



Bio-Signal Data Gathering, Management and Analysis
within a Patient-Centred Health Care Context

being a thesis submitted for the degree of
Doctor of Philosophy
at the University of Hull

by
ROBERT ALEXANDER MUNNOCH MEng (University of Leeds)

December, 2017

Abstract

The healthcare service is under pressure to do more with less, and changing the way the service is modelled could be the key to saving resources and increasing efficacy. This change could be possible using patient-centric care models. This model would include straightforward and easy-to-use telemonitoring devices and a flexible data management structure. The structure would maintain its state by ingesting many sources of data, then tracking this data through cleaning and processing into models and estimates to obtaining values from data which could be used by the patient. The system can become less disease-focused and more health-focused by being preventative in nature and allowing patients to be more proactive and involved in their care by automating the data management. This work presents the development of a new device and a data management and analysis system to utilise the data from this device and support data processing along with two examples of its use. These are signal quality and blood pressure estimation. This system could aid in the creation of patient-centric telecare systems.

Acknowledgement

This thesis has been a journey, and I have been very fortunate with the people I have had around me.

First I would like to thank my family. My wife Ashley Merrithew, and mum Hilary Munnoch and of course my father Richard Munnoch who passed away 23rd January 2016; he is greatly missed.

Secondly, I have had the privilege to have known two supervisors. The first who helped me begin this journey was Professor Ping Jiang. I have enjoyed my time with him and wished it could have continued. Sadly he passed away three days before my father. He too will be missed, and I hope he would be happy with how the work has turned out. Lastly but certainly not least, I would like to thank Dr. Darryl Davis, who took me on after Ping. It has been a pleasure and privilege to work with him. His patience and criticism of my work have been invaluable.

Authors Declaration

I declare that the material contained in this thesis represents original work undertaken by the author, except for the later development of the device as part of chapter 6. This chapter was first presented in a conference paper by Munnoch & Jiang (2015) following a technology strategy board (TSB) project in collaboration with Imonsys and Tritec to prototype and test the device. The device was developed further after the project was completed with help from the author. The author also made the data gathering programs and supported the Yorkshire innovation forward (YIF) grant¹ to conduct the study and gather the data used in this thesis. Ethical approval was sought and given for the YIF study and the signal quality assessment with outside assessors conducted in chapter 7.

¹The Yorkshire Innovation Fund (YIF) was led by the University of Bradford on behalf of a consortium of Yorkshire and Humber universities. It was part-financed by the European Regional Development Fund (ERDF) under the Yorkshire and Humber ERDF Programme 2007-13. The £5m programme attracted £3.1m of ERDF to support the regions economic development. YIF support was secured within this programme for two stages of R&D and innovation support - YIF project reference: ERDF 904618; University of Hull collaborative project references: SIP02.01 and RDP52

Contents

1	Introduction	13
1.1	Introduction	13
1.2	Problem Identification	14
1.3	Research Aims	16
1.4	Thesis Contributions	17
1.5	Organisation	19
I	The Data Management Framework	21
2	Background	22
2.1	Overview	22
2.2	System Architectural Design	22
2.3	Big data Systems	24
2.3.1	Distributed Architectures	25
2.3.2	Communications	26
2.4	Application Specific Storage	27
2.5	Database Systems	28
2.6	Summary	32
3	Data Management Framework	33
3.1	Overview	33
3.2	Framework Structure	33
3.2.1	System Objectives	34
3.2.2	System Overview	35
3.2.3	Service Communication and API	38
3.3	NodeSR - Storage System	42
3.3.1	Overview	42
3.3.2	API interfaces	45
3.3.3	Storage Manager	47
3.3.4	User Interface and Management	51
3.4	ProcessCR - Compute System	51
3.4.1	Overview	51
3.4.2	API Interface	53
3.4.3	Programs as Data	55
3.4.4	Asynchronous Task System	56
3.4.5	User Interface and Management	59
3.5	Application Specific Service Interface (ASSI)	59
3.5.1	Overview	59
3.5.2	API Interface	60
3.5.3	Domain Components	61

3.5.4	User Interface and Management	61
3.6	Summary	62
4	Discussion and Future Work	63
4.1	Overview	63
4.2	Discussion	63
4.2.1	Current State	64
4.2.2	Limitations	65
4.3	Further Work	67
4.4	Summary	68
II	Application of the Framework	69
5	Background and Literature	70
5.1	Overview	70
5.2	Health Service Pressures	70
5.3	Investigating Models of Healthcare	71
5.3.1	Clinician-centred	73
5.3.2	Patient-centred	75
5.4	Telehealth Support Systems	77
5.5	Summary	82
6	EIMO Data Gathering System	83
6.1	Overview	83
6.2	Review of Personal Medical Devices	84
6.3	Device Development	87
6.3.1	Device Overview	87
6.3.2	Processing and Communications	87
6.3.3	Signal Processing	89
6.3.4	Device Interface and Logging Software	91
6.4	Device Testing Study	95
6.4.1	Study Protocol	95
6.4.2	Comparison Devices	96
6.4.3	Processing and Synchronisation	99
6.4.4	Comparison Analysis	103
6.5	Summary	109
7	Signal Quality Assessment	110
7.1	Overview	110
7.2	Background	111
7.2.1	Signal Analysis and Quality Assessment Survey	112
7.2.2	Classification Strategy	117
7.2.3	Summary	122
7.3	Classification System	122
7.3.1	Main Structure	123
7.3.2	Data Inputs	124
7.3.3	Processing Elements	140
7.3.4	Classification and Learning Architectures	144
7.3.5	Testing and Results Strategy	146
7.4	Case Study - Human Annotations	147

7.4.1	Overview	147
7.4.2	Methodology	147
7.4.3	Results	148
7.4.4	Discussion	155
7.5	Summary	155
8	Automating Signal Quality Analysis	157
8.1	Overview	157
8.2	Background	157
8.2.1	Further Machine Learning	158
8.2.2	Feature Descriptions	162
8.2.3	Subjective and Objective Signal Analysis	163
8.3	Feature Extraction and Modelling Systems	166
8.3.1	Extracting Signal Features	166
8.3.2	Modelling Test Determination	167
8.4	Exploring Signal Features	168
8.4.1	Overview	168
8.4.2	Defining Classes of Features	169
8.4.3	Discussion	178
8.5	Case Study - Subjective Signal Quality With Objective Features . . .	179
8.5.1	Overview	179
8.5.2	Methodology	180
8.5.3	Results	182
8.5.4	Discussion	192
8.6	Framework Data Flow Example	193
8.7	Summary	196
9	Blood Pressure Model Analysis	197
9.1	Overview	197
9.2	Review of Blood Pressure	197
9.2.1	Blood Pressure Measurement	198
9.2.2	Blood Pressure Estimation and Modelling	199
9.3	Blood Pressure Estimation Structure	201
9.3.1	Prepare and clean the data	201
9.3.2	Extract the good features	202
9.3.3	Model analysis	203
9.4	Case Study BP Feature Estimation Using Raw and Cleansed Data . .	203
9.4.1	Overview	203
9.4.2	Methodology	204
9.4.3	Results	207
9.4.4	Discussion	217
9.5	Framework Data Flow Example	218
9.6	Summary	220
10	Discussion and Future Work	221
10.1	Overview	221
10.2	Data Gathering	222
10.2.1	Discussion	222
10.2.2	Current State	223
10.2.3	Limitations and Future Work	224
10.3	Data Analysis	224

10.3.1 Discussion	224
10.3.2 Limitations and Future Work	225
10.4 Summary	227
III Conclusion and References	228
11 Conclusions	229
11.1 Overview	229
11.2 Thesis Contributions	229
11.2.1 Aim 1 - Develop a Data Management and Analysis Framework.	229
11.2.2 Aim 2 - Application of the Framework for Data Capture and Analysis	230
11.3 Closing Statement	232
12 References	233
IV Appendix	249
A Mathematical Notation	250
B Device Logs	256
B.1 Data Logs for devices	256
B.1.1 EIMO Mobile Monitor	257
B.1.2 CM400 Patient Monitor	261
B.1.3 Unified Log File Listing	265
C Study Protocol Information	269
C.1 Device Study	269
C.2 Quality Assessment Study	271
D Framework Implementation	279
D.1 Overview	279
D.1.1 NodeSR	279
D.1.2 NodeSR Internal Representational Structures	280
D.1.3 MedicalDB	285
D.2 Medical Database Structures	291
D.2.1 Annotation types	291

List of Figures

3.1	Framework diagram	36
3.2	Framework network map	37
3.3	URL Route Tree	40
3.4	NodeSR service architecture	43
3.5	Internal graph data structure	44
3.6	Storage manager data flow diagram	48
3.7	Storage manager component mapping and access	50
3.8	ProcessCR service architecture	52
3.9	Abstract processing system	56
3.10	ProcessCR graph representation of a process	57
3.11	ASSI service architecture	60
5.1	Models of Health Care Control.	72
6.1	EIMO system overview	87
6.2	EIMO being held in use	88
6.3	Device system overview	89
6.4	EIMO PPG Signal processing block diagram	90
6.5	EIMO ECG Signal processing block diagram	91
6.6	Development application signal display screen	92
6.7	Screenshot of the iPad App for the study	93
6.8	CM400 connected to a laptop	97
6.9	CM400 interface application diagram	98
6.10	Unified log file data flow system	100
6.11	Signal correlation between EIMO, CM400 and Case-GE	102
6.12	Synchronisation system diagrams	103
6.13	Synchronisation signal results	103
6.14	Best device comparison	105
6.15	Average device comparison	106
6.16	Best device comparison	107
6.17	Average device comparison	108
6.18	Best device comparison timing alignment	108
7.1	Overview of signal analysis system structure	124
7.2	Segment boundary comparison	129
7.3	PPG adaptive peak detection algorithm	132
7.4	Annotation type hierarchy diagram	134
7.5	Example annotation to quality trace mapping diagram	135
7.6	Idealised signal morphology labelling	136
7.7	Web based annotation interface	138
7.8	Sample signal quality trace calculation	139

7.9	Basic contextual signal features	141
7.10	Sample signal segmentation and event quality assignment	143
7.11	Scanned annotation with measurements	149
7.12	ROC performance of human and machine annotation origins	152
7.13	Assessors annotation summaries	154
8.1	ROC field diagram	161
8.2	Example segment graph with the median filter and residual traces	171
8.3	Performance distance verses AUC	184
8.4	Feature class performance	185
8.5	Model Performance for single features representing each feature class	187
8.6	ROC model performance contour	189
8.7	Framework dependency graph for the signal quality analysis	195
9.1	Blood pressure analysis data flow diagram	201
9.2	ECG PPG fusion feature extraction	202
9.3	Sample extraction data section before the BP datapoint	204
9.4	Sample extraction data section after the BP datapoint	205
9.5	Sample extraction data section early after the BP datapoint	206
9.6	Distribution of measurement support split for each user	209
9.7	Feature relationships with increasing support showing V_{PWV} vs P_s , P_d , P_p	211
9.8	Feature relationships with increasing support showing $\ln(T_{PTT})$ vs P_s , P_d , P_p	212
9.9	Feature relationships with increasing support showing T_{HR} vs P_s , P_d , P_p	213
9.10	Modelling error verses support threshold	215
9.11	Framework dependency graph for the blood pressure analysis.	219
C.1	Coloured annotation assessment example	276
C.2	Scanned annotation example filled out	277
D.1	NodeSR web screenshot	281
D.2	NodeSR process node rendered graph	282
D.3	MedicalDB graph representation of medical data	286
D.4	MedicalDB web annotation screenshot	290

List of Tables

3.1	The main structures used to query and store data within the service.	41
3.2	NodeSR storage interface verbs	45
3.3	NodeSR management interface verbs	47
3.4	ProcessCR service component list	53
3.5	ProcessCR compute interface verbs	54
3.6	ProcessCR management interface verbs	55
3.7	ASSI example service components list	61
6.1	EIMO Feature data packet layout	88
6.2	EIMO Signal integrated data packet layout	89
6.3	EIMO signal comparison	104
7.1	The model confusion matrix.	120
7.2	Example table of annotations	135
7.3	Annotation time verses recording time for each user	150
7.4	Annotation agreement between annotation origin	151
7.5	Agreement of each assessor	153
8.1	Summary of features from the literature	163
8.2	Top model results for ECG	190
8.3	Top model results for PPG	190
8.4	Feature set of the best models	191
8.5	Model parameters of the best models	191
9.1	Summary of the feature windows extracted	208
9.2	Statistical description of measurement instances table of the feature windows extracted	208
9.3	Blood pressure estimate summary with a support threshold of above 0 events	214
9.4	Blood pressure estimate summary with a support threshold of above 22 events	214
9.5	Individual user modelling and estimation for User 6	216
9.6	Individual user modelling and estimation for User 8	216
9.7	Individual user modelling and estimation for User 10	216
9.8	Individual user modelling and estimation for User 16	217
A.1	Mathematical notation used in Chapter 5.	250
A.2	Mathematical notation used in Chapter 6.	252
A.3	Mathematical notation used in Chapter 7.	254
B.1	Field description for the log file and data produced by the EIMO device as event data and summaries.	259

B.2	Field description for the log file and data produced by the EIMO device as a time series of logged samples.	260
B.3	Field description for the log file and data produced by the CM400 patient monitor as event data and summaries.	263
B.4	Field description for the log file and data produced by the CM400 patient monitor as a time series of logged samples.	264
C.1	EIMO study participant characteristics	270
D.1	Example field description for the node model within the NodeSR system when saving to MySQL.	283
D.2	Example field description for the bulk model which is derived from the node model within the NodeSR system when saving to MySQL.	283
D.3	Example field description for the link model within the NodeSR system when saving to MySQL.	284
D.4	ProcessCR component implementation attributes	287
D.5	MedicalDB component implementation attributes	288
D.6	A listing of the annotation types used in the MedicalDB.	292

List of Abbreviations

ANN	Artificial Neural Network.
API	Application Peripheral Interface.
ASIC	Application Specific Integrated Circuit.
ASSI	Application Specific Service Interface.
AUC	Area Under the Curve.
BP	Blood Pressure.
CDS	Clinical Decision Support.
CDSS	Clinical Decision Support System.
DDD	Domain-Derived Design.
DFR	Distance from Random Line.
ECG	Electrocardiograph.
EPIC	Electric Potential Integrated Circuit.
GUID	Generic Universal Identity.
HR	Heart Rate.
JSON	JavaScript Object Notation.
PPG	Photoplethysmography.
ROC	Receiver Operating Characteristic.
SHES	Sport, Health and Exercise Science.
SPO2	Saturation of Peripheral Oxygen.
SVM	Support Vector Machine.
SVMc	Support Vector Machine Classifier.
SVMoc	Support Vector Machine One Class Classifier.
SVMr	Support Vector Machine Regression.
URI	Uniform Resource Identifier.
URL	Uniform Resource Locator.
XML	Extensible Markup Language.

1 Introduction

1.1 Introduction

The current issues surrounding medical monitoring and data management within a healthcare setting inspire a need to create a better, generic, data understanding and management solution. These issues are potentially far reaching, since most people have an interest in their health, and some may require external healthcare intervention. This intervention needs to be timely and effective. To do this new healthcare models will need to be built and proven to work, which can assess the individual needs of the patient through monitoring, making sure the best data is available to be used when estimating a person's state of health and providing these insights back to the users of the system.

In broad scope, this thesis describes elements of a framework which follows the data flow of the care system above. There are two parts presented in this thesis:

- Data management and processing, the major component of the thesis, by designing and implementing the system which supports data analysis through graph based chains of processing and analysis, following a block chain system of linking data through modular processes into results and then on to further processing.
- Data gathering, analysis and processing, by using the framework and elements above to gather and store data from a device for comparison and analysis. The data stored was annotated to benchmark and classify the quality of the signals gathered. The signal quality assessment was used to compare the device and features used for blood pressure estimation.

The components of the system and framework described in this thesis could be used as operational parts or for the design of a patient-centred healthcare system.

Next, the chapter will examine the problems and identify the surrounding issues, that will set the scope of this work. Then the aims for the research as a whole will be described, along with their contributions. Lastly, this chapter will detail the overall organisation of this thesis.

1.2 Problem Identification

The issues briefly described above can be broken down into four main areas. Firstly, the gathering and recording of medical data from patients. Secondly is the management and synthesis of this data to gain the most benefit through analytics and modelling with the integration into a useful care system. Thirdly, the problem of using the data to model and predict signal quality and vital signs is addressed. Lastly, the prediction of the system and the risk and benefits that the users or patients face are then discussed, thus completing the system's life cycle.

Problem 1

There are a growing number of devices, which are being designed to capture an increasing amount of information generated by patients or users. This issue is discussed in more detail in section 6.2. These devices need to not only capture the right kind of diagnostic information about vital signs but also to do it quickly and efficiently so that the devices can have a high adoption and compliance rate from its users. When considering these devices, it is not enough to own a device, but one has to use it (Greenhalgh *et al.* , 2013).

Problem 2

The design of the management and data processing framework needs to be addressed. The primary concerns are that the design should try to solve the general problems and not let the design be limited to solving empirical problems that could

arise at the design or implementation stage of the system, being mindful of the future development of the system. The last issue is, to be able to easily run data transformations and processing tasks to turn the raw data into useful and informative results, while maintaining the data integrity (Sittig *et al.* , 2008; Fox *et al.* , 2010).

Problem 3

Mobile medical devices, when used in modern telehealth systems can allow the patients to potentially generate a large quantity of data. This data influx needs to be supported by a system that can manage it. The quality of the data these devices capture can suffer from problems, including movement and environmental noise artefacts for example. Therefore, finding and maintaining methods of keeping the data quality high is a growing concern as the potential data input volumes start to scale up with the increasing number of devices on the market. The cleaned signal data should be able to be utilised for further analysis and estimation of parameters used to inform and educate the users of the system about their health and the recommended options. By solving this issue, it may assist the users to follow a clear trajectory for the maintenance of their health, long-term outcomes, and goals (Epstein & Street, 2011; May *et al.* , 2003; Cusack *et al.* , 2008).

Problem 4

The last and crucial part of the system is the feedback of the collected and processed information back to the users of the system. However, this requires extensive knowledge of the best way to inform and interact with users. Work has been done by Boll & Brune (2015) suggesting that people's needs change depending on the age group as they studied people aged 55-75 and 75+. These factors and others need to be considered when providing robust and efficient user interfaces. The author acknowledges this as being important and a large subject in itself and it is currently outside the scope of the thesis. This thesis is primarily concerned with providing the tools and frameworks to support such systems, where a well designed

user interface could be added. The user interface is more of a psychological and sociological question based on the best design for a system to convey the appropriate information to the patient in a way that they can readily understand.

1.3 Research Aims

The main research objectives of this thesis are to find potential solutions to the first three problems stated above. The aims are presented in more detail below. The fourth problem of creating a useful health related user interface for presenting information back to the user is beyond the scope of this thesis and should be addressed in future work.

Aim 1 - Develop a Data Management and Analysis Framework.

This goal addresses the design and development of a framework to ingest the data from variable data sources. The system should be designed with change in mind, and be able to fit the users and the different applications of these systems. Data from different sources needs to be combined to create new augmented results in a reliable way. The design of the framework needs to be able to track the history of the stored elements and their dependencies and to put process management and the data genealogy at the heart of the system. This will allow a fluid approach to problem solving while keeping the steps re-traceable. The system also required a data structure which can store many different types and sizes of information in a structure which can be interrogated with speed and efficiency. Together this should create a system which can dynamically and easily adapt to change, while maintaining data integrity.

Aim 2 - Develop a data capture device and process high quality information from data streams.

This aim is to design and develop a portable medical data capture device that is easy to use, which can be compared to other devices to show that it can indeed

produce useful signals for further processing. The device also needs to have an accurate and simple data logging system to allow the data to be sent to the data management system.

The next task is to process the data from the device to produce high-quality information. Thus improving the data management and analysis system substantially by locating and promoting only the high-quality data. The data management solution from the previous aim will be used. This will allow the integrated analysis of the data collected.

To show an example of further analysis and data chaining the signal quality measures, will aid an investigation into blood pressure (BP). Blood pressure is the one vital sign where it usually requires a cuff based measurement which can be time consuming and uncomfortable to obtain. This could be improved by using an electrocardiogram (ECG) and photoplethysmography (PPG) signals to estimate it but this requires the formulation and testing of models for estimation.

1.4 Thesis Contributions

The contributions of this thesis based on the research aims described previously, are to develop the following:

- A Data Management and Analysis Framework.

The data generated must be managed by a framework that best harnesses the data provided so that it can be intelligently utilised. The framework supports a block chain approach to managing data processing and analysis by creating graph based data flows, created dynamically by the running program performing the analysis.

- EIMO Vital Sign Capture Device.

The design and development of this device, as shown in chapter 6, highlights the contribution of recording and reading ECG from a handheld portable device with second level synchronisation and direct comparison with a four and six lead ECG. This includes the improved sampling of the PPG with low jitter

to measure the pulse transit time (PTT) on a beat-by-beat basis thus enabling the estimation of the BP measurement.

- ECG and PPG Signal Quality Analysis.

The signals captured should be assessed for their quality so that the most efficient use of bandwidth and storage can be made. To this end, an investigation must be performed as to the best ways to assess the quality of a signal, then to finally use the annotated signals to find the best models and features to accurately estimate BP based on device data captured from a study.

- BP Feature Comparison and Estimation.

This illustrates a full pathway from data ingestion through data preparation and signal quality estimation to features which can be selected based on their quality to achieve two things. First to compare commonly used features in the literature to allow the data mining aspect of the system to be utilised and second, to perform regressive model analysis using the framework to estimate the BP from the best of those features.

1.5 Organisation

The thesis has been organised in to four main parts. The first and second parts present the data management framework, and the application of this framework by gathering and analysing data from a device which was designed and tested. The second two parts form the conclusions, references and further information in the appendices. The four sections will be described below.

Part I describes the data management framework. The background and context for the framework described in chapter 2. This leads to the main design of the system describing the three main services and their interaction in chapter 3. Finally, chapter 4 describes the current state, limitations and suggestions for further work. More information about the implemented prototype can be found in appendix D.

Part II then takes the framework designed in the previous part and applies it to a prototype data-flow in a health care context. First some background is presented in chapter 5 about the main type of health care methodologies and how information technology can play a large role in patient-centred care. Chapter 6 describes the development and testing of a device able to record the basic vital signs of users for further study in the next three chapters. Chapter 7 discusses a signal processing framework to segment and assess the quality using human annotations, validating the annotations with a placebo controlled trial to establish a signal quality baseline. The signal processing system is then further developed in chapter 8 to add objectively defined features. The features and models are then tested in a model analysis framework. The best signal quality models are then used in chapter 9 to perform an analysis on the data from the device, to estimate the modelling accuracy of the patient's BP, comparing the features commonly used in the literature, and the performance of the device. Chapter 10 discusses the work done in this part with the current state, explaining the limitations and where further work could be directed.

Part III concludes the thesis in chapter 11 by bringing the work together and discusses the final outcomes against the aims above and concludes the thesis. The bibliography is also provided in this part.

Part IV provides supplementary information with four appendices. Appendix A contains the mathematical notations used within the thesis. Appendix B provides information on the log files and data structures used in the device study. Appendix C provides information about the two main studies in order to aid the documentation and replication. Appendix D is a brief description of the prototype implementation of the framework.

Part I

The Data Management Framework

2 Background

2.1 Overview

Data management issues arise from the amount of raw data systems have access to. This is evidenced by the development of the EIMO device in chapter 6, along with current developments in the internet of things (IOT) and the advent of industry 4.0. This has increased the requirements for managing and analysing the data produced from a more heavily connected world, which can simultaneously generate more data.

This has led to the development of a general data management and analysis framework, with an architecture that can be used to store, process and link many types of information. The system has been prototyped and used for the experiments in this thesis. This chapter explains the background and concepts surrounding the system design, with the next chapter discussing its design and implementation.

Data storage and management systems have been reviewed to uncover their strengths and weaknesses, so that they can be addressed in the design stage. The main themes are discussed in the next sections. Commonly used database applications are then compared, as understanding these systems would allow the utilisation of already present storage systems.

2.2 System Architectural Design

The formulation and implementation of practical, robust architectures for the analysis and management of large datasets can be illustrated by work done by: Zualkernan & Shouman (2008), Al Saiyd *et al.* (2009), Ong & Khaddaj (2010) and Yang *et al.* (2008), where they discuss different aspects of the frameworks. The main points

are that these frameworks are intended to be distributed systems to promote concurrency and reliability, while some deal with the ontological aspects of the data that they store allowing better searches. This shows that there has historically been interest in developing systems that can combine ontologies with data.

Chen *et al.* (2016) published a review of frameworks for data storage and processing. In this review, they analysed different ways of storing data by describing tensor networks and graph-based structures in more detail. Deserno *et al.* (2012) describes a dynamic design for a data storage and retrieval system prototyped for medical imaging data, with associated ground truths for that system. The interesting aspects of these, are that the systems deal with the storage of the data to be used in the machine learning system and that this processing is performed within the framework.

Taking these ideas further, data agnostic objects and the management of the process's results could be added into the scheme. Further to this, the graph-based data structures are useful for describing data as well as ontologies, and can be combined with tensor networks as seen in work by Cichocki (2014). Combining these structures allows a single graph structure to represent data, program and process relationships. This graph structure of the relationship between the data and programs represents a data genealogy as it would show the family tree of the data and its analysis.

The processing and management capabilities could be extended by allowing the results of a process to be placed in context with the methods used to calculate them and the input data source. The context surrounding the process and result allows, for example, an inaccurate program and any nodes influenced by the program to be removed from the system without the prior knowledge of the characteristics of the process. Further, if a program is superseded by an improved version, it would be easy to identify the results that needed to be recalculated.

Some of these principles have been put together as a project undertaken by the Apache Software Foundation called Spark¹; this project normally runs within

¹Information on Apache spark can be found at <<https://spark.apache.org>>. Author: Apache, Retrieved: 2016/10/24

distributed cluster systems, like Hadoop and provides functional programming elements operating on a data element called a resilient distributed dataset (RDD). The RDDs are created by importing data from a file system into one or more of these elements and scheduling operations within them. These operations can be cached by using in-memory buffers or new files that are created to preserve this information outside of the system's operation. The movement of files offers many ways to lose or confuse data sets or results, therefore potentially losing the consistency of the data. The system allows the operation to be performed as tensors on the data, but it does not track the run time of the operations and how the data results evolve. This system can perform interesting processing structures, for example: running map-reduce functions easily, but the data still has to be managed manually before and after the analysis. The framework below addresses this by using the flexible storage in the NodeSR and by using the graph structures for data organisation. This allows operations to be built out of reusable pieces and allow work-flows to evolve as the data comes in and new transformations are defined.

This flexibility of storage types requires management; this could be aided by using the domain-driven design of the application specific interfaces applied in order to deal with the data within, in the right contextual way (Al Saiyd *et al.* , 2009).

2.3 Big data Systems

Discussions on storage and data management tend to have a lot in common with big data systems, where 'big data' simply refers to processes dealing with massive datasets (Hu *et al.* , 2014; Sagiroglu & Sinanc, 2013). These systems can be described using three terms: Volume, Variety and Velocity. These terms are a convenient short hand for describing big data systems (Sagiroglu & Sinanc, 2013; Cichocki, 2014). Volume refers to the total size of the data being stored and retrieved. Variety refers to the many different types of data requiring storage along with the task of storing structured, semi-structured and unstructured datasets. Velocity can be harder to define since, like its physical counterpart, a high velocity could mean a lot of data moving slowly, or a little data moving fast. Cichocki (2014) adds a fourth, veracity,

which refers to the quality of the data. This is a property of the data itself and not necessarily to big data systems uniquely.

Different systems have been developed in order to address these properties. Most systems address the volume aspect directly, since having a fast storage system can allow for the storage of a variety of data with data encapsulations. This indirectly tends to improve the data velocity limit of the system. The discussions have been broken up into two sections. The first explores architectures and issues for distributed systems, the following section looks at communication protocols.

2.3.1 Distributed Architectures

An important aspect to consider for the design of the framework, is where the computer resources and control will be located. This question prompts a split of the debate into two broad kinds of architectures: centrally managed or fully distributed. Wu *et al.* (2009) uses a modular service-based system, where the data is gathered and processed through a central service, acting as a manager. However Lu *et al.* (2010), Isern *et al.* (2011) and Duque *et al.* (2003) have chosen a far more distributed route with grid-based architectures; this allows for better scaling and improved robustness. Modular systems can be very flexible and this is what underlies the power seen in grid-based computing. These systems have to be modular in order to cross computer resource boundaries. By contrast, centrally managed systems tend to have bottlenecks and low redundancy which are points of weakness. These bottlenecks should be kept to a minimum and designed out in the early stages if at all possible. The main reason to use centrally managed systems seems to be that they are easier to administrate and keep consistent.

Grid-based systems could have their responsibilities separated into storage and processing parts, which could be run on separate hardware, making the resultant system more flexible as it could utilise heterogeneous hardware. Heterogeneous systems use computers which do not have the same resources or capabilities. There are two main types of resources: data storage and data processing. The storage resource can be defined as the available data storage capacity available on a com-

puter. The compute resource can be defined as the available processing capacity on a computer. An example of this is that a computer can be designed to be efficient for both storage and retrieval of data, but poor at high demand algorithms; this would have a high storage resource capacity. Another computer might have many high-speed cores and plenty of memory, but very little attached storage; this would be a computer with a high compute resource capacity.

The graph data representation could be a point of weakness if this system could not be similarly scaled up for large and distributed data sets. Work done by Spyropoulos *et al.* (2016) shows a system for the distribution and co-location of graph data across multiple graph database structures by using query rewriting techniques. This work shows that distribution and combination graph databases are possible; even showing performance improvements. This illustrates one case where utilising this representation should not be a barrier for its central use as the chosen representation of the framework below.

2.3.2 Communications

An interesting overview of communication systems and protocols is described by Varshney (2007) where he points out that the desirable qualities of a wireless network can be used to implement a pervasive medical system. These systems consist of many mobile devices, and the benefits of using ad-hoc networks and the use of more than one type of protocol network keeps the system flexible. The author refers to the communication systems and how it offers both robustness and provides low-level context awareness. Other forms of communication, as described by Lu *et al.* (2010) demonstrates, as with any communication system, that the design should be structured as a service based architecture to allow the various devices and components to self-organise. Service based systems can allow automatic service discovery, interfacing and automatic fail-over, which can be made transparent to the user. This can be realised by using pre-existing open communication systems such as MQTT, ZeroMQ and the representational state transfer (REST) architecture. This opens up many possible lines of communication between programs and services.

The REST architecture, first defined by Fielding (2000), has been used by many companies to drive their API interfaces such as Dropbox², Neo4j³ and others⁴. This architectural design defines a client-server relationship where each action is atomic and should be completely defined in the request and no state is held in the server. This allows different servers to handle requests, so that the number of servers can be scaled up to match the incoming requests through load balancing without having to transfer the state of a process between request servers.

2.4 Application Specific Storage

There have been a number of different storage solutions developed for distributed systems (Lu *et al.* , 2010; Wu *et al.* , 2009; Ma *et al.* , 2010; Duque *et al.* , 2003). Types of storage systems used in the literature range from the simple system described by Wu *et al.* (2009), where it is explained as an internal database with component classes written to provide service-wide access and security, to much more intricate systems. An example of a more intricate system is discussed by El-Sappagh & El-Masri (2014), with data sharing between the distributed aspects of their system using a decision query, which can be sent to other hospitals to recover records for patients. This makes the knowledge base and the medical records easy to access, but this system is run between hospitals which requires synchronising techniques between data centres. Lu *et al.* (2010) shows a more classical distributed storage system based on Chord (Stoica *et al.* , 2001), but is improved by using a two layered distributed search algorithm to speed up the recovery of files within the system.

The systems with the greatest scope are the fully distributed grid based or Peer-2-Peer based storage solutions, for example Chord (Stoica *et al.* , 2001) and bit-torrent, because they provide necessary redundancy, and can be tailored to have

²The Dropbox REST-API can be seen at <<https://www.dropbox.com/developers-v1/core/docs>>. Author: Dropbox, Retrieved: 2016/09/05

³The Neo4j REST-API can be seen at <<http://neo4j.com/docs/rest-docs/current/>>. Author: Neo3j, Retrieved: 2016/09/05

⁴Google Docs REST API <<https://developers.google.com/drive/v2/web/about-sdk>>. Author: Google, Retrieved: 2016/09/05 and the MongoDB REST API <<https://docs.mongodb.com/ecosystem/tools/http-interfaces/>>. Author: MongoDB, Retrieved: 2016/09/05

high performance with a small resource footprint per node when scaled to big data volumes. An example of this is bit-torrent, where each node only sources the file it currently holds, while the clients only pull in data to build the files required, but over the whole network, millions of files can be shared together. Other distributed systems like ‘The Onion Router’ (TOR), while not used for storage, showcases the robustness and scalability of a distributed system. In this regard, these storage solutions offer high redundancy but have greater costs associated with their setup and infrastructure, as more care needs to be taken when creating the architecture. Other commercial database solutions are described next; these provide profitable examples of data integrity and distribution as well as being easy to setup and access.

2.5 Database Systems

There are many databases available for data storage. The systems described below are general purpose data storage systems, each having their advantages and disadvantages which are summarised below.

MySQL

MySQL⁵ is a functional and well-tested relational database server, based on tables of data. This makes it fast for queries as they are internally optimised, but it can be hard to modify the table’s structure quickly and efficiently. The table structure requires that all records are the same, so this requires that columns be added or removed for all, when maybe only a small subset of rows in the table require that particular attribute. MySQL, being a tried and tested server system, supports concurrency and multiple client connections and is used to provide the storage to many websites.

⁵The MySQL specification can be found at <https://www.mysql.com/products/enterprise/techspec.html>. Author: MySQL, Retrieved: 2016/09/04

SQLite

SQLite⁶ offers local flat file data storage without the overhead of a server to administrate. It does however, suffer from some performance issues, as it is not normally thread-safe and cannot support multiple clients easily. It also requires operating system file locking, with the threading support to be required at compile time. In its favour, it is simple to use and does not require a server to setup; it also offers reasonable performance on single thread queries, as compared to other flat file storage systems. Most importantly, SQLite allows for the use of SQL data queries, familiar to many developers, in a local storage medium.

Neo4j

Neo4j⁷ is a popular graph database server system. It has its own query language called Cypher, which makes accessing data within the graph and graph traversal easy. However, the nodes are optimised for graph traversal and search, but are not designed for holding great amounts of data within each node. This can lead to the system struggling with large datasets within each node, as the system assumes that the number and size of the properties within each node are small. The system also returns all of the data associated with each node. This is inconvenient because not all of the data might be required which would lead to inefficiencies, in the overall storage system.

Titan

Titan⁸ is a distributed graph database which allows a graph representation to be stored within different storage back-ends, it seems to only use one of these at any one time. The framework described below is designed to be able to distribute the data between many different types of storage, utilising the best depending on the data.

⁶Information on the limits of SQLite can be found at <<https://www.sqlite.org/limits.html>>. Author: SQLite, Retrieved: 2016/09/04

⁷More information on Neo4J version 3.0 can be found at <<https://neo4j.com/blog/neo4j-3-0-massive-scale-developer-productivity/>>. Author: Neo4J, Retrieved: 2016/09/04

⁸More information on Titan can be found at <<http://titan.thinkaurelius.com>>. Author: Titan, Retrieved: 2017/02/19

The Titan database however, does claim currently a large number of concurrent users and distributed server performance. The language used is the Gremlin query language and the database itself also claims good graph traversal properties. The main drawback compared to others and the framework described below is that it does not possess data processing or data management qualities other than the basics afforded by a graph database.

MongoDB

MongoDB⁹ is a document-orientated storage database server system that uses a variant of the JSON language called BSON to serialise and store data without a table structure and record schemas. It also accommodates a very flexible storage system, but due to the constraints of the BSON documents, there are internal issues that means a document cannot grow past 16Mb as per its manual. To counter this, developers use a file system storage for bulk data called GridFS, which has to be coordinated with the document store by the client manually.

Storj

Storj¹⁰ is a distributed peer-2-peer storage system. It is currently only in the beta testing stage but it allows its users to share storage. The system breaks data up into shards, encrypts the information within a shard, then distributes the shards across the network. This keeps the data secure, reminiscent of the data sharing methods of bit-torrent. This creates the same type of storage system similar to companies like Amazon S3 and Dropbox, but without having to trust the companies with protecting the data they hold.

⁹More information on MongoDB can be found at <<https://docs.mongodb.org/manual/core/document/>>. Author: MongoDB, Retrieved: 2016/09/04

¹⁰More information on Storj can be found at <<https://storj.io/index.html>>. Author: Storj, Retrieved: 2016/10/03

CouchDB

CouchDB¹¹ is a document store similar to MongoDB. This system created by the Apache software foundation, is based on JSON documents and handles requests by using queries to select the documents required and serving the documents found. The notable point about this system is that queries use view functions in JavaScript to run map reduce processes of the data within the database.

