel, and M. De Vos, "Automated EEG sleep staging in the term-age baby using a generative modelling approach," *Journal of Neural Engineering*, vol. 15, no. 3, 2018.

[42] D. M. Murray, G. B. Boylan, C. A. Ryan, and S. Connolly, "Early EEG Findings in Hypoxic-Ischemic Encephalopathy Predict Outcomes at 2 Years," *Pediatrics*, vol. 124, no. 3, pp. 459–467, 2009.

[43] M. Toet, L. Hellström-Westas, F. Groenendaal, P. Eken, and L. De Vries, "Amplitude integrated EEG 3 and 6 hours after birth in full term neonates with hypoxic – ischaemic encephalopathy," *Archives of Disease in Childhood-Fetal and Neonatal Edition*, vol. 81, no. 1, pp. 19–23, 1999.

[44] I. Korotchikova, N. J. Stevenson, B. H. Walsh, D. M. Murray, and G. B. Boylan, "Clinical Neurophysiology Quantitative EEG analysis in neonatal hypoxic ischaemic encephalopathy," *Clinical Neurophysiology*, vol. 122, no. 8, pp. 1671–1678, 2011.

[45] N. Stevenson, I. Korotchikova, A. Temko, G. Lightbody, W. Marnane, and G. Boylan, "An automated system for grading EEG abnormality in term neonates with hypoxic-ischaemic encephalopathy," *Annals of biomedical engineering*, vol. 41, no. 4, pp. 775–785, 2013.

[46] R. Ahmed, A. Temko, W. Marnane, G. Lightbody, and G. Boylan, "Grading hypoxic-ischemic encephalopathy severity in neonatal EEG using GMM

supervectors and the support vector machine," *Clinical Neurophysiology*, vol. 127, no. 1, pp. 297–309, 2016.

[47] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," *Knowledge and information systems*, vol. 7, pp. 358–386, 2005.

[48] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. J. Keogh, "Querying and mining of time series data: experimental comparison of representations and distance measures," *Proceedings of the Very Large Database Endowment*, vol. 1, no. 2, pp. 1542–1552, 2008.

[49] C. Luo and A. Shrivastava, "SSH (Sketch, Shingle, & Hash) for Indexing Massive-Scale Time Series," in *Proceedings of the Time Series Workshop at NIPS 2016*, vol. 55, pp. 38–58, 2017.

[50] Y. Bryce Kim and U. M. O'Reilly, "Large-scale physiological waveform retrieval via locality-sensitive hashing," in *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, (Milan, Italy), pp. 5829–5833, IEEE, 2015.

[51] D. C. Kale, D. Gong, Z. Che, Y. Liu, G. Medioni, R. Wetzel, and P. Ross, "An Examination of Multivariate Time Series Hashing with Applications to Health Care," in *2014 IEEE International Conference on Data Mining*, (Shenzhen, China), pp. 260–269, IEEE, 2014.

[52] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, (Beijing, China), pp. 262 – 270, ACM, 2012.

[53] B. M. Murphy, G. B. Boylan, G. Lightbody, and W. P. Marnane, "Distance Metric Approach for Nearest Neighbour Recall of Neonatal EEG," in *2018 29th Irish Signals and Systems Conference*, (Belfast), pp. 1–6, IEEE, 2018.

[54] B. M. Murphy, G. B. Boylan, G. Lightbody, and W. P. Marnane, "Bag of Patterns for Nearest Neighbour Neonatal EEG Recall," *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 4558–4561, 2019.

[55] B. M. Murphy, C. O'Driscoll, I. Korotchikova, G. B. Boylan, G. Lightbody, and W. P. Marnane, "Application of audio fingerprinting to Neonatal EEG," in *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 912–915, IEEE, 2016.

[56] B. M. Murphy, G. B. Boylan, G. Lightbody, and W. P. Marnane, "An Approximate Nearest Neighbour System for Neonatal EEG Recall," *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 2018-July, pp. 283–286, 2018.

[57] S. Ackerman, *Discovering the Brain*. Washington: National Academies Press, 1992.

[58] E. Niedermeyer and F. L. da Silva, *Electroencephalography: Basic Principles, Clinical Applications, and Related Fields*. Lippincott Williams Wilkins, 2005.

[59] M. De Vos, W. Deburchgraeve, P. Cherian, V. Matic, R. Swarte, P. Govaert, G. Visser, and S. Van Huffel, "Automated artifact removal as preprocessing refines neonatal seizure detection," *Clinical Neurophysiology*, vol. 122, no. 12, pp. 2345–2354, 2011.

[60] K. T. Tapani, S. Vanhatalo, and N. J. Stevenson, "Time-Varying EEG Correlations Improve Automated Neonatal Seizure Detection," *International Journal of Neural Systems*, vol. 28, 2018.

[61] S. Faul, G. Boylan, S. Connolly, L. Marnane, and G. Lightbody, "An evaluation of automated neonatal seizure detection methods," *Clinical Neurophysiology*, vol. 116, no. 7, pp. 1533–1541, 2005.

[62] H. H. Jasper, "The ten-twenty electrode system of the International Federation," *Electroencephalography and Clinical Neurophysiology*, vol. 10, pp. 370–375, 1958.

[63] K. Tufenkjian, "EEG Instrumentation, Montage, Polarity, and Localization," 2017.

[64] J. M. Rennie, C. F. Hagmann, and N. J. Robertson, *Neonatal cerebral investigation*. Cambridge University Press, 2008.

[65] T. N. Tsuchida, C. J. Wusthoff, R. A. Shellhaas, N. S. Abend, C. D. Hahn, J. E. Sullivan, S. Nguyen, S. Weinstein, M. S. Scher, J. J. Riviello, and R. R. Clancy, "American clinical neurophysiology society standardized EEG terminology and categorization for the description of continuous eeg monitoring in neonates: Report of the american clinical neurophysiology society critical care monitoring committee," *Journal of Clinical Neurophysiology*, vol. 30, no. 2, pp. 161–173, 2013.