PostgreSQL

Postgresql¹² is a SQL database similar to MySQL. The internal storage system is more flexible than other SQL databases as it can easily run stored procedures in other languages. This allows code to be run either when asked or when a trigger event is seen such as a table update or row deletion. There is no oversight as to what these procedures do and how they interact and so must be documented individually. The environment used within the procedure is rather restricted so mapping and translations could be done easily but there is not much support for more complicated data processing algorithms, even though, in theory they could be run.

Summary

There are more data storage systems in existence and in use. For example Redis¹³ and Blazegraph¹⁴ or the big data systems mentioned earlier. Those described above are just a sample of the different types of storage systems available.

These different databases and data storage systems each have their own feature sets so the design and data-flow in the framework presented below is designed to be able to hybridise them. This allows the ability to harness multiple databases

¹¹More information can be found at <<http://couchdb.apache.org>>. Author: CouchDB, Retrieved: 2016/10/03

¹²More information can be found at <<https://www.postgresql.org>>. Author: PostgreSQL, Retrieved: 2016/10/03

¹³Is a memory based key-value store, more information can be found at <<http://redis.io/topics/introduction>>. Author: Redis, Retrieved: 2016/10/03

¹⁴Is a property graph using RDF for the semantic web with commercial licences, more information can be found at <<https://www.blazegraph.com>>. Author: Blazegraph, Retrieved: 2017/06/03

transparently and flexibly, in order to get the best out of them, without having to lock into any particular database.

2.6 Summary

This chapter has detailed the background surrounding the framework's design. The main issues raised here are that there are many different storage techniques, each with its own advantages and disadvantages, and the overall architecture should put the data genealogy and integrity first with the storage patterns used. The framework needs to have an approach that balances the current storage solutions, while gaining the best out of them, and allowing the use of new storage systems using well defined interface methodologies. The design and implementation of the data management and analysis framework is discussed in chapter 3.

3 Data Management Framework

3.1 Overview

The previous chapter explained the background and concepts surrounding the design of a general data storage and processing framework. The proposed structure, data flows and designs are discussed next. The chapter ends with a summary of the system with the current progress of the prototype system which has been used for the experiments in this thesis described in appendix D. The components described have been used to process and estimate the signal quality and blood pressure later in this thesis, but could equally well be used for managing the processing of images, text or other signal data. The main contributions are the proposed designs for services and modules that will enable the combination of data storage, computation and management, enabling the production of a data genealogy and data-flow graph within the system.

3.2 Framework Structure

The development of the framework has three main motivations for the design of the framework system. It:

- Was an immediate way to manage the analysis of the study data from chapter 6 device.
- Removes obstacles when designing systems which require flexibility in the use and location of storage and computational resources, to take advantage of available computer resources, new systems and better techniques.

- Is used to manage the data processing process creating better visibility and chaining of data through to the results. Programs are run against the data within the database while managing different lines of enquiry and process influence.

In summary, the framework is designed to allow flexibility of how and where the system is distributed across computers and resources, and to embed process management strongly into the design. This method extends and combines some of the ideas about graphs, block chains and tensor systems like Apache Spark mentioned above increasing the accountability and computational flexibility of the system. This improves the data and process management within the system, creating the history of the data and results, building a data genealogy within the graph. The data sources for the framework are purposefully left open as the system should be data agnostic so that it can work with time-series signal data or text through to images or video.

The system developed below is a prescriptive design framework detailing the roles and responsibilities for the parts of the ecosystem. This allows implementations to be developed based on this to suit the technology and purpose. The current prototype system implementation used to run the exploration and analysis later in this thesis is described and summarised in appendix D.

3.2.1 System Objectives

There are important objectives to be met by a system to allow it to manage a large variety and volume of data. The objectives for the system are to be:

1. Internally data agnostic in their management and organisation.

This objective addresses the data variability, since data types for medical systems can vary so much, from signal data through to images, there should be a method of encapsulation within the system, so that it can take and store any data required. This allows a partial relaxation of the data types stored in the nodes because it can be encapsulated, and the system preserves the

overall integrity of the data block, no matter what it contains. The encapsulation would allow the storage and use of structured, semi-structured and unstructured data, within the same overall data storage representation.

2. Designed for horizontal scaling.

This objective is for the system to be designed with internal structures and processes that allow for growth horizontally by adding more nodes rather than vertical scaling, where each node has to be improved. The design should also allow the reshaping of the system as requirements change and technology allows. Big data systems deal with this by allowing multiple computer nodes to be harnessed together. This could be improved by having system concepts that are designed from the ground up to be scalable, such as graph based relationships and node locking for immutability. This also allows for different data scales to be accommodated and to improve data and code dependencies.

3. Designed with a data processing and analysis ecosystem.

This objective looks directly at the management and use of the data contained within the system. The first two objectives allow for a great variety and scalable volumes to be stored. Data is of limited usefulness if it is difficult to access, so methods of processing the data internally should be added to the system so that the data can be left in place and managed throughout the analysis. However allowing data processing and mutation to happen unguarded can run into problems without structures in place to manage and monitor this internal analysis; data integrity can suffer if the processes are not checked and monitored. The framework should have methods of process management designed in, which can translate the use and processing of the data into trackable assets of the system itself. Thus allow programs to be monitored while in use.

3.2.2 System Overview

The framework that has been developed is structurally divided into two domains and three defined types of services based on a micro-service architecture as seen in

figure 3.2 and figure 3.1. The first domain consists of the NodeSR and ProcessCR services. These are general infrastructure services for storage and compute resources respectively. The second domain contains the third service, which is an application specific front-end service defined here as a template in section 3.5 to support and encourage new services to be added to extend the system. This domain can have more than one application specific service or none, if the infrastructure services are sufficient. This allows for application specific user interfaces and data structures to be built into system components to improve the interaction speed for clients and users. The creation of these domains allows for a modular approach with defined interfaces between the services. The modules can then be more easily documented, improved and upgraded as required, with well definable responsibilities.

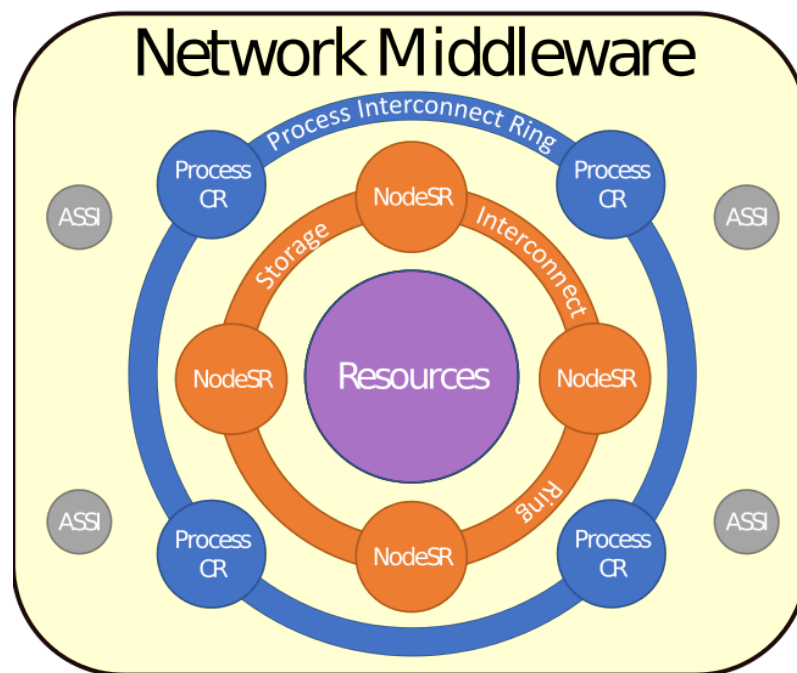


Figure 3.1: Diagram showing the Jelly DL system. Each service exists in an open structure, all sharing and embedded in the network middleware as represented by the yellow area. Each service serves as an interface framework to utilise the central resources appropriately. The rings between the service types show that each service should be able to pass messages between siblings, enabling the system to intelligently solve storage and processing problems while keeping the structure open and distributed.

The first service is the base storage system called NodeSR. Its responsibility is for the storage and retrieval of data and the graph structure it is contained in. The chosen structure, as discussed earlier in section 2.2, will be a property graph

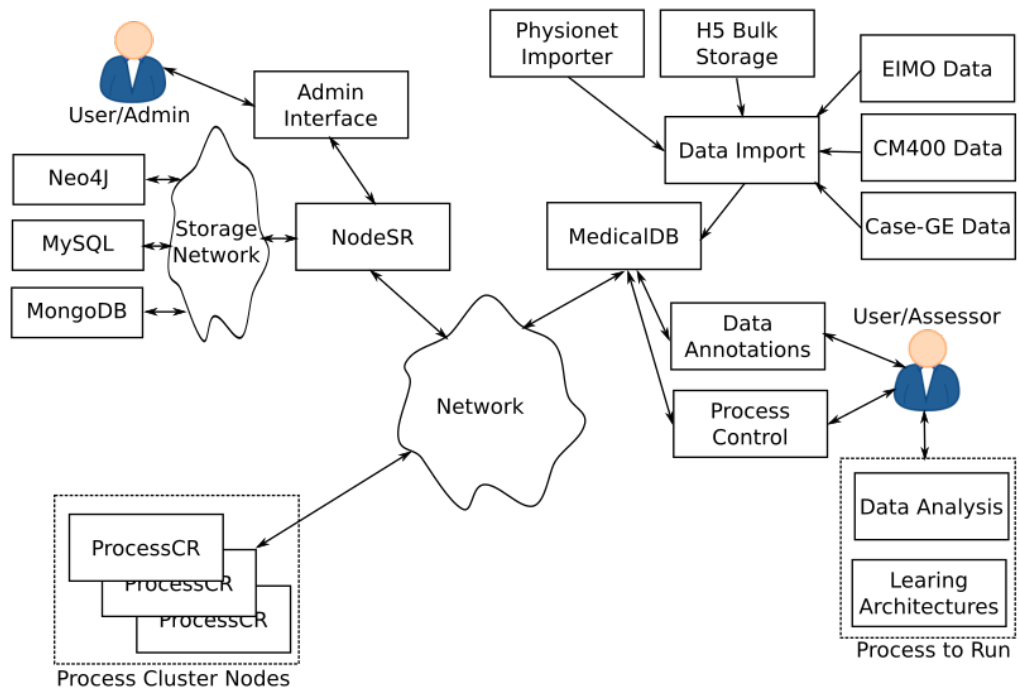


Figure 3.2: Diagram showing the network map of the system services. Each service exists in an open structure, all sharing open REST interfaces over the network. Each service has its own specialities and management capabilities.

structure because it is the most flexible. The graph data structure facilitates the storage of the data as well as the process management by building in a tensor representation into the graph structure. The structure leads to a strong internal data structure and storage framework which can deal with incoming and outgoing data as repurposable nodes in the graph, then links them together to capture the knowledge of the data relationships and the analysis that has been carried out on it. This satisfies the first objective for dealing with data variety. The second objective requires it to be designed as a lightweight service for existing scalable databases, so that many service nodes can work together with other common systems that would be well suited to balancing requests to the service's API. A simple example would be systems like Nginx¹. The service abstraction allows internal signalling if required, between copies of this service. The objects can be then stored and recalled through a standard API protocol available at each NodeSR service, creating an easily reusable and flexible system, as described further in section 3.3.

The second service is the compute service called ProcessCR. Its responsibility is

¹More information can be found <<https://nginx.org/en/>>. Author: Nginx, Retrieved 2017/03/05

for initiating the running and control of the process tasks, as well as for reporting the status of the resources consumed by those processes, while managing the allocation of resources and environments to new processes. This builds on to the graph structure of the NodeSR service and the tensor representation to store the knowledge of processes and their results. The design, structure and operational aspects are explained in section 3.4.

The third and last service is the application specific service interface (ASSI). These are definable server systems or micro-services that handle the specific requests for data and user interfaces. In section 3.5 the application interface design ideas are discussed with examples of the current implementation seen in appendix D, which was used for the experiments undertaken in this thesis.

3.2.3 Service Communication and API

The interfaces to the services all use the REST pattern over the HTTP protocol. The REST system, as describe earlier in section 2.3.2, is being used in many applications. The REST architecture allows all of the services to be easily accessed from many different types of application. Many programs, programming languages and systems now have the capability to access web resources.

All service requests use uniform resource identifier (URI) endpoints with structures and queries designed to be simple to document and consistent across the different services because they are derived from a common template explained below. The system uses the URI to resolve two issues in web based application programming interfaces or APIs. The first is knowing the version of the interface, so as to understand the dialect it uses and the requests it is capable of. This is defined in the URI path using the "<version>" tag, which means the service could run different versions of the interface API simultaneously. The second is to have an intuitive organisation of resources that allows the description of the endpoint to be partially inferred from the terms used in its URI. The main template used for the endpoint route in this framework is shown and described in template 3.1, where the template

parts can be replaced by the service or interface required. The "<tokenID>" is a process management identity which will be explained in section 3.3.2.

URL Template:

```
/<service_name>/api/<interface>/<version>/<graphdb>/<verb> (cont)
  [</resource>[</identity>]][.<type>] (cont)
```

With GET or POST query string parameters as follows:

```
[? [<attribute_name>=<attribute_value>]* [&limit=<limits>] [&token=<tokenID>]
```

Template 3.1: The URL template for the framework API. Where: "[]" and "[]*" are optional parameters of the route, the star means that the option can be repeated as required; "<text>" are replaceable template tokens; "<service_name>" is the service name within the system for example for the NodeSR service "node" for the ProcessCR "process" and for the MedicalDB "medical" can be used; "<interface>" is the interface type of the service for the NodeSR this could be given as "storage" or "management"; "<version>" is a path compliant version code e.g. V1_0_1 for version 1.0.1; "<graphdb>" is a unique identifier of the graph database to query; "<verb>" is the current request operation e.g. GET, DELETE or MERGE; "<resource>" is dependant on the verb given usually it is the resource requested e.g. Node, Line or Subgraph or can be the component name; "<identity>" is a unique identity which is usually the generic universal identity (GUID) of a resource such as a node or link, or the name if kept as unique, either of which can then act as a permanent URI for resource; "<type>" is to allow the request to be answered in different dialects or formats such as JSON, XML, HTML or PY, allowing the resources to be packaged up to suit the type specified. Other parameters can be sent as a query string or posted form data. The query string is organised into: "<attribute_name>,<attribute_value>" which is the name and value of resource attributes respectively, with value strings treated as regular expressions; "<limits>" is the definition of the limits on the set of resources to be recovered e.g. "limit=5,20" or return with 20 resources starting from 5 using the query and finally "<tokenID>" is the specified token ID passed as a string to use for the request.

The structure of the endpoint template above is used to allow the service to coexist in the same URL if necessary or allow them to utilise a proxy to redirect requests to the appropriate service. It is organised as a tree structure as shown in figure 3.3 so more endpoints could easily be added without disrupting the overall scheme as the new leaves can be made independent of the rest of the structure if required.

Query parameters like those above can also be sent encoded as "form" data in the POST request if the query is large. Table 3.1 shows the kinds of objects the URI expects. Similar to the query string above, these objects can be sent as some key-value attributes, like in the above case or as JavaScript object notation JSON or

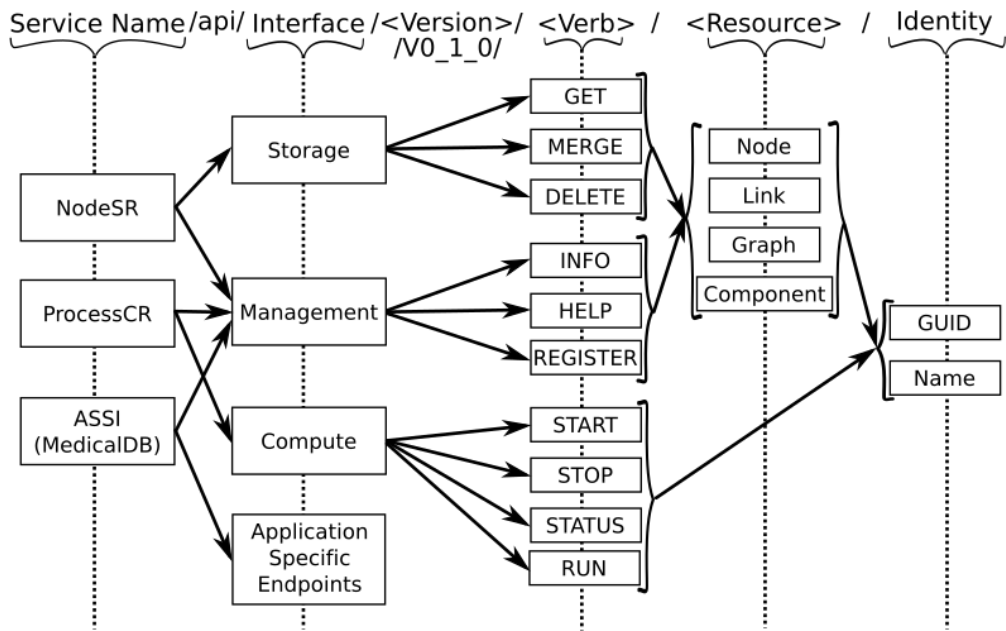


Figure 3.3: Diagram showing the route structure as a tree where each level is assembled as parts of a path. This predictable scheme allows for further development as extra interfaces or verbs can be added while knowing how the others fit together. The application specific service interface (ASSI) is a place holder for other services. The MedicalDB can be used as an example.

extensible markup language XML encoded form data. These objects allow queries to be specified and data to be exchanged.

These endpoints are also defined to run for certain HTTP methods. The common HTTP methods used for REST include GET, POST, PUT and DELETE. These method types need to be preserved for their use to maintain compatibility with other REST services, for example the GET verb uses the GET HTTP method, but can also use the POST method to allow more complicated queries to be sent to the endpoint. Similarly for MERGE, both POST and PUT are accepted as valid methods in keeping with commonly used REST architectures. The URI template acts as the framework for the services, they then specialise this by adding verbs or resources to suit the application as is defined in the services below in sections 3.3.2, 3.4.2 and 3.5.2.

The other main communication subsystem is the methodology for component discovery and registration. Components defined by other services needs to be registered with the NodeSR service so that the defined components can then be used as native types. The system for registration will vary dependant on the final implemen-

Table 3.1: The main structures used to query and store data within the service.

Name	Resource	Description
Query		Objects containing fixed search criteria for a Graph, Node or Link, setup in a graph structure.
	Graph	A sparse description of a graph structure e.g. JSON or XML
	Node	A sparse description of a node e.g. JSON or XML
	Link	A sparse description of a link e.g. JSON or XML
Resource Structure		Objects containing the data for a Graph, Node, Link or registered component.
	Graph	The full or sparse data for a graph e.g. JSON or XML
	Node	The full or sparse data for a node e.g. JSON or XML
	Link	The full or sparse data for a link e.g. JSON or XML
Limit		A limit on the number of resources to return if not given, an internal default is used. X and Y are Integers.
	'all'	String to allow all the resources found to be returned.
	Y	Integer will return Y resources.
	X,Y	Integers will return Y resources starting at X.
Token	GUID	An node ID that must be exist within the main graph requested.
Relationship	String	An optional text string that allows the type of link to be set, Default to 'Influence' when not set.
Component Definition		Containing the definition and mapping of a component to register.
	Mapping	A set of mappings whose key form the property names noting how each property should map to a node resource, along with definitions of defaults and descriptions for each, given as a JSON object. The default can be string procedures already defined in a library, through to a custom program script, the final complexity of the mapping will depend on the implementation.

tation. An example of component use and registration is described in section 3.3, when discussing the API communications.

Service discovery and management is handled by each service instance maintaining a node presence in the graph through their own components. So on creation of a service node it creates or updates a component with a unique name and ID. The resources available to the system can then be queried like any other node in the system. The meta information that ProcessCR service manages includes the process components and data relationships. This information is contained within the links and nodes of the graph and as explained below, these links can be changed, representing the changing responsibilities and data processing history.

3.3 NodeSR - Storage System

3.3.1 Overview

The NodeSR handles the storage of the data elements of the system and their arrangement in a connected graph network. This service addresses the problems of the variety and flexibility of storage, along with the issues and disadvantages of the data systems as discussed in section 2.5, by making one data ontology flexible enough to cater for many needs, managed on many different types of back-end storage. This service is solely responsible for the storage and retrieval of the data structures given to it. The API and user interfaces are explained in section 3.3.2 and in section 3.3.4 respectively. The service currently implemented is described in appendix D.1.1.

The modules seen in figure 3.4 show the API front end interface coupled to the storage manager and the resources. The API receives a query to either find or save some data to the system. Once the query is decoded and understood, the storage manager then translates the query into operations that can be carried out on the connected resources. This will be explained in section 3.3.3.

The modular structure can be scaled, while also allowing room for optimisation and intelligent handling of the requests. In its simplest form, the query could be

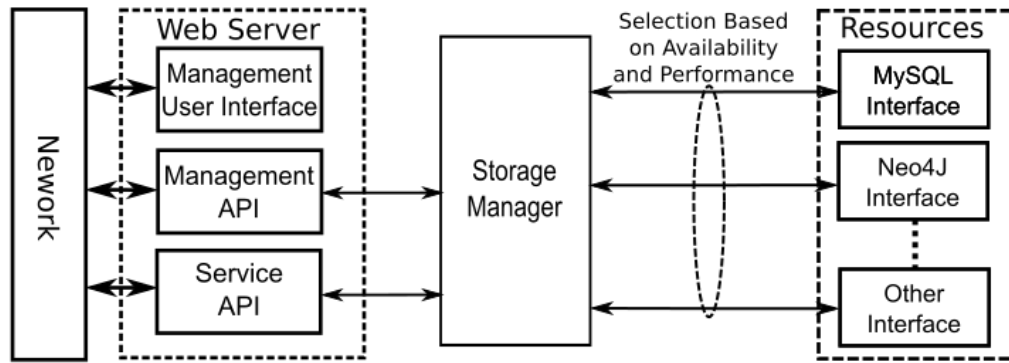


Figure 3.4: Diagram showing the basic architecture of the NodeSR service. This shows the common network interface on the left through to the storage resources on the right. The storage manager in the middle takes requests served by the API through to the most appropriate resource interface.

directly relayed to the resource with minimum alteration, so the system behaves as a resource aggregator, allowing the system to grow in internal complexity. The storage manager discussed below could be made to assess each query and use internal machine learning capabilities to learn which resources would optimally handle each type and size of data given to it. The greatest advantage of this is to the user and developer, as the internal architecture of the system should not impact the end result. The clients can continue to save and load graph like objects, but the system can internally be improving and optimising itself.

The storage architecture is logically represented as a linked node graph as seen in figure 3.5, where graphs and nodes are given their normal representation, the links are extended by using three-link-parts as described in the figure. The node types form the basis of the components that are derived and mapped from the base nodes as defined and used by the other services as described later. Links have an A to B connection as normal, and the C connection allows for the link to have a dynamic node attached to it, orthogonal to the normal link path. Parts of the graph can then serve as ontological markers or modifiers of other graphs while staying within the same queryable environment. This allows for the modelling of many different structures and representations, aiding knowledge capture by the storage system. All of this is continuously searchable without structure boundaries.

The data can then be queried or saved using this structure. This mirrors the operation of graph databases currently released, such as Neo4j, but unlike Neo4j the

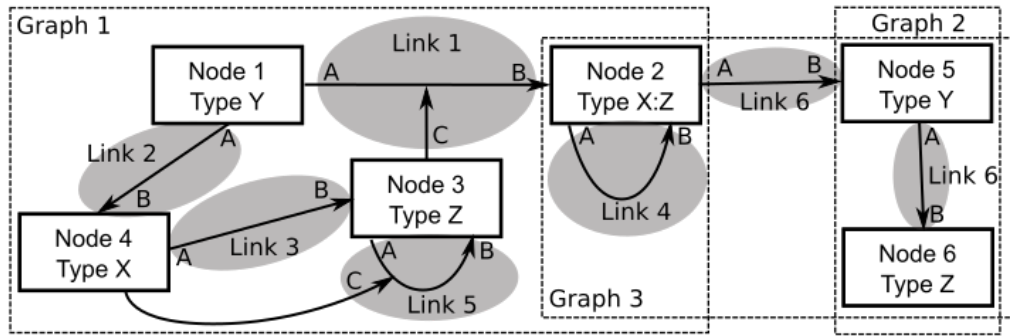


Figure 3.5: Diagram showing the top level representations. This displays the nodes in the the database along with the links connecting them; the dashed boxes show some of the possible sub-graphs, that could be queried. A, B and C are the three parts of a link, as a link normally goes from Node A to Node B but optionally can be linked to a third Node C. This allows part C of a Link to act as Link or Node resource modifiers, dynamic types, or allows responsibility chains. X, Y and Z are type categories, these can be used to type nodes within the system, aiding querying, which allows for the definition of both hyper-graphs and hyper-edges. More than one type can be applied to a node to show its membership to different component types or type groups. Both node and link types and three-part-links aid in preserving ontological representations with the system.

resources used are not fixed. Nodes can be expanded upon greatly, with larger bulk data stored in special bulk storage areas. Unlike Neo4j, there are extra capabilities using the C connection in the links to store more complex relationships. This system has the ability to use different types of storage resources to take advantage of what the users already have at their disposal, such as: Neo4j, MySQL, MongoDB CouchDB or Postgresql. This frees developers from being locked into any particular database system, allowing the translation and migration of information between connected resources, if the NodeSR implementation and current interface allows it.

The storage manager provides objects level translation for resources and components. This can be adjusted or new translations can be added to create flexibility and decoherence from the internal structure as seen through the API to the backend storage systems. Since the logical representations can stay consistent, new storage architectures can be made to improve the performance without breaking compatibility. All that is required is the building of a new resource interface into the database or storage medium of choice. The API interface remains the same because it is defined without direct reference to the resources available and is discussed next. This is followed by a description of the storage manager. The service is then rounded off

Table 3.2: The main verbs used for request to query and store data within the service. These refer to resources or components within the service such as Nodes, Links, Graphs or defined components.

Name	Structures Required	Description
GET	Query, Limit	For the recovery of resources matching the query structure
DELETE	Query	For the removal of resources matching the query structure
MERGE	Query, Merge Type, Resource Structure	For the insertion or merging of resources given by the store structure matching the query structure

by the description of the service management user interface.

3.3.2 API interfaces

The interface can be split into two areas: the data storage and retrieval API and the management and informational API. These areas require defining 5 verbs seen in tables 3.2 and 3.3. The storage, retrieval and management requests once understood by the API endpoints, are delivered to the storage manager within the service. The query and store structures are described in table 3.1.

Storage API

The data access interface is derived from the URI template and tree structure defined in section 3.2.3. Three verbs are required as shown in table 3.2 for the purpose of storage. The query structure allows resources to be found. This is straight forward in the GET and DELETE verbs as only a query structure is required. MERGE uses this to allow the node found to be merged or replaced on the server. The MERGE verb has provision for an options list in the request. It is a system to inform the storage manager of how the merge should take place. If the option is set to “replace” for example the resource structure sent should be used to wholly rewrite the node found using the query. If set to ‘edit’, the resource structure being sent as a difference to the graph resource queried.

To address the process and data management within this system, the service uses process tokens, which are implemented in the API and storage design. The token used can be any normal node within the graph network as only an ID is required to form a link. However for normal use, the type of approved nodes which can be used as tokens can be restricted.

A client requesting a token sends identifying information, which is saved into the meta data for the token and the ID of the token created is returned to the client. When the user queries the service for a set of nodes, the token must be given to the service. Links are then added between the found nodes and the token using a link with a “USED” type. This means that the system knows what data that token, and therefore that process or client has had access to. Similarly when saving data into the service, the token must be given. The service links all created nodes to the token with a “PRODUCED” type. This system allows the information about the operation of a token or client to be monitored without a large overhead or disconnected logs as the links are built into the main data structure. The ProcessCR service extends on this principle to allow distributed compute resources as in section 3.4.

Management API

Apart from the verbs used for storage above, the API of the NodeSR also has three other verbs, as shown in table 3.3 for the purpose of management. INFO and HELP are used to aid management capabilities so that the service can be managed through a web interface. REGISTER allows a component definition to be sent and registered within the service.

The user interface, discussed in section 3.3.4 works by using both the management and storage API. The service can be requested to provide information about the current storage and resource setup, as well as the current status of these resources, through the INFO verb. The HELP verb allows resource or service information to be requested in order to document the interfaces dynamically. This allows both the documentation to be provided to clients when required and further API's which can self discover the new endpoints. The documentation could be rendered as an

Table 3.3: The main verbs used for the management of the service. These can refer to resource and component objects within the service such as Nodes, Links or Graphs.

Name	Structures Required	Description
INFO	Query	To gain information on the service or attribute types for the query.
HELP	Query	To gain help on the endpoints available for this resource, acting as part of the documentation of the service.
REGISTER	Component Definition	To gain help on the endpoints available for this resource, acting as part of the documentation of the service.

HTML page for display or a structured language such as JSON or XML as objects to be read and interpreted by another program. The REGISTER verb works by being given a component definition as described in table 3.1. This definition could be as simple as a JSON or a python class which can be registered and imported within the NodeSR service. In the implementation given in appendix D a python script is submitted to the service. This defines a class with the properties listed and mappings given. Since it is a full python script, access methods can also be defined, to allow more intelligent property access through the storage manger, as discussed below.

3.3.3 Storage Manager

The main module of the service is the storage manager, which takes queries from the API and translates them into the necessary queries for the underlying resources. The processing done at this level involves the routing and translation of information to the resources which are best at storing those types of data. The block diagram of this process is seen in figure 3.6. The interface that is presented to outside systems is based around three main objects, which are Graphs, Nodes and Links as explained above. The purpose of this module is to translate the logical graph representation into structures that are better able to be stored in or retrieved from the connected resources.

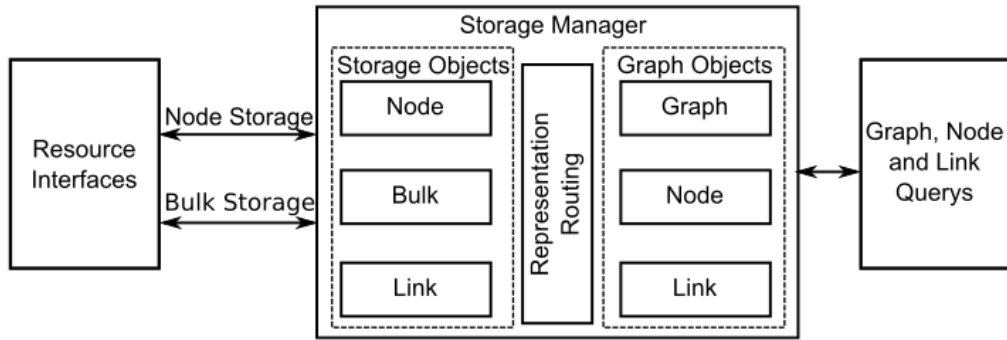


Figure 3.6: Diagram showing the architecture of the storage manager. API add or merge queries are presented to the storage manager, which are translated into storage representations and dispatched to the resource interfaces. This sequence is reversible for data searches.

The storage structure of the system is a form of graph, with two major resources called Nodes and Links with a third internal resource called “Bulk”. These are also shown in figure 3.6. These model components are, in principle, very simple in structure. These will need certain known attributes such as a: unique ID; name; type; object hash and time stamps (creation, modification and access) for each Node and Link. The object hash allows node merging easily; it can check one attribute and see if the basic attributes match, such as ID, name and type. Attributes are managed through Node components defined in the other two services, which can then be added and removed through the API REGISTER verb as above.

The storage resources that are used are dependent on the underlying resources made available to the NodeSR system and on internal choices that can be made by the system. More storage resource managers can be plugged in, as they only need to understand how to store or retrieve the internal representations of the Node, Link and Bulk objects. An example of the structure used for the MySQL connection are detailed in appendix D.1.2 in tables D.1 to D.3 for the internal structure of these objects because they are used in the NodeSR’s implementation at the end of this chapter.

The translation from the external to internal representations are done by breaking down the object to a set of the lowest common denominators which can be either found and returned, or merged by using the resource manager that would be controlling each connected resource. The links can be decoupled from the graph

objects and related nodes to make atomic links to storage in tables or left with the nodes for storage in a graph database. The nodes are pulled together from the graph or collected from the nodes in the query. These are then processed to make node sets with all the bulk attributes, separated into implicitly linked bulk objects. The optimised nodes and links can then be sent to the graph storage resource, similarly the bulk items are sent to the bulk storage resource. Each of these steps require decisions based on the best way to optimise the node for speed and where to store them. The decision logic at this point has access to the multiple levels of representation, so intelligent decisions can be made as to the classification of bulk objects and the best resource to use, or even to start up new resources or optimise search strategies for increased performance.

The details of the search strategies, such as implementing a layered search scheme similar in approach to the one used for CHORD (Stoica *et al.* , 2001) or Google's big table (Chang *et al.* , 2008), and to show how they could be used to improve the performance of the storage system, is a large subject in itself; and is beyond the scope of the current framework. The framework does however bring together the storage of data, along with contextual information, which allows for further improvements to be included within this module.

The translation of the nodes discussed above into components is also handled here so when a request is made for a specific resource, a query is given along with the type of resource specified. A query is constructed to make sure that the component type is among the types that the query is searching for. A diagram explaining how the component properties map onto the under-lying node properties is given in figure 3.7. The component definitions are given to the NodeSR service by the other services. These can then be checked internally for consistency. How this exchange of component types takes place would be dependant on the specific implementation of the system. The current implementation is given in appendix D for example, the MedicalDB and ProcessCR services send the components to the NodeSR using the REGISTER verb as they load up. The components sent to the NodeSR get overwritten so always the last submitted component by each service is used when

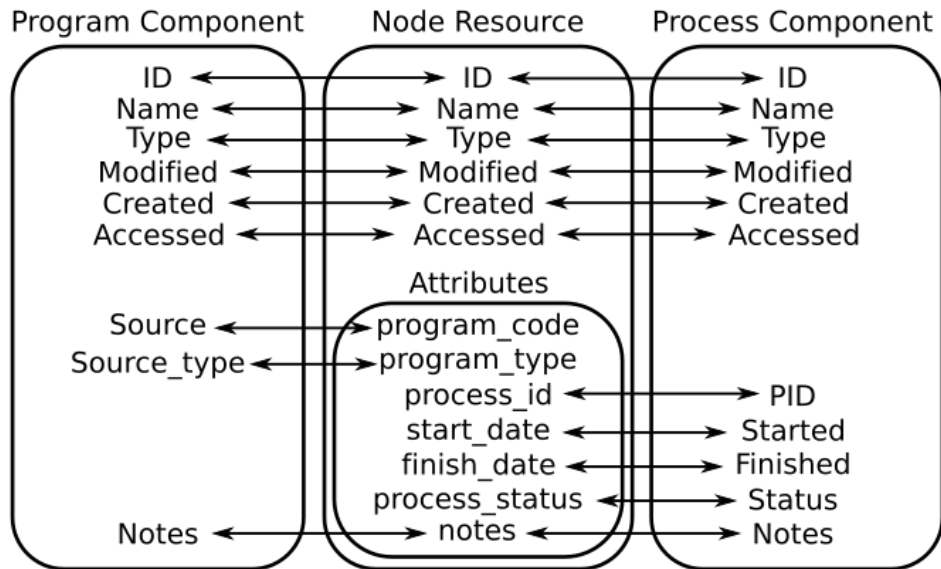


Figure 3.7: Diagram showing the component property mapping on to a node resource in the storage manager. The middle block shows the Node resource properties to the left and right of this are two example components and how properties from these map on to the node properties. Component properties are defined mapping on the node resource through a tree hierarchy, with properties able to map to the same node attributes. This means that these attributes would be shared between the component properties, the example being a note attribute as shown, multiple component types can be active on any particular node based on the type given.

doing the property mappings.

The node types represented within this system allow for component based views to show context specific information within the node, dependent on the component types that the node has been given. The component models then act as the gatekeepers and maps for the data and keep the internal structure consistent. Different components can then either be made to work together and generate emergent properties, or keep the function they serve as independent attributes.

These components can be combined to create the different types and views of the information within the graph that the external systems require. A node could be a 'Raw:Data:Annotation' type for example, which would mean that the node would have raw, unprocessed data and also have annotation information available. The properties and capabilities of these components are stored within the component objects, along with the implementation of how to encapsulate and un-encapsulate the data. When storing these types, the contents can be packed and unpacked to create the base resource types of nodes and links so that they are saved into the

NodeSR storage system graph.

3.3.4 User Interface and Management

For easy management of the service, separate management programs lead to restrictions for, when and where the system can be monitored. The system design is intended to use web based management built into the server itself, with APIs and client side scripts. This can allow for easy management and configuration with a responsive interface, taking advantage of the latest website scripting developments. This also aids in operational consistency; if the server is upgraded, all of the management consoles will be consistent. The web based API interface allows a browser to be used to control the system and review results. This is a system of dynamic web pages with built-in JavaScript to allow the interface to restructure itself in the browser client and provide a native feel.