[66] J. Kohyama, "Sleep as a window on the developing brain," *Current Problems in Pediatrics*, vol. 28, no. 3, pp. 73–92, 1998.

[67] J. Rennie and G. Boylan, "Treatment of neonatal seizures," *Archives of Disease in Childhood-Fetal and Neonatal Edition*, vol. 92, no. 2, pp. 148 – 150, 2007.

[68] J. Shen, S. Mao, Z. Yuan, Y. Yu, Z. Xia, and F. Gao, "Different Outcome with Different Aetiologies : The Prognosis Follow-up in 13 Infants with Burst-suppression Pattern," *HK J Paediatr (new series)*, vol. 21, no. 1, pp. 7–13, 2016.

[69] J. Löfhede, M. Thordstein, N. Löfgren, A. Flisberg, M. Rosa-Zurera, I. Kjellmer, and K. Lindecrantz, "Automatic classification of background EEG activity in healthy and sick neonates," *Journal of Neural Engineering*, vol. 7, no. 1, 2010.

[70] L. M. Douglass, J. Y. Wu, N. P. Rosman, and C. E. Stafstrom, "Burst suppression electroencephalogram pattern in the newborn: Predicting the outcome," *Journal of Child Neurology*, vol. 17, no. 6, pp. 403–408, 2002.

[71] D. Schomer and F. da Silva, *Niedermeyer's Electroencephalography: Basic Principles, Clinical Applications, and Related Fields*. Lippincott Williams & Wilkins, 2012, 2012.

[72] A. Temko, G. Boylan, W. Marnane, and G. Lightbody, "Robust Neonatal Eeg Seizure Detection Through Adaptive Background Modeling," *International Journal of Neural Systems*, vol. 23, no. 04, 2013.

[73] T. Balli and R. Palaniappan, "Classification of biological signals using linear and nonlinear features," *Physiological Measurement*, vol. 31, no. 7, pp. 903–920, 2010.

[74] P. Zarjam, J. Epps, and F. Chen, "Spectral EEG features for evaluating cognitive load," in *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 3841–3844, IEEE, 2011.

[75] P. J. Cherian, W. Deburchgraeve, R. M. Swarte, M. De Vos, P. Govaert, S. Van Huffel, and G. H. Visser, "Validation of a new automated neonatal seizure detection system: A clinician's perspective," *Clinical Neurophysiology*, vol. 122, no. 8, pp. 1490–1499, 2011.

[76] S. D. Muthukumaraswamy, "High-frequency brain activity and muscle artifacts in MEG/EEG: a review and recommendations," *Frontiers in Human Neuroscience*, vol. 7, no. 138, pp. 1–11, 2013.

[77] R. G. Lyons, *Understanding Digital Signal Processing*. Pearson Education, 3 ed., 2010.

[78] J. Lin and Y. Li, "Finding structural similarity in time series data using bag-of-patterns representation," in *International Conference on Scientific and Statistical Database Managemen*, pp. 461–477, Springer, 2009.

[79] A. Bhattacharyya, Sourya and Biswas, J. Mukherjee, A. K. Majumdar, B. Majumdar, S. Mukherjee, and A. K. Singh, "Detection of artifacts from high energy bursts in neonatal EEG," *Computers in Biology and Medicine*, vol. 43, no. 11, pp. 1804–1814, 2013.

[80] S. O'Regan, S. Faul, and W. Marnane, "Automatic detection of EEG artefacts arising from head movements using EEG and gyroscope signals," *Medical Engineering and Physics*, vol. 35, no. 7, pp. 867–874, 2013.

[81] N. S. Abend and C. J. Wusthoff, "Neonatal seizures and status epilepticus," *Journal of clinical neurophysiology : official publication of the American Electroencephalographic Society*, vol. 29, no. 5, pp. 441–448, 2012.

[82] R. Esteller, J. Echauz, T. Tcheng, B. Litt, and B. Pless, "Line length: an efficient feature for seizure onset detection," in *Proceedings of the International*

Conference of the IEEE Engineering in Medicine and Biology Society, vol. 2, (Istanbul, Turkey), pp. 1707–1710, IEEE, 2001.

[83] K. Patel, C.-P. Chua, S. Faul, and C. J. Bleakley, "Low power real-time seizure detection for ambulatory EEG," in *3rd International Conference on Pervasive Computing Technologies for Healthcare*, pp. 1–7, IEEE, 2009.

[84] R. G. Lyons, *Understanding Digital Signal Processing*. Pearson Education, 2 ed., 2012.

[85] P. Z. Peebles, *Probability, random variables, and random signal principles*. McGraw-Hill, 3 ed., 2001.

[86] M. R. Spiegel and L. J. Stephens, *Schaum's outline of statistics*, vol. 4. McGraw Hill Professional, 2017.

[87] P. H. Westfall, "Kurtosis as Peakedness, 1905–2014. R.I.P.," *The American Statistician*, vol. 68, no. 3, pp. 191–195, 2014.

[88] B. Hjorth, "EEG analysis based on time domain properties," *Electroencephalography and clinical neurophysiology*, vol. 29, no. 3, pp. 306–310, 1970.

[89] M. J. Van Putten, T. Kind, F. Visser, and V. Lagerburg, "Detecting temporal lobe seizures from scalp EEG recordings: A comparison of various features," *Clinical Neurophysiology*, vol. 116, no. 10, pp. 2480–2489, 2005.

[90] J. F. Kaiser, "On a simple algorithm to calculate the 'energy' of a signal," in *International Conference on Acoustics, Speech, and Signal Processing*, pp. 381–384, IEEE, 1990.

[91] M. D'Alessandro, R. Esteller, J. Echauz, G. Vachtsevanos, A. Hinson, and B. Litt, "Epileptic Seizure Prediction Using Hybrid Feature Selection Over Multiple Intracranial EEG Electrode Contacts: A Report of Four Patients," *IEEE Transactions on Biomedical Engineering*, vol. 50, no. 5, pp. 603–615, 2003.

[92] P. J. Brockwell and R. A. Davis, *Introduction to Time Series and Forecasting*. Springer, 2 ed., 2002.

[93] G. Mohammadi, P. Shoushtari, B. Molaee Ardekani, and M. B. Shamsollahi, "Person Identification by Using AR Model for EEG Signals," in *Proceeding of World Academy of Science, Engineering and Technology*, vol. 11, pp. 281–285, 2006.

[94] N. B. Karayiannis, A. Mukherjee, J. R. Glover, P. Y. Ktonas, J. D. Frost, R. A. Hrachovy, and E. M. Mizrahi, "Detection of pseudosinusoidal epileptic seizure segments in the neonatal EEG by cascading a rule-based algorithm with a neural network," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 4, pp. 633–641, 2006.

[95] A. Shoeb, H. Edwards, J. Connolly, B. Bourgeois, S. Ted Treves, and J. Guttag, "Patient-specific seizure onset detection," *Epilepsy and Behavior*, vol. 5, no. 4, pp. 483–498, 2004.

[96] M. Fani, G. Azemi, and B. Boashash, "EEG-based automatic epilepsy diagnosis using the instantaneous frequency with sub-band energies," in *7th International Workshop on Systems, Signal Processing and their Applications*, pp. 187–190, IEEE, 2011.

[97] H. Qu and J. Gotman, "A patient-specific algorithm for the detection of seizure onset in long-term EEG monitoring: Possible use as a warning device," *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 2, pp. 115–122, 1997.

[98] M. Smit, J. A. Dawson, A. Ganzeboom, S. B. Hooper, J. Van Roosmalen, and A. B. Te Pas, "Pulse oximetry in newborns with delayed cord clamping and immediate skin-to-skin contact," *Archives of Disease in Childhood: Fetal and Neonatal Edition*, vol. 99, pp. 309–314, 2014.

[99] I. Rampil, F. Sasse, N. T. Smith, B. Hoff, and D. Flemming, "Spectral edge frequency - a new correlate of anesthetic depth," *Anesthesiology: The Journal of the American Society of Anesthesiologists*, vol. 53, no. 3, 1980.

[100] C. E. Shannon, "A mathematical theory of communication," *Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[101] S. J. Roberts, W. Penny, and I. Rezek, "Temporal and spatial complexity measures for electroencephalogram based brain-computer interfacing," *Medical and Biological Engineering and Computing*, vol. 37, no. 1, pp. 93–98, 1999.

[102] R. A. Fisher, "Theory of Statistical Estimation," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 22, pp. 700–725, Cambridge University Press, 1925.

[103] C. J. James and D. Lowe, "Extracting multisource brain activity from a single electromagnetic channel," *Artificial Intelligence in Medicine*, vol. 28, no. 1, pp. 89–104, 2003.

[104] A. Vakkuri, A. Yli-Hankala, S. Talja, P and Mustola, H. Tolvanen-Laakso, and H. Sampson, Tl and ViertiÖ-Oja, "Time-frequency balanced spectral entropy as a measure of anesthetic drug effect in central nervous system during sevoflurane, propofol, and thiopental anesthesia," *Acta Anaesthesiologica Scandinavica*, vol. 48, no. 2, pp. 145–153, 2004.

[105] S. O'Regan and W. Marnane, "Multimodal detection of head-movement artefacts in EEG," *Journal of Neuroscience Methods*, vol. 218, no. 1, pp. 110–120, 2013.

[106] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[107] K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[108] L. I. Smith, "A Tutorial on Principal Component Analysis," tech. rep., 2002.

[109] B. Czigler, D. Csikós, Z. Hidasi, Z. Anna Gaál, É. Csibri, É. Kiss, P. Salacz, and M. Molnár, "Quantitative EEG in early Alzheimer's disease patients - Power spectrum and complexity features," *International Journal of Psychophysiology*, vol. 68, no. 1, pp. 75–80, 2008.

[110] M. J. Berryman, S. W. Coussens, Y. Pamula, D. Kennedy, K. Lushington, C. Shalizi, A. Allison, A. J. Martin, D. Saint, and D. Abbott, "Nonlinear aspects of the EEG during sleep in children," in *Fluctuations and Noise in*

Biological, Biophysical, and Biomedical Systems III, pp. 40–49, International Society for Optics and Photonics, 2005.

[111] Z. Huang, E. Ayday, J. Fellay, J. P. Hubaux, and A. Juels, "GenoGuard: Protecting genomic data against brute-force attacks," in *Proceedings - IEEE Symposium on Security and Privacy*, pp. 447–462, IEEE, 2015.

[112] M. Sutton, A. Greene, and P. Amini, *Fuzzing Brute Force Vulnerability Discovery*. Pearson Education, 1 ed., 2007.

[113] A. Mueen and N. Chavoshi, "Enumeration of time series motifs of all lengths," *Knowledge and Information Systems*, vol. 45, no. 1, pp. 105–132, 2015.

[114] T. Shajina and P. B. Sivakumar, "Human gait recognition and classification using time series Shapelets," in *Proceedings - International Conference on Advances in Computing and Communications*, pp. 31–34, IEEE, 2012.

[115] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI magazine*, vol. 17, no. 3, pp. 37–54, 1996.

[116] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2 ed., 2011.