Within this interface there is provision for a low level storage search and editing as well as testing facilities to allow for online setup and configuration of the resources available to the system. This can also serve to present and partially document some of the management API endpoints to the users and developers. An example of the user interface is given in appendix D.1.1, where the current implementation is discussed.

3.4 ProcessCR - Compute System

3.4.1 Overview

The compliment to the storage service described above is the ability to run procedures automatically using the managed data. The service requires the capability to start and stop processes, along with checking their resource usage. The service's job is to run and monitor the programs given to it while providing an interface for status updates and to allow the system to monitor the jobs in progress. This adds the 'compute' service to the system, allowing the framework to both store and process data. The service currently implemented is described in appendix D.1.3. The

design architecture of the ProcessCR is shown in figure 3.8, this shows the connections to the network and the internal logic. The process logic and management will be explained in section 3.4.4.

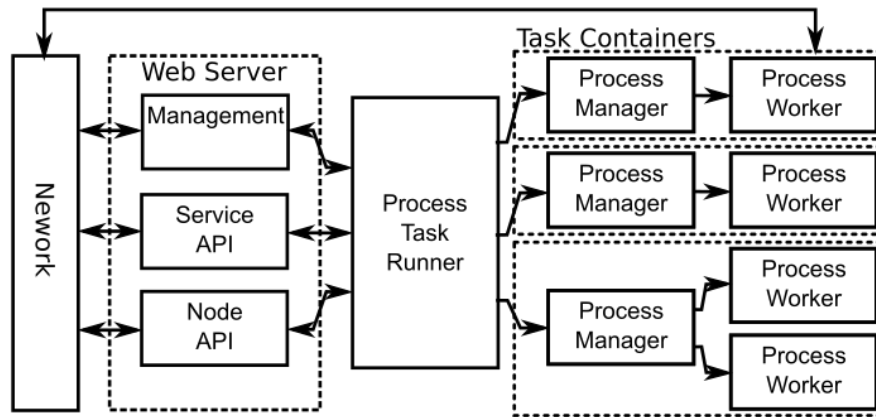


Figure 3.8: Diagram showing the basic architecture of the ProcessCR service. The shared network is on the left, with the process workers being managed on the right. The management and service API allows for interaction with the service. The NodeAPI shows the interface with the storage system for recovering programs and processes. The processes spawned have independent connections to the network and run their own interfaces. The process manager setup is a basic process running environment and passes the process ID as a token into the worker.

The service requires the special components and attributes of nodes to be defined and stored in the NodeSR. The structures shown in table 3.4 are defined for the storage of the information used in this service. Program nodes store the source code or scripts with the dependencies to allow for a complete program structure to be built. This defined structure is then used by the process manager which saves its current state into a process node. This serves two purposes; the first is to allow the parts of the system to see the current state of a process and the second to act as the organising principle to better manage the running of the processes, as mentioned before in section 3.3.2 when discussing the NodeSR tokens.

Results components are a general node to encapsulate the various report messages, files and process logs produced by a running program. They store a report of the program's operation and such files as required to be saved, during the programs operation. More than one result node can be created by a process along with any other node types the process needs to create.

Table 3.4: A list of the components required and defined by the ProcessCR.

Name	Description
ProcessCR	Denotes the ProcessCR instance node.
Program	Denotes a program node to use to store a program source, along with its trigger and estimated duration.
Process	Denotes a process node to use to store information about the internal processing worker.
Result	Denotes a results node to store program result such as files, program logs or intermediary result digests.

3.4.2 API Interface

The interface is constructed in a similar fashion to the NodeSR service above, again using the URI templates defined, in section 3.2.3. There are two main areas which are: the compute and management interfaces.

The first is used to allow other parts of the system to interact with ‘compute’ or processing tasks. The request could be, for example, to start or stop a process or to see the status of current or selected processes. The second is the management of the resources defined by this service. This interface starts with the same verbs and resource as in the NodeSR, then extends this set to the local components defined by this service. This service then responds to 8 verbs shown in tables 3.5 and 3.6 and 6 resources which are the Graph, Node and Link from the NodeSR along with the components defined within this service which are shown in table 3.4. The query and store structures are described in table 3.1.

The searches and requests for program, process and result components are undertaken by the NodeSR service, with the results and queries pre and post, processed by the ProcessCR. Communication and updates from the process managers are routed through the ProcessCR service that called them, and saved using the NodeSR. The API areas are both used throughout the web based user interface for the service as described in section 3.4.5.

Table 3.5: The main verbs used for request to query and start or stop process within the service.

Name	Structures Required	Description
START	Query	When given the ID of a prepared process node, it will be started
STOP	Query	When given the ID of a running process node, it will be stopped
STATUS	Query	To find the current running status and resource usage of the process's queried
RUN	Query	Requires a query of a program node, this will then build the program environment which can then be started automatically, or through the START verb on the returned process ID

Compute API

This part of the API is used for the manipulation of the processes themselves. The endpoints access the starting, stopping and the process status enquiries through the verbs START, STOP and STATUS as defined in table 3.5 under the group of 'compute'. The only requirements for these are the ID of the process token resource node in question, given in "<ID>". These endpoints return simple JSON, XML or HTML responses as defined by the "<type>" property in the URI. The responses can contain pertinent node IDs with the request. This allows other pathways to be used to gather the information, as these ID's could be used to query for that resource. The last endpoint is the RUN verb, which can be sent with a program node ID so that the process service sets up the task environment and runs the task concerned, which is explained in the next sections.

Management API

The second part of the API is used for the management of the service. Similar to the NodeSR, the main endpoint verbs are GET, DELETE and SAVE as these encapsulate the functionality of the NodeSR storage API system above in section 3.3.2. The system should be self documenting and easy to investigate, as each component can be queried as to the properties and defaults to be expected from the service

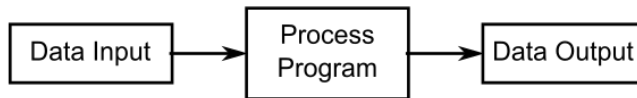
Table 3.6: The main verbs used for request to query and store data within the service. These can refer to Nodes, Links or Graphs.

Name	Structures Required	Description
GET	Query, Limit	For the recovery of resources matching the query structure
DELETE	Query	For the removal of resources matching the query structure
SAVE	Query, Store	For the insertion or merging of resources given by the store structure matching the query structure
INFO	Query	To gain information on the service or attribute types for the query
HELP	Query	To gain help on the endpoints available for this resource, acting as part of the documentation of the service

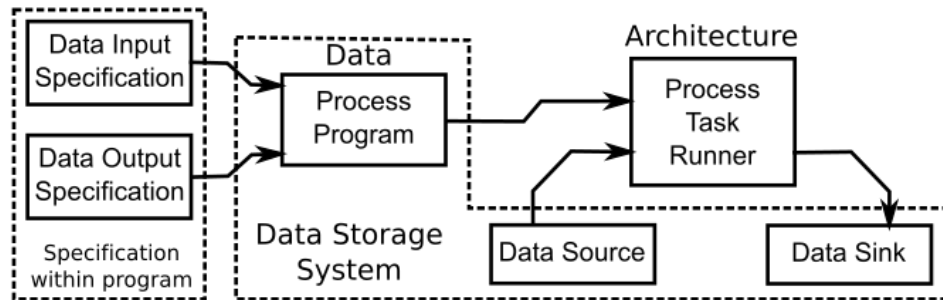
itself using the INFO and HELP verbs as in the NodeSR. The same basic interface is used to keep the overall interface simple and consistent which is described in table 3.6. This shows the verbs and their descriptions of the type of information to be expected.

3.4.3 Programs as Data

Data processing tasks share an underlying structure shown in figure 3.9a which is also used in tensor networks (Cichocki, 2014); this displays the basic structure of a functional process, which applies a program to the inputs and then produces a set of outputs. The process program can be turned into a data input as an abstraction of this process, shown in figure 3.9b, and stored with the rest of the data. This means data processing could be automated and the results automatically collected and correctly associated with the appropriate program source, environment and starting data. This would also allow the tracking of the process's activities and the results produced, which is described in the next section. This abstracted process lends itself to a linked node graph representation that the NodeSR uses.



(a) General data processing concept shown as a connected graph structure.



(b) Abstract system framework for data processing, showing how the programs as well as the data input and outputs can be stored and managed together.

Figure 3.9: Diagrams to show the (a) General and (b) Abstract processing system framework.

3.4.4 Asynchronous Task System

Programs within the ProcessCR service are treated as node resources and are part of the whole graph database with the results linked to them as described above. This makes data processing using this system, self-documenting. Reports can then be constructed with the result nodes, which are linked with the program and architecture that were used to create them. This documentation also allows the system the ability to track data through its lifespan within the storage system; audit trails are then built into the system. The way the graph environment processes are setup and run, is shown in figure 3.10. While the program is running, the graph representation is being updated and evolving within the graph storage system.

Since the programs get access to data in their own environment and dependencies, this satisfies the objective to keep the storage system data agnostic. Only the programs need to know what data they are expecting, and be able to understand it; the data moving through the system can be multivariate medical signals, event log data or even images. The results of the processes are then able to be stored back into the database, whatever type they are, using component types to help with the storage and is automatically linked to the programs that created it.

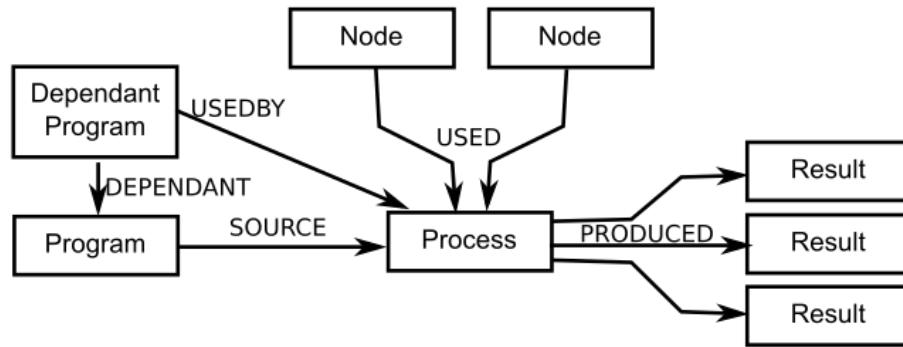


Figure 3.10: Diagram showing the process graph representation. This displays the nodes and links used in a process graph structure. The programs are connected to the process by the ProcessCR as the program environment is built. The data nodes used and result nodes are linked by the NodeSR service as the data is gathered and processed. This shows the total sphere of influence of a program.

There are two main methods for a task process to be initiated. The first is the most accessible as it uses the compute API's RUN verb, given a valid program ID. The second method is internal; the ProcessCR can be pointed to programs within the database system and all programs can have trigger and trigger duration attributes. The return value of these trigger programs determines whether the programs the trigger belongs to, will be called. The trigger programs have a duration given, so that the system has an estimate on how long they may take to run. This can be assessed and inhibited from running if the trigger duration is too high or killed if found to be over running, so that it may free up resources. The program trigger could be a system query looking for a certain type or state of a node or nodes or simply a given date and time as an alarm. The trigger can be left empty so that it cannot be triggered and must be run manually. This system allows programs and processes to be created, which can allow for the automatic chaining of functions and tasks, as the result of the first task could satisfy the query for the trigger of the next. This allows insulated programs to form complex relationships while still having their operations managed.

When a program is activated by either method, the task system is initiated and the program is fetched along with its dependencies which are also stored as program nodes. Once a program and its dependencies are collected, the process manager sets up the running environment and a worker process to execute the program code.

This worker and environment is based on the programs and process information, given or defined in the request. Each worker gets an environment and resources. These can include:

- Service API URLs

These are resource locations for the local NodeSR or other services attached to the system, as allocated by the manager.

- Temporary file storage

This is a temporary directory to act as a scratch pad as the process runs. None of the information here is preserved after the process finishes.

- Process ID

The ID of the process token created by the manager to represent this worker is used as an identity when making requests.

The worker process itself could be simple: where a python process dynamically evaluates the program; or spawns an entire virtual machine, by using either a cloud service, e.g. Amazon or Azura, or just a local operating system container such as the one provided by Docker. These containers are lightweight Linux virtual machines with isolated private environments. The environments can be kept and managed with the programs and the data that they require. The operation of a process within this system is open as it could invoke external larger process execution environments such as Apache Spark. The worker can act as the intermediary and interface between these systems. If the process running within a worker requires resources from the system it sends a request to the NodeSR using the NodeSR API, giving it's environment process ID token as identification and authorisation, similarly this token system works for storing resources. Each worker is assigned to a ProcessCR manager, although this can be dependent on the implementation, one manager could be set to monitor multiple workers if required depending on the level of monitoring required. The job of the managers is to monitor the workers and to keep the process nodes associated with each, updated with the current status of

the worker and process. This gathered information also allows the workers to be stopped and cleaned up after use.

3.4.5 User Interface and Management

The user interface for this service allows the service's current status and resources to be displayed. It is important as it provides feedback for the users of the system as to its current setup and resources. The user interface also allows the API to be seen in use. This helps clarify the documentation by providing working and useful examples of the endpoint used. However, it is important to understand that this interface is only important to some users, which actively need to manage or interact with the tasks running. The APIs described above allow for the integration of the facilities offered into other user interfaces; especially those of the next part of this system, the application specific service interface (ASSI), which is described next.

3.5 Application Specific Service Interface (ASSI)

3.5.1 Overview

This is a template service design which can be built upon and tailored to suit specific purposes. This creates the services where the main users interactions with the system would be located. The system acts as the main hub and micro-service template for creating application specific UI's and interface and internal logic, by using the other two services through their API. This component could be one of many within the framework, and allows for domain specific knowledge to be used at a system level, borrowed from the domain-derived design (DDD) system for software architecture. The architecture is shown and described in figure 3.11. The ASSI connects and configures the framework allowing specific application logic, such as the data import and data annotation interfaces to be used and displayed while providing the logic internally to allow the defined domain components below to be used.

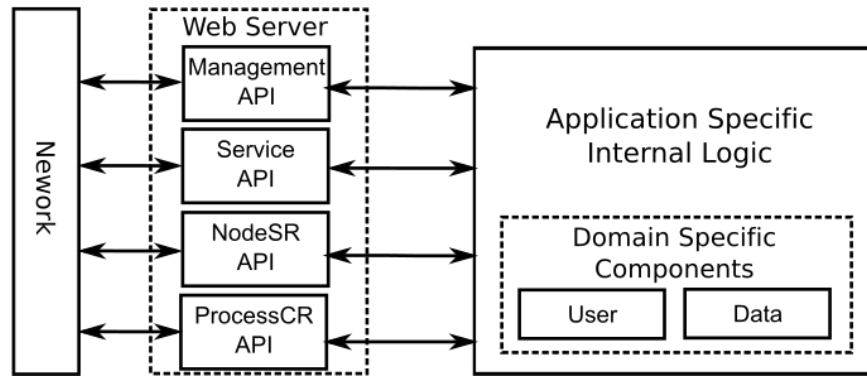


Figure 3.11: Diagram showing the basic architecture of the application specific service interface (ASSI) service. The connections are shown from the network to the service using the service API as well as the other service APIs for the NodeSR and ProcessCR. These connect the network and the specific application logic. The component pool can then be registered with the NodeSR through its API.

3.5.2 API Interface

As in the other two services, the API is derived from the templates endpoints in section 3.2.3. Example domain components defined by this service, are shown below. These are registered with the NodeSR service and so requests can be passed to that service either through a proxy in this service or by directing the request to the API's for the other services and allowing them to handle the request. The basic verbs of GET, DELETE and SAVE are used for the access of these resources if required and INFO and HELP are used to help document features of the service. Other verbs can be defined to suit the application or replicated from either of the services.

Considerations for the final interface are described in section 3.5.4. The main design constraint for the service interface and endpoints, are the requirements of the particular application it is to fulfil. The design of the ASSI services is aimed to streamline the whole system to create a consistent application specific API and service interfaces for integration with other server systems or user front-ends to be built upon. These services act as the main architecture customisation points for its role. This then can deliver an interface that keeps the intended application simple and efficient.

Table 3.7: An example list of the components required and defined by the MedicalDB - an application specific interface.

Name	Description
User	Denotes a user node component.
Device	Denotes a device node component.
Meta	Denote a meta node component. This holds the circumstantial information stored as a semi-structured dataset to tie other data nodes together.
Raw	Denote the data stored are raw signals from a device.
Data	For storage of data within the node so it should have a type and data attribute.
Annotation	For storage of event annotations coupled to data.

3.5.3 Domain Components

The structure of the data internally follows the basic graph database architecture and all elements are based of nodes in a property graph. However the nodes can have multiple type simultaneously, which allows organisational components to stack on to them as in the ProcessCR, thus creating an easy and well structured architecture for further development and structured sharing of some of the properties and components. This allows that each node can be “viewed” as different resource components, using the component objects defined within the service and registered with the NodeSR service. The component types defined are for the application that the ASSI is designed for. The medical implementation discussed in appendix D can be used as an example of the form that these components can take. The main components required for this can be seen in table 3.7.

3.5.4 User Interface and Management

The system management interface is constructed as a web based application as in the previous two services, utilising the APIs described above to drive the controls. These controls are loosely coupled to interface logic controllers in the website to keep it simple and flexible. The site can be designed to suit the particular application, utilising many different ASSIs if required and the two services detailed above;

the open nature of the APIs allow many capabilities for making interface solutions tailored to the specific application required forming a micro-service framework.

3.6 Summary

This chapter has detailed the design of the data management and analysis framework. A prototype has been implemented as briefly described in appendix D to perform the data management and analysis in the next part of this thesis, along with other processing tasks and experiments outside of this thesis. Using and comparing data from the EIMO device in chapter 6 to then annotate, assess and estimate the signal quality in chapters 7 and 8 and blood pressure in chapter 9. The current implementation has three services: the first called NodeSR, which delivers graph data storage to all who require it. The second, called ProcessCR, provides compute resources to stored procedures within the system, while managing data access. The third ASSI called MedicalDB, delivers application-specific functionality and user interfaces.

These have been built into two main projects called the NodeSR and the MedicalDB. These projects can and have been duplicated and distributed across multiple computers to take best advantage of local resources while using this system. The system has been used to perform the experiments described in the next part of this thesis, along with other processing tasks and experiments outside of this thesis. There are partial visualisations of the data-flow graph showing the data genealogy produced by the framework at the end of chapters 8 and 9 to show the analysis framework in action. The implementation described in appendix D has been found to run well and at an adequate speed, although optimisations are possible and would be encouraged for future work as is discussed next.

4 Discussion and Future Work

4.1 Overview

To support the growing wealth of data being captured around us both in the health care sector and the emerging sectors of Internet of Things (IoT) and industry 4.0. There is now scope to be able to gather data at volume. This thesis has shown that progress can be made to design and prototype a data management framework which allows transparency and reproducibility for the analysis performed on the data within the system. The current state and developments, limitations of the current work and ideas for further research for the framework and prototype will be discussed.

4.2 Discussion

The framework described in chapter 3 has been used to perform and manage all of the experiments mentioned in this thesis. Further implementation details can be found in appendix D. By its very nature, it chains the result to the programs and the data that created them. The data analysis performed was used to show that the framework as implemented, can be used to run meaningful data analysis for both classification and regression models, while splitting the stored data across multiple database systems if required utilising the best from each. It was also used to combine the results from different experiments within the framework which can ensure appropriate linking of resources for analysis purposes. The structure allows the work to be reproduced or investigated many months or years after, with snapshots of all of the resources either by being version-controlled or immutable, thus

producing a unique traceability where each piece of information can be tracked back to its import origin or entry. This has been used throughout to aid the development, exploration of the processes and system contained within this thesis. The results can be produced in the form of latex or HTML reports, including tables and graphs embedded within them. The framework can allow information to be combined with data in static values with a program script as well as time series data contained in attached comma separated variable (CSV) files held in the bulk system. This dynamic use of the data and processes within the system builds an ontological network, and the programs can query this storehouse of saved metadata opening up many possibilities for meta-learning in the future. The exact nature of the data contained within these elements is not fixed, meaning information from images or videos could be added to the network and brought into these processing networks. This has not been utilised in the current version but could be included for future analysis.

4.2.1 Current State

Most of the current projects were discussed in chapter 2. There are other frameworks which allow the chaining of processes which have had increased interest lately. The most interesting is Dask which is described by Rocklin (2015). It is a library system to enable routines and programs to be built up using its graph base descriptions which Dask can then distribute out to other processes to allow data processing accelerations (Dask Development Team, 2016). The graph system used is similar to Tensorflow where functions and methods are chained together to form a dependency tree, then the graph is interrogated to find ways of distributing the work. The graph system is similar to what the framework develops over time while data is being processed.

The main difference is the scale and the scope. Dask can run many processes but they are short lived as it is built to work with interactive work loads. The framework is built around long term storage, management and flexibility, where processes might run from seconds to days and the results are semi-permanent. The

scope that the Dask and Tensorflow library graphs work at function and operation level by paralleling array operations. The framework creates a graph for whole libraries, programs and projects. There is little conflict since the framework was built to have its process control open ended. One process could load data from the framework where the system would capture its interactions. The process could run a Dask task to speed up its performance or run a Tensorflow deep neural network dependant on its setup and environment. After the task completes, the results are then moved back into the framework by the process, which the framework captures and stores for further analysis by other programs. The framework creates graphs of multiscale data transformations and ontologies built up as independent programs to process the data. The processes within the ProcessCR can achieve their goals however the program thinks is best and has the computer resources to run.

KNIME (Berthold *et al.* , 2007) application shares some apparent similarity with the current framework. KNIME allows users to build programs out of known function blocks with some scope to build new blocks to process data. The current framework differs from this since it is designed to allow the management of any process using this system for data storage by building a graph of the data consumed and produced by that process, along with recording the process's dependancies, if the framework runs the program. If the process is run externally, the framework can be informed and nodes linked by passing a process token. In KNIME a graph of nodes can be connected by the user to build the program, however in the current framework the user's programs build the graph as they are executed. The framework holds this information to allow later programs to learn and use this ontological knowledge for their operation or this allows the data to be managed using the process information.

4.2.2 Limitations

The framework is limited in its implementation, as described in appendix D, due to time constraints and some features not being directly required to complete the experiments performed. One such feature was the node trigger system. This would allow a trigger query to be specified. If met, the program would be turned into a

process and run based on the value of the query. The effects of triggering could allow for automatic and reactive behaviour in the data analysis. For example, if a node sees new data without it being connected to a process that is connected to itself, it could trigger the processing of that data. This, in turn, could trigger other programs, and a self-organising network of data processing and analysis would ripple through the graph. However, at any time the old versions of the results would still be present, and all paths could be re-traced like the data flow graphs at the end of the previous two chapters either by human or other processes in the system running as daemons. Lines of inquiry could be made without having to worry about the documentation as that is being built along with the processing elements. So different lines of inquiry could be setup and concurrently pursued. Fruitful lines of inquiry could be expanded, and dead ends could be documented as sign posts to others exploring the data.

Another limitation was the degree of distribution and scalability of the system. It has been wholly run on a single laptop but also scaled to use three Linux machines, running three services, all connected to one MySQL and one Neo4J database resource for the bulk and graph data sections of the system used to increase the resource utilisation and distribution. The processes running were able to read, load and understand the processes being computed on the other computer elements if required. However, testing and utilisation of the system on an extensive network has not been tested. This might require a few modifications to the NodeSR service to remove some of the possible bottlenecks in the writing of graph nodes into the back-end storage resource.

The final limitation is the capability of the user interface and the security of the system. The prototype system was implemented emphasising function and to allow the framework to be used, tested and to produce results. This has left usability and aesthetics to be slightly less important, due to time constraints. The interface, therefore, could be greatly improved with a redesign which would significantly improve the user's ability to leverage the power of the scheme as a whole. The security of the framework has got some basic level security built in, for example, the

processes can be run in separated Docker containers thus limiting the damage they could do. The token system for process tracking can be used as an authentication key to decide whether a resource can be seen and loaded by a particular token. The tokens and the ontological network they produce allows the influence of that token to be known and measured. If in the future a token and process were found to be a problem, the nodes it touched could be found and corrected. This however can and should be improved now that the basic system has been developed.

4.3 Further Work

The framework design above has been implemented as is described in appendix D. This follows the designed interfaces but simplifies the interfaces and the physical location of the micro-services. This framework is sufficiently fast enough and has been proven reliable enough to have been used as the main method of processing and analysis throughout this thesis. However there are many opportunities for optimisations and extensions. It has been designed to welcome changes and optimisations by having a modular system. New extensions, upgrades to existing modules or plugins can be created as needed to either upgrade module parts within a service or be added to new services to handle different information and transformations. The main path for extension of the system would be to create or employ an extendable query language to allow the more powerful requests to be given to the system. The current plan is to utilise OpenCypher¹ over others such as Gremlin². OpenCypher is the open sourced extension to the Cypher language used in Neo4j and allows for a very fluid definition of the nodes or sub-graph to be searched. Utilising a more comprehensive query language can simplify the data searches and so can then invite larger more complicated questions to be asked of the graph based representation. As the queries become more powerful, optimisation might have to be used to keep the

¹More information can be found on the OpenCypher projects website <<http://www.opencypher.org>>. Author: OpenCypher, Retrieved: 2017/02/20

²More information can be found on the gremlin website <<http://gremlindocs.spmallete.documentup.com>>. Author: Gremlin, Retrieved: 2017/02/20

speed of data access high. The system has been designed to give good optimisation points for this extra intelligence to be added.

4.4 Summary

This concludes the discussion of the data management framework. The current state has been explored, along with its limitations. From these, suggestions have been made for interesting further work to improve or extend the concepts that have been explored in this part of the thesis. The next part takes the framework described and discussed in the last three chapters and shows how a prototype implementation, can be applied to manage and process data captured from a novel data capture device for signal quality and blood pressure described next.

Part II

Application of the Framework

5 Background and Literature

5.1 Overview

The second part of this thesis applies the framework described in the first part to manage and process data from a experimental data gathering device, in order to achieve the second aim from the introduction in chapter 1. This application aims to support patient centred health care by building a prototype data management system as described in the next few chapters.

This chapter first looks at the issues facing the healthcare system and the opportunities that technology can bring. Then the two main models of healthcare are described in section 5.3. Finally, an analysis of how these systems support the healthcare service and models of delivery through information management, analysis and automation are discussed.

5.2 Health Service Pressures

Healthcare systems are under pressure to do more with less, and so increases in efficiency and quality are required. Technology can play a large role in these improvements by both helping patients and the healthcare service (Stewart *et al.* , 2003). This can be accomplished through data gathering and analysis along side the automation of current medical knowledge. García-Lizana & Sarría-Santamera (2007) describe a systematic review of the ICT technology used to support chronic illness management and education. Although the evidence is limited to specific management aspects, the benefits include: improved clinical outcomes, mortality

reduction and encouragingly, lower health service utilisation, particularly with cardiovascular disease. The improvements observed were mostly due to the improved education and self-management of the condition by the patient.

The changes to the service in both the overall model and how that model is delivered are more welcome as the pressure increases (Eijk *et al.* , 2013; García-Lizana & Sarría-Santamera, 2007). The advent of new technology for monitoring, such as mobile and wearable devices, as discussed in section 6.2, along with the current trends in individualised healthcare and treatments, creates hope that the continued introduction and extension of telecare systems, could have a favourable impact on the situation.

The considerations of the treatment are becoming increasingly more personalised for the individual patient. This personalisation requires more time from the doctor to understand each patient and the issues facing them (May *et al.* , 2006). To reduce the time requirements of the clinician, Eijk *et al.* (2013) proposes a need for a collaborative healthcare model that encourages the patient not to be a passive bystander in their own healthcare, but to work with the clinician to resolve the issues encountered.

5.3 Investigating Models of Healthcare

The models of healthcare delivery in the UK, USA and Canada are changing (NICE, 2012; Morgan & Yoder, 2012; Deshpande *et al.* , 2008). Modern medicine (from the last 100 years specifically) operates based on the knowledge that humans and human biology are very uniform across the species. This is based on scientific observation and evidence-based reasoning (McWhinney, 2003). This idea has since developed into the modern clinician-centred view that states that: if people are the same, the treatment given should treat all with similar symptoms. This assumption can be detrimental as it underpins a medical system that may omit a patient's uniqueness and circumstance and very possibly misdiagnose an individual. Therefore, a move from treating the species to the individual patient is an important step.

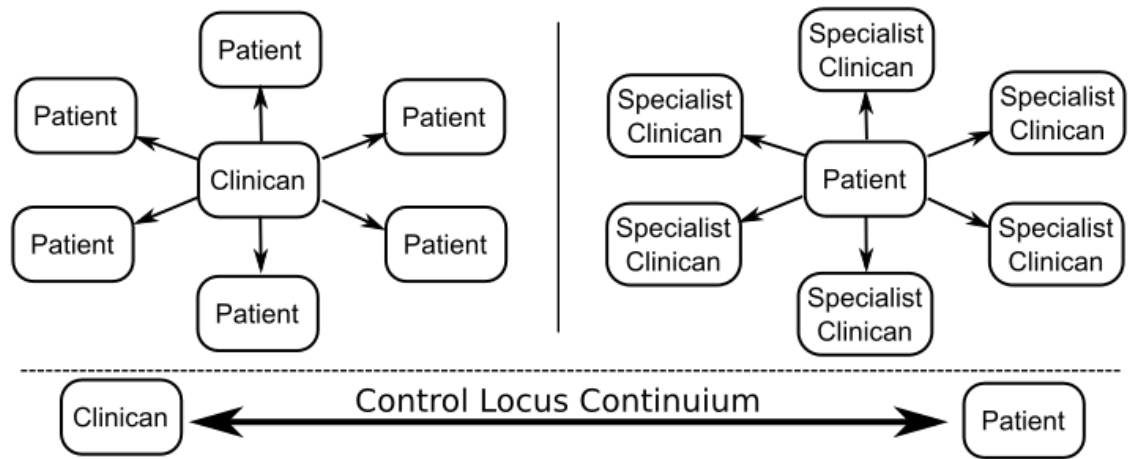


Figure 5.1: The organisational graph of clinician centre control (left) to patient centred (right).

Two main models exist with the difference being where the locus of control sits for the individual. This is shown in figure 5.1. On one hand the locus of control is placed with the clinician with a fully clinician dependent system. The patient would have no choice other than to do what the health-professional says. A more practical system is the traditional clinician-centred model, sometimes called disease centred (Morgan & Yoder, 2012). The decisions are made by a clinician about how to treat a particular disease resulting in a conversation with the patient to find out what a patient wants so that can be taken into account in the patient’s treatment where possible. On the other hand, the locus of control would be with the patient. This is the patient-centred model, where the information is relayed to the patient along with a framework for understanding the decisions to be made. This would be a totally independent system where the individual’s healthcare would be controlled by the patient. The patient would be able to request medical assistance from the service. These options illustrate the extremes of the hypothetical healthcare systems and neither would be realised completely in any practical system. The first would not be very ethical as it would not have informed patient consent. The second would not have enough resources for every patient to demand immediate personal attention for any minor ailment.

A practical more patient-centred model could be manifested by a self-managed system for example, where the clinicians and professionals can be on hand to help with requests or changes in health status where issues can be dealt with pro-actively,

led by the patient and supported by an appropriate telecare system as mediator. This would give the patient more of an opportunity to participate in their healthcare, and help to remove some of the bottlenecks that monopolise the clinicians valuable time. These opportunities just mentioned are made possible by the use of telecare and telemonitoring systems, which allows the care to be shifted from a disease focus to a preventative focus (Morgan & Yoder, 2012).

These models will be discussed in more detail in the next two sections. Both of these models benefit from telecare systems which are described further in section 5.4.

5.3.1 Clinician-centred

Clinician-centred or evidence-based medical models have been the gold standard of the medical profession. It is viewed scientifically as the ideal model because it is based solely on empirically observed and generalisable information. This traditional healthcare approach allows clinicians to lead a patient's healthcare. The clinician has the training, knowledge, experience and access to a patient's medical records. Their task is one of translating this information for the patient, to make it understandable for the lay person, and finding out what would help the patients under their care.

Most clinical systems are based on this traditional model. So much so that health systems are not usually referred to by the term "clinician-centred" when being described. The work described by Bhaskaran *et al.* (2012) and Prasad *et al.* (2013) are very much clinician-centred systems but neither makes any mention of this fact. These systems and other examples of clinician-centred systems will be discussed next.

Bhaskaran *et al.* (2012), describes first that the doctor-patient relationship has a large impact on the quality of the care that the patient perceives. Here the paper looks at facilitating a counselling session by using a visualisation method for the doctor to help communicate the risk the patient is under from the current condition they are in. They then process the data to classify the information the doctor has to visualise. The details of the visualisation ideas and the system, along with a fictional scenarios were then investigated. The doctor has to make the grouping of

parameters and also show the diagnosis by setting up an ideal case and take the patient on the journey of their health.

The other case study by Prasad *et al.* (2013), describes a clinician-centred service that connects doctors to patients in real-time for mobile doctor's appointments. This system was built to allow a mobile doctor to care for patients in a hospital. They were provided access to the patient's medical data and current status along with a video feed on their mobile device. This further removes the doctors from the patient and if anything, makes the doctor-patient relationship and management more difficult. This brings attention to the work done by Smith & Vela (2001); highlighting that memory and working knowledge varies with the context that the subjects find themselves in. If the doctor is mobile their context and environment will change, there is the possibility that in the event of an emergency, the patient information displayed remotely does not give the doctor the same visual cues or context as if the doctor was in the same environment as the patient. However, this remote management of patients could allow one doctor to handle more patients as there is less travel time and the patients are accessible to them where-ever they are. These systems have the benefit of allowing more patients access to the judgement of a doctor but could potentially lead to dangerous errors of that judgement if overloaded.

A further case study was conducted by Gambling & Long (2006) where the medical professionals delivered medical advice and carried out regular appointments over the telephone with patients that were diagnosed with type 2 diabetes. The study concluded that the use of the telehealth system was effective, but that the overall impact of the telecare sessions could have been much more effective if the caregivers had first worked out with the patient more specific goals and motivations before undertaking the trial. They discovered that there needed to be a more collaborative or even a patient-centred approach to fully maximise the benefit of a telehealth delivery system.

Although most of the world's current medical systems rely almost entirely on the clinician-centred model of healthcare delivery, there are shortcomings which

must be discussed. Firstly, the system was designed to be robust and practical (i.e. regimented, short, routine visits) and easy to administer to the masses because it relied on the health professional staying in one place, with access to medical equipment and knowledge needed to diagnose and triage patients. As time has progressed, it has become apparent that the continuous strain and ever increasing demand on the health professional's time and energy further taxes their resources, which could be more efficiently used. The clinician managed system can also lead to a single point of failure. The patients who do not have the knowledge or the experience to make educated decisions based on their current health, have to seek medical advice before even looking to manage their own health. Some of these issues could be addressed by incorporating some of the best aspects of patient-centred healthcare delivery into a distributed collaborative paradigm that would satisfy more fully the patient and the medical delivery system alike. Using both machine learning and experts would allow for a better managed triage and management system. These revelations lead to the discussion of patient-centred healthcare systems in the next section.

5.3.2 Patient-centred

In a patient-centred healthcare system, the patient is responsible and can take an active role in the state of their health and health-related goals. This is in direct contrast to the clinician-centred approach and finds some friction in the medical world for a few reasons as briefly discussed by May *et al.* (2006); Eijk *et al.* (2013). Both cite that the patient-centred approach is beneficial because of the involvement of the individual, but with some short-comings, similar to the clinician-centred model. These short-comings are listed by Eijk *et al.* (2013) in particular and are fairly extensive, although they are still advocating for the use of a collaborative, patient-centred approach. Some of the concerns that the author lists are: a lack of interest in the subject by the patient, the fuzzy definition of "patient-centered" therapy, the decrease in clinician commission for delivering specific treatments, and age and cognitive ability become complicated issues when dealing with specific diseases, children

or the infirm. Lastly, some situations may need some interpretation on the part of the physician to determine an appropriate level of involvement, which could lead to misinterpretation.

Epstein & Street (2011) noted that there is also a need for appropriate measures of success to properly evaluate the effectiveness of the patient centred model, and also more specifically, telehealth system effectiveness. They state that by having a more effective evaluation method, it could increase the rate of adoption on a larger scale. This has also been noted by May *et al.* (2003), as well as by Epstein & Street (2011) that the need for more well-defined measures of assessment to increase the rate of adoption would be welcomed by the medical community. This success could be measured in the number of telehealth appointments which would mean that the patient did not need a face-to-face contact session with a doctor and so better utilise the resources from both the medical and patient's perspectives (Cusack *et al.* , 2008). Epstein & Street (2011) do propose a solution to at least one of the concerns voiced by the papers above, where there is considerable concern about the slow adoption rate of this revolutionary form of the model due to lack of specific definitions that are often associated with patient-centred healthcare.