[117] G. Fiol-Roig, M. Miró-Juliá, and A. P. Isern-Deyá, "Applying data mining techniques to stock market analysis," in *8th International Conference on Practical Applications of Agents and Multiagent Systems*, pp. 519–527, Springer, 2010.

[118] M. A. Hearst, "Untangling Text Data Mining," in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pp. 3–10, Association for Computational Linguistics, 1999.

[119] E. W. Ngai, Y. Hu, Y. Wong, Y. Chen, and X. Sun, "The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature," *Decision Support Systems*, vol. 50, no. 3, pp. 559–569, 2011.

[120] T. Li, M. Ogihara, and G. Tzanetakis, *Music Data Mining*. CRC Press, 1 ed., 2011.

[121] T. C. Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 164–181, 2011.

[122] P. Esling and C. Agon, "Time Series Data Mining," *ACM Computing Surveys*, vol. 45, no. 1, p. 12, 2012.

[123] C. Damle and A. Yalcin, "Flood prediction using Time Series Data Mining," *Journal of Hydrology*, vol. 333, no. 2-4, pp. 305–316, 2007.

[124] A. Grigorievskiy, Y. Miche, A.-M. Ventelá, E. Séverin, and A. Lendasse, "Long-term time series prediction using OP-ELM," *Neural Networks*, vol. 51, pp. 50–56, 2014.

[125] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast subsequence matching in time-series databases," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, vol. 23, pp. 419–429, ACM, 2005.

[126] A. Movchan and M. Zymbler, "Time series subsequence similarity search under dynamic time warping distance on the intel many-core accelerators," in *International conference on similarity search and applications*, pp. 295–306, Springer, 2015.

[127] B. Liu, J. Li, C. Chen, W. Tan, Q. Chen, and M. Zhou, "Efficient motif discovery for large-scale time series in healthcare," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 583–590, 2015.

[128] S. Torkamani and V. Lohweg, "Survey on time series motif discovery," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 7, no. 2, 2017.

[129] B. D. Fulcher and N. S. Jones, "Highly comparative feature-based time-series classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 12, pp. 3026–3037, 2014.

[130] S. Rani and G. Sikka, "Recent Techniques of Clustering of Time Series Data: A Survey," *International Journal of Computer Applications*, vol. 52, no. 15, pp. 1–9, 2012.

[131] S. Aghabozorgi, A. Seyed Shirkhorshidi, and T. Ying Wah, "Time-series clustering - A decade review," *Information Systems*, vol. 53, pp. 16–38, 2015.

[132] P. K. Chan, P. K. Chan, and M. V. Mahoney, "Modeling Multiple Time Series for Anomaly Modeling Multiple Time Series for Anomaly Detection," in *Fifth IEEE International Conference on Data Mining*, IEEE, 2005.

[133] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.

[134] Q. Gui, Z. Jin, M. V. R. Blondet, S. Laszlo, and W. Xu, "Towards EEG Biometrics : Pattern Matching Approaches for User Identification," in *IEEE International Conference on Identity, Security and Behavior Analysis*, pp. 1–6, IEEE, 2015.

[135] B. Doğan and M. Korürek, "A new ECG beat clustering method based on kernelized fuzzy c-means and hybrid ant colony optimization for continuous domains," *Applied Soft Computing*, vol. 12, no. 11, pp. 3442–3451, 2012.

[136] R. Agrawal, C. Faloutsos, and A. Swami, "Efficient similarity search in sequence databases," in *International conference on foundations of data organization and algorithms*, pp. 69–84, Springer, 1993.

[137] V. Perlibakas, "Distance measures for PCA-based face recognition," *Pattern Recognition Letters*, vol. 25, no. 6, pp. 711–724, 2004.

[138] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the Surprising Behavior of Distance Metrics in High Dimensional Space," in *International conference on database theory*, pp. 420–434, Springer, 2001.

[139] O. Maimon and L. Rokach, *Data mining and knowledge discovery handbook*. Springer, 2 ed., 2005.

[140] K. C. Chang, R. G. Lee, C. Wen, and M. F. Yeh, "Comparison of similarity measures for clustering electrocardiogram complexes," in *Computers in Cardiology*, pp. 759–762, IEEE, 2005.

[141] S. H. Lee, J. S. Lim, J. K. Kim, J. Yang, and Y. Lee, "Classification of normal and epileptic seizure EEG signals using wavelet transform, phase-space

reconstruction, and Euclidean distance," *Computer Methods and Programs in Biomedicine*, vol. 116, no. 1, pp. 10–25, 2014.

[142] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[143] P.-f. Marteau, "Time Warp Edit Distance with Stiffness Adjustment for Time Series Matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 306–318, 2009.

[144] G. Das, D. Gunopulos, and H. Mannila, "Finding Similar Time Series," in *European Symposium on Principles of Data Mining and Knowledge Discovery*, pp. 88–100, Springer, 1997.

[145] K. Wang and T. Gasser, "Alignment of Curves by Dynamic Time Warping," *The annals of Statistics*, vol. 25, no. 3, pp. 1251–1276, 1997.

[146] G. Al-Naymat, S. Chawla, and J. Taheri, "SparseDTW: A novel approach to speed up dynamic time warping," in *Proceedings of the Eighth Australasian Data Mining Conference*, pp. 117–127, 2009.

[147] S. Salvador and P. Chan, "FastDTW : Toward Accurate Dynamic Time Warping in Linear Time and Space," *Intelligent Data Analysis*, vol. 11, no. 5, pp. 561–580, 2007.

[148] E. S. Ristad and P. N. Yianilos, "Learning String-Edit Distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 5, pp. 522–532, 1998.

[149] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering Similar Multidimensional Trajectories," in *Proceedings 18th International Conference on Data Engineering*, pp. 673–684, IEEE, 2002.

[150] V. Kurbalija, M. Radovanović, Z. Geler, and M. Ivanović, "The Influence of Global Constraints on Similarity Measures for Time-Series Databases," *Knowledge-Based Systems*, vol. 56, pp. 49–67, 2014.

[151] R. J. Alcock and Y. Manolopoulos, "Time-series similarity queries employing a feature-based approach," in *7th Hellenic Conference on Informatics*, pp. 27–29, 1999.

[152] M. Thomas, M. K. Das, and S. Ari, "Automatic ECG arrhythmia classification using dual tree complex wavelet based features," *AEU - International Journal of Electronics and Communications*, vol. 69, no. 4, pp. 715–721, 2015.

[153] E. Scheme and K. Englehart, "On the robustness of EMG features for pattern recognition based myoelectric control; A multi-dataset comparison," *36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 650–653, 2014.

[154] R. Jenke, A. Peer, and M. Buss, "Feature extraction and selection for emotion recognition from EEG," *IEEE Transactions on Affective Computing*, vol. 5, no. 3, pp. 327–339, 2014.

[155] B. Horst and K. Abraham, *Data mining in time series databases*. World scientific, 57 ed., 2004.

[156] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases," *Knowl. Inf. Syst.*, vol. 3, no. 3, pp. 263–286, 2001.

[157] J. Lin, E. Keogh, L. Wei, and S. Lonardi, "Experiencing SAX: A novel symbolic representation of time series," *Data Mining and Knowledge Discovery*, vol. 15, no. 2, pp. 107–144, 2007.

[158] S. Hari, F. Agrafioti, and D. Hatzinakos, "Design of a Hamming-distance classifier for ECG biometrics," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3009–3012, IEEE, 2013.

[159] L. Wang, Y. Zhang, and J. Feng, "On the Euclidean Distances of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1334–1339, 2005.

[160] M. Randić, M. Vračko, N. Lerš, and D. Plavšić, "Analysis of similarity / dissimilarity of DNA sequences based on novel 2-D graphical representation," *Chemical Physics Letters*, vol. 371, pp. 202–207, 2003.

[161] A. S. Thakur and N. Sahayam, "Speech Recognition Using Euclidean Distance," *International Journal of Emerging Technology and Advanced Engineering*, vol. 3, no. 3, pp. 587–590, 2013.

[162] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 565–592, 2015.

[163] M. Müller, "Dynamic Time Warping," in *Information Retrieval for Music and Motion*, pp. 69–84, Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[164] F. Itakura, "Minimum Prediction Residual Principle Applied to Speech Recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, no. 1, pp. 67–72, 1975.

[165] C. Ratanamahatana and E. Keogh, "Everything you know about dynamic time warping is wrong," in *Third Workshop on Mining Temporal and Sequential Data*, 2004.

[166] S. W. Kim, S. Park, and W. W. Chu, "An Index-Based Approach for Similarity Search Supporting Time Warping in Large Sequence Databases," in *Proceedings - International Conference on Data Engineering*, pp. 607–614, IEEE, 2001.

[167] D. Lemire, "Faster retrieval with a two-pass dynamic-time-warping lower bound," *Pattern Recognition*, vol. 42, no. 9, pp. 2169–2180, 2009.

[168] E. Keogh, L. Wei, X. Xi, M. Vlachos, S. H. Lee, and P. Protopapas, "Supporting exact indexing of arbitrarily rotated shapes and periodic time series under Euclidean and warping distance measures," *VLDB Journal*, vol. 18, no. 3, pp. 611–630, 2009.

[169] E. Keogh and S. Kasetty, "On the need for time series data mining benchmarks: a survey and empirical demonstration," *Data Mining and knowledge discovery*, vol. 7, no. 4, p. 102, 2003.

[170] H. C. Huang and B. H. Jansen, "EEG waveform analysis by means of dynamic time-warping," *International Journal of Bio-Medical Computing*, vol. 17, no. 2, pp. 135–144, 1985.

[171] R. Ahmed, A. Temko, W. P. Marnane, G. Boylan, and G. Lightbody, "Exploring temporal information in neonatal seizures using a dynamic time warping based SVM kernel," *Computers in Biology and Medicine*, vol. 82, pp. 100–110, 2017.

[172] J. Lin, R. Khade, and Y. Li, "Rotation-invariant similarity in time series using bag-of-patterns representation," *Journal of Intelligent Information Systems*, vol. 39, no. 2, pp. 287–315, 2012.

[173] S. Scott and S. Matwin, "Feature engineering for text classification," in *International Conference on Machine Learning*, vol. 6, (Bled, Slovenia), pp. 379–388, 1999.

[174] S.-H. Kim, C. B. Jeong, H. K. Kwag, and C. Y. Suen, "Gap Clustering and Special Symbol Detection," in *Proceedings 16th International Conference on Pattern Recognition*, (Quebec, Canada), pp. 320–323, IEEE, 2002.

[175] S. Soderland, "Learning to Extract Text-Based Information from the World Wide Web," *Knowledge Discovery and Data Mining*, pp. 251–254, 1997.

[176] A. Mueen and E. Keogh, "Extracting Optimal Performance from Dynamic Time Warping," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 2129–2130, ACM, 2016.

[177] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, "Learning time-series shapelets," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 392–401, ACM, 2014.

[178] G. Gupta and S. Sharma, "Enhanced SHA-192 algorithm with larger bit difference," in *International Conference on Communication Systems and Network Technologies*, pp. 152–156, IEEE, 2013.

[179] A. L.-C. Wang, "An Industrial Strength Audio Search Algorithm," in *Proceedings of the 4th International Society for Music Information Retrieval Conference (ISMIR 203), Baltimore, Maryland (USA), 26-30 October 2003*, pp. 7–13, 2003.

[180] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. No. 3, MIT press, 3 ed., 2009.