Another important issue is how the service model is perceived by the patients. Morgan & Yoder (2012) have produced a concept analysis of patient-centred medical care that unearths the positive aspects that patients feel when they are engaged in their own care. They conclude that patients feel empowered, enthusiastic and have increased compliance with medical treatments that are agreed to collaboratively. The patient's overall satisfaction is increased and even the efficacy of the treatment is enhanced by participating in patient-centred medical treatments. These are some of the immediate benefits that can be gained from patient-centred models of healthcare delivery.

An example of a patient-centred system in use is discussed by Gambling & Long (2010). This paper describes the exploration of a patient-centred study in a tele-care behavioural change intervention. This highlighted the need for patient-centred systems to be flexible and to keep in consistent contact with the staff providing

the telephone support, as the study showed through their contact over the three year period. It has to be realised that patients have differing levels of background knowledge; following this, the help needs to be tailored to the patient separately. It is encouraging that they advocate increasing the active role patients take in their health.

Patient-centred methods of medical care delivery are increasingly becoming the preferred healthcare model of service delivery. It could be perceived that the patient-centred model of healthcare delivery operates in a way that is in contrast and contention with the traditional model of healthcare delivery; but this is inaccurate according to Stewart *et al.* (2003). An ideal medical model for the delivery of healthcare is a combination of the two approaches that combine the strengths of both to create a collaborative approach that addresses some of the shortcomings of both of these systems. However, such schemes require the support of a strong telehealth system, to aid by automating and maximising the utility of the resources within the system¹. The system allows greater flexibility and the ability to have medical understanding on hand in more flexible ways. Evidence may be built up and used over many months, with initial diagnosis by a patient, which can be automatically verified by a machine learning system. Only when confirmed as suspected, is it then forwarded straight to a specialist clinician for that condition.

5.4 Telehealth Support Systems

A telehealth support system will be defined here by the descriptions used by Adeogun *et al.* (2011) and NHS (2014), which, in summary states that: a healthcare system using ICT technology as the mediator between patients and the clinicians to support the provision of healthcare and education at a distance is useful. Examples of telehealth systems include home medical monitoring devices, using Skype or telephone interviews to discuss a patient's general or specific health problems, or using

¹Independent NHS report "Making IT work: harnessing the power of health information technology to improve care in England", published online on 2016/09/07 at <<https://www.gov.uk/government/publications/using-information-technology-to-improve-the-nhs/making-it-work-harnessing-the-power-of-health-information-technology-to-improve-care-in-england>>. Retrieved: 2017/06/05

clinical decision support systems to help make a more well informed or accurate diagnosis of an issue, while being supported by a clinician. Cusack *et al.* (2008) discussed three models of telehealth that could be delivered as examples. They then look at the cost effectiveness of each model. The three models that they propose for comparison of cost are: “store-and-forward”, “real-time video”, and “hybrid systems”.

As discussed in section 5.3, telehealth has been theorised to be a cost effective method for delivery of patient-centred healthcare, but it is slow to be adopted because of the relatively young nature and the vastly theoretically based estimations as postulated by Eijk *et al.* (2013). Cusack *et al.* (2008) have carried out theoretical model cost estimations, and although it is theoretical, and does not carry as much weight to the general public as an empirical study, the idea could potentially save the American healthcare system \$3.4 billion dollars per year, if only they would adopt a more significant telehealth approach to delivering medical services to the public. This is one of the many reasons that the telehealth system is quickly becoming the sought after model of healthcare delivery.

To deliver useful telehealth systems, it is pertinent when designing the overall system, to consider some of the most desirable features from the literature, that define both effective and relevant telehealth systems and clinical decision support systems (CDSS). These systems, working together can bring all the threads of information together and present this to an end user such as a GP, patient or specialist; this could also be all three, in a form that the user finds informative and helpful when monitoring or even diagnosing illness. These system features and design principles have been discussed first by Sittig *et al.* (2008), stating that the technology has been slow to be adopted by medical professionals and institutions. They also postulated that to improve this situation, empirical designs would, in their opinion, be essential in creating increasingly useful and more popular systems. This was later responded to by Fox *et al.* (2010) who has argued that the technology has been adopted in the past, but systems should concentrate more on theory led developments to begin the design of a framework. Given these seemingly contradictory points of view, it

seems clear that although both agree that the systems are critically important, both design strategies have flaws. However, it seems that the difference resides in the scope of its use in the short and long term. Empirically driven systems can have many benefits as they were designed to solve empirical issues in the short term. However, in the long term, a well designed system can be more easily modified and adjusted to accommodate change.

When looking into how these systems are used in the current age of mobile technology, it seems natural to combine their methodologies and features to give context to which the aim of this research is directed. A system should have a strong theory led design. However, that design should be used to allow the system to be flexible and adapt to meet the empirical problems that are encountered throughout its lifetime. The essential characteristics for the design and implementation of an effective telecare delivery system are described by Sittig *et al.* (2008), Fox *et al.* (2010) and Kawamoto *et al.* (2005). These will be summarised below:

- Visualisation and summary of the information, presented effectively for patients and clinicians.

This can include the evidence or the sources used for the decisions. Also keeping track of the information presented and the decisions taken by the users so that the system can learn from its users.

- Dissemination of best practices in design, development, and implementation.

This should include having transparency in the system architecture when required so that decisions can be seen and traced, by using ontologies and formal representations for the systems internal processes.

- Prioritisation, filtering and combining recommendations for patients.

To take into account all of the conditions exhibited by a patient then make the recommendation, not just assessments of the state of health of a patient. Also, to promote the possible actions that could be taken.

- Using big data analytics and mining to increase the internal knowledge within the system.

Use natural language processing and free text to augment the internal information to drive clinical decision support and its knowledge base.

These points are the main desirable features and principles that should be used to develop an effective telehealth system based on the research observed presently. Overall, the features found in this list should also include an appreciation that the system should have methodologies for self-assessment of its current performance so that the system can evolve and adapt to the changing environment.

As stated previously, there is a real need for assessment that is meaningful and universal - which demonstrates the true value of not only patient-centred healthcare models but also for the delivery and monitoring of telehealth. A paper by May *et al.* (2003), describes an evaluation of the assessment techniques, using telehealth as an illustrative case study. They stress the need to normalise the evaluation of the different telecare systems, to allow the different systems to be compared in the NHS, which relies, for policy decisions, on empirical evidence based studies. The normalised evaluation proposed could then be used to compare systems and also to allow the structure of the system to be improved to maximise the measures used. The paper was only looking at interviews and qualitative outcomes in isolation from a discussion but provides an interesting template to take these ideas and merge them with the quantitative result of a system to find a workable, normalised measure by which systems could be evaluated for effectiveness.

A systematic review by Ward *et al.* (2015) was conducted on 38 studies which have been evaluated. They took an in-depth look at the use of telehealth applications in the Emergency Room for specialist access to information, for minor injury and illness evaluation and for diffuse patient populations that do not have access to in-depth emergency care. Their main conclusions were that although the studies they found were limited in what they reported in detail, the evaluation did point to telemedicine being helpful and making an improvement in care. This means evaluations of these systems can be carried out but that the reporting is not standardised, so as mentioned before, normalisation of the evaluation is still necessary.

As articulated by Farberow *et al.* (2008), they discuss the regulations and criteria that could be used for certification of its reliability and performance. The five criteria that they discuss are:

- Effectiveness of patient management
- Evidence-based outcomes
- Regulation
- Cost, including cost effectiveness and reimbursement
- Certification to ensure reliability

As these systems are used more, they have the potential for costly mistakes in diagnosis, care or treatment plans to escalate, and could lead to malpractice lawsuits and similar. The criteria mentioned above from the paper by Farberow *et al.* (2008) could be used to mitigate some of these issues.

Surveys by Koch (2006) and DelliFraine & Dansky (2008), provides good overviews of different telehealth systems drawn from 578 and 29 published articles respectively, where the trends and the technologies used to provide telehealth services, including the internet, video monitors, data monitors and telephones were described.

In the review by Koch (2006), they showed that there are trends in the systems found and they were built to be used by both patients and clinicians. This indicates that the collaborative point of view between patients and clinicians, as discussed before in section 5.3.2 is useful and presents evidence that these types of systems are being trialled and found to be working. They did report however that there was a lack of studies for special groups such as the elderly or disabled, and that the studies show problems with standards of reporting, evaluation and regulation, which is further evidence for the issues discussed above.

DelliFraine & Dansky (2008) showed that the telehealth systems showed the most benefits regarding effect size when being used to care for cardiovascular or psychiatric conditions, but conditions like diabetes showed a low effect size; they concluded no relationship. Again they mention a lack of report information from

the studies as problems leading to the removal of 12 articles from their original pool of articles, which showed positive findings. This could be improved by having standards of evaluation of these types of systems as discussed previously above.

5.5 Summary

The main outcomes of this chapter can be summarised by stating that the health service is under pressure to change. This change is directed towards looking at the modification of the basic structure of the health service. The traditional clinician-centred care system could be improved by allowing and encouraging patients to take a more active role and even being the leader of their own healthcare. This changing role has to be facilitated by telecare and telemonitoring systems to better take advantage of contemporary medical knowledge and make the best use of the current resources, through easy and consistent monitoring and useful data management and analysis. Systems have been examined above, by looking at the desirable features of such systems for delivering healthcare services remotely in a telecare environment to patients by clinicians. An important point is that for systems to be successful, they have to be designed to incorporate and welcome change. The evaluation and regulation of these systems have also been discussed, as well as examining surveys of these systems throughout the chapter. This sets the context for the problems as presented in chapter 1. The next chapter focuses on the data gathering aspects of the data flow framework being developed, which will address the problems in data gathering and management in a telecare context.

6 EIMO Data Gathering System

6.1 Overview

For telehealth systems to function, they require their users to have their health and vital statistics recorded on a large scale and over the long term. There have been monitors produced as described in section 6.2 that are able to record and track their users. These devices have limitations in the type of information that they can record or are otherwise restricted in their use. Device applications normally require specialised equipment and consumables like electrocardiogram (ECG) pads or blood pressure cuffs, thus making their users wear them all the time or they need to be reapplied regularly. This renders them inconvenient and restrictive for the periodic monitoring required for a long-term telehealth system. The purpose of this chapter is to show the current state of development and testing of a new medical device called EIMO, which is self-contained and easy-to-use. The important points of the device for signal recording and capture are described in section 6.3. In order to compare the data recorded, the EIMO device and two other medical data recording devices were used in a study described in section 6.4, discussed at the end of this chapter. The aim of the study was to compare the data recorded by the devices and to gather further information for the estimation of blood pressure which is described in chapter 9. This data was also used for the signal quality assessment in chapters 7 and 8. In order to run the study, data management and analysis became an important issue. Ideas for data management were explored in the unification and analysis software made for the study that is discussed at the end of this chapter and is expanded upon in chapter 3.

6.2 Review of Personal Medical Devices

There is increased interest by the public in personal and wearable devices that measure, track and assess health-related information. (Pantelopoulos & Bourbakis, 2010; May *et al.* , 2005; Neuman *et al.* , 2012; Etemadi *et al.* , 2016; Ghamari *et al.* , 2016). The functionality of these devices can be arranged into two basic layers.

The first layer are the basic health trackers made for the retail market such as the Fitbit® series - a personal accelerometer and activity tracker, and a range of other devices from Samsung, Sony, LG, Motorola, Microsoft and Apple. These take the form of wrist-mounted devices, which can track the user's activity level, heart rate and position. Some take the next step and add a band around the chest such as the Polar Tracker for a more accurate measurement of heart rate when exercising. In the new watches, heart rate and location tracking using sensors built into the devices, along with a more convenient interface and questions for the user, can also be recorded. These devices, while interesting and easy to use, are limited to the vital sign signals that they can record, and so limit their use as complete telemonitors for managing and caring for patients (Munnoch & Jiang, 2015).

The second layer consists of more involved devices; these have more readings available, but they require to either be worn all of the time or use sticky pads to maintain contact with the skin. These include devices devised by Etemadi *et al.* (2016) and Jakoby *et al.* (2010). These are placed on the user's chest. They combine many cardiovascular measurements into one device. However, the fact that it is attached to the centre of the user's chest by self-adhesive pads or a belt could be very inconvenient for long-term use by non-critically ill patients as a more general health monitor. Two other example devices will be described. The first called AMON, developed by Anliker *et al.* (2004), measures electrocardiogram (ECG), photoplethysmography (PPG) and blood pressure with a cuff. The authors mention that they have a reduced ECG signal quality and saturation of peripheral oxygen (SpO₂) accuracy. The second designed by Al-Ali *et al.* (2003), measures cuff-based blood pressure and temperature, collects the information and transmits it to a server

system through the global system for mobile communications (GSM) network. These devices show that multi-parameter measurements using a single device have been developed, but lack a single demonstrable solution that can collate multiple signals into a reliable and easy-to-use system, convenient for users. Therefore, a device that could be applied as needed, without any consumables, would be highly beneficial (Munnoch & Jiang, 2015).

Further devices have been developed utilising various wireless technologies as discussed by Ghamari *et al.* (2016) and Pantelopoulos & Bourbakis (2010), where they discuss many techniques for transmitting data for a body area network as well as surveying other telehealth systems. They mention Bluetooth® and Bluetooth® low energy (BLE) to establish easy and ubiquitous interfaces from recording devices through to mobile phones or computers. One example device that has been developed by Tahat *et al.* (2011) is a Bluetooth® connected blood pressure cuff that monitors patients. A second device uses a wrist-mounted sensor system that was designed by Winkley *et al.* (2012); this uses BLE communication to connect the wrist-mounted device to a mobile phone. The wrist-mounted sensor that is described is limited by the signals it can pick up; currently, it captures acceleration, PPG and temperature. This work is interesting as it shows a device designed to aid users by using these signals to monitor and inform a telemonitoring system (Jiang *et al.* , 2016). The device is engaging for users by having an intelligent monitoring program built into a smartphone. The wearable nature of this device, improves the accessibility for the users. This device however, lacks the multiple vital sign measurements to assess the health of the user in a telemonitoring environment (Munnoch & Jiang, 2015).

Greenhalgh *et al.* (2013) have investigated the perceptions of the users, and what aspects matter to the elderly when using telehealth monitoring equipment. The problems highlighted include reduced compliance in taking measurements and that the devices were seen as being intimidating to use, and it being viewed as one step closer to being in hospital. Some cases required the device or system to be set up by a carer or family member, then require users to forward the collected data

into a healthcare setting. It can be concluded that a device will be more successful with the elderly and the general population if it can be more accessible, less invasive, smaller and less onerous in recording the person's vital statistics so that they can take part in and increase ownership of their own health monitoring. A device could accomplish this by collecting data without accessories or residue, then distributing this gathered data to the appropriate carer or system with minimal intervention by the user. In this way the users are helped by allowing themselves to both learn about their health and send the data to systems which could provide valuable insights into the current and future health of the user or users of the system (Munnoch & Jiang, 2015).

The data collected by these devices requires management and storage. The devices discussed above have their methodologies that dictate how and where their data gets stored and treated. An example of these systems with a specific handling routine can be seen in the work by Anliker *et al.* (2004) and the embedded system by Al-Ali *et al.* (2003), where they connected their devices to an external data storage centre. This data is analysed by human operators in the control centre. While this is a very accurate way of analysing data, it is highly labour intensive and costly in both time and money, more so when the system is scaled to deal with large data sets from many such monitors. Manually monitoring the signals is not an efficient way of dealing with this sort of high volume of data. Although both are simple to implement, there is also a low chance of losing critical data as all data is captured and stored but this wastes storage capacity. Automated monitoring and filtering of data could allow for time and cost savings for large systems.

This interconnection and further automated processing would allow alerts to be generated and forwarded to the right parties. Clinicians can then have a better picture of the day-to-day historical readings for a patient by being able to focus on the time around the alerts and are able to spend more time on the data that matters.

6.3 Device Development

6.3.1 Device Overview

A prototype device developed by the author has been progressed further by taking into account the issues found during the prototyping process. This translated to recommendations on the power supply, processing power and sensor placement and advice given by the author to the parties developing the CE marked device, called EIMO (Munnock & Jiang, 2015). The device description below is an overview and the important details about the device that affect the context of the device and signals directly are discussed. Subsequently, the data collected in the study following, is used in the later chapters of this thesis. The complete system overview is shown in figure 6.1 which illustrates the data flow through the system. The device is pictured in figure 6.2 which shows the device being held while recording.

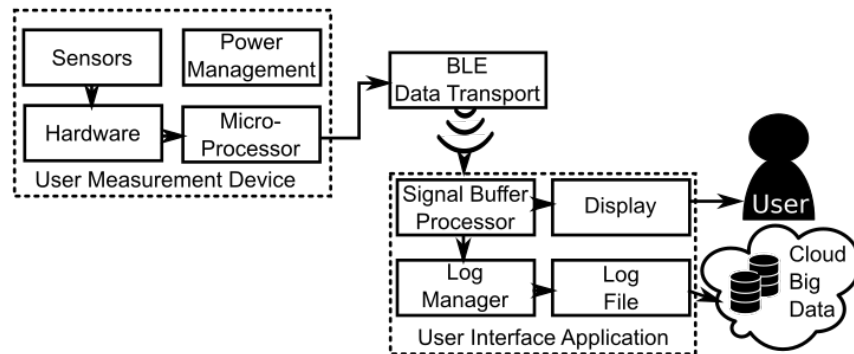


Figure 6.1: System overview and data flow throughout the EIMO device. The data flow runs from the sensors through to the interface application on either an iPad or PC (Munnock & Jiang, 2015).

6.3.2 Processing and Communications

The device makes use of the BLE protocol within a CC2540 microprocessor as in the prototype, with the signal processing being done in a MSP430 microprocessor. The ECG and PPG signals are sampled at 1KHz for the front end software filters, however, these signals are then decimated to 100Hz and turned into 20-byte packets for the BLE communications. The decimation is required due to the processing



Figure 6.2: The EIMO device being held in use. The left hand thumb presses against the outside ECG pad. The right hand first and second fingers touch the inside ECG pad and the two PPG sensors in the insert (Munnock & Jiang, 2015). This normally takes place when the device is in a relaxed position for the user.

and bandwidth of the BLE data transport. The remainder of the calculated measurements are combined into a single broadcast packet as shown in table 6.1 and transmitted once per second. The raw signals are grouped into channel pairs. This gives rise to data packets shown in table 6.2 which are used for two pairs of signals; the first pair was the PPG and ECG signals for timing comparison and analysis. The second pair was the red and infrared PPG signal from the AFE4490 chip used for oxygen saturation monitoring as shown in figure 6.4.

Table 6.1: Feature data packet, issued once every second from the device used in the study. H,M and L denote the High, Medium and Low bytes respectively.

Name	Type	Byte	Range	Description
SPO2 % O_2	H	0	0-100%	The SPO2 measurement is sent as a whole number and must be divided by 100 to acquire the true value.
	L	1		
Pulse transit time (PTT)	H	2	0-100%	The PTT measurement sent as multiple of 16uS.
	L	3		
Heart rate	H	4	0.00 - 350.00%	The heart rate as a whole number must be divided by 100 to acquire the true value.
	L	5		
Infrared Temperature	H	6	17 bits	The raw bits from the infrared detector temperature probe object.
	M	7		
	L	8		
Case Temperature	H	9	14 bits	The raw bits from the infrared detector temperature probe case.
	L	10		
		11 to 19		Not Used

Table 6.2: Signal data packets from the device, are issued 10 times a second to maintain the 100Hz sampling rate. This combines two signal channels into one 20 byte BLE packet, with the first 10 bytes used for the first signal and the last 10 bytes used for the second signal. This packet structure is used for the the ECG and PPG for timing and for the red and infrared for the raw SPO2 signal.

Name	Byte	Range	Description
Channel 1	0 to 9	0-127	Samples are ordered from oldest to newest. Bits 7-1 of each byte are used for the signal
Channel 2	10 to 19	0-127	amplitude. Bit 0 is used to signal that a peak has been detected.

The recorded data from the device and the application are combined and saved. This consists of basic vital sign information, both recorded and calculated, along with user questions. The logging system and format is described in section 6.3.4.

6.3.3 Signal Processing

The system diagram in figure 6.3 illustrates the signal path of each measurement and the signal data fusion links, starting with the temperature, PPG and ECG signals through to the results of the signal data and event markers such as peaks, signal timing and heights. The temperature is transported from the infrared detector IC through the micro controller and into the broadcast packet in table 6.1. No post processing is done on these values, as this is then done and calibrated within the connected application. The PPG and ECG signal are processed internally and are explained in more detail below.

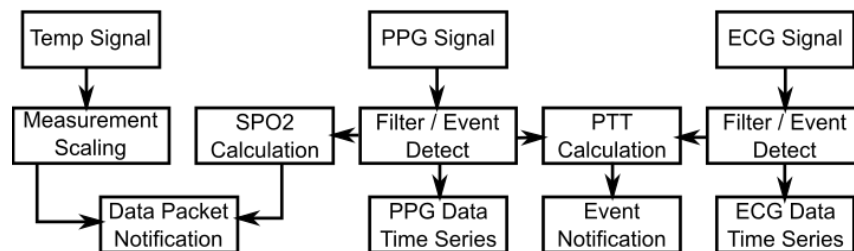


Figure 6.3: Signal system overview showing the data flow and top level processing elements within the device from Munnoch & Jiang (2015).

6.3.3.1 PPG signal

Figure 6.4 shows in more detail the processing steps that the PPG signal has within the device. There are two PPG sensors arranged in the finger insert, which when looking into the insert, Sensor A is on the left and Sensor B is on the right. There are two sensors as these serve different purposes and leave an option for redundancy in signal capture. Sensor A is connected directly to an AFE4490 application specific integrated circuit (ASIC). This was to allow the simultaneous recording of a calibrated and filtered PPG signal, with both red and infrared components for the measurement of SPO₂. Sensor B only uses infrared light and records a PPG signal directly using the onboard analogue to digital converter (ADC) allowing the micro-controller to control the sampling of the PPG along with the ECG for the measurement of the pulse transit time (PTT). These signals run through hardware filters then software filters within the MSP430, an internal peak detection algorithm, is run on the signals to find the signal peaks and valleys for the Sensor B signal and the red and infrared signals from Sensor A. The signals are then decimated and placed into the packets as mentioned above. The peaks and valleys are used for the internal measure of PTT from Sensor B and the R value for oxygen concentration from Sensor A with help from the AFE4490 chip.

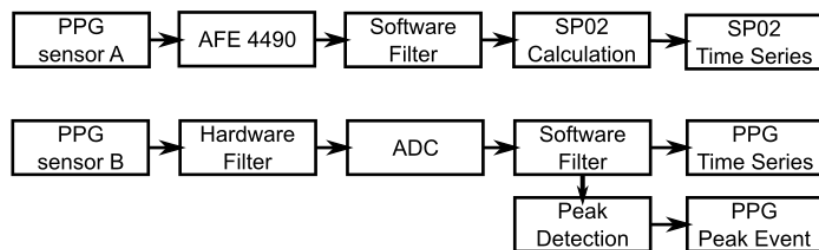


Figure 6.4: Photoplethysmography (PPG) Signal processing block diagram and data flow from Munnoch & Jiang (2015).

6.3.3.2 ECG Signal

The ECG signal system of the device is shown in figure 6.5. The final ECG signal is the electronic difference between two ECG sensor pads, which are electric potential

integrated circuits (EPIC) sensors (PS25201) from Plessey Semiconductors. The noise received by these sensors has been reduced by the inclusion of a 50Hz notch filter and then bandpass filter before the ADC of the MSP430 and internal software filters are applied at 1KHz. The filtered signal is then processed internally for peak information, before being decimated and put into packets as mentioned earlier.

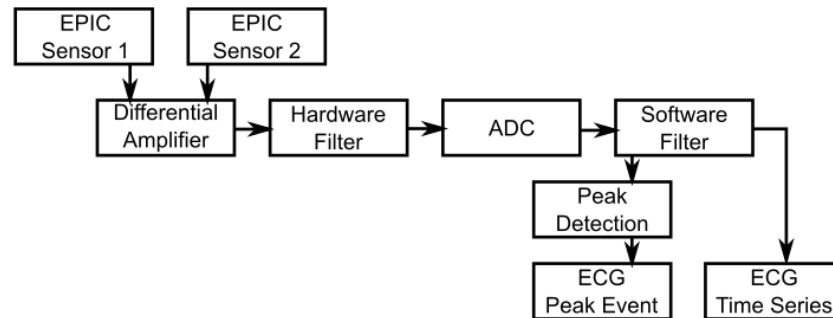


Figure 6.5: Electrocardiograph (ECG) Signal processing block diagram and data flow through device from Munnoch & Jiang (2015).

6.3.4 Device Interface and Logging Software

The software for EIMO has two main interface types. The first is the C# application which was made for debugging and logging for long term and formal data gathering. A screenshot of the device interface is shown in figure 6.6, where the ECG and PPG waveforms are displayed, along with the temperature and the SPO2 reading. The layout of the interface is close to that of ICU patient monitors, with the ECG trace on top, then the PPG below. Along the bottom, the calculated measurements are listed next to icons allowing the user to more easily grasp the meaning of each measurement.

The second is an iPad application for general use. The iPad application was modified to produce a program for the EIMO testing study as it was the most easy to use interface for the assistant taking the measurements. The iPad application and C# interface above, were designed to share the same interface design aesthetics.

The study that follows this section required a simplified version of the iPad application. The application produced the raw signal data from the EIMO device along with a timestamp. This was then controlled using two enlarged buttons. While the study was progressing, an assistant could easily tap the “Start record”

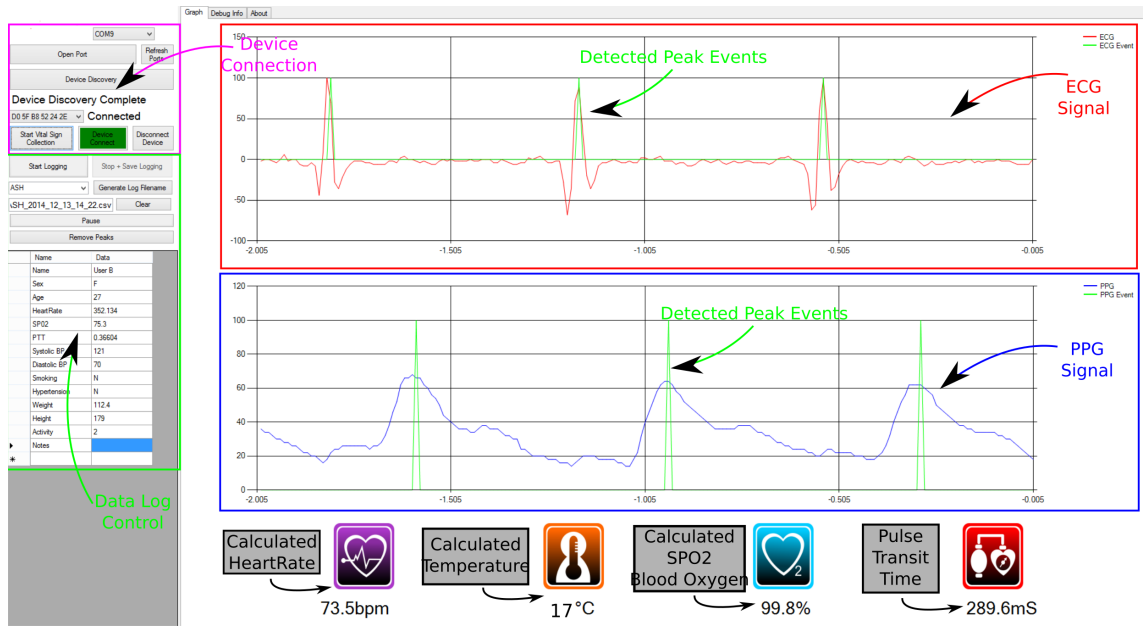


Figure 6.6: Screenshot of the platform application showing measured live vital sign signal data.

button to begin and again to end a recording session, which put the data straight into a dropbox shared folder. The “Sync” button added a known PPG sequence (two ramps from 0 to 10 counts are recorded in the signal), this could then be used to mark known positions in the recording so the data would have independent synchronisation marks if required. A screenshot can be seen in figure 6.7.

The log file produced is summarised in log file listing 6.1 with the full version in appendix B.1.1.1 in log file listing B.1. It presents the JavaScript object notation (JSON) header for the file parameters, averages, and questions along with the comma separated variable (CSV) data set under it for the sampled signals that were recorded. Tables B.1 and B.2 explains the field names, the type expected, the range or value set options, along with a description of the recorded field.

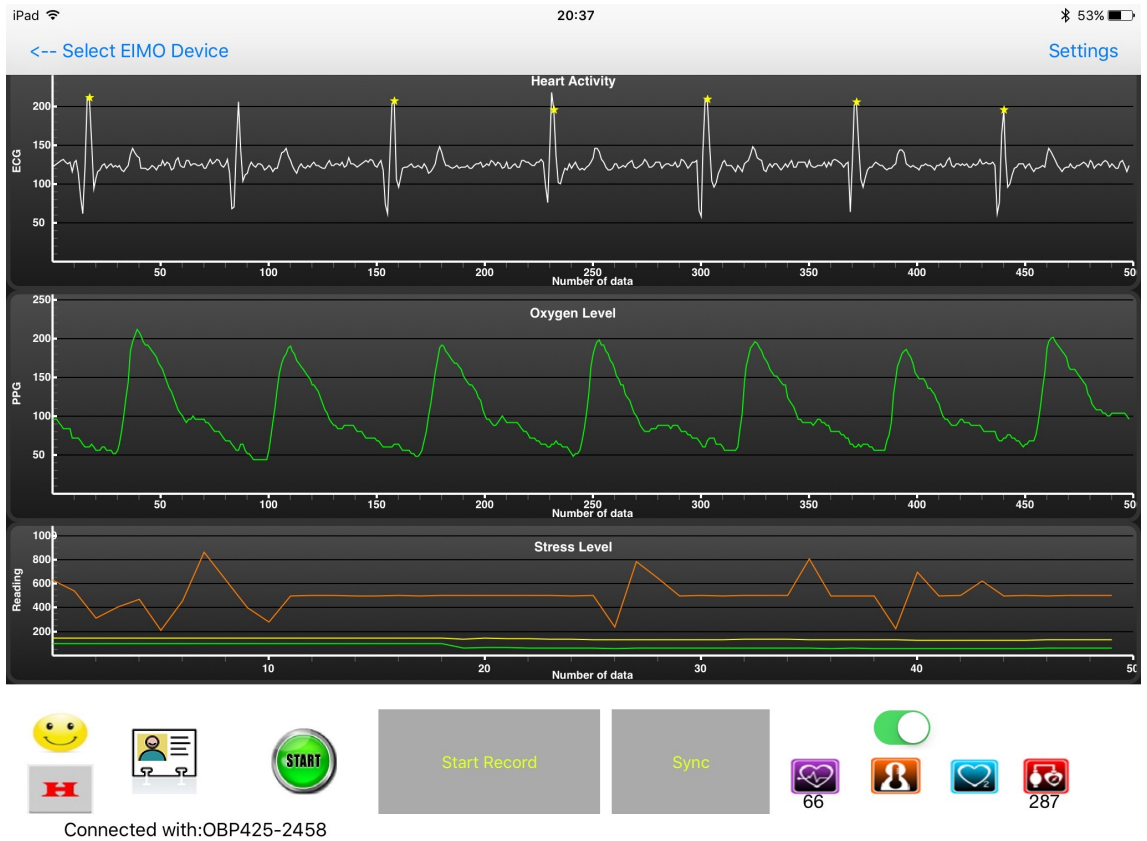


Figure 6.7: Screenshot from the iPad App for the modified testing user interface showing the real time data and the calculated measurements. This includes the enlarged 'Start record' and "Sync" buttons for starting and stopping the recording and adding a synchronisation mark respectively.

```

{
  "Program" : "iPad",
  "Version" : "1.0.7",
  "DeviceAddress" : "A370B607-A0DE-6C26-B1E8-B9FC35CF877A",
  "Logstarttime" : "2015-05-01-09-36-28",
  "Name" : "YIF008_S1_2",
  "Heartrate" : "70",
  ...
  "Notes" : "screen froze battery slipped"
}
Timestamp,ECG,PPG,SP02,PTT,HR
2015-05-01-09-36-28.251,120,182,91,367,70
2015-05-01-09-36-28.261,126,182,91,367,70
2015-05-01-09-36-28.271,124,180,91,367,70
...

```

Log File 6.1: Summarised log for the raw EIMO logging file. The upper part represents the key-value questions and summaries in JavaScript object notation (JSON) format. The JSON section provides the summary and context of the file with the device that recorded it as "iPad" and the identity of the device, along side the version on the software and user name and dates. The lower part holds the data recorded as timestamped samples with each signal given a column in the comma separated variable (CSV) format to keep the data density high. The timestamp is given first as a UTC formatted date string. The electrocardiograph (ECG), photoplethysamography (PPG), saturation of peripheral oxygen (SPO2), pulse transit time (PTT) and finally heart rate (HR).

6.4 Device Testing Study

A study was conducted by the Sport, Health and Exercise Science Group (SHES, University of Hull) with help from the author to compare the data gathered by the EIMO device with two other medical data capture devices. The study's primary focus was to build up a dataset from 3 different devices and be able to synchronise these devices for further data mining analysis and comparison. The secondary focus was on the accuracy of the blood pressure estimate that the EIMO device produces. The next three sections discuss the study protocol, the nature of the comparison devices and the required management and logging programs created. Lastly in this chapter a device comparison is presented using the data gained from the study.

6.4.1 Study Protocol

The experimental setup for the study was conceived by the SHES research assistant and the author. More information on the parameters for this study are in appendix C.1. There was:

- One file per machine.
- Three recorded sections per session.
- Three total sessions separated by 24 hours.
- 27 files per participant.
- One additional measurement file containing other information about the participants.

The raw data set was 19 GB when complete for 18 participants with a minimum of 487 files generated 588 were reported as some of the sessions end up being broken into multiple files due to computer errors, also two of the starting participants did not finish. The management tools and analysis of this data are discussed in sections 6.4.3 and 6.4.4.

6.4.2 Comparison Devices

The devices to be compared were the EIMO device discussed above, along with the CM400 and Case-GE exercise monitor discussed below. These devices were chosen to be able to collect the same type of signals as the EIMO device. Both could record ECG with the Case-GE being the device used within the department, but only the CM400 could record the PPG signal.

6.4.2.1 CM400 Patient Monitor System

The CM400 patient monitor from Contec Medical Systems CO. Ltd, is a PC based medical monitor which, instead of being self-contained with a display, has a USB connection and a PC program supplied to show the signal data captured on a consumer computer. The software supplied with the device only exhibits limited data logging capabilities, which extends to the periodic recording of the heart rate and oxygen saturation (SPO2) with measurements from the blood pressure cuff, in intervals ranging from a few seconds to a minute. This is the case for all medical monitors that have been investigated by the author up to the present. For the purposes of the current study, where the morphology and timing analysis of the real-time signals were required, an application was created which has the capacity, using the protocol supplied by the company, to record the raw signals of ECG and PPG to allow further in-depth study and analysis.

Signals and Data Recording Program

The software built was constructed in C# using the company's propriety DLL program library for communication to the CM400 patient monitor, with a C++ to C# wrapper around the API to leverage the same basic construction and robust logging as the EIMO debug interface used in development. This software was made to stream and buffer the signals with primary focus on the recording of the data and not on the display of the data. This trade-off had to be made to make sure the software would be as stable as possible on the laptop used for the study. The



Figure 6.8: Picture of the CM400 connected to a laptop ready for use.

architecture of this software is shown in figure 6.9. The considerations that were taken into account when designing the modifications include:

- Resource consumption - The target computer was not a high performance machine thus the final performance required efficient tailoring to maintain the data rates and the signal order in the log file.
- Ease of use - The assistant recording the data needed to be able to interact with the patient and not have to interact significantly with the software apart from telling it when to start, stop and the name of the file.

The software had to successfully record the raw data from the eight signals that are currently required for recording by the device; these being the ECG signals and the PPG raw and processed signals. The ECG signals include I, II, III, aVR, aVL, aVF and V, which are the basic signals from the five lead ECG and represents the electrical activity of the heart as seen through 60-degree cross sections. The PPG signal is the raw signal from the finger sensor, along with the calculated measurements of the SPO₂ and heart rate generated by the device while measuring the

SPO2. These are streamed at two different frequencies. The ECG signals consist of 12-bit values internally scaled and sampled at 500Hz across the seven signal streams. The PPG signal consists of a bundle of data sent at 60Hz, comprising the raw PPG signal at 7 bit resolution along with SPO2 and heart rate.