[181] P. Cano, E. Batlle, T. Kalker, and J. Haitsma, "A review of audio fingerprinting," *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, vol. 41, no. 3, pp. 271–284, 2005.

[182] H. Özer, B. Sankur, and N. Memon, "Robust audio hashing for audio identification," in *12th European Signal Processing Conference*, pp. 2091–2094, IEEE, 2004.

[183] E. Canhasi, "Fast document summarization using locality sensitive hashing and memory access efficient node ranking," *International Journal of Electrical and Computer Engineering*, vol. 6, no. 3, pp. 945–954, 2016.

[184] B. Stein, "Fuzzy-fingerprints for text-based information retrieval," in *Proceedings of the 5th International Conference on Knowledge Management*, pp. 572–579, 2005.

[185] H. Li, W. Liu, and H. Ji, "Two-Stage Hashing for Fast Document Retrieval," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 495–500, 2014.

[186] D. Ceglarek and K. Haniewicz, "Fast plagiarism detection by Sentence Hashing," in *LInternational Conference on Artificial Intelligence and Soft Computing*, pp. 30–37, Springer, 2012.

[187] M. Hammad, G. Luo, and K. Wang, "Cancelable Biometric Authentication System Based on," *Multimedia Tools and Applications*, pp. 1857–1887, 2019.

[188] D. Nguyen, D. Tran, D. Sharma, and W. Ma, "Investigating the Impact of Epilepsy on EEG-based Cryptographic Key Generation Systems," *Procedia Computer Science*, vol. 112, pp. 177–185, 2017.

[189] C. He, X. Lv, and Z. J. Wang, "Hashing the mAR coefficients from EEG data for person authentication," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1445–1448, IEEE, 2009.

[190] P. T. Xuan and D. T. Anh, "An Efficient Hash-Based Method for Time Series Motif Discovery," in *International Conference on Multi-disciplinary Trends in Artificial Intelligence*, pp. 205–211, Springer, 2018.

[191] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2916–2929, 2013.

[192] R. A. P. Hernandez, M. N. Miyatake, and B. M. Kurkoski, "Robust image hashing using image normalization and SVD decomposition," in *Midwest Symposium on Circuits and Systems*, pp. 1–4, IEEE, 2011.

[193] J. Wang, S. Kumar, and S. F. Chang, "Semi-supervised hashing for large-scale search," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 12, pp. 2393–2406, 2012.

[194] G. Grutzek, J. Strobl, B. Mainka, F. Kurth, C. Pörschmann, and H. Knospe, "Perceptual Hashing for the Identification of Telephone Speech," in *Speech Communication; 10. ITG Symposium*, pp. 1–4, 2012.

[195] Y.-b. Huang, Q.-y. Zhang, and Z.-t. Yuan, "Perceptual Speech Hashing Authentication Algorithm Based on Linear Prediction Analysis," *TELKOMNIKA Indonesian Journal of Electrical Engineering*, vol. 12, no. 4, pp. 3214–3223, 2014.

[196] L. Schmidt, M. Sharifi, and I. L. Moreno, "Large-scale speaker identification," in *IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 1650–1654, IEEE, 2014.

[197] J. Haitsma and T. Kalker, "A highly robust audio fingerprinting system," in *Proceedings of the 3rd International Society for Music Information Retrieval Conference (ISMIR'02)*, pp. 107–115, 2002.

[198] S. Fenet, G. Richard, and Y. Grenier, "A Scalable Audio Fingerprint Method with Robustness to Pitch-Shifting.," in *International Society for Music Information Retrieval Conference*, pp. 121–126, 2011.

[199] R. Hasan, M. Jamil, G. Rabbani, and S. Rahman, "Speaker Identification Using Mel Frequency," in *3rd International Conference on Electrical & Computer Engineering*, vol. 1, pp. 565–568, 2004.

[200] P. Indyk and R. Motwani, "Approximate nearest neighbors," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing - STOC '98*, (Dallas, Texas, USA), pp. 604–613, ACM Press, 1998.

[201] A. Rajaraman and J. D. Ullman, *Mining of Massive Datasets*. Cambridge University Press, 2011.

[202] V. Venkatachalam, L. Cazzanti, N. Dhillon, and M. Wells, "Automatic identification of sound recordings," *IEEE Signal Processing Magazine*, vol. 21, no. 2, pp. 92–99, 2004.

[203] P. Cano, E. Batle, T. Kalker, and J. Haitsma, "A review of algorithms for audio fingerprinting," *Proceedings of 2002 IEEE Workshop on Multimedia Signal Processing, MMSP 2002*, pp. 169–173, 2002.

[204] P. Cano, E. Batlle, H. Mayer, and H. Neuschmied, "Robust Sound Modeling for Song Detection in Broadcast Audio," *Proc. AES 112th International Convention*, vol. pp, pp. 1–7, 2002.

[205] Yu Liu, Kiho Cho, Hwan Sik Yun, Jong Won Shin, and Nam Soo Kim, "DCT based multiple hashing technique for robust audio fingerprinting," in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 61–64, IEEE, 2009.

[206] S. Baluja and M. Covell, "Waveprint: Efficient wavelet-based audio fingerprinting," *Pattern Recognition*, vol. 41, no. 11, pp. 3467–3480, 2008.

[207] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. D. Ullman, and C. Yang, "Finding Interesting without Support Pruning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 1, pp. 64–78, 2001.

[208] H. Kekre, N. Bhandari, N. Nair, P. Padmanabhan, and S. Bhandari, "A Review of Audio Fingerprinting and Comparison of Algorithms," *International Journal of Computer Applications*, vol. 70, no. 13, 2013.

[209] J. Haitsma and T. Kalker, "A Highly Robust Audio Fingerprinting System With an Efficient Search Strategy," *Journal of New Music Research*, vol. 32, no. 2, pp. 211–221, 2003.