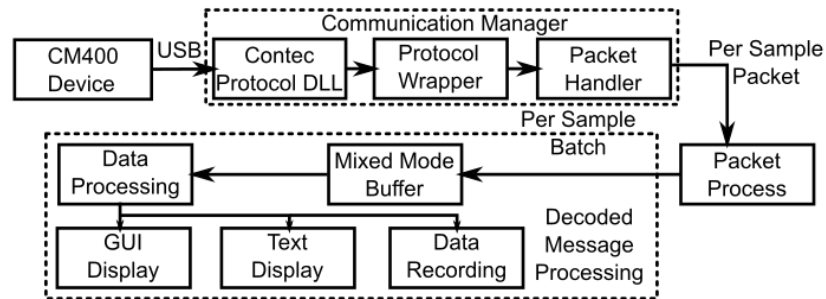


Figure 6.9: Diagram showing the control process and the data buffers between them for the CM400 patient monitor. Each dotted lined shows a package with a controlling thread so each part could run at its own pace while maintaining the consistency of the buffers at each of the data rates used by the CM400.

The log files produced share common patterns with the EIMO log files; it presents the JSON header with the comma separated variable (CSV) data set under it for the sampled signals that were recorded. A summarised sample is shown in log file listing 6.2, with the full file for reference shown in log file listing B.2, where the data fields are listed in tables B.3 and B.4 with more information on this to be found in appendix B.1.2.

```

{
  "DeviceAddress": "CM400",
  "Version": "1.0.3",
  "Program": "CM400_Interface",
  "Name": "YIF008_S1_2",
  "Sex": "M",
  ...
  "Height": "180.6",
  ...
  "Notes": "second rest phase"
}
Timestamp,I,II,III,aVR,aVL,aVF,V,PPG,SP02,PTT,HR,seq
2015-05-01-09-36-24.610,0.096,0.176,0.079,-0.136,0.009,0.129,0.248,18.000,99.0,0.486,64.0,42880
2015-05-01-09-36-24.612,0.104,0.186,0.081,-0.144,0.012,0.134,0.265,17.000,99.0,0.486,64.0,42881
2015-05-01-09-36-24.614,0.106,0.201,0.094,-0.153,0.007,0.148,0.437,17.000,99.0,0.486,64.0,42882
...
  
```

Log File 6.2: A Summarised log extract for the raw CM400

6.4.2.2 Case-GE ECG Machine

The Case-GE exercise monitor allowed the measurement of an independently verified ECG signal from a known clinical class device. The ECG signals recorded were the same as for the CM400 discussed previously. The device also ran the automated blood pressure cuff monitor (SunTech Tango automated monitor). This information was recorded into an XML file for export by the machine.

The file contains the blood pressure readings from the cuff, for each stage in the study protocol. The six channel ECG recordings are only added to the file when the full disclosure option is selected at the time of data export. As this machine is a standalone Embedded XP machine, the time synchronisation is problematic as the manufacturer does not recommend keeping the machine attached to the network while monitoring a user. As a result, the time could only be occasionally synchronised. Final synchronisation was achieved by running a correlation between the CM400 ECG signals and the Case-GE ECG signal. Since these signals contain high-frequency impulses around the QRS complexes, good localisation could be achieved.

6.4.3 Processing and Synchronisation

As part of the data analysis of this study and for the data and model analysis performed later in this thesis in chapters 7 to 9, the log files created need to be unified and synchronised. To this end, two tasks were undertaken. The first being the unification of the data logs to build one style of data log from the three devices. This is described in the next section. The second issue to address is to correct the synchronisation of the signal to allow direct comparison, the method used and results are discussed in section 6.4.3.2.

6.4.3.1 Data Log Processing and Unification

There are three slightly different forms of data log file generated by the devices, so there are many different types of raw log files saved during the study. In order to create an effective analysis pipeline, domain knowledge of the devices should stop

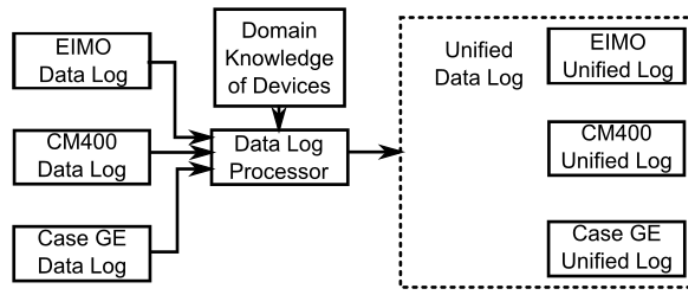


Figure 6.10: Diagram showing the data flow of the log file into the unified log file.

at the data import, and a data standard should be imposed. A device independent log file can then be generated with fields normalised if possible, or signal ranges set pragmatically into the file. This removes any inconsistencies and errors at a low level.

Figure 6.10 shows the architecture of producing the unified files. Each raw log file gets read and domain specific knowledge can then be used to normalise and unify the data in the log file. Correct signal names, ranges and other device version quirks can be identified. The format of the resultant log file is similar to the original files described above, with a JSON object above a CSV sample table. The unification happens by specifying in the JSON object the signal ranges and ontologies for the signals in the log. The CSV sample set column names are then corrected to a standard set that fits with the signal ontology. The translations can be easily scripted, and new translations can be built on top of old ones. Any JSON attributes that can be answered from the sample set are calculated and updated. Once this is all complete, the file is written back to disk with a corrected file name based on its location and contents.

A summarised, unified log file is in log file listing 6.3, where the signal information additions are shown for the EIMO device. Samples of the full, unified log files for each device are shown in appendix B.1.3, and shown in log file listings B.3 to B.5. These listings include the signal ranges and normalised values. The records per participant can be more easily managed by packing the correct information and domain knowledge into each of the files.

The main benefits of the normalisation is that all data can be kept and archived in a single style, which now includes the signal limits, and signal name ontologies.

```

{
  "Program": "iPad",
  "Version": "1.0.7",
  "DeviceAddress": "A370B607-A0DE-6C26-B1E8-B9FC35CF877A",
  "Logstarttime": "2015-05-01-09-36-28.000000",
  "Name": "YIF008_S1_2",
  ...
  "Signal_Limits": {"ECG": [ 0, 1 ], "HR": [ 0, 300 ], "PPG": [ 0, 1 ], "SP02": [ 0, 100 ]},
  "Signal_Ontology": {"ECG": [ "ECG" ], "PPG": [ "PPG" ]},
  ...
  "Notes": "screen froze battery slipped"
}
Timestamp,ECG,PPG,SP02,PTT,HR,seq
2015-05-01-09-36-28.251000,0.471,0.714,91.0,0.367,70.0,55060
2015-05-01-09-36-28.261000,0.494,0.714,91.0,0.367,70.0,55061
2015-05-01-09-36-28.271000,0.486,0.706,91.0,0.367,70.0,55062
...

```

Log File 6.3: A example log extract for the unified EIMO device. This shows the signal correction and ranges, along with the signal ontologies added to the file by the unification system.

The signal synchronisation and comparison analysis discussed next, only had to deal with one standardised file log, which lets the comparison program and following programs be simpler and more consistent when using this unified data.

6.4.3.2 Data Stream Timing Synchronisation

The solution for the synchronisation of the three devices for signal morphology comparison was to use the correlation and pattern matching between the signals. The process started by interpolating the signals from the Case-GE, CM400 and EIMO to the sample rates of the highest device; in this case, the CM400, which runs at 500Hz, then a cross-correlation between the signals is performed. The system clocks of the devices recording the signals could be synced to within 2 or 3 seconds. Advancing the Case-GE and EIMO signal using a sample offset from -2000 samples behind the CM400 to +2000 samples ahead over a clean piece of the signal gives a correlation graph. This sample range was chosen since at 500Hz, this corresponds to a maximum of 4 seconds of timing drift. This is illustrated in figure 6.11. Figure 6.12 shows a block diagram of the synchronisation method with the processing steps involved.

Figure 6.13 displays a sample of the signals received by the devices and the synchronisation possible using the unification program described above. One can

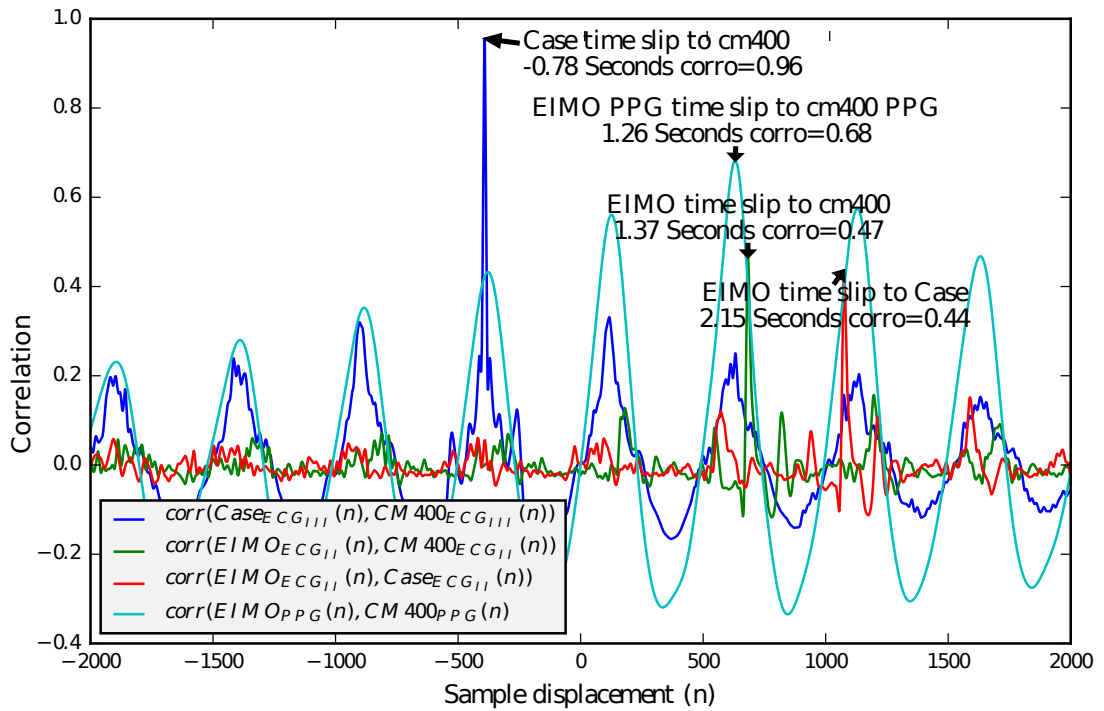


Figure 6.11: Showing the correlation function between the sampled signals, the time displayed is the synchronisation time offset. The mark also contains the correlation of the corrected signal and the reference. This synchronisation was taken over a 90 second window from 30-120 seconds in the underlying signals for subject YIF008.

see that when comparing the top to the bottom graph, the timing has been improved. The morphological features, such as the peaks and valleys are now coinciding over the period of the sample. The amplitude on this graph has not been manipulated since only synchronisation was the priority. The alignment of the signals is used in the device comparison analysis in the next section.

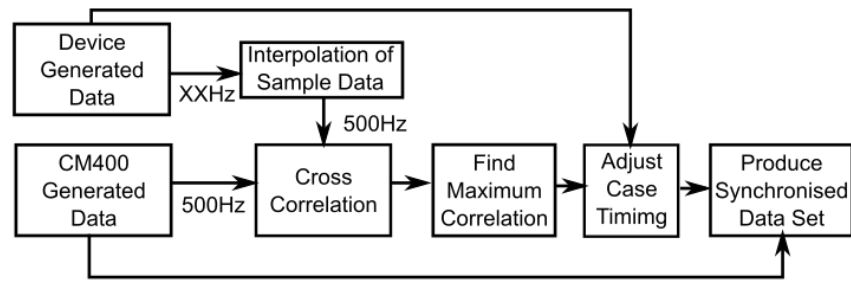
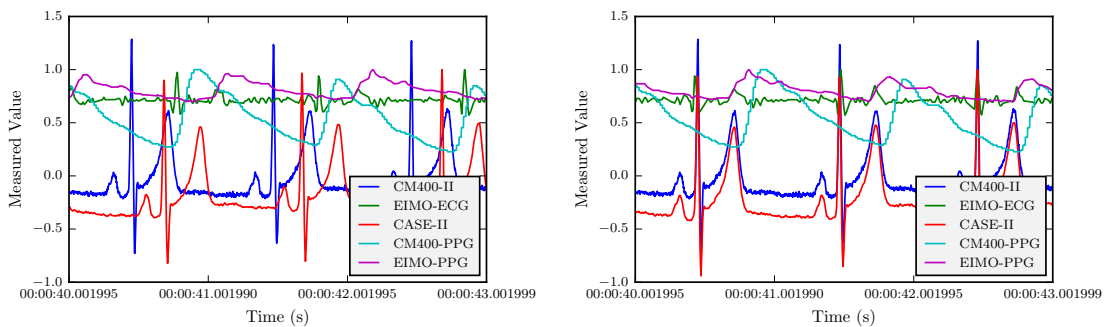


Figure 6.12: Synchronisation diagram illustrating the methods used for the devices. The diagrams show the data flow and processes to create a synchronised data set and log files for analysis. The process uses the CM400 as the main references signal since it has the highest resolution. The frequency marked as XXHz is the stand in for the EIMO device at 100Hz and the Case-GE at 250Hz.



(a) Signal traces to show the ECG channel II and PPG trace from CM400, Case-GE and EIMO before Synchronisation. (b) Signal traces to show the ECG channel II and PPG trace from CM400, Case-GE and EIMO after Synchronisation.

Figure 6.13: Signal traces to show the ECG channel II and PPG trace from CM400, Case-GE and EIMO (a) before and (b) after Synchronisation. Only the signal timing was modified; the amplitude and shape of the signals was left untouched.

6.4.4 Comparison Analysis

6.4.4.1 Methodology

The plan to compare the signals generated from the three devices discussed above is described here. First the recorded data is synchronised and aligned using the system mentioned above for timing and normalised for amplitude. The signals can then be compared to find an objective measurement of the similarity between the signals. To do this, 50 one-minute samples for comparison were chosen from 5 participants (users 5, 6, 7, 8 and 14), using the first session. 10 samples were randomly chosen from each. Participants 5, 6, 7 and 8 were chosen as they were recorded in the middle of the study, representing a steady state of data collection with all of the sessions completed. Participant 14 was added to this set as a representation of data

recorded towards the end of the study. The Pearson correlation coefficient was used as the main metric for comparing the morphology of the signals. The metric was run over each one-minute sample. The test is such that the higher the correlation recorded, the more alike the signals are. The best and average correlation samples have been included for each signal. These will show how well the signals match in the best and average case, these figures are discussed below, along with the findings.

6.4.4.2 Results

The correlations between EIMO and the other two devices are statistically summarised below. Table 6.3 shows the main statistics of the signal comparison over 50 one-minute samples. First, the ECG was compared to the Case-GE then the CM400. Second for the PPG, when the EIMO device is compared to the CM400.

Table 6.3: The summary of the correlation measurements between the EIMO device and the two other devices, for the ECG and PPG signals, over 50 one-minute samples.

	Case-GE	CM400	
	ECG	ECG	PPG
Maximum	0.464	0.488	0.920
Minimum	-0.005	0.012	-0.019
Mean	0.210	0.228	0.378
STD	0.137	0.144	0.274

The table shows that the best correlation match is 0.49 with an average of 0.23 for the ECG. Samples of these are shown in figures 6.14 and 6.15. The first shows the best correlated sample segment in the 50 samples used. The second shows the closest match for a sample that scored the average. Inset graphs on both show that there can be great variation within these samples, with the best and worst 5 second sections marked. The lower ECG correlation most probably stems from the difference in the morphology and base line drift when compared to the CM400 and Case-GE. These differences could be due to the filters used to limit the noise in the signal. A hardware and software bandpass filter on the EIMO device removes most of the low frequencies responsible for this drift. The drift itself is due to other muscle contractions such as breathing and posture. The visual loss of the ECG signal for

the Case-GE on the first graph was caused by the pads detaching as the participant adjusted posture.

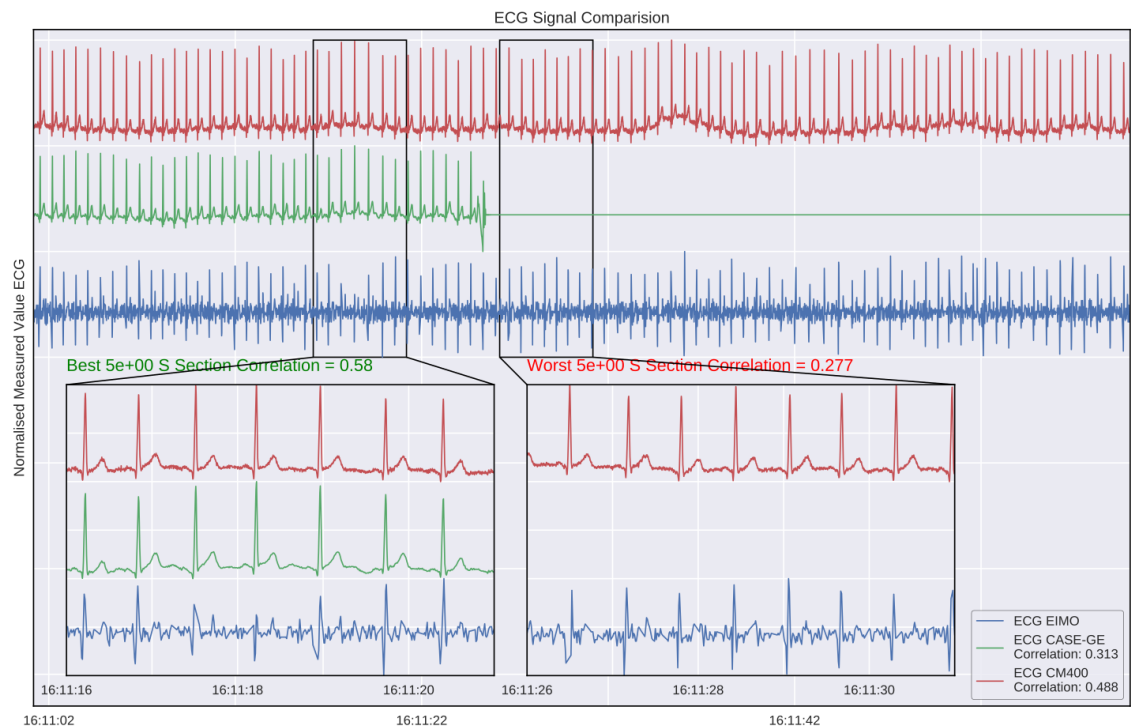


Figure 6.14: ECG signal comparison between the EIMO, CM400 and Case-GE devices with the best correlation for the whole sample between the three devices. Inset plots showing the best correlated 5 second section and the worse 5 second sections show that within the sample, shorter sections can report much higher and lower correlations.

The best correlation observed in the table was 0.92 with an average of 0.37 for PPG. Samples are shown in figures 6.16 and 6.17 where similarly to ECG, the first graph is for the best sample and the second is for the closest match to the average, where it is noticeable that the signal was lost part way through the sample and is reflected in the inset graphs. The inset graphs also mark the best and worst correlations found in each 5 second segment in the sample. The difference between the best and worst shows a large variation in the second case, when the signal was lost. The lost signal was most probably caused by the participant adjusting his or her fingers in the device.

The PPG can achieve a high correlation by having a smoother morphology with the lower frequencies allowing a large margin for alignment. However, for the ECG the briefness of the QRS complex shows, a small misalignment can drop the correla-

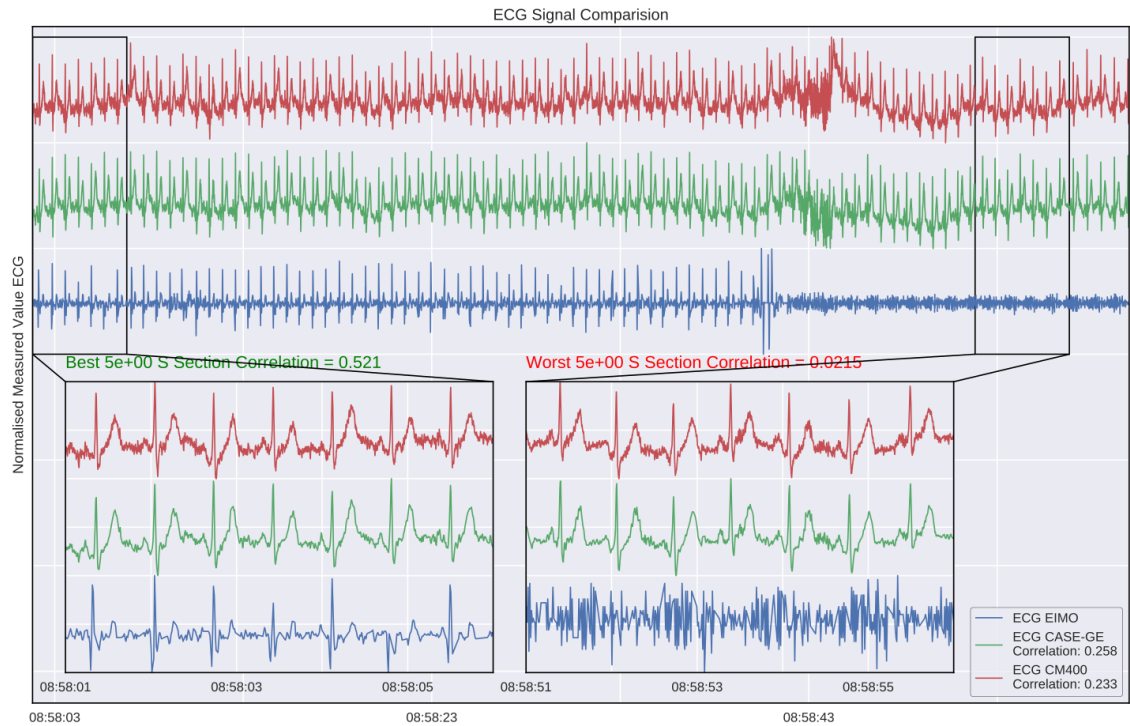


Figure 6.15: ECG signal comparison between the EIMO, CM400 and Case-GE devices with the closest to average correlation for the whole sample between the three devices. Inset plots showing the best correlated 5 second section and the worse 5 second sections, show that within the sample, shorter sections can report much higher and lower correlations.

tion metric drastically. The shape of the correlation curves are shown in figure 6.18 and this illustrates the sharpness of the ECG alignment as compared to the PPG.

The graphs also show that the timing of the ECG and PPG peaks and valleys are consistent with the other devices. This allows for the accurate measurement of the heart rate and the pulse transit time (PTT) between them. The heart rate and PTT are used in the blood pressure estimation in chapter 9. The signal timing appears consistent, but the morphology of the ECG signal shows visual differences. This is due to the more aggressive internal filtering to reduce the noise. The PPG signal does have a very similar morphology across the CM400 and the EIMO devices. However, there is more noise evident on the EIMO signal. The timing consistency and the similar morphologies allow the device to act as it was intended to; allowing a quick check of the user's rate and rhythm using the ECG and PPG as often as they like.

The other feature of interest in the comparison graphs is that the correlation metric does not always give a faithful representation of the device's compared per-

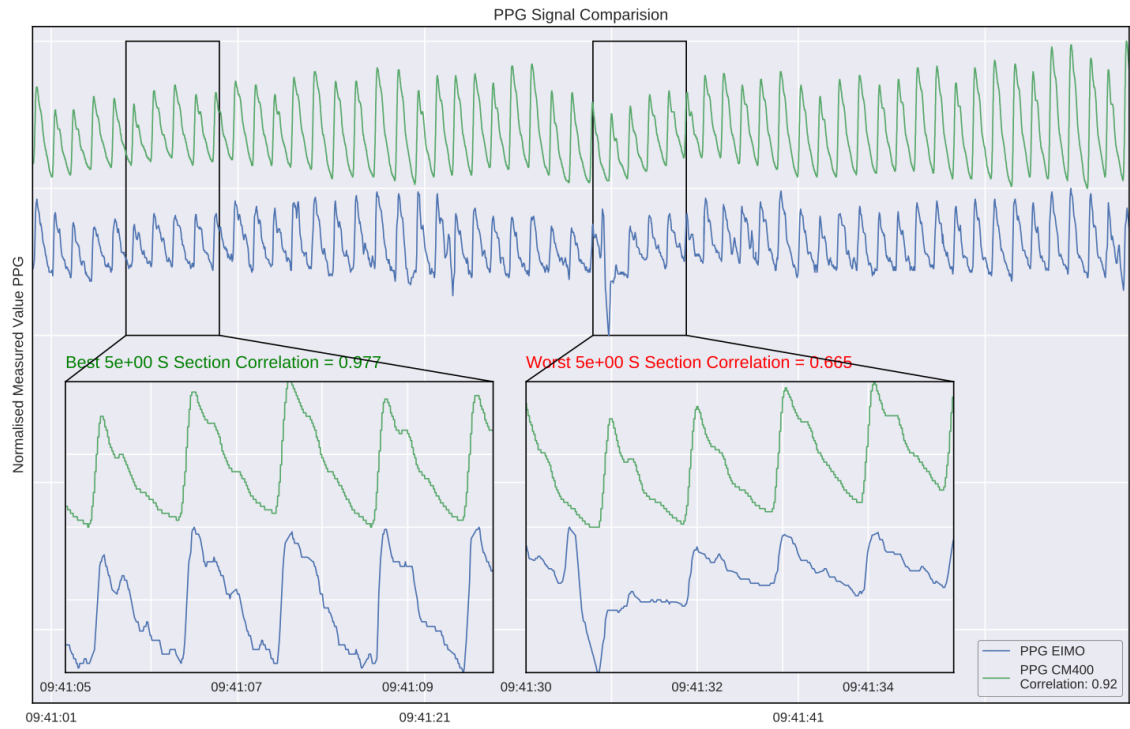


Figure 6.16: PPG signal comparison between the EIMO and CM400 devices with the best correlations for the whole sample between the two devices. Inset plots showing the best correlated 5 second section and the worse 5 second sections, show that within the sample, shorter sections can report much higher and lower correlations.

formance. This seems to be because the signals vary a lot in basic shape and quality throughout the samples. If a signal is lost or misaligned due to noise or signal loss in the ECG or PPG signal, the resultant signal morphology correlation is directly impacted.

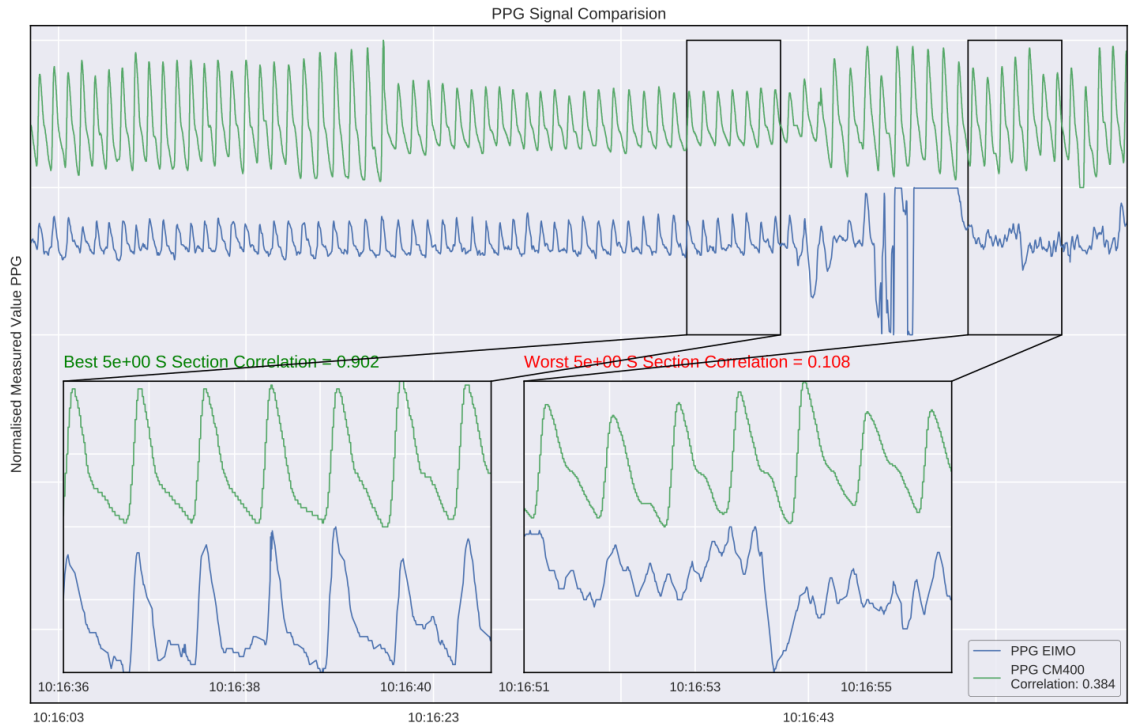


Figure 6.17: PPG signal comparison between the EIMO and CM400 devices with the closest to average correlations for the whole sample between the two devices. Inset plots showing the best correlated 5 second section and the worse 5 second sections, show that within the sample, shorter sections can report much higher and lower correlations.

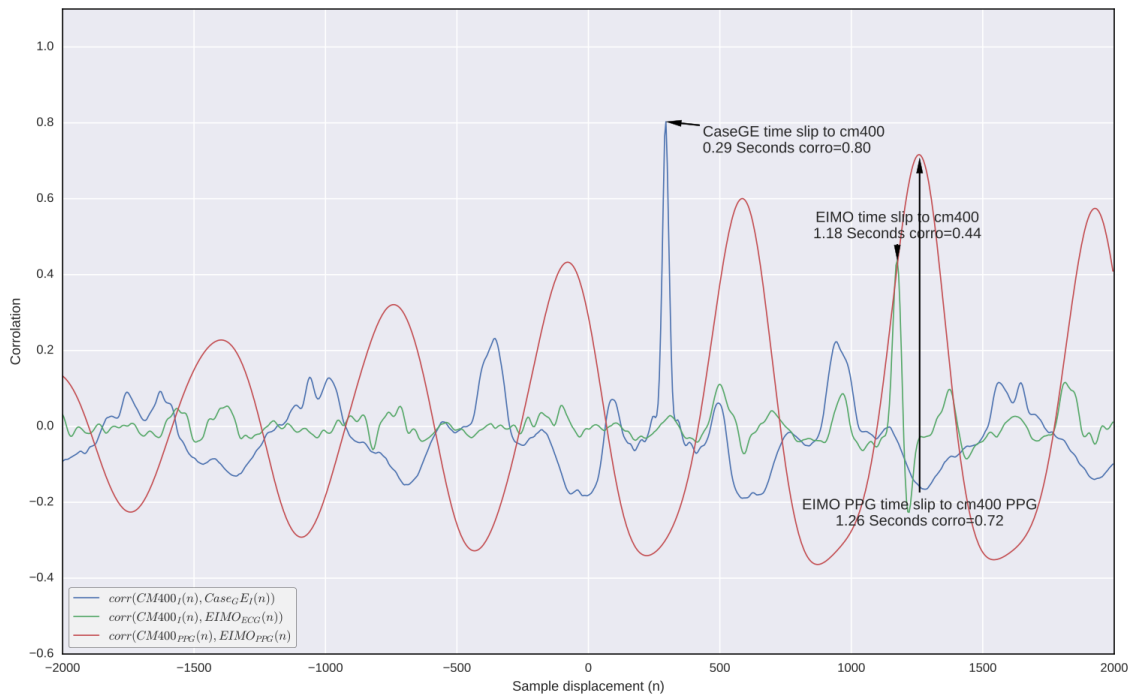


Figure 6.18: Signal correlation between EIMO, CM400 and Case-GE, showing the correlation function between the sampled signals, the time displayed is the synchronisation time offset. The mark also contains the best correlation between the signal and the reference. This alignment graph was taken for the best ECG sample for participant 6.

6.5 Summary

The two contributions in this chapter are first, the design of the EIMO device, which allows the simple gathering of vital sign information normally only available to much more invasive monitors requiring consumable ECG pads to be applied. This device has been used for data recording in a trial in order to ascertain its accuracy when compared to two other monitors. This has gathered a new dataset of vital sign data from the participants in the study to investigate the signal quality assessment in chapters 7 and 8, and model the blood pressure estimation in chapter 9. The secondary contribution of this section is the development of the log management tools and synchronisation methods. It is useful not only for the processing and testing of the EIMO device, but also has wider applications where the domain knowledge and a more in-depth ontology can be applied and the result scaled out for both unification and synchronisation of the other device data. The unified data log can then be stored in the data management framework as described in chapter 3 and appendix D.

7 Signal Quality Assessment

7.1 Overview

It would be beneficial to have a reliable and stable method for ascertaining the quality of the signals produced by the devices discussed in chapter 6. Having a reliable method would lower the storage and device load resources and increase the efficiency of data capture. The analysis system needs to classify data by using features, derived from the signal alone to be most effective, then it can be combined as required with other signals and is not dependent on them. Most data needs to be filtered before analysis. The incomplete and inaccurate data is removed or corrected either manually, or by using statistical measures. After these steps are taken, the dataset is ready for analysis. A robust but simple classification algorithm would be useful because the decision of ‘Good’ or ‘Bad’ classes of data would be best made at the point of collection in lower powered devices, saving time, storage space and ultimately, money.

The determination of signal quality to aid in process efficiency is similar to the work done by Orphanidou *et al.* (2015). The determination of signal quality can allow for more efficient data gathering methods where sensors, rather than being turned on for a length of time, could be activated selectively to gather a certain quantity of high-quality signals for a task. The same quality metrics could also be used to clean up data already recorded, readying it for further processing.

The first problem encountered when finding the quality of a signal is to determine how to classify a signal’s quality by human assessors. This will be the subject of the study at the end of this chapter. The second problem centres around being able to automate an accurate signal quality classification. This begins by extracting

features from the signal, then selecting the right features and model to allow the accurate classification of the signal quality using the assessment performed here as the benchmark, which will be presented in chapter 8. The discussion presented below aims to address the first problem in three steps.

The first step is to look at the literature concerning the quality assessment of the two vital sign signals in question, which are the photoplethysmogram (PPG) and the electrocardiogram (ECG); this is described in section 7.2. The second step, based on this information, was to outline a plan in section 7.3 for the segmentation of the signal, extracting features and adding annotations to the signal. The third step uses this system in section 7.4 to both add quality annotations to the signals and create a placebo set of annotations for a double-blind assessment of those qualities.

Together, the events, description and the peer-reviewed quality assessments can then be used in the further automation and estimation of signal quality by utilising machine learning algorithms, which will be discussed further in the next chapter.

7.2 Background

The assessment of signal quality first requires an appreciation of what is ‘Good’ and ‘Bad’ signal quality. This must be derived from an understanding of the signals in question through a search of the previous literature. The results of this are shown in section 7.2.1, which shows two main outcomes. The first explores where the signal data has been sourced from, as discussed in section 7.2.1.1. The second focuses on how signal data sources have been annotated in the literature; this is discussed in section 7.2.1.2.

The next part of the background presented in section 7.2.2 is a review of the issues and strategy for the framework described below. This is then used to classify the signals for the placebo control in study 1 as described in section 7.4. The system is then extended in chapter 8, which explores the best way to model and select features for accurate signal quality classification.

7.2.1 Signal Analysis and Quality Assessment Survey

A survey was done by Nizami *et al.* (2013) which shows that there is much disagreement and variability among the researchers working on this problem when finding signal quality indicators and classifying what seems to be ‘Good’ signal quality. The fact that there are many possible indicators that can be used to mark quality, shows that there is a certain ambiguity between which indicators are best to use. This resource lists many other works, as well as the systems they used; this style has been used for summarising the work done within this field, which follows below.

To classify the signals captured by devices such as EIMO, a conclusion needs to be drawn from the previous work, in order to setup an assessed database of ‘Good’ and ‘Bad’ classified signal data which can be used for training the models. The major points to note across the previous works were, the origin of the data used, and how they defined or found their signal quality baseline on this data in order to create a useful dataset for supervised machine learning.

Each of the points above have had their sources summarised in the following sections. The sources are grouped and then discussed with reference to the papers of note, starting with the source of the data used in their analysis in section 7.2.1.1. The annotation methods used and the various ways of gathering the quality standards for these signals are described in section 7.2.1.2.

7.2.1.1 Data Sources For Analysis

Two main sources of data have been identified for research comparison:

- Signal databases.
- Studies conducted by the individual authors.

The first source of signal data came from databases; most commonly the Physionet databases (Goldberger *et al.* , 2000). They include the MIMIC (Moody & Mark, 1996) and MIMIC II (Saeed *et al.* , 2011) databases and the Cardiology Challenges, notably 2011 and 2015. They have been used by many authors for determining signal quality (Li & Clifford, 2012; Behar *et al.* , 2013; Silva *et al.* , 2012;

Behar *et al.* , 2013; Orphanidou *et al.* , 2015; Martinez-Tabares *et al.* , 2012; Behar *et al.* , 2013). Less well known is the Capnobase Database (Karlen *et al.* , 2013), which contains respiratory rate information derived from the PPG data. This has been used by Karlen *et al.* (2012) to examine the signal quality of the PPG signal, and to assess a peak detection algorithm for determining a stable heart rate reading.

The data contained in these databases varies from multi-parameter (ECG, PPG, Arterial blood pressure, Respiration, etc. . .) in the MIMIC and MIMIC II databases, through to specifically focused datasets; the use of ECG in the MIT-BIH and the Cardiology Challenge, for example. The latter produced a yearly dataset to tackle certain problem areas. The 2011 Challenge (Silva *et al.* , 2011) was designed to improve the determination of signal quality in ECG signals in 10 second segments, from intensive care unit monitors. The 2015 Challenge was aimed at reducing false arrhythmia alarms, based on a segment of multi-parameter data around the alarm.

The benchmark for these databases have the added advantage that other researchers can use the same data for comparison. The disadvantages are that the databases contain very few annotations on the data which they are providing, so it is difficult to determine under what circumstances the data was collected. This can make assessing solutions problematic because it can be hard to tell what artefacts are due to the person or the circumstances, be it motion, electrical interference or mechanical failure. Also the signals they provide are not an exact match for the signals received by the EIMO device.

The second source of data in the work by Orphanidou *et al.* (2015), Sun *et al.* (2012), Aboy *et al.* (2005) and Sukor *et al.* (2011) is from the studies that the researchers did themselves, either using their own devices or externally manufactured and purchased devices. The advantages of this are that the circumstances in which the data recording takes place can be more specifically controlled. Specific protocols for exercise can be set up so that the results could be relatively easy to reproduce if the study was rerun, and as the protocol was developed for the question that the researchers had in mind, they had the ability to scrutinise directly the issues that their papers addressed. The main disadvantage of these data sources is that

the data from these studies are unique and so no comparison benchmarks can be employed to address and test the same issues by subsequent research. The data recorded from the EIMO device falls under this category, with no other benchmarks currently published on this new captured data. The signal comparison in section 6.4, shows that the signals captured are highly correlated to the other sources, but they are not an exact match. This will need to be taken into account, as this data from the study is the best test of the signal quality assessment on the actual data from the device.

The data sources discussed above need to be annotated for the quality of the signal. Some databases already have some form of annotation, but the independently recorded study data has no signal quality annotations. The data sources require sufficient signal quality annotations to be useful in training models. The issues around the quality annotation of these signals is discussed in the next section.

7.2.1.2 Annotations and Benchmarking for Quality

To build a classification and benchmarking system, a method for comparison must be established with a set of ground-truth annotations for the data that can be used as a standard for training and testing. These annotations must be added to the signal recording by having the signal assessed either by a human or a machine. The literature has been examined to find methods of creating the ground-truth baseline for the signal. There are three main methods of doing this classification.

The first, and it would seem the most useful, is to use annotations and cues from an already existing database to build and assess signal quality models. The main databases used in the literature are the MIT-BIH Arrhythmia Database (Moody & Mark, 2001), this carries beat annotations and alarm classification of the ECG signals only, also they keep signal quality annotations as timed marks in the signal. The MIMIC II database has ‘True’ and ‘False’ alarm data listed but no information on the signal quality itself. The 2011 Cardiology Challenge reviewed by Silva *et al.* (2011) has 10 second ECG signal segments marked as acceptable or unacceptable.

The data from these databases have been reused in past signal quality studies.

The signal quality annotations given in these databases have not been adopted and many papers have reported to have re-labelled the data in their studies. For example: in work by Behar *et al.* (2012), they state that they do use the ECG feature annotations from the database, but manually re-classify the quality, and Li *et al.* (2014) mention that they have re-annotated the quality for the purposes of that paper, most probably because of the lack of detail. Experience with the MIT-BIH dataset shows that there are few signal quality marks, although the exact reason the data was re-annotated is not reported. However, it is probable that they re-annotated the dataset to create more detailed training and test-sets for machine learning. Orphanidou *et al.* (2015) shows that they did indeed use the data from a few data sources and they manually relabelled the segments to give a better resolution to their dataset. There seems to be other databases when one explores the Physionet search system, few of which have meaningful signal quality marks attached or are as well used and known as the MIT-BIT or the Cardiology Challenge 2011 databases.

The second method and evidently most popular, is to take signal data from either published databases like Physionet or from a device that has either been built for the task or purchased, then manually annotate sections of this data as above. Some of the previous work does not give details on how this was done, and what methods they used for classification, like (Behar *et al.* , 2013; Sun *et al.* , 2012; Behar *et al.* , 2012; Li *et al.* , 2014). Others state that two assessors were nominated to annotate the signal segments with a third to decide on the discrepancies between the first two and give the final verdict (Li & Clifford, 2012; Orphanidou *et al.* , 2015; Silva *et al.* , 2012; Sukor *et al.* , 2011). Karlen *et al.* (2012) mention that a research assistant annotated the quality of the signals used.

The most notable reference for annotation information is the work done by Clifford *et al.* (2011). The authors took the base ‘True’ or ‘False’ annotations from the Cardiology Challenge but added more detailed signal quality data to this. They defined an eight step scale with a continuous value (from +1 to -1), and a letter (from A to F) to label the classifications. They dealt with the quality by having a

statement and description about what should be visible. Most of the descriptions hinge on whether it was thought that the signal could give accurate results or be ‘reasonable’. It is notable that they miss out the zero area and label each side at + or - 0.25, as the 0 quality signal could be difficult to define. For example if +1 was a ‘Good’ signal and -1 was a ‘Bad’ signal, it could suggest that 0 would be no signal but a bad signal might also be considered to not have a signal either. This might be a reason why they miss the 0 definition. To avoid this ambiguity, the definitions can be simplified to a two-class discontinuity for marking signal quality, which is ‘Good’ or ‘Bad’ or alternatively ‘Useable’ or ‘Unusable’. The terms ‘Good’ or ‘Bad’ will be used to describe these classes, and a definition of the heuristic features used for ‘Good’ and ‘Bad’ signal quality are described later in the chapter (see section 7.3.2.2).

The third method is not as widely used, but very interesting because this method looks for standards in features and signals and holds them as a template for classification (Aboy *et al.* , 2005; Mahri *et al.* , 2012). This could be in the form of a template signal that is kept and subsequent new signals are then compared to the initial template signal. Only signals which correlate closely are considered ‘good quality’ (Fu *et al.* , 2010). The primary question arising from this scenario is ‘how does one find the perfect signal to compare or the most descriptive feature statistics?’ This issue should be kept in mind for the remainder of the chapter and is discussed directly in chapter 8.

Since the direct purpose is to assess the signal data from capture devices like the EIMO device, the data captured from the study in section 6.4 should be annotated to make a viable training dataset as in the work discussed above. Since there can be ambiguity in what constitutes a ‘Good’ and ‘Bad’ signal segment, the assessment will be divided between two layers. The first layer of assessment will be done where one human annotator adds annotations to the data for ‘Good’ and ‘Bad’ using the scheme above. Then in the study detailed in section 7.4, these annotations are compared by other assessors as the second layer of assessment.

A signal data repository with both data signals and with annotations would

be the preferable solution, since it would allow simple benchmarking for quality algorithms, with the ability to find a standard for categorisation of signal quality. The main reason this may not have been addressed previously seems to be that there is great difficulty with permission to open and publish all of the data from a study while maintaining data protection for the participants. There needs to be a way for people to annotate data easily and preserve all the different types of annotations. As the quality given might be disputed between different researchers, each of their annotations should be preserved so comparisons can be drawn. These issues, although understandable, leave an opening for a system which can address these issues and keep the results of signals and annotations clean and presentable. Further discussion on this topic can be found in section 7.3.2.2 and ultimately this question has led to the work described in chapter 8 which looks at methods to assess and automate the signal quality assessment. The primary system developed for doing this is the framework discussed in chapter 3, which has been utilised for the signal analysis explained in section 7.3.

7.2.2 Classification Strategy

In order to produce a classified set of signal quality annotations for recorded signals, a system of classification needs to be described. A review was carried out to find strategies and issues that will need to be addressed. Formulation of the ‘Good’ and ‘Bad’ signal quality system should incorporate the important aspects of novelty or anomaly detection.

Work has been done to review the field of novelty detection by Pimentel *et al.* (2014). They present a comprehensive review of the state-of-the-art methods for novelty detection, highlighting the wide variety of methods and evaluations used, as there is no ‘best case’ or ‘optimal’ system for all data sets. Therefore, the type and distribution of the data needs to be considered when choosing a methodology. Further surveys by Chandola *et al.* (2007, 2009), describe techniques for anomaly or outlier detection. The authors expand on the details along with describing possible applications that the methods could be used for. The applications that they discuss

are: detecting network intrusion, fraud, image processing and novel topics in text mining.

Novelty detection is a process designed to find uncharacteristic patterns when there is not enough data to assemble a reasonable training set (Pimentel *et al.* , 2014). Anomaly or outlier detection however, is the process of finding the patterns that are outside of normal behaviour (Chandola *et al.* , 2009). They do share a lot in common in terms of classification and data flow, except that novelty detection is more common when the construction of an outlier training class is difficult. In both cases, anomaly and novelty detection are forms of classification problems (Pimentel *et al.* , 2014; Ding *et al.* , 2014; Chandola *et al.* , 2007). This classification is normally determined by the features derived from the data at particular event boundaries. There are three issues raised above, these are:

- Determining effective learning models to use.
- Methods for the segmentation of the data.
- Determining the features to describe the segments.

These issues are expanded upon in the next sections.

7.2.2.1 Simple Learning Models

The classification system used requires a model which is reliable to train. To be reliable, a model should have a cost function that can be optimised without hitting local optimums. Support vector machines (SVM) (Kononenko & Kukar, 2007) are perfect for this task since they have the advantage of a convex error surface which makes training them straight forward. Further to this, they can use kernel functions to vary the metric of the boundary space. This allows SVMs to emulate many different forms of model boundary by careful selection of the kernel function. With these advantages, SVMs can be used to provide an easy model for classification, which will give the best classification boundary for any given set of data. More on machine learning methods will be discussed in chapter 8, when examining other

methods to explore the classification strategies. Here, the model generalisation and classification accuracy will be of utmost importance.

7.2.2.2 Segmentation

Useful segments are defined as decision boundaries at points through-out the signal, where there is enough information to make descriptive features. The meaning of the events or marks themselves is not the important part of their determination however. The segment boundaries should be an easily definable part of the signal and robust to signal changes and occur reasonably frequently for the problem at hand. A number of these segmentation events have been used in previous work by Orphanidou *et al.* (2015) and Behar *et al.* (2013). For example, they use a fixed segment length of 10 seconds, as was used in the Cardiology Challenge 2011 (Silva *et al.* , 2011), which was their source of signal data. Others include Silva *et al.* (2012) which trained on and assessed a 10 minute epoch before an alarm. This created a continuous simulation of the signal for quality assessment but only one sample was used over the 10 minute epoch. The most promising efforts were made by Li & Clifford (2012) and Sukor *et al.* (2011), where they used the cycle valleys of the waveform to classify the signal as ‘Good’ or ‘Bad’. This created a variable length segment. The cycle periods are the most important part of the ECG and PPG signals as they occur approximately every second or so. This leads to reasonable granularity in the signal events. This will be explored further in section 7.3.2.1.

7.2.2.3 Features and Descriptions

The process starts by finding a set of characteristics that best describes the segment in the feature space defined, which enables the correct outcome prediction. Features can be found using multiple sources, and can include statistical measures (for example mean, standard deviation or skew), morphological measures (such as correlation, peak-to-valley amplitude distance, cycle time or amplitude), along with signal mapping algorithms (including curve mapping methods (Kumar *et al.* , 2012), as well as time warping methods (Li & Clifford, 2012)). These measures can also be

classified as context insensitive measures, as they are evaluated for the signal or event by mapping algorithms.

The final choice of the features will be explained further when describing the classification system below in section 7.3.3.1. These features will be expanded in the next chapter.

7.2.2.4 Performance Metrics

There are many different metrics, which could be used to assess the accuracy of a model. They are split into two main types. The first is to assess the performance on continuous outcomes, such as a correlation coefficient between the estimated and the actual outcomes, or the L1/L2 norm error between the estimate and the actual outcomes. The second are metrics used to assess discreet outcomes, where there are a finite number of classes that the result could be chosen from. The two that can be focused on here are the strategies used for classification; primarily using the confusion matrix and the receiver operator characteristic (ROC) graphs, discussed below.

Confusion matrix

The confusion matrix is an excellent way of comparing the results from classification models by displaying four groups called: True Positives (TP), True Negative (TN), False Positive (FP) and False Negative (FN). These are laid out as shown in table 7.1 (Witten *et al.* , 2011).

Table 7.1: The model confusion matrix.

		Estimated	
		True	False
Actual	True	TP	FN
	False	FP	TN

These groups can be used to estimate three main metrics, these metrics are defined as:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \quad (7.1)$$

$$Sensitivity = \frac{TP}{(TP + FN)} \quad (7.2)$$

$$Specificity = \frac{TN}{(TN + FP)} \quad (7.3)$$

Accuracy defined by equation (7.1) shows how well an estimator estimated the actual classes, for both ‘True’ and ‘False’ classes. However this can be skewed if the number of the classes are imbalanced. The two other metrics are less subject to imbalances. They are the sensitivity (defined in equation (7.2)) and specificity (defined in equation (7.3)). These metrics look at the accuracy of the ‘True’ or ‘False’ class respectively, thus showing the true accuracy of each, given any number of ‘True’ or ‘False’ classes.

Receiver Operating Characteristics (ROC)

A good introduction for Receiver Operating Characteristics (ROC) analysis was given by Fawcett (2006), where they explained the best approach to using the ROC charts to assess model performance. Here they describe the algorithm for calculating and plotting the graph, including other parameters discussed further in the next chapter. They also show that many operating points can be plotted on a single ROC graph as seen in figure 8.1, allowing easy comparison between them. This compound graph, or ROC field, can be used to display and compare the results of many models together as it is used to display the results of the studies conducted in section 7.4. Each point on the graph can show the operating point of a model. The vertical axis of the field shows the specificity which, as shown above, indicates how accurate the model is at correctly estimating the ‘True’ instances. The horizontal axis shows the sensitivity as defined above and is the estimation accuracy of the model for ‘False’ instances. This is usually reversed by either subtracting it from one or reversing the axis, to bring the highly sensitive models towards the left edge. Therefore, a point in the top left of the field would be the optimum, showing that the model has a high accuracy with both ‘True’ and ‘False’ classes.

7.2.3 Summary

The survey mentioned above (Nizami *et al.* , 2013) demonstrates that there appears to be much disagreement between the researchers working on the problem of finding reliable signal quality indicators. There seems to be certain databases used in other works, most notably the Physionet MIMIC II, MIT-BIH and Cardiology Challenge 2011 databases and other signal repositories, but limited ground-truth information is associated for signal quality, and when there is, it is not generally used in the literature. Signal quality assessment seems to be mediated by two human assessors looking at segmented signals, with a third to pick out the discrepancies between the first two. It has been noticed that the third assessor might have more experience with classifying signals since the opinion of the third assessor is the one held to be the final truth. There are two main ways to gain useful signal quality classifications. The first is to use a peer review standard, and so even if the annotations are not perfect, all of the algorithms can, at least, be compared. This standard is the subject of the rest of this chapter. The data from the EIMO device recorded in section 6.4 can then be used, with annotations produced for it. These can then be assessed by other human annotators as is discussed in section 7.4.

Looking into the issue of benchmarking, there is an opening to create a database and analysis system for medical signals. Starting with this data and defining the ground truth based on the same database scheme as used for the Physionet, allowing annotations to be added to the data and then these can be compared. For more detail on the framework that the system is built on, see chapter 3. A proposed analysis and annotation system has been developed to facilitate the work done here and is explained in section 7.3. The annotations and annotating interface used in the system are described in section 7.3.2.2.

7.3 Classification System

The system proposed here is used to aid in the classification of time-based signals, which can be broken down into four parts; these are detailed in the primary system

structure below in section 7.3.1.

The structure of the classification and learning analyser is the main consideration for this architecture to aid in the discovery and testing of classification and regression knowledge models, for the real-time analysis of time-based signals. The main purpose in describing the framework for the system is that, with a system in place for the processing of time based signals, the analysis can start out more simply. New learning methods, segmentation and features could be easily added to the system by replacing and extending the modules defined below and is described in chapter 8.

The ECG and PPG signals, as recorded by EIMO and the CM400, which are described in section 6.4.2, shall be used as a case study for testing and illustration of the design where appropriate. Other signals, however, could be used for segmentation, feature extraction and quality assignment modules, with the modules upgraded as required to keep them accurate.

7.3.1 Main Structure

The main principle driving the quality analysis of the data is that the data might not all be of sufficient quality, and the contribution of the data could be regulated based on the quality of the data seen. The system has been designed to emphasise that the primary processing elements for feature extraction, outcome assignment and the classification or regression of the estimated outcomes have been designed using a modular architecture with interchangeable blocks.

The structure being developed to classify the stream of events, as briefly described above, has been designed to classify and remove events in a hierarchical fashion. Figure 7.1 shows this modular structure and operation of the signal analysis system. The four main modules shown are as follows:

1. Input - The inputs to the system in the form of a set of signal annotations for quality and the signal events which segregate the signals into decision boundaries, either found automatically or manually, are described in section 7.3.2.

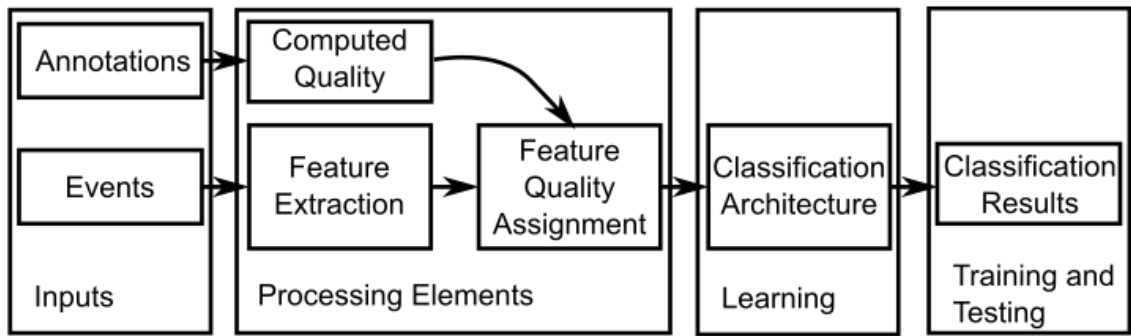


Figure 7.1: System overview and data flow throughout the system showing the four main module types. These can be further broken down into the different sub-modules. The arrows denote the interactions between these elements, the data flows from left to right. Each module can be defined and is replaceable.

2. Processing Elements - The modular processing elements of the system where the event and annotations are processed to describe the event features and assign them an outcome. The process is described in section 7.3.3.
3. Classification Architecture and Training - The main replaceable module of the system, this contains, trains and runs the model processing architecture, such as ANNs or SVMs, using the instance event data as described in section 7.3.4.
4. Testing and Result Analysis - Within this module are the elements that assess the prediction accuracy of the classifier module, where the outcomes of the learning module can be analysed and measures of accuracy generated. This is discussed in section 7.3.5.

The full signal quality analysis architecture is described logically below. This builds onto the analysis framework as laid out in section 3.4, where the low level implementation of the processing elements is described in section 7.3.3. The following modules and processes described below are run within that framework and so take advantage of the process management and data flow system described there.

7.3.2 Data Inputs

There are two main inputs to the system. The first input is a series of events defining decision boundaries on the time-based signals. The acquisition of these events will

be discussed in section 7.3.2.1. The second is a series of annotations provided for the signal, which will be described in more detail in section 7.3.2.2.

7.3.2.1 Signal Event Segmentation

The signal events are points in the signal marked by a reliable feature point. The markers, therefore, allow for a system where classification is performed in real time and assessments happen without unreasonable delay. The time-based markers can be used as a decision boundary. This allows the framework to know when sufficient conditions, necessary for analysis and classification, have been reached. As found above, they can either be defined over a fixed time irrespective of the signal or based on a defined point in the signal. They are manifested in general as events, or notable local key points. The marker's exact nature would depend on the signal in question, so an example would serve to illustrate the segmentation of the signal into events, then to assess the stability of these events.

Event Marker Stability

The events in continuous cyclic signals such as the PPG or ECG can either be the peaks or valleys of the signal, as both would be repeating points in the signal cycles. The peak of the QRS complex would be a natural place for the event point to be assigned to the ECG signal since this is the most notable feature. By contrast, the event placement is not as clear cut in the PPG signal. There are peaks (maximum pulse pressure), and valleys (minimum pulse pressure), both of which are interesting features of the cycle. The valley is a more interesting feature since this point of minimum pressure is the point at which the heartbeat pressure wave arrives at the area of measurement. Therefore, the most consistent feature between the two should be selected, to keep the descriptions as consistent as possible. In the previous studies described above in section 7.2.2.2, the valley was chosen as it is the start of the pressure wave, but for the quality and classification of the signal, the most stable point of the wave should be used as the boundary.

A thought experiment and hypothesis can be deduced at this point: it would stipulate that the peak would be the most consistent feature since the energy of the system changes under different pressure characteristics. When the peak is found, the vessel is under high pressure. This will mean that a signal will require more energy to distort the volume within the vessel under measurement. Conversely, when the vessel is under low pressure, and therefore low energy in the valley of the signal, the same level of noise energy can have a bigger impact, thus distorting the signal more significantly, and distorting the feature's position due to the noise.

Given this is an issue about stability, what is required is the point at which the highest signal-to-noise ratio (SNR) is seen in the system. To calculate the potential signal to noise in the system, one can start by examining the energy of the vessel when at the highest and lowest pressure. It can be shown that deriving from Hook's law and constant \mathcal{K} and the 'work done' equations that the energy of an elastic deformation is given by equation (7.4) and for a closed walled elastic tube is then given by equation (7.7) (Landau *et al.* , 1986). This defines that the energy U is proportional to the change in radius dy squared.

$$U = \frac{1}{2}\mathcal{K}dr^2 \quad (7.4)$$

$$\mathcal{K} = \frac{EA}{C} \quad (7.5)$$

$$C = 2\pi r \quad (7.6)$$

$$U = \frac{EAdr^2}{4\pi r} \quad (7.7)$$

Where, dr is the change in radius r of the artery. Subject to an Youngs modulus E for lateral expansion of the vessel and A is the cross-sectional area of the vessel's wall and finally C is the circumference of the vessel or the length of the spring and the vessel is of a unit length L .

Relating this potential energy to the pressure using the vessel distension formula as seen in equation (7.8) from work presented by Bramwell & Hill (1922) and Roylance (2001) on the pulse wave velocity in humans. This equation indicates that for a given pressure rise P_p , the radius increases, which in turn increases the volume capacity of the vessel. This cyclic shape shows in the PPG signal as the pulsatile

part of the signal, familiar on blood oxygen monitors.

$$dr = \frac{r^2 P_p}{EA} \quad (7.8)$$

Substituting 7.8 into 7.7 and simplifying gives:

$$U(P_p^2) = \frac{r^3}{4\pi EA} P_p^2 \quad (7.9)$$

If $c = \frac{r^3}{4\pi EA}$, substituting it in to equation (7.9) gives equation (7.10). This shows simply that the potential energy of the elastic vessel is proportional to the square of the pressure difference.

$$U(P_p) = cP_p^2 \quad (7.10)$$

$$U(P_p) \propto P_p^2 \quad (7.11)$$

Setting P_p to the peak pressure difference $d\hat{P}$ and valley pressure difference $d\check{P}$ gives:

$$U(d\hat{P}) \propto d\hat{P}^2 \quad (7.12)$$

$$U(d\check{P}) \propto d\check{P}^2 \quad (7.13)$$

and since the pressure at the peak is greater than in the valley, $d\hat{P} > d\check{P}$.

Finding the signal to noise power ratio SNR at the peak pressure SNR_P and at the valley SNR_V .

$$SNR_P = \log \left(\frac{U(d\hat{P})}{n} \right) \quad (7.14)$$

$$SNR_V = \log \left(\frac{U(d\check{P})}{n} \right) \quad (7.15)$$

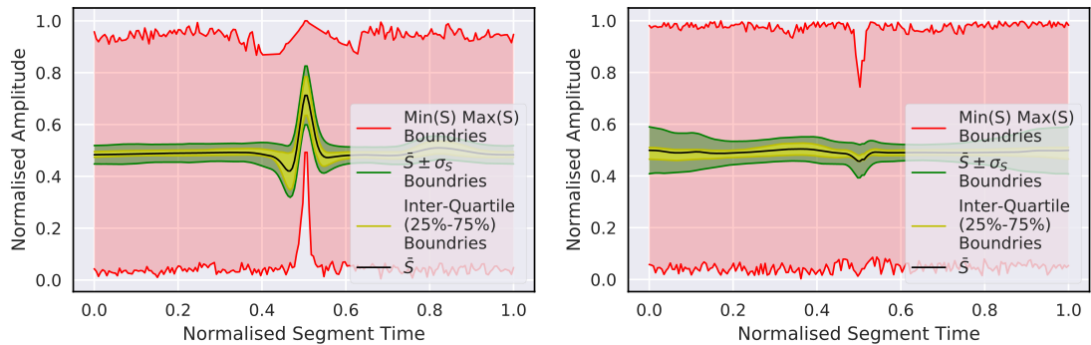
where n is the noise energy in the vessel area.

$$\therefore SNR_P > SNR_V \quad (7.16)$$

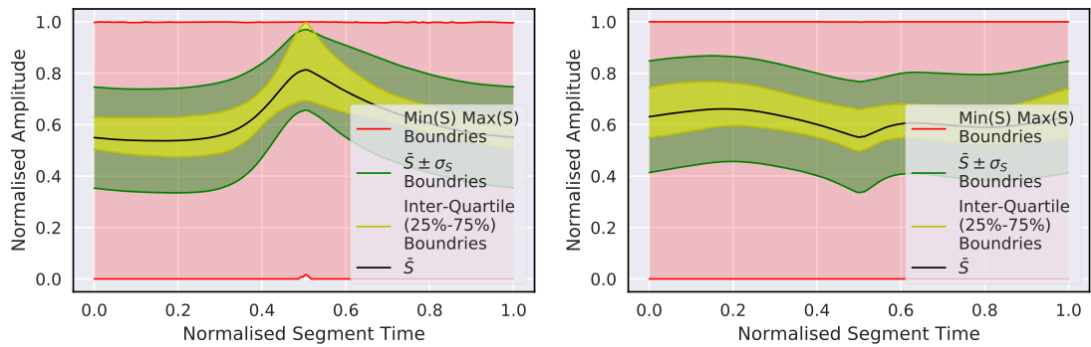
Given equations (7.14) and (7.15), the SNR is proportional to the pressure. If taken at the peak, the energy of the system will be larger than at the valley. Therefore the SNR will be bigger than if taken at the valley, under a given amount of noise power as given in equation (7.16). This suggests that the peak event at higher energy should be more stable, and more consistent for triggering an event.

An experiment was done to show the framing of the PPG and ECG signals for morphology analysis. Using the first 2 recording periods of data from Users 5 and 6 in the study described in section 6.4 was used to illustrate the differences in signal segmentation using peaks and valleys. Figures 7.2c and 7.2d shows the overlapped signal frames for the PPG signal, figures 7.2a and 7.2b shows the overlapped signal for the ECG. The first is overlapped using peaks as the points of reference. The second shows the same data, but using valleys as a point of reference. The graphs illustrate that using the peak for signal framing, yields the most consistent signal frames. When using the valleys, the noise allows the position of these to drift, thus distorting the signal morphology.

Given a PPG signal waveform, such as one recorded from the EIMO device, the signal events, as discussed above, are to be the detected peaks of the PPG waveform. The primary considerations for their definition are that they are in a stable position within the signal and that they are generated with sufficient frequency to serve as a point of classification. The choice of using the peaks of the PPG or QRS in the ECG waveform, satisfy the points mentioned above since the peaks occur at least once a cycle and when classified correctly, form a classification frame for the incoming data.



(a) Segment peak versus Valley comparison for ECG using peaks as the event boundaries. (b) Segment peak versus Valley comparison for ECG using valleys as the event boundaries.



(c) Segment peak versus Valley comparison for PPG using peaks as the event boundaries. (d) Segment peak versus Valley comparison for PPG using valleys as the event boundaries.

Figure 7.2: Statistical display of the data logged for each segment with peak-to-peak or valley-to-valley boundaries for both electrocardiogram (ECG) (a) and (b) and photoplethysmogram (PPG) (c) and (d) respectively. The same signal data was used in both cases. The scales shown are both normalised for amplitude and for the cycle time to allow the morphology to be shown more clearly. The signal segments are taken from Users 5, 6, 7, 8 and 14 using segments found from the first two periods from the first session using 24 minutes of a signal per user. The black lines shows the mean segment levels. The yellow area shows the inter-quartile range and the green shows the first standard deviation from the mean. Lastly the red area shows the minimum and maximum of the signal segments reached. (a) and (c) show the most pronounce average signal which were when the boundaries were from peak-to-peak. In (b) the QRS peak has been averaged away and in (d) the defined PPG segment as also been averaged away suggesting that the signals are less consistent in the valley-to-valley arrangement.

Signal Segmentation

Shin *et al.* (2009) describes the PPG waveform signal $I(i_n)$ where i_n is the sample index as ‘very dynamic’ due to the breathing rhythm, motion artefacts and of course, local BP changes, which causes significant variations in the transmitted light intensity between cycles. They suggested a method of peak detection that uses an adaptive threshold technique. They proposed a threshold which decays from the previous peak with time; so in effect, the detector becomes more sensitive the older the original peak is.

$$D_{i_s} = D_{i_s-1} + \tau \frac{\hat{I}(i_s - 1) + std_{PPG}}{F_s} \quad (7.17)$$

Where, i_s is the current peak index, $i_s - 1$ is the last peak index, D_{i_s} is the current decay constant, D_{i_s-1} is the previous decay constant, τ is the slope changing rate fixed at +/- 0.6 heuristically in their paper, $\hat{I}(i_s - 1)$ is the last peak, std_{PPG} is the standard deviation of the PPG signal and F_s is the nominal sample rate of the signal.

The signal adaptation in their method, which is shown in equation (7.17), was to allow the decay constant D_{i_s} to vary with the amplitude of the previous peak $I_{pk}(i_s - 1)$ from cycle to cycle. The sampling frequency F_s and the standard deviation of the signal std_{PPG} are practically constant for any given sampled signal. No adaptation for the cycle timing of the signal was used.

The adaptive decay constant was extended to improve the performance of the algorithm by allowing the decay constant to vary with the previous peak height, cycle period and the estimate of the noise floor of the signal. This was to enable the previous cycle characteristics to predict a peak height and the threshold at the next peak arrival time. This sets the decay constant D_{i_s} accordingly and adaptively at every cycle. The new definitions are:

$$T_c(i_s - 1, i_s - 2) = T(P_G(i_s - 1), P_G(i_s - 2)) \quad (7.18)$$

$$I_{min}(i_s - 1) = \frac{1}{l} \sum_{i=1}^l (P_B(i_s - (l + i))) \quad (7.19)$$

$$D_{i_s} = \frac{(P_G(i_s - 1) - I_{min}(i_s - 1))}{(T_{IHB}(i_s - 1, i_s - 2))} H \quad (7.20)$$

Where, $T(A, B)$ is the time difference between peak A and peak B , $P_G(\cdot)$ is the set of the ‘Good’ detected peaks above the threshold. $I_{min}(i_s - 1, i_s)$ is the estimated noise floor from peak. $(i_s - 1)$ to i_s , $T_c(i_s - 1, i_s - 2)$ is the period of the previous heartbeat. $P_B(i_s)$ is the set of ‘Bad’ peaks, which are peaks that did not make the threshold previously, H is the coefficient to determine the decay rate and L is the number of ‘Bad’ peaks to look back over.

There are only two parameters to choose with this model. H is chosen to be the desired estimate of the next peak at the time that the previous cycle period is over. Setting $H = 1$ would make the threshold drop to the noise floor and a setting of 0.5 would make the threshold drop by half of the peak to noise floor distance by the estimated end of the period. The choice of l and thus the noise floor is based on the desired performance as the more bad peaks that are averaged over, the better the estimate. The threshold $Th(n)$ is therefore decayed with time $T_{pk}(i_s - 1, n)$ as is the period from the last good peak $(i_s - 1)$ until the current sample n and defined as in equation (7.21).

$$Th(n) = P_G(i_s - 1) - D_{i_s-1}T_{pk}(i_s - 1, n) \quad (7.21)$$

Any detected peaks lower than $Th(n)$ are classified as ‘Bad’ peaks and are used to estimate the noise floor in equation (7.19). The first peak greater than the threshold $Th(n)$ and over the minimum peak duration, $0.3s \leq t \leq 2.0s$ is classified as a ‘Good’ peak. These would translate into a possible heart rates of 30bpm for 2 s and 200 bpm for 0.3 s. To recalculate the decay constant D_k , equations (7.19) and (7.20) are used; the period between the current and last ‘Good’ peak T_{IHB} . This can then be used as an estimate of the period of the next cycle, therefore it can also be used as an estimation of the instantaneous heart rate.

The adaptive threshold is robust to noise because it reduces the decay rate if the estimated noise floor I_{min} is high, which means a peak has to overcome the higher threshold to be considered as a valid peak. It is also robust to false peaks because it adjusts the sensitivities of the peak detector threshold by modelling the dynamics of the signal regarding peak level, noise floor and cycle period.

There is a simple signal quality measure within the peak detection algorithm.

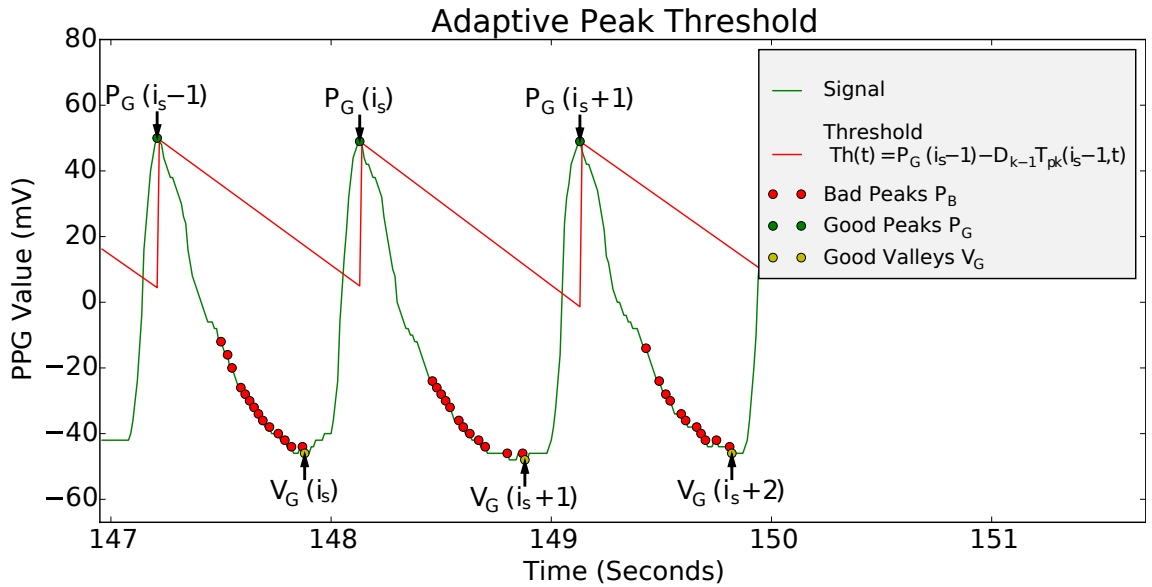


Figure 7.3: The action of the adjustable threshold in the peak detection algorithm is shown in red on an example PPG signal (green), showing the adaptive threshold from Munnoch & Jiang (2015). The 'Good' P_G and 'Bad' peaks are marked with green and red markers respectively. The 'Good' valleys are marked in yellow. The i_s is the index of the current last peak seen.

This uses the number of consecutive 'Good' peaks within 10% of the last period to maintain the quality. On the first 'Bad' peak, the signal quality is reduced to 0. This measure effectively looks for consistently timed periods within the signal.

7.3.2.2 Signal Annotations

To allow the training of supervised learning architectures, the baseline quality of the signal is required to form a standard for the signals in question. In the currently implemented system, this can be deduced from the annotations given about the signal, with events found from the segmentation, dynamically allocating a quality result class, by interpreting the signal quality transitions. First the types of annotations used within the framework are discussed. Next, the interface for adding annotations and markers to the signals is described. Once the interface is presented, the definition of what is meant by the 'Good' and 'Bad' classifications can be made. Lastly the process is described that interprets the annotations.