[210] J. Woodbridge, B. Mortazavi, A. A. Bui, and M. Sarrafzadeh, "Improving biomedical signal search results in big data case-based reasoning environments," *Pervasive and Mobile Computing*, vol. 28, pp. 69–80, 2016.

[211] W. H. Gomaa and A. A. Fahmy, "A Survey of Text Similarity Approaches," *International Journal of Computer Applications*, vol. 68, no. 13, pp. 13–18, 2013.

[212] W. B. Cavnar and J. M. Trenkle, "N-gram-based text categorization," in *Proceedings of 3rd Annual Symposium on Document Analysis and Information Retrieval*, pp. 161–175, Citeseer, 1994.

[213] R. L. Cilibrasi and P. M. Vitányi, "The Google similarity distance," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 3, pp. 370–383, 2007.

[214] C. Sadowski and G. Levin, "SimHash : Hash-based Similarity Detection," 2007.

[215] A. Shrivastava, "Simple and Efficient Weighted Minwise Hashing," in *Advances In Neural Information Processing Systems*, no. 2, pp. 1498–1506, 2016.

[216] S. Buyrukbilen and S. Bakiras, "Secure Similar Document Detection with Simhash," in *Workshop on Secure Data Management*, pp. 61–75, Springer, 2013.

[217] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proceedings of the 34th Annual ACM Symposium on Theory of computing*, (Montreal, Quebec, Canada), pp. 380 – 388, ACM, 2002.

[218] K. Williams, J. Wu, and C. L. Giles, "SimSeerX: A Similar Document Search Engine," in *Proceedings of the ACM Symposium on Document Engineering*, pp. 143–146, 2014.

[219] K. Williams, L. Li, M. Khabsa, J. Wu, P. C. Shih, and C. L. Giles, "A web service for scholarly big data information extraction," in *Proceedings - IEEE International Conference on Web Services*, pp. 105–112, IEEE, 2014.

[220] D. Zhang, J. Wang, D. Cai, and J. Lu, "Laplacian co-hashing of terms and documents," in *European Conference on Information Retrieval*, pp. 577–580, Springer, 2010.

[221] B. Logan, "Mel Frequency Cepstral Coefficients for Music Modeling," in *Proceeding of the International Symposium on Music Information Retrieval (ISMIR)*, (Plymouth, USA), oct 2000.

[222] E. R. Dougherty and R. A. Lotufo, *Hands-on Morphological*. SPIE press, 2003.

[223] M. Kalos and P. Whitlock, *Monte Carlo Methods*, vol. 18. John Wiley & Sons, 2009, 2 ed., 2009.

[224] X.-s. Yang, *Nature-inspired Metaheuristic Algorithms*. Luniver Press, illustrate ed., 2010.

[225] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[226] T. Back, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, 1996.

[227] C. Blum and X. Li, "Swarm Intelligence in Optimization," *Swarm Intelligence Introduction and Applications*, pp. 43–85, 2008.

[228] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, (Perth, WA, Australia), pp. 1942–1948, IEEE, 1995.

[229] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence*, (Anchorage, AK, USA, USA), pp. 69–73, IEEE, 1998.

[230] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[231] J. Kennedy, J. Kennedy, R. Eberhart, and Y. Shi, *Swarm Intelligence*. Morgan Kaufmann, illustrate ed., 2001.

[232] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," (La Jolla, CA, USA), pp. 84–88, IEEE, 2000.

[233] M. Bosi and R. E. Goldberg, *Introduction to Digital Audio Coding and Standards*. Springer US, 2002.

[234] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement of the psychological magnitude pitch," *The Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 185–190, 1937.

[235] L. Paulevé, H. Jégou, and L. Amsaleg, "Locality sensitive hashing: A comparison of hash function types and querying mechanisms," *Pattern Recognition Letters*, vol. 31, no. 11, pp. 1348–1358, 2010.

[236] P. Indyk, N. Koudas, and S. Muthukrishnan, "Identifying Representative Trends in Massive Time Series Data Sets Using Sketches," in *International Conference on Very Large Data Bases*, pp. 363–372, 2000.

[237] C. Luo, A. Shrivastava, O. Anava, M. Cuturi, A. Khaleghi, V. Kuznetsov, and A. Rakhlin, "SSH (Sketch, Shingle, & Hash) for Indexing Massive-Scale Time Series," in *NIPS 2016 Time Series Workshop*, pp. 38–58, 2017.

[238] S. Ioffe, "Improved consistent sampling, Weighted Minhash and L1 Sketching," in *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 246–255, 2010.

[239] I. Marinchev and G. Agre, "On Speeding Up the Implementation of Nearest Neighbour Search and Classification," in *Proceedings of the 16th International Conference on Computer Systems and Technologies*, pp. 207–213, 2015.

[240] A. Colin Cameron and F. A. Windmeijer, "An R-squared measure of goodness of fit for some common nonlinear regression models," *Journal of Econometrics*, vol. 77, no. 2, pp. 329–342, 1997.

[241] J. H. Hansen and T. Hasan, "Speaker recognition by machines and humans: A tutorial review," *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 74–99, 2015.

[242] N. A. S. I. Series, *Fundamentals in Handwriting Recognition*. Springer Science \& Business Media, 1 ed., 2012.

[243] W.-Y. Lin, Y.-H. Hu, and C.-F. Tsai, "Machine Learning in Financial Crisis Prediction: A Survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 4, pp. 421–436, 2012.

[244] O. Faust, Y. Hagiwara, T. J. Hong, O. S. Lih, and U. R. Acharya, "Deep learning for healthcare applications based on physiological signals: A review," *Computer Methods and Programs in Biomedicine*, vol. 161, pp. 1–13, 2018.