Annotation Types

The annotation style used for this research is derived from the Physionet database and the past work, discussed above as a starting point; see table D.6a in the appendix D.2.1. They use annotations to add metadata to their recorded signals as others have done before them. The annotations used were a mark showing the assigned quality of a section of the database’s ECG signal. Their annotation categories will be used as a staging point, with extra types added by the author to extend the types of annotations possible. Other annotation categories were added to gain the ability to gather better resolution and categorisation, which can be seen in table D.6b. These new annotation types allow for more expressive marking within sections by either a general mark, or a named signal. The name is stored with the annotation, and details of the storage and interface can be found in section 7.3.2.2. The annotation scheme implemented, collects and stores this list of annotations for the signal as a real floating point score. This was derived from the classification table in the work done by Clifford *et al.* (2011). This means that, without modification, the signal quality can be given at any point in a continuous set if required. The quality is currently assigned as discrete -1 or ‘Bad Quality’ and +1 or ‘Good Quality’ creating a two class binary problem as mentioned above. By holding the quality at the two limits the user annotating the signal cannot impose a perceived or emotional weighting to parts of the signal. The annotations are defined during a transition in quality, marking them as either 1 or -1, marking all of the signal’s events by using inferences from the annotation transitions, or marks. Internally, it must be recognised that every signal quality annotator who assesses the signal must have their own thresholds of what is ‘Good’ and what is ‘Bad’, by restricting the quality to the two extremes, a simple decision can be made by the Assessors. The variation in this decision for each person is explored in the study described in section 7.4.

The annotation types added come in two main types: general and named forms, as can be seen in figure 7.4. The first group contains the quality marks for both named and general signals. These are used to assign the signal by default to a

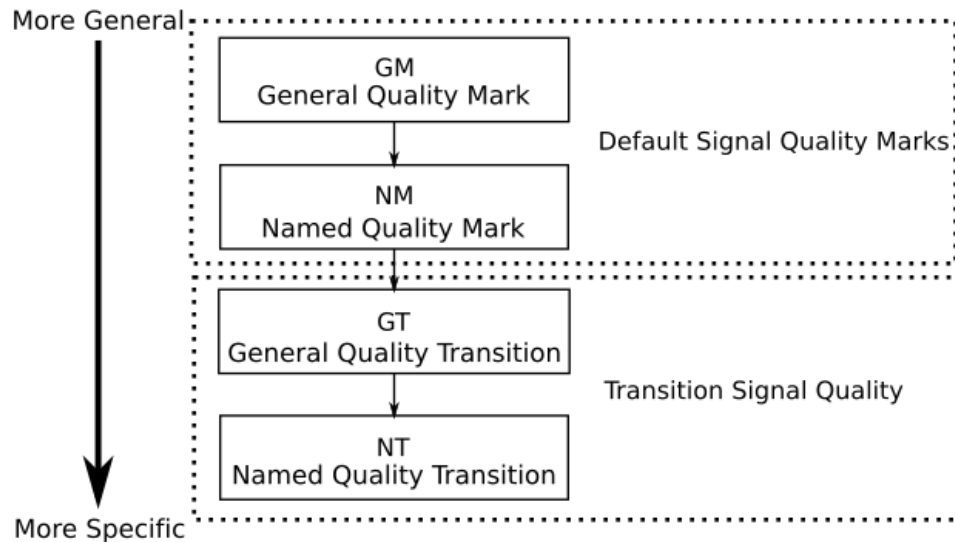


Figure 7.4: Annotation type hierarchy for signal quality marks and transitions. The quality annotations come in two groups: first, a quality mark to fix a quality for the signal section with or without a timestamp; and second, a quality transitions which shows that at the timestamp, the signal quality transitioned between the prior and post quality values. Both of these annotations come as general and named. General annotations apply to all the signals, and named ones are specific for only the named signal. This forms a hierarchy from the most general, a general signal quality mark (GM), with no specific time or signal through to the most specific, a named quality transition (NT).

quality score; no time is given for these, as it is used as a default for generating the signal. A good example of their use is to set default quality for a segment of the signal. The whole set can be allocated this quality as the general form, or each signal can be given its own corresponding mark.

The second group contains signal quality transitions. These transitions are more descriptive, as there is a general and named form. These data points have to have a time associated with them, as they mark a position along the signal. The quality can then transition between two continuous levels for all or some of the signals, depending on whether the transition was a general or named signal respectively.

Table 7.2 and figure 7.5 shows an example annotation set and the quality trace it creates, illustrating the quality changing at each of the annotations events.

Table 7.2: Example annotation table. This consists of a made up set of signal quality annotations, showing a General Mark (GM), Named Mark (NM), Named Transition (NT) and General Transition (GT) in use and the other information that they require (N/A)s are parameters which are not required for that annotation).

figure 7.5.

Time	Type	Quality Before	Quality After	Signal Name
N/A	GM	N/A	1.0	N/A
N/A	NM	N/A	-1.0	Signal 1
T1	NT	-1.0	1.0	Signal 1
T2	NT	1.0	-1.0	Signal 1
T2	NT	1.0	-1.0	Signal 3
T3	NT	1.0	-1.0	Signal 2
T4	GT	-1.0	1.0	N/A

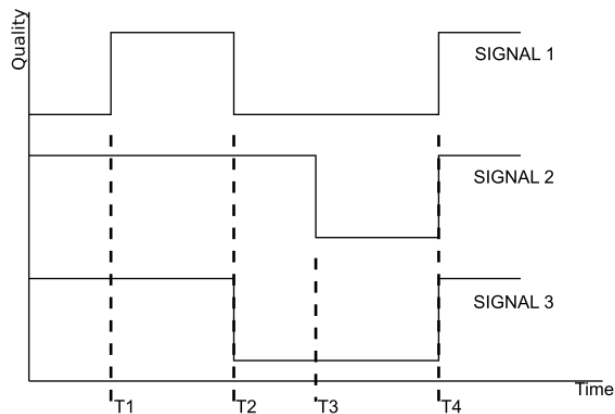
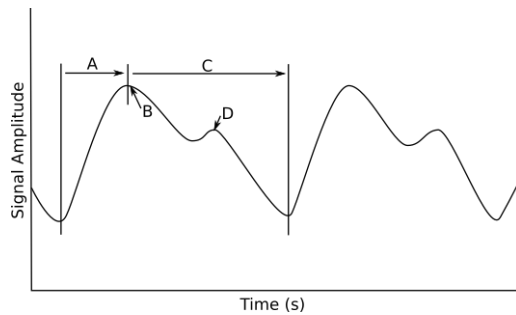


Figure 7.5: An example of how the annotations from table 7.2 are interpreted into a signal quality trace. The dotted lines are the time stamped annotations. The traces for the three signals are the signal quality traces made by these annotations.

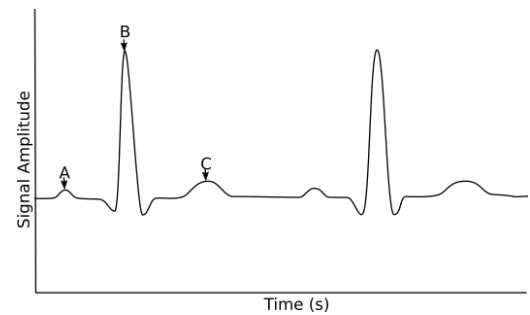
Signal Quality Class Characteristics

The signal quality of the waveforms and the annotations were recorded to give signal events with a ‘Good’ objective signal morphology. The main characteristics used to determine ‘Good’ morphology are shown in figures 7.6a and 7.6b, where they show diagrams of the PPG and ECG signals respectively and illustrate the main features of the signal. These figures are used below for visual reference when describing the characteristics of ‘Good’ PPG and ECG signals

The ideal PPG waveform, as shown in figure 7.6a can be best described by the following characteristics:



(a) An idealised 'Good' quality PPG signal. Region A is where the vessel volume raises. Point B is the Maximum vessel volume. Region C is where the vessel volume recovers. Point D is the dicrotic notch as a secondary peak, common but not always visible in the signal.



(b) An idealised 'Good' quality ECG signal. Point A is commonly called the P wave. Point B is in the middle of a space called the QRS complex which is from the negative deflection through the peak to the second negative deflection. Point C is commonly called the T wave.

Figure 7.6: Idealised signal morphology annotations for both PPG (left) and ECG (right). Explained above are marked points and regions denoting the interesting features and morphology of the signals.

- A fast rise time in region A lasting approximately $1/3$ of the period, with a slow recovery in region C lasting for $2/3$ approximately of the period.
- A smooth trace with an identifiable peak point B with low to no signal clipping.
- An extra peak in position D can be expected but not required as this is caused by the reflection of the pulse on the capillary beds and varies from person to person, commonly it is found a quarter to halfway through region C.
- The peak of the next wave needs to be consistent to give a good estimate for calculated parameters such as heart rate.

The ideal ECG waveform as shown in figure 7.6b can be best described by the following characteristics:

- Points A and C are not always noticeable or are hidden by muscle noise.
- Points A and C should always be lower than the main QRS point B.
- The waveform should be smooth other than the points marked.
- The peak of the next wave needs to be consistent to give a good heart rate estimate.

If any of the above points are not met, the wave might not give an accurate result and, therefore, should be marked as a ‘Bad’ quality signal, otherwise if the points are met the signal segment is labelled as ‘Good’ quality. These are assigned numerical values of -1 and +1 respectively for the two classes.

Annotating Data Interface

The interface for adding this information is a purpose built web system, as shown in figure 7.7 and explained in the caption. This illustrates the current interface. The style is a combination of annotation systems from the work done by Sukor *et al.* (2011) and the LabelMe system¹ for the ImageNet database. The web based system allows remote data annotation and review.

¹Label Me Image annotation system can be found at <<http://labelme.csail.mit.edu>>. Author: MIT, Retrieved 2016/12/01

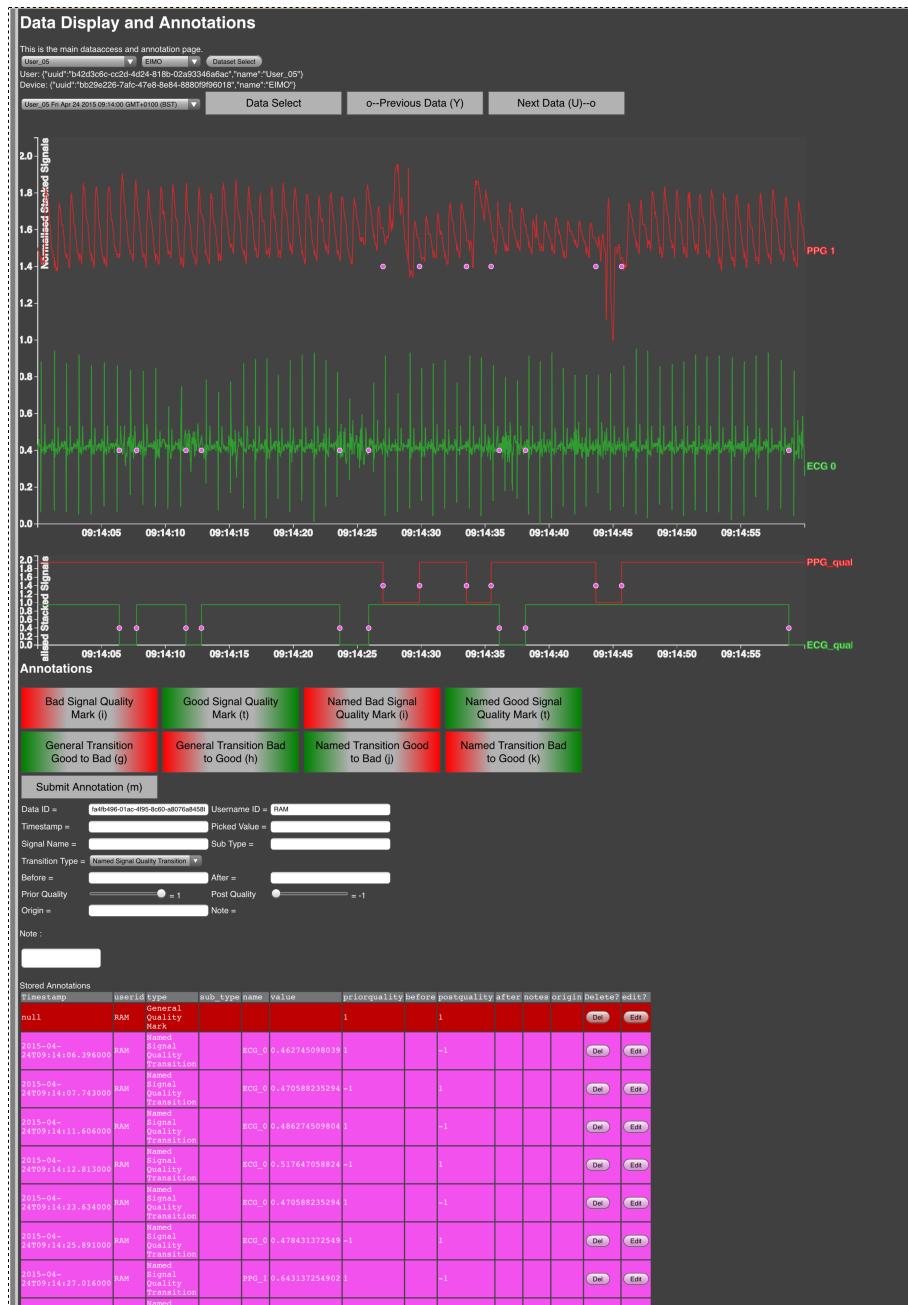


Figure 7.7: The web-based signal annotation user interface system demonstrating the main sections for selecting a user, device and the current data segment with the annotation types colour-coded below. The computer determined signal quality is shown with the same time base as the displayed signal. The drop-down boxes allow the user to select a current user and device to review. A 1 minute data segment can be selected from the next drop-down box from the selected user. Once selected, the main area displays the signal data for the selected user and this can be zoomed and panned using the controls below the signal graph. Below this, the smaller graph shows the currently computed signal quality graph using the annotations. All of the annotations can be marked straight onto the signal by clicking on the appropriate point, then a choice of the annotation types can be made and the point or mark is assigned. All of the other annotation types can be added in this way; so that the peaks of the signals, or diagnostic information, can be added or modified. The interface automatically updates and shows the current state of the signal quality to give feedback to the annotator.

Processing the Annotations

When a segment of data requires processing, the signal and annotation data are loaded, then the quality annotations are assessed. The most general marks are used first, and then more specific marks and transitions are used to build the quality trace, following the annotation hierarchy described above. These are strung together in time order to create a time-series of quality over the same time span as the data signal annotated. This can be seen in figure 7.8, where the circular mark indicates the annotation points, and the line spanning them indicates how the quality signals are calculated between them. Quality marks are used as the signal default, with quality transitions used to modify the signal at known points. An example of this can be found earlier in table 7.2 and figure 7.5 where the diagram shows the result of the table of annotations.

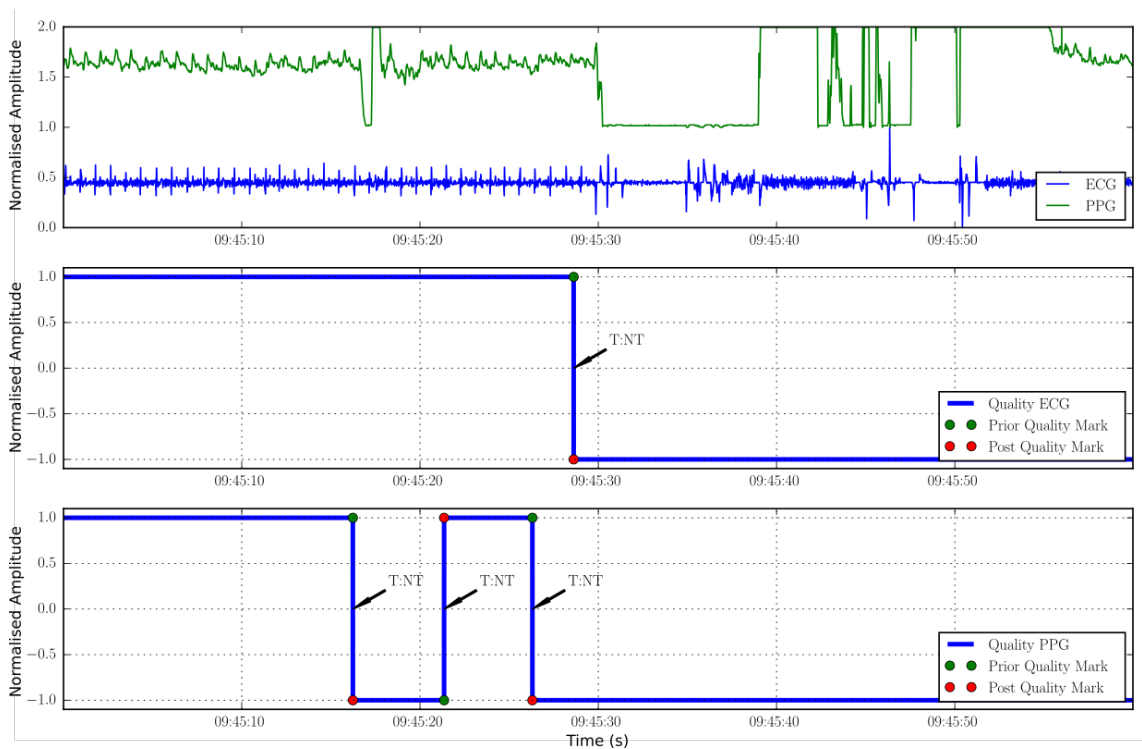


Figure 7.8: Graph showing a sample of signal data with the coloured circles marking where there are quality annotations and the lines indicate the calculated quality signal using these points as data for interpolation. The ECG and PPG signal data is shown on the top axis, then the bottom two graphs take the annotations for this segment, which can effect each signal, and calculate the quality seen at that point. T:NT is referring to a quality named transition (NT) which means this was a quality transition for a particular signal. The T is a short code to differentiate transitions (T) from marks (M).

7.3.3 Processing Elements

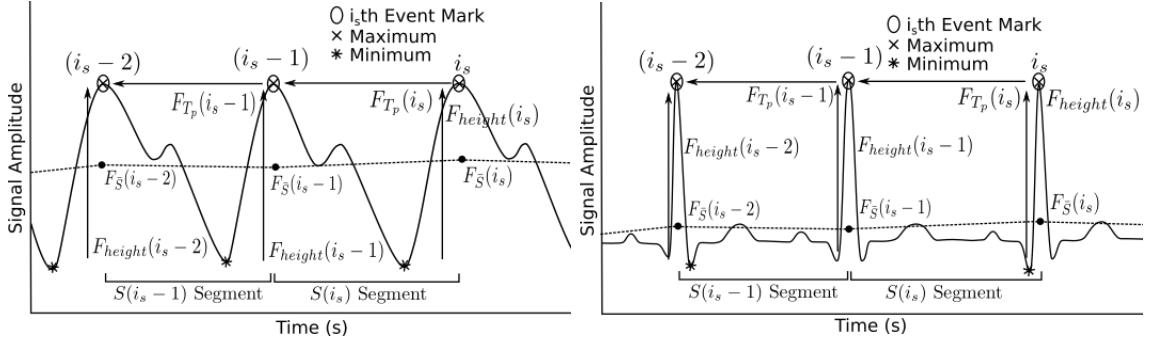
The other processing blocks which are seen in the figure described in figure 7.1 show that the events and annotations are used to extract desired descriptive features and how those features are then given a quality mark using the annotations. The desired classification architecture can then be applied to learning and be assessed on subsets of the quality assigned events provided by the framework. The full description of this can be found in section 7.3.4.

7.3.3.1 Contextual Feature Extraction from Events

This processing element augments a set of signal events $\mathcal{E}(i_s)$, which have been indexed in time, with the feature descriptions based on the contextual data surrounding the event and the segment of data between events. As referred to previously, the simple statistical measures are easily calculated from the data between event $\mathcal{E}(i_s)$ and the previous event $\mathcal{E}(i_s - 1)$, which bound the signal data segment $S(i_s)$ at segment index i_s . Features that are of marked note are:

- $F_{height}(i_s)$ - The segment i_s height from peak to valley between the current $\mathcal{E}(i_s)$ and last $\mathcal{E}(i_s - 1)$ events.
- $F_{T_p}(i_s)$ - The time difference or length of segment i_s between the current $\mathcal{E}(i_s)$ and the last $\mathcal{E}(i_s - 1)$ events.
- $F_{\bar{S}}(i_s)$ - Mean of the signal segment i_s .
- $F_{\sigma_S}(i_s)$ - The standard deviation of the signal segment i_s .

The second group of features used are more sensitive to the morphology of the signals. These are derived from the signal data within the current-to-previous event time frame. However, using these signal features raises problems due to time and amplitude variance across samples. Since the peak height and cycle height can be better captured elsewhere, the data for morphology can be then normalised, to lay between -1 and 1, to allow the routine to be amplitude invariant. The time variance between segments is a slightly harder proposition, but can be treated similarly to the



(a) An example PPG signal with the event features marked. (b) An example ECG signal with the event feature marked.

Figure 7.9: These show example PPG and ECG signals annotated with the signal features. These are used to describe the signal events and are marked to aid an explanation of their meaning on the signal. The $F_{T_p}(i_s)$ is the calculated period between the peak $P_G(i_s)$ and $P_G(i_s - 1)$ of the signal. $F_{height}(i_s)$ is the calculated signal height between the maximum and minimum seen in segment i_s . $F_{\bar{S}}(i_s)$ is the mean of the signal data $S(i_s)$ within segment i_s . Finally $F_{\sigma_S}(i_s)$ is the standard deviation σ over the signal data $S(i_s)$ in segment i_s .

amplitude. The cycle timing can be captured by a single feature, as above, and then the morphology can be normalised in time. In this case, the signal is re-sampled to give N evenly spaced samples or signal taps, which should represent one cycle starting from the top of a peak, spanning to the next peak and the values calculated are then set as features able to be used in the learning methods. This method can then be augmented dependent on what features are required, to provide the first and second differential of the sampled values, with the boundaries wrapped around.

Morphological features include:

- $F_{last}(i_s)$ - Correlation coefficient from current event, using the N tapped signal segment $S(i_s)$ to previous event signal segment $S(i_s - 1)$.
- V_N - N Tapped normalised signal values for segment $S(i_s)$.
- dV_N - N First consecutive sample difference as a toroidal array for segment $S(i_s)$.
- ddV_N - N Second consecutive sample difference as a toroidal array for segment $S(i_s)$.

These features, calculated for the events, can then be added to the event times to create a descriptive event in the signal time. This contains the descriptive features, and will allow a routine to classify on the routines as a set of event descriptions. This set of events can be defined for the features above by:

$$\mathcal{E}(k) = \{F_{height}, F_{period}, F_{\bar{D}}, F_{\sigma_D}, F_{last}\} \cup V_N \cup dV_N \cup ddV_N \quad (7.22)$$

Where, $\mathcal{E}(i_s)$ is the event description of the i_s th segment, each of the features are evaluated for this segment using the past signal data before the current boundary.

7.3.3.2 Assignment of Annotated Quality to the Event Descriptions

To utilise the human annotated signals, the signal quality calculated from the annotation needs to be interpolated to calculate the quality outcomes and the timing of each event. This calculation is done in continuous time, to keep the timing of the detected events relative to the signal time. The process is shown in figure 7.10; this figure shows the segmentation and events from the signal on the top graph, and how those events are allocated. The assigned quality for the detected events is shown on the lower two graphs for the ECG and PPG respectively. The assignments are given to the signal quality transition closest in time to the event using the following:

$$dt(i_s, e) = T(\mathcal{E}(i_s)) - T(Q(e)) \quad (7.23)$$

$$A(i_s) = \min_l dt(i_s, e) \quad (7.24)$$

$$\text{subject to } dt(i_s, l) \leq 0, i_s \in |E|, e \in |Q|$$

Where, $T(\cdot)$ is an operator to recover the timestamp of the event (\cdot), $\mathcal{E}(i_s)$ is the event at time i_s , $Q(e)$ is the quality at time e , dt is the difference in time between the event $\mathcal{E}(i_s)$ and the quality point $Q(e)$. This finds the closest quality transition $Q(e)$ at time e from the past transitions and assigned it to the actual event set $A(i_s)$ at time i_s .

The dynamic approach used to assign the actual quality can allow the prediction horizon to be varied to suit the problem at hand. The following two sections describe how the event description and actual values could be remapped to give descriptive or predictive assignments, with arbitrary offsets.

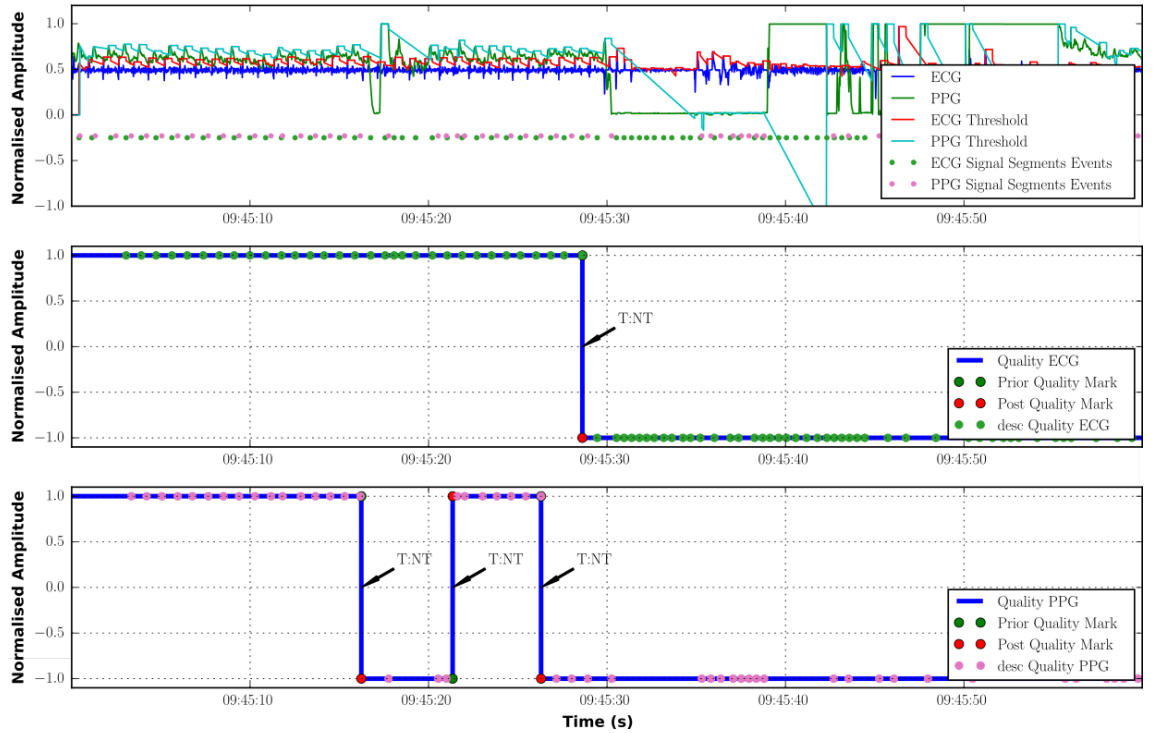


Figure 7.10: Graphs showing an example selection of signal data and its quality value assignments. The data is segmented in the top graph into a set of events for ECG and PPG, then the bottom two graphs take each signal and assign the quality seen at that point. T:NT is referring to a quality named transition (NT) which means that this was a quality transition for a particular signal. The T is a short code to differentiate transitions (T) from marks (M).

Descriptive Assignments

The actual quality or base standard assignments are given to the current events. The actual value to be determined is mapped to the same or past events given by:

$$D(i_s) = \{\mathcal{E}(i_s - h_d), A(i_s)\} \quad (7.25)$$

where, $D(i_s)$ is the i_s th full description with actual quality assigned, $\mathcal{E}(i_s)$ is the i_s th event with a description, $A(i_s)$ is the i_s th actual quality assignment, with h_d being the descriptive horizon offset. Varying h_d allows the classification to be moved from the current event into the past if required.

Predictive Assignments

The assignments are given to the current event, based on the prediction horizon h_p , the actual value to be determined is reassigned along h places. This, in essence,

moves the actual value to be determined into the future of the events series. Actual outputs are matched to the feature events of older samples for it to predict, given by

$$D(i_s) = \{\mathcal{E}(i_s), A(i_s - h_p)\} \quad (7.26)$$

where h_p is the prediction horizon offset.

The final event descriptions now have an interpolated quality value assigned to them. Therefore, this mapped set of events with outcomes are suitable for either supervised or unsupervised learning as required. This allows the list of events to be then sent to classification modules that can test the calculated set of features on the descriptions in the learning architectures available.

7.3.4 Classification and Learning Architectures

The main modules for classification and learning in the system are not directly specified; only the interfaces are specified because the modular nature consumes the finished descriptive event lists and other settings for the internal structures. The final selection for this structure can be chosen when an experiment is run. The implementation of the underlying processes and structures is described in section 3.4.

The nature of the model and performance analysis can be selected dependant on the analysis being performed. This will be described in the next section. The architectures that have been used and documented are detailed in section 7.3.4.1. These run within the analysis framework as discussed in chapter 3. The modular system described here allows different models and processing structures to be created and used within the framework. These can be easily reused or upgraded as requirements and resources change.

7.3.4.1 Implemented Models and Algorithms

There are many learning architectures that can be used within the framework discussed here. Only a few of the architectures will be introduced here, since only a simple model is required for the study below. In the next chapter, machine learning

models will be explained and compared in more depth as the modelling and analysis is the main focus of chapter 8.

There are several classes of models and algorithms that have been implemented and could be used within the framework. These include:

- Two-class, Regressive SVM.
- Single and multi layer perceptron SLP / MLP.
- Dimensional embedding (e.g. T-SNE, Principle component analysis (PCA), Independent component analysis (ICA)).
- Clustering (e.g. DBSCAN (Duan *et al.* , 2006), k-means) (Jain, 2010).
- Autoencoders.

These routines and more are available by utilising libraries like Theano, pyBrain and scikit-learn, with others available in Python like Tensorflow² and Caffe³. More routines can be created and added to the above list using Python or other languages. An excellent resource of machine learning methods and algorithms, which include some of those mentioned above, are presented by Kononenko & Kukar (2007). Other schemes mentioned in this book or elsewhere, could be implemented within this scheme since this system and the ProcessSR framework it is built on, could implement or call through to any other algorithm or system, including parts of itself, to aid in the data processing.

The architecture, as discussed so far, describes a system where any set of definable events in time on a signal, can be mapped with any signal annotation output class. This currently works well for the signal quality classes, but equally would work for other features of signals that need to be measured.

These lists of features can then be used for feature learning and classifier training. To allow for flexibility in the type of architecture used and to ensure, as far as

²A parallel processing framework useful for deep learning network. More information can be found at <<https://www.tensorflow.org>>. Retrieved 2016/12/01

³A deep learning neural network framework, with the ability to speed up deep convolutional neural networks. More information can be found at <<http://caffe.berkeleyvision.org>>. Retrieved 2016/12/01

possible, an unbiased test will be the result. The system is set up so that the models, segmentation or feature descriptions can be replaced in a modular fashion. Further details on the framework that this system runs in can be found in section 3.4.

7.3.5 Testing and Results Strategy

The model accuracy can be measured using two methods as described in section 7.2.2.4. The first, a correlation coefficient, is set up between the actual assigned quality and the estimated quality. The result of the models, therefore, can vary continuously between -1 and 1. A correlation coefficient is used because it is a way of benchmarking the accuracy that the estimated learning model can replicate; it can do this without clear classification categories or with a continuous output score. The second method is to turn the continuous output into a two class discrete system by using equation (7.27) setting a threshold ϵ_c at 0 and anything above is set to 1 and anything below is set to -1. Then usual classification metrics can be used on C_e . These can include ROC fields, accuracy, sensitivity and specificity.

$$C_e = \begin{cases} O_e \geq \epsilon_c & 1 \\ O_e < \epsilon_c & -1 \end{cases} \quad (7.27)$$

Where, C_e is the final classed output set, O_e is the continuous estimated quality set, ϵ_c is the threshold at which to differentiate.

The analysis system has an internal results and documentation system which, as the program is run, the output can be generated and logged. It includes dumps of the dataset, graphs and written text as well as explanations. Once the chosen program is finished, the results report is processed and an HTML file is created to act as a report of the process. This allows a brief digest of all the output produced and added to an automatically created file folder system. The result report generates a web-based page which can be loaded remotely to view the output and documentation produced. This system allows the extension of the documentation throughout the processing of the data. Intermediary results can be captured as they are being processed and the decision as to what to save can be made while the program is in the current section. For example if a result is higher than expected, a program log or current data source

could be saved for that specific event. The saved intermediary and final results can be placed into new nodes and linked to others using data ontologies for that process. This in-built management system allows the analysis system as a whole to be able to run, test and assess machine learning and meta-machine learning activities.

7.4 Case Study - Human Annotations

7.4.1 Overview

The aim of this study was to look into the process and consistency of manually annotating the signals, along with assessing the cost and benefits of those annotations. The determination of signal quality would be most useful, in order to reduce the effects of noise and motion artefacts for the device described in the previous chapter. The plan for achieving this can be found in section 7.4.2, with further details in appendix C.2. The results of the study are explained in section 7.4.3.

7.4.2 Methodology

The datasets used here, were from the study described in section 6.4 and appendix C.1. For the purposes of this study, only the PPG and ECG signal data from the EIMO device will be used.

The study relies on five main phases which are further explained in appendix C.2. The phases can be summarised as follows:

1. To select five users from the data set and then annotate the first session of signal data (approximately 74 minutes) to create a pool of annotated ECG and PPG signals in terms of ‘Good’ and ‘Bad’ signal quality. These are based on the definitions in section 7.3.2.2, this is described further in appendix C.2.0.1.
2. To use these annotations to train a two-class SVM using the features as detailed in section 7.3.3.1 to produce annotations to act as a control group for the study as described in appendix C.2.0.2.

3. To take these annotations and create 50 randomly chosen samples consisting of 25 samples each of human and machine annotations.
4. Pass the samples to volunteer assessors. This allows the Assessors to add their own assessment of the quality as ‘Good’ and ‘Bad’ segments of the signals as described further in appendix C.2.0.3.
5. Collect and measure the Assessors responses, to create agreement scores for each assessor for the given annotations from the human annotator and machine control. The sections are marked as either being:
 - GG - True Positive (presented annotation was ‘Good’ assessor marked it as ‘Good’ - agreement)
 - GB - False Negative (presented annotation was ‘Good’ assessor marked it as ‘Bad’ - disagreement)
 - BG - False Positive (presented annotation was ‘Bad’ assessor marked it as ‘Good’ - disagreement)
 - BB - True Negative (presented annotation was ‘Bad’ assessor marked it as ‘Bad’ - agreement)

The preparation of the results are explained further in appendix C.2.0.4. Figure 7.11 shows an a marked up and measured example of the annotations.

7.4.3 Results

The results are presented as follows. Table 7.3 shows the recording duration, followed by the time it took for the signal to be annotated. The time that was taken for the human annotator to manually classify the signals in the set was approximately the same or longer than the sample originally took to record the signal data. This was estimated by recording the time that the first sample was started and recording the time when the last sample in that section was complete.