[245] M. Schedl, E. Gómez, and J. Urbano, "Music Information Retrieval: Recent Developments and Applications," *Foundations and Trends in Information Retrieval*, vol. 8, no. 2-3, pp. 127–261, 2014.

[246] B. Blankertz, G. Dornhege, M. Krauledat, V. Kunzmann, F. Losch, G. Curio, and K.-R. Müller, "The Berlin Brain-Computer Interface: Machine learning based detection of user specific brain states," *Journal of Universal Computer Science*, vol. 12, no. 6, pp. 581–607, 2006.

[247] B. Obermaier, C. Guger, C. Neuper, and G. Pfurtscheller, "Hidden Markov models for online classification of single trial EEG data," *Pattern Recognition Letters*, vol. 22, no. 12, pp. 1299–1309, 2001.

[248] S. Marcel and J. d. R. Millan, "Person authentication using brainwaves (EEG) and maximum a posteriori model adaptation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 4, pp. 743–748, 2007.

[249] J. A. Nasiri, M. Naghibzadeh, H. S. Yazdi, and B. Naghibzadeh, "ECG arrhythmia classification with support vector machines and genetic algorithm," in *3rd European Modelling Symposium on Computer Modelling and Simulation*, pp. 187–192, IEEE, 2009.

[250] N. Emanet, "ECG beat classification by using discrete wavelet transform and random forest algorithm," in *5th International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control*, pp. 1–4, IEEE, 2009.

[251] K. S. Kim, H. H. Choi, C. S. Moon, and C. W. Mun, "Comparison of k-nearest neighbor, quadratic discriminant and linear discriminant analysis in classification of electromyogram signals based on the wrist-motion directions," *Current Applied Physics*, vol. 11, no. 3, pp. 740–745, 2011.

[252] A. Banerjee, S. Datta, M. Pal, A. Konar, D. Tibarewala, and R. Janarthanan, "Classifying Electrooculogram to Detect Directional Eye Movements," *Procedia Technology*, vol. 10, pp. 67–75, 2013.

[253] V. Vapnik, *Statistical learning theory*. Wiley, New York, 1998.

[254] R. Nallapati, "Discriminative models for information retrieval," in *Proceedings of the 27th Annual International Conference on Research and Development in Information Retrieval*, pp. 64–71, ACM, 2004.

[255] E. Fix and J. Hodges, "Nonparametric discrimination: consistency properties," tech. rep., 1951.

[256] A. Tharwat, T. Gaber, A. Ibrahim, and A. E. Hassanien, "Linear discriminant analysis: A detailed tutorial," *AI Communications*, vol. 30, no. 2, pp. 169–190, 2017.

[257] D. Reynolds, "Gaussian mixture models," *Encyclopedia of Biometrics*, pp. 827–832, 2015.

[258] V. Vapnik, *The nature of statistical learning theory*, vol. 1. Springer, 1 ed., 1995.

[259] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[260] X. Wu, V. Kumar, Q. J. Ross, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z. H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg, *Top 10 Algorithms in Data Mining*, vol. 14. Springer, 2008.

[261] S. Balakrishnama and A. Ganapathiraju, "Linear discriminant analysis - a brief tutorial," *Institute for Signal and Information Processing*, vol. 18, pp. 1–8, 1998.

[262] R. Ahmed, *Dynamic classifiers for neonatal brain monitoring*. PhD thesis, University College Cork, 2016.

[263] C. J. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[264] A. O'Shea, G. Lightbody, G. Boylan, and A. Temko, "Investigating the Impact of CNN Depth on Neonatal Seizure Detection Performance," in *Proceedings of the International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 2018, pp. 5862–5865, IEEE, jul 2018.

[265] A. Yazdani, T. Ebrahimi, and U. Hoffmann, "Classification of EEG signals using Dempster Shafer theory and a K-nearest neighbor classifier," in *4th International IEEE Conference on Neural Engineering*, no. 1, pp. 327–330, IEEE, 2009.

[266] B. A. S. Hasan and J. Q. Gan, "Unsupervised adaptive GMM for BCI," in *International Conference on Neural Engineering*, pp. 295–298, IEEE, 2009.

[267] D. Reynolds, "Universal Background Models," in *Encyclopedia of Biometrics* (S. Z. Li and A. Jain, eds.), pp. 1349–1352, Boston, MA: Springer, 2009.

[268] K. Chomboon, P. Chujai, P. Teerarassammee, K. Kerdprasop, and N. Kerdprasop, "An Empirical Study of Distance Metrics for k-Nearest Neighbor Algorithm," in *Proceedings of the 3rd International Conference on Industrial Application Engineering*, no. December, pp. 280–285, 2015.

[269] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[270] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," in *Advances in Neural Information Processing Systems*, pp. 841–848, 2002.

[271] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, and B. Arnaldi, "A review of classification algorithms for EEG-based brain-computer interfaces," *Journal of Neural Engineering*, vol. 4, no. 2, 2007.

[272] L. Liu and J. He, "On the use of orthogonal GMM in speaker recognition," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 845–848, IEEE, 1999.

[273] A. Dempster, N. M. Laird, D.B. Rubin, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 1–38, 1977.

[274] S. P. Lloyd, "Least Squares Quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[275] D. Arthur and S. Vassilvitskii, "k-means ++ : The Advantages of Careful Seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035, 2007.

[276] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the 14th international joint conference on Artificial intelligence*, (Los Altos, CA), pp. 1137–1143, 1995.

[277] M. H. Zweig and G. Campbell, "Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine," *Clinical Chemistry*, vol. 39, no. 4, pp. 561–77, 1993.

[278] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[279] S. Gunn, "Support vector machines for classification and regression," *ISIS technical report*, vol. 14, no. 1, pp. 5–16, 1998.