The annotated samples from each assessor were returned and measured. There were 50 sample pages to review with 20 seconds of signals on each page so the assessor

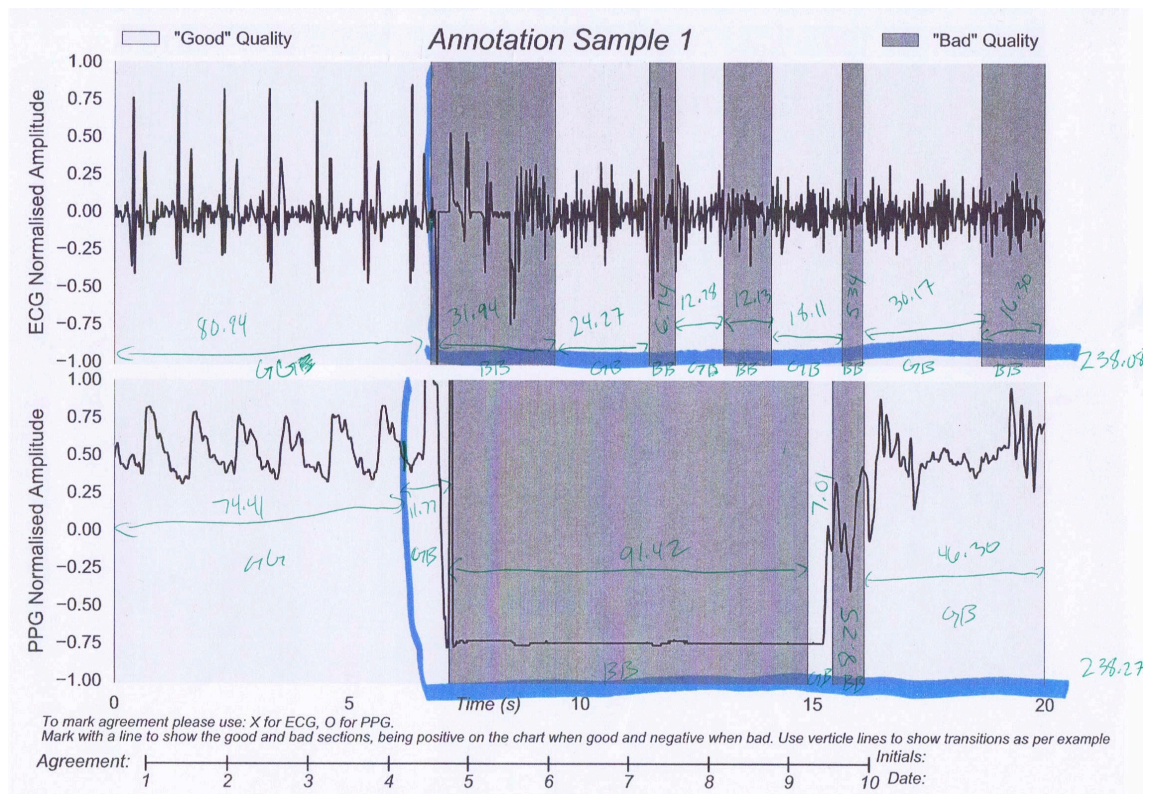


Figure 7.11: This shows an example annotation assessment sample as it was scanned in after being assessed. This is the grayscale version of the graph with the light grey for the 'Good' signal and the darker grey the 'Bad' signal quality. The blue highlighter line follows the signal quality as the opinion of the assessor. A line at the top denoted 'Good' quality areas and a line along the bottom denotes 'Bad' areas. The green pen shows the measurement in millimetres and section classifications as GG, GB, BG, BB as the first letter denotes the presented quality, the second letter the assessed quality.

looked through approximately 17 minutes of printed signal, randomly selected from 50 minutes of annotated signal trace. Upon the return of the samples, each assessor was asked how long it had taken to complete the sample pages. Their answers varied between 40 to 60 minutes. The best and worst case scenarios for the annotation times were:

- Best case - 2.35 minutes per minute of signal seen (40 minutes / 17 minutes of signal seen or 48 seconds per sample page)
- Worst case - 3.52 minutes per minute of signal seen (60 minutes / 17 minutes of signal seen or 72 seconds per sample page)

The assessment time is approximately double the time it took to annotate the original signals, as seen in table 7.3. As the signal quality got more ambiguous,

the length of time taken to accurately assess the signal quality could significantly increase for the human that is annotating or assessing the signals. The assessment was done by considering a printed paper signal trace, then drawing two lines on a piece of paper. The Assessors did not always use the agreement scale or initial them, so it is doubtful that the annotations could be done much quicker.

The basic agreement of the full set of 200 samples is presented in table 7.4. These groups have been expanded in table 7.5, which breaks the agreement down into each of the Assessors. On average it can be seen how often the annotations matched. The sensitivity and specificity measures are used to examine the mix of ‘Good’ and ‘Bad’ classifications, which is discussed next.

Table 7.4 also shows the classification scores, which were given by the Assessors collectively for the human and machine annotated signals. These tables have grouped the performance over all of the assessor’s samples for both ECG and PPG signals, to show the differences between the human and machine annotations more clearly. It should be evident that there is a marked difference between the human and machine assessed signal quality with the human assessed annotations getting 15% higher accuracy. The results get more interesting when the sensitivity and specificity are taken into account. The specificity was higher than the sensitivity for both human and machine annotations suggesting that the assessors were happier to agree with ‘Bad’ than ‘Good’ signal segments, with the machine annotations having a greater margin, showing the assessors were being conservative when classifying. However it should be noted that both the sensitivity (+19%) and specificity (+7%) are higher with human over machine classified annotations, suggesting there was

Table 7.3: The length of the recording sample set as compared to the time to annotate them for each user.

User Name	User 005	User 006	User 007	User 008	User 014
Section Lengths (mins)	12/12/50	11/11/49	12/12/50	12/12/50	12/12/48
Recording length (mins)	(74)	(71)	(74)	(74)	(72)
Annotation length (mins)	75	77	92	87	90

Table 7.4: The human verses machine annotator origin, averaged across the Assessors and signals selected.

Annotation Origin	Agreement	Sensitivity	Specificity
Human	87%	84%	90%
Machine	72%	65%	83%

more agreement with the ‘Good’ and ‘Bad’ quality segments when a human annotated the signal showing that the human annotations were preferred, as expected. The machine annotations did lead the assessors slightly, with an overall 72% agreement, however the assessors defaulted to ‘Bad’ signal quality as shown by the higher specificity.

This also follows in practice, so that to gain a high quality signal set, an algorithm should also be conservative, and since there can be many segments to chose from, those which are only marginal should be classed as ‘Bad’ and should be avoided. The false positive segments would pollute the ‘Good’ dataset with more marginal ‘Bad’ signal segments.

This is visually shown in figure 7.12 as the metrics used allow the assessor agreement to be plotted as part of a ROC graph. In the graph it is shown that there are two clusters with the human annotated samples markedly above the machine annotated as was predicted in the hypothesis. Interestingly, the human annotated cluster is tighter than the machine annotated cluster, suggesting that the limits were being reached as there are only a few ways to get it right, which might show better consistency between the samples. The larger machine cluster shows that this is more variable and if performed again, those points may move considerably whereas the human annotated cluster might not.

If the categories are expanded to show the agreement by assessor and annotation origin as seen in table 7.5, there is an apparent difference in agreement between the Assessors. It is useful to note the experience of the Assessors to find common groups. Assessor 1 was a cardiology consultant, Assessors 2, 3 and 4 were computer scientists with experience in recognising the ECG and PPG signals.

Assessor 1 depicted on the table was the clinician with all three performance

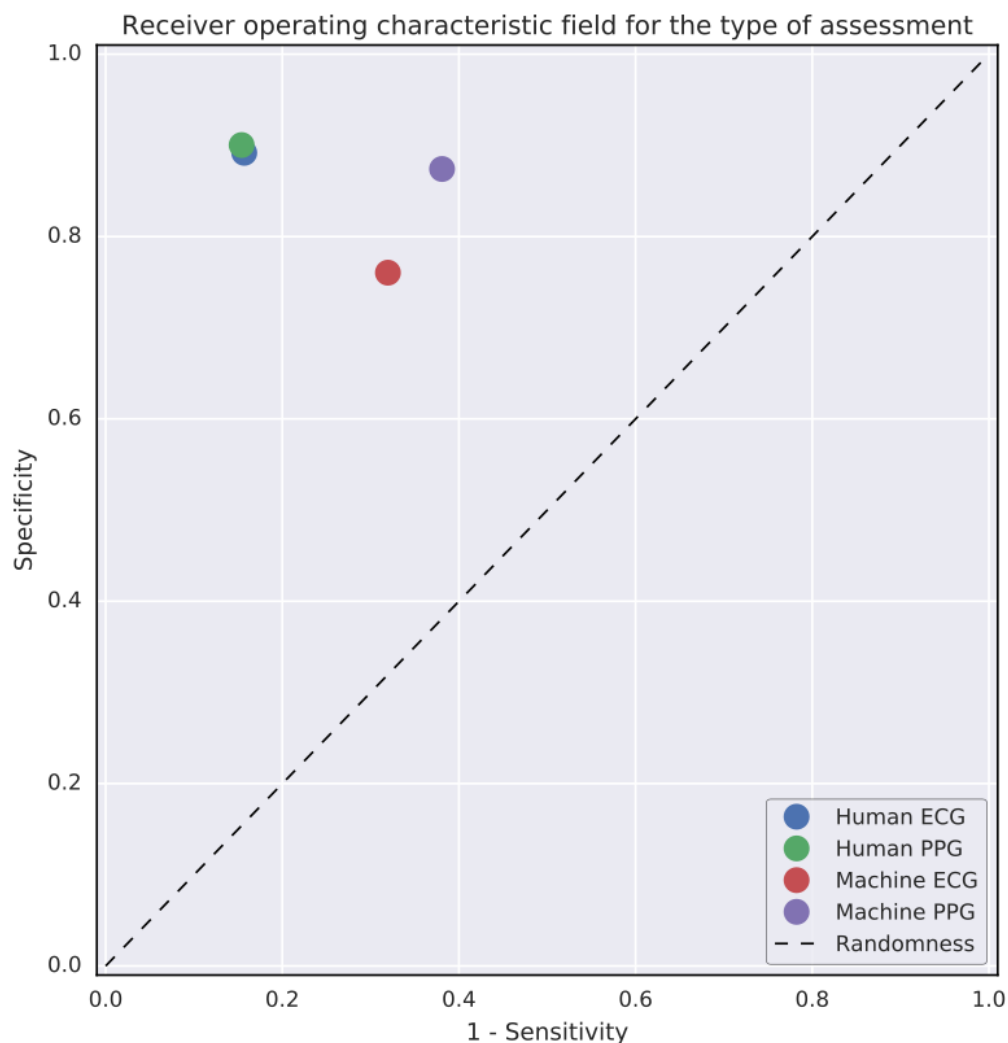


Figure 7.12: This shows a sensitivity and specificity performance receiver operating characteristic (ROC) graph, displaying the performance of the Assessors when the annotations were obtained from a human or machine source. This shows the agreement on the human annotation origin was higher and more consistent with the points being close to each other, than for the machine annotation origin as those points are more spread out and close to the random line.

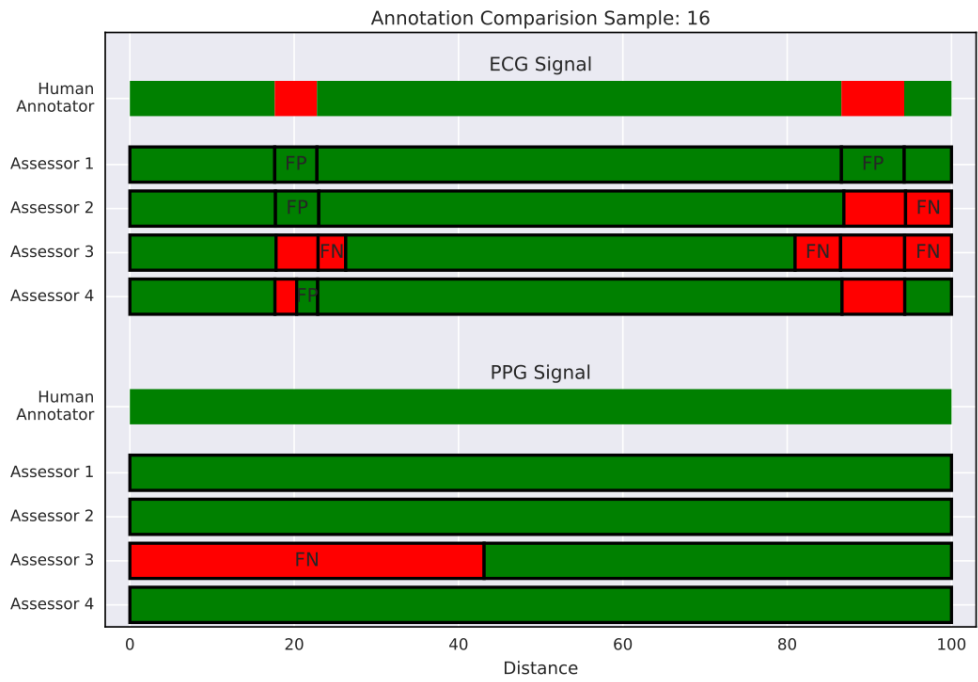
metrics at or above 90% on the human annotations. Assessor 4 followed closely with all of the metrics over 89%. This suggests that Assessor 1, Assessor 4 and the human annotator must share the same ideas of what ‘Good’ and ‘Bad’ signals should look like over 89% of the time. Assessor 1 also had the greatest separation between the machine and human annotations suggesting they were led less by the machine which would make sense based on their experience. Assessors 2 and 3 dropped to 71% but rose up to 96% when looking at the agreement of the ‘Bad’ sections as noted by the specificity, reflecting that they defaulted to ‘Bad’ when unsure. Both assessors showed a reluctance to classify the signal as ‘Good’, with

Table 7.5: The agreement scores grouped separately for each of the Assessors and annotation origins.

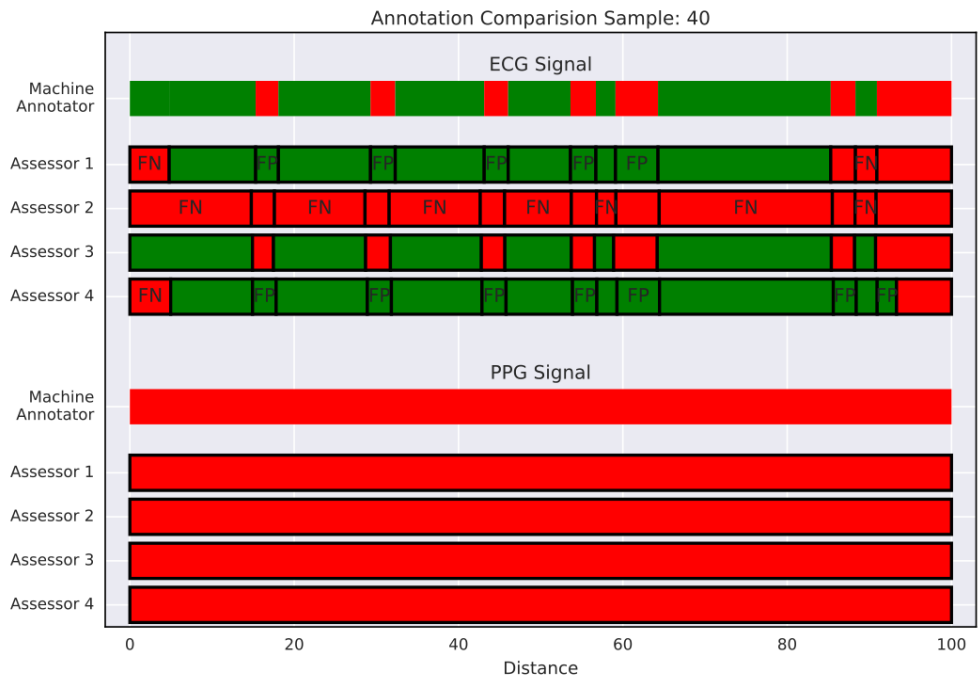
Assessor	Annotation Origin	Agreement	Sensitivity	Specificity
1	Human	92%	95%	90%
	Machine	70%	67%	77%
2	Human	87%	76%	96%
	Machine	69%	58%	87%
3	Human	77%	71%	83%
	Machine	68%	56%	87%
4	Human	93%	97%	89%
	Machine	81%	81%	80%

Assessor 4 departing from this, showing less separation between the human and machine produced annotations and a less conservative approach.

The annotations on the sample can be compared segment by segment between the Assessors. Figure 7.13 shows two diagrams which are examples that show interesting patterns in annotating behaviour, both from a human annotated sample in figure 7.13a, and a machine annotated sample in figure 7.13b. The first shows a sample where the ECG has been annotated as being ‘Good’ apart from two ‘Bad’ segments, most of the Assessors agree, but there is some variation over the exact start and end of the ‘Bad’ segments. The PPG is interesting as almost every one has labelled it ‘Good’ apart from one assessor, who has labelled the start of that segment ‘Bad’. This example shows that there is variation between people evaluating the same patch of data. The second shows a sample where the machine classified the signal as mostly ‘Good’ with 7 ‘Bad’ segments. Assessors 1 and 4 both classed it as mainly ‘Good’ with no bad segments in the middle. However, Assessors 2 classed the whole sample as ‘Bad’ showing a conservative approach and Assessor 3 followed the classification of the machine, showing a tendency to follow the annotations given.



(a) Example of great variation between the Assessors on human annotated sample 16. The figure shows consistency but there is some disagreement between the position of the transitions on the ECG signal. One assessor on the PPG is being more conservative than the others, showing the choice of assessor could make a difference to results.



(b) Example of the Assessors being influence by the annotations on machine annotated sample 40. Some Assessors followed the given annotations with some going to each extreme of mostly 'Good' or mostly 'Bad'. This shows that Assessors can have radical disagreement in annotations for the ECG, but total agreement for the PPG signal.

Figure 7.13: These show the annotation given by the four Assessors and the original annotator, colour coded for the 'Good' class in green and the 'Bad' class in red. The outlines in black denote the changes of classification of the segment e.g. from TP to FP. The FP and FN are marked to show segment disagreements.

7.4.4 Discussion

It can be seen above that when looking at signal annotations, it takes just as long to annotate the signal manually as it does to capture the signal in the first place. This length of time increases as the signal gets noisier and harder to discern. This is a reflection based on the number of samples and transitions between acceptable and unacceptable signal quality. The time span stated is only an estimate of the annotation time taken for both the original human annotator and the Assessors. However the time could rise practically because the annotator was very familiar with the annotation system and the Assessors were presented with a simplified printed segment, and required to draw two lines on the printed graph. It would be very prohibitive for clinicians to spend that amount of time to annotate and interpret the signals recorded from the patients if they were not already defined as ‘Good’ and ‘Bad’. The results of the assessed samples suggest that the Assessors do not always agree with the same signal quality outcome, marking parts of the same signal differently. The other outcome is that the Assessors as a whole agreed with 87% of the samples produced by the human annotator, and if the clinical assessor was taken alone, the agreement would be 92%. This indicates that the standard of annotation, although not perfect, could be sufficient for further modelling and quality assessment. The next chapter considers how the signal classification could be accurately modelled.

7.5 Summary

This chapter has described the primary issues and opportunities for signal quality assessment for medical devices. It has then gone on to examine the literature to gain a better context and background on the subject. The proposed signal analysis system was defined, along with the current testing strategy for the study. The results have been presented and discussed on the final agreement of the Assessors and the signal quality annotations, and the time it takes to manually annotate signals. The results seem to suggest that the annotations given for the data, show

good agreement with all of the Assessors and especially with the clinician alone of 87% to 92% respectively. The time it takes to annotate and assess these signals is very prohibitive, which increases when the decisions are harder to make, when the Assessors seem to get more conservative, defaulting to 'Bad' signal quality. The classification scheme described above will be expanded upon in chapter 8, using the human annotated signals assessed here to find automated ways to accurately determine the signal quality quickly and efficiently.

8 Automating Signal Quality Analysis

8.1 Overview

The last chapter defined a classification framework and investigated human annotated signal quality. It was found that not only do the human annotations take a lot of time, but there is variation in the resultant signal quality markings given from the assessors. The purpose of this chapter is to investigate accurate ways of automatically classifying the quality of the signal, through exploring and comparing the different features derived from those used in the literature for signal quality estimation. The system for the determination of modelling tests consisting of feature processing and selection along with model definitions, are explained in section 8.3. The features are expanded using the methodology discussed in section 8.4 to create more independent and objective features. These features are then tested and evaluated in the case study presented in section 8.5 using the test procedures presented in section 7.3.1.

8.2 Background

The structure of machine learning was described in the previous chapter. However, the details on modelling algorithms, performance metrics and features were only briefly discussed. The algorithms and metrics are expanded upon and evaluated since in this chapter, the accuracy and performance of the automated classification is paramount. First, the topic of machine learning is expanded upon in section 8.2.1 by discussing and comparing the different models. Then the different performance metrics, introduced in the last chapter, are extended in section 8.2.1.2. A review of

the different features which have been used and discovered in the literature so far for identifying the quality of these signals is presented in section 8.2.2. Finally, a discussion of the difference between subjective and objective features is discussed to lay the foundations for the case study described in section 8.2.3.

8.2.1 Further Machine Learning

The last chapter introduced the basic strategy used to classify the signals for quality. Although the system is capable of much more, only a simple classifier was used to gain the control group for the assessment placebo. The decision for the style of learning algorithm and the types of features used will make much more of a difference now that accuracy and robustness feature prominently. The various learning methods will be compared using their formulation in the literature. Useful models, features and performance metrics are then examined and used to find and select the features that define the models that allow for the best classification accuracy.

8.2.1.1 Machine Learning Methods

There are many different machine learning algorithms currently in existence. The formulation of most involve setting up a decision boundary. The description of the instance can be given and a classification is able to be drawn from this. The basic version of this is the linear discrimination function and is given by:

$$y = W\hat{x} + c \tag{8.1}$$

where y is the result, x is the features for the current example, W is the direction of the decision boundary or regression line, and c is the offset threshold.

Equation (8.1) is a simple learning function and can be solved or trained analytically by using a training set for y and x . Since this is a linear relationship, this defines a straight hyperplane. Other models, discussed below, improve on this by allowing the decision surface to curve and deform around more complex topological landscapes and by using a non-linear function to map the values.

These functions form the basic principle of many learning functions such as artificial neural networks (ANN), both single and multi-layer perceptions (SLP, MLP)

or support vector machines (SVM). The features are combined with a vector to define a function line and given an offset. The function separates the features into the positive and negative sets.

The main difference in systems, e.g. the ANN and SVM algorithms, is the way that they are trained. ANNs use a stochastic gradient descent method by back propagating the error through the network. The gradient descent searches to minimise the weighting parameters between mapping functions in the junctions. However, some SVMs use the Lagrangian method to create a single differentiable error surface. Quadratic programming (QP) can then be used for their optimisation and so can be solved more analytically.

Support Vector Machine (SVM) Variations

Support vector machines (SVM), are invaluable for novelty or outlier detection due to their capability for calculating decision boundaries using flexible kernel functions which transform the surface into a high dimensional linear surface. Boser *et al.* (1992) extends on their previous work by formulating an SVM with a soft margin, allowing a ‘Best Guess’ boundary to be determined when the classes might not be separable. The derivation presented provides a domain based cost function that is very similar in structure to a radial basis function classifier, or a partial neural network. However, due to the kernel function and formulation, it produces a global convex cost function and optimisation employing sequential minimal optimisation in order to find the parameters. There are many constructions of SVMs to complement different data sets and situations. Some structures are examined in the following examples; these can be theoretical or real-life datasets and produce useful outcomes on most occasions so that they improve event detection rates and outlier detection rates as discussed by Gómez-Verdejo *et al.* (2011), and Shahid *et al.* (2012).

There are different formulations of the SVM algorithm which can be used depending on what the final outcome is required to be. The formulations and examples of their use are presented below.

- Class based (SVMc).

The basic formulation is the ‘class-based’ support vector machine (SVMc) which uses the basic discriminatory function to find the separation between classes. This formulation has been described by Huang *et al.* (2011).

- Regressive (SVMr).

Regression support vector machine (SVMr) algorithms formulate the SVM margin as a tube through the instance data points which is well explained by Smola & Schölkopf (2004). A probabilistic formulation of the SVM was produced by Gao *et al.* (2002).

- One-Class (SVMoc).

One-Class support vector machines (SVMoc) are a different formulation which can be trained using only one class of instances by contrast with the other two methods mentioned above. A hypersphere is arranged over the training instances’s centre of mass. The goal of the algorithm is to select the radius of the sphere so that all of the instances given can be found within the hypersphere. This can also be used with a soft margin so that a proportion can be left outside. This can be very useful for novelty detection as only the normal class is needed. This has been described by Gómez-Verdejo *et al.* (2011) who also allows the system to be adaptive, based on the new data coming in.

8.2.1.2 Performance Metrics

The metrics described in section 7.2.2.4 can be used to assess the performance of the models; however, in order to rate the models accurately, both sensitivity and specificity measures are required. Assessing models in this two-dimensional field can be difficult, therefore to gain a better model ranking, a measure using a signal dimension would be useful. Ordinarily this could be the accuracy of the model, but this does not take into account class imbalances. There are two other metrics which indirectly depend on the sensitivity and the specificity which are insensitive to class imbalances. These are shown graphically in figure 8.1 and are described next.

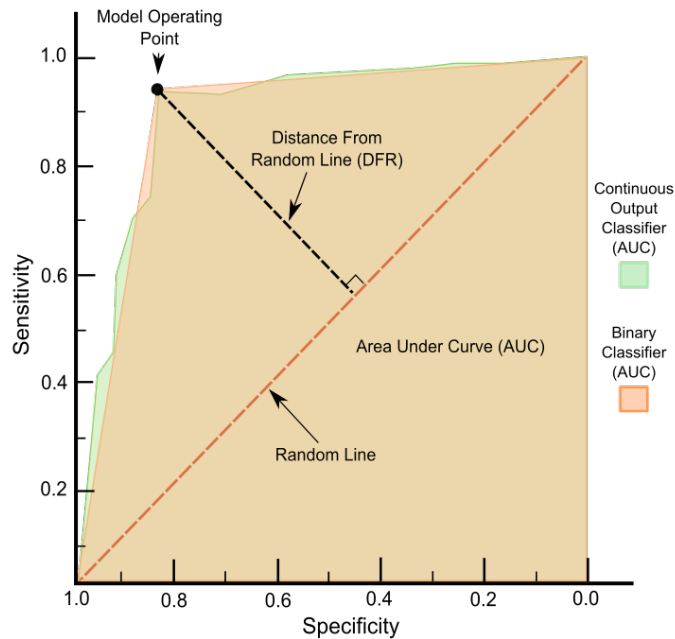


Figure 8.1: Shows the representation of the receiver operating characteristic (ROC) field. The graph shows the performance of the models and estimators with various operating points. The vertical axis shows increasing sensitivity which is the accuracy of the model at correctly classifying 'Good' instances. The horizontal axis in contrast shows the accuracy at correctly classifying 'Bad' instances. The dashed line running corner to corner denotes the performance of a random estimator. The point marked shows the best operating point of a model with the perpendicular distance from the random line (DFR) marked. The line through this point shows the performance at other operating points with the sum of the area under the curve (AUC) being used as another metric, as it shows how well the models can be separated by the classifier.

The first is the area under the curve (AUC), which is the total area swept out by the ROC graph; a greater area indicates a greater separation of the classes in the model. This metric looks at the class separability and not the best performing operating point (Bradley, 1997).

The second metric used is the signed distance from the random line (DFR), which is defined as the perpendicular euclidean distance between the optimum operating point for a model and the ROC random estimation line. The location and usefulness of the ROC operating points are discussed by Fawcett (2006). Here they mention that, the further to the top left an operating point is, the greater its performance is on both 'Good' and 'Bad' classes. This means that if the distance between the operating point and the random line was measured, it would reach a maximum when the point was in the top left, zero when the point falls on the random line and negative when under the random line. This metric looks for the best performance

at the operating point, but relaxes the class separability constraint, which the AUC keeps by measuring the whole curve.

8.2.2 Feature Descriptions

The features found and used in the literature are summarised in table 8.1. The features can be split up into a taxonomy of four feature areas. Each should address a different characteristics about the signal, which can then be used to define groups of features to analytically explore the feature description space.

- Signal.

Dimensions of any apparent, useful signal. The signal statistics are used, along with the amplitude, which are used in work by Aboy *et al.* (2005), Sun *et al.* (2012) and Elgendi (2016). This is used only to estimate the amount of signal present, not to examine the shape of the segments.

- Noise.

Estimates the level of noise in the signal. The statistics and measures applied to the noise or non-useful signal look at the variation between amplitudes and time that the signal crosses the zero axis as used by Sukor *et al.* (2011) and Elgendi (2016) respectively.

- Morphology.

Estimates the shape and character of the signal as used by Li & Clifford (2012), Orphanidou *et al.* (2015) and Karlen *et al.* (2012).

- Entropy.

Estimates based on the differences between the probability distributions of the signal segments, as used by Elgendi (2016).

These feature types are explored further in section 8.4. A consistent set of features are defined and compared with the signal's quality.

Table 8.1: A summary of the features found in the literature showing the characteristic groups for the features and the sources that have used them.

Feature Class	Feature	References
Signal	Peak height	Aboy <i>et al.</i> (2005), Sun <i>et al.</i> (2012)
	Cycle height	Mahri <i>et al.</i> (2012), Sukor <i>et al.</i> (2011)
	Cycle Period	Mahri <i>et al.</i> (2012), Aboy <i>et al.</i> (2005), Orphanidou <i>et al.</i> (2015), Sukor <i>et al.</i> (2011)
Noise	Difference in valley bottoms	Sukor <i>et al.</i> (2011)
	Zero Crossing Count	Silva <i>et al.</i> (2012)
Morphology	Auto correlation	Fu <i>et al.</i> (2010)
	Template correlation	Li & Clifford (2012), Orphanidou <i>et al.</i> (2015), Karlen <i>et al.</i> (2012)
	Dicrotic notch	Sun <i>et al.</i> (2012)
	Fast Fourier Transform (FFT)	Karlen <i>et al.</i> (2012)
	Wavelet Transform/Filtering	Silva <i>et al.</i> (2012)
Entropy	Multiscale Entropy	Zhang <i>et al.</i> (2015), Elgendi (2016)

8.2.3 Subjective and Objective Signal Analysis

In order to understand how to improve the assessed signal quality, it is helpful to define what subjective and objective analysis of information is, and the differences between them, through their presentation in the literature.

Verburgt (2015), describes the history of objective and subjective probability throughout the mid 19th century and also included the modern understanding of the terms. They describe ‘subjective’ as ‘an epistemic state of knowledge’ and ‘objective’ as ‘an ontological state of the world’. Two main points can be taken from this. The first is a definition of ‘subjective’, because it is in the knowledge of people and given by people’s opinion. By contrast, their definition of ‘objective’ is described by knowledge of elements or characteristics of the world. These can be understood to be descriptions that objective measures are the study of characteristics that do not

vary based on knowledge and opinion. The second is that in mathematics, the more objective and quantifiable something is, the more useful and consistent it becomes because it can be relied on and reasoned over.

Thung & Raveendran (2009) describes a collection of various measurements of image quality with a section on subjective and then on objective features. The subjective section describes these measures to be opinion based and found through opinion ratings. However, the other description, addressing objective signal quality, is much more clear in terms of defining the measure using text and mathematical equations to precisely define the metrics.

Subjective Analysis

For the purpose of this work subjective analysis can be defined as the features or elements that rely on the knowledge and experience of people and is gathered through interviews or questionnaires (Verburgt, 2015; Chow *et al.* , 2016; Merat *et al.* , 2011).

‘Subjective assessment is the ratings given by human subjects based on their judgment...’ (Chow *et al.* , 2016).

They consist mostly of:

- Qualifications.

Qualifying an item based on others such as: ‘Item 1 must be good because it looks like Item 2’.

- Opinion.

A statement based on a person’s experience or feelings such as: ‘A belief that Item 1 is a good one’.

- Ratings.

A statement built on a ranking system of options. One can rank a signal based on opinion such as: ‘Item 1 rated a 4 out of 10.’

Objective Analysis

For the purpose of this work objective analysis can be defined as those features that can be observed independently and measured as inherent characteristics (Verburgt, 2015; Chow *et al.* , 2016; Merat *et al.* , 2011).

‘Objective assessment is an alternative method defined mathematically’
(Chow *et al.* , 2016).

They are usually given as:

- Derived formulae.

Derivation of the definition by using other defined principles such as: ‘Deriving a noise measure of the signal using the principle of entropy as the basis of the definition’.

- Functional definitions.

A definition like a derived formulae, which is based on a functional relationship such as: ‘Using the difference between the maximum and minimum to define the size of a signal’.

- Heuristic formulae.

A person’s opinion and experience defined in a formal notation of algorithm such as: ‘Item 1 is useful when the signal is 2 standard deviations above the mean’.

To define objective characteristics of a time based signal, one needs to look for objective characteristics, then look for possible definitions that could be used as working definitions of that characteristic. For time based signals, some characteristics could include:

- Noise Level.

The AC or DC component of a signal trace are not associated with a known signal pattern.

- Signal Level.

The pulsatile and potentially constant components that represents a possible signal pattern for the signal in question.

- Signal Morphology.

The average shape of the signal and how it changes as the signal is gathered.

There are two points to note from the above; the first is that subjective measures tend to have more qualifiable features. The more objective a feature is, the more quantifiable it becomes. However the trend extends further than just assigning numbers to qualities and enumerating or defining a number scale. It is done by creating definitions and quantifiable relationships. Creating objectively defined features then allows one to reason over them, continuously ordering them and automatically mapping traits to values. Therefore the best objective features should be based on quantifiable features with continuous mapping. The objective features will be developed below as the signals are examined for these characteristics in section 8.4.

8.3 Feature Extraction and Modelling Systems

The methodology to be used for building accurate classifiers can be broken down into three smaller goals: feature extraction, test planning and modelling assessment. The system for doing this was designed to be as generalisable as possible to allow for further modification within the management framework. Each system will be discussed next.

8.3.1 Extracting Signal Features

The first task is to take raw data and find useful features that can best describe the patterns and trends in the data. The system for doing this is expanded upon in the feature extraction system from section 7.3.3.1. This is extended to allow mapping functions to be applied to the raw data in a segment, taking into account the data's context. The aim is to create a system that can apply feature mapping

routines automatically to the raw data and assess the features produced to see which would be best to consider further. The data mining system for doing this requires a data content definition. The feature sequences created requires a naming scheme to keep the features deterministically named. The scheme used for the feature mapping process is shown in section 8.2.2. The naming scheme is based on keeping the feature sharing qualities together with an index for uniquely identifying them in the form:

$$f_{i_{fc}, i_{fi}} := \{S(i_s), S(i_s - 1), \mathcal{E}(i_s - 1)\} \mapsto F_{i_{fc}, i_{fi}}(i_s) \quad (8.2)$$

$$\mathcal{E}(i_s) = \{F_{i_{fc}, i_{fi}}(i_s) \mid \text{for } i_{fc} \in F_{class}, i_{fi} \in F_{class}(i_{fc})\} \quad (8.3)$$

Where $f_{i_{fc}, i_{fi}}$ is defined as a mapping function to translate the signal for the current segment $S(i_s)$, previous segment $S(i_s - 1)$ and previous event feature $E(i_s - 1)$. $F_{i_{fc}, i_{fi}}(i_s)$ is the feature measurement for a particular feature class i_{fc} and index i_{fi} , $\mathcal{E}(i_s)$ is the set of all mapped features grouped together for the event. $F_{class}(i_{fc})$ is the features in class type i_{fc}

This feature set holds all of the features available, to be used for signal mapping. The features are subject to selection, as not all of the features that can be generated can be reasonably tested. The feature selection $\phi_{i_{fs}}(i_s)$ can be defined as

$$\phi_{i_{fs}}(i_s) = \{x \mid x \in E(i_s) \wedge P(*_x)\} \quad (8.4)$$

where $\phi_{i_{fs}}(i_s)$ is the feature set indexed for $i_{fs} \in \mathbb{Z}$ which can be unique for all of the event indexes i_s , $P(*_x)$ is a selection function for a particular feature. This can be probabilistic or deterministic. This creates combination sets of features which can be use for definition of the tests when combined with the learners. The selection function $P(*_x)$ can also be selected based on previous feature performance to set up genetic algorithms.

8.3.2 Modelling Test Determination

The second task is to form a strategy to build and test machine learning models. A formal approach for defining the test and models can then be used to automate the testing and analysis process, allowing for a thorough investigation of the features and models currently under test. Importantly for the definitions, these can be used

for the automatic generation and analysis of new models and features for this and other outcomes as the framework is expanded.

A test consists of two main parts of a feature set $\phi_{i_{fs}}(i_s)$ from the previous section and a model definition $M_{i_{mc}, i_{mi}}$ to test it on, where i_{mc} is the index for the class of the model SVM or ANN for example and i_{mi} is an integer index to allow the model to have a unique name for further reference. The model class is chosen and parameters are selected for it. Example parameters useful for an SVM model are shown below to illustrate the point:

- Kernel type :- {'Radial basis function (RBF)', 'linear'}.
- Gamma := { 0.5, 0.2, 0.1, 0.01}.
- Cost function weighting :- {0.5, 1.0}.
- Soft margin weighting :- {0.1, 0.2}.

If these parameters were used as a grid function, they could define 20 models $i_{mi} \in (1, 20)$ of each class $i_{mc} \in a, b, c, \dots$ where these parameters are applicable.

$$\Psi_{i_{ti}} = M_{i_{mc}, i_{mi}}(\phi_{i_{fs}}(i_s)) \quad (8.5)$$

To understand the performance across a range of learning machines, each test $\Psi_{i_{ti}}$ can be computed and plotted on a unified field to allow comparison between the classification models. Thus we are able to see the shape of the solution space using the tests as landmarks.

8.4 Exploring Signal Features

8.4.1 Overview

The purpose of this study is to use the feature extraction system above to build up a set of features which could be used for the assessment of the signal quality of the ECG and PPG signals. The features defined and assessed here will then be used to create and assess models in the next study.

8.4.2 Defining Classes of Features

The objective is to create a mapping function to find the main characteristics of a raw signal starting with metrics already in the literature. Four main feature classes will be considered, including: signal, noise, morphology, and entropy as discussed in section 8.2.2. The next section will take each of these components, describe them in more detail and construct possible definitions for them. In order to be the most useful, each component should concentrate on just one measure. Keeping the measures separate allows them to be orthogonally mixed up in a given raw signal and to define a feature landscape for exploration in the next study.

8.4.2.1 Signal strength

To find a useful measure for signal strength, consideration needs to be given as to whether there is a useful signal present, but not necessarily that the signal possesses the correct shape or morphology. The first task in signal estimation is to try to separate the signal from the noise. If the noise is modelled by sample outliers, a median filter can be used. This type of filter is very robust to outliers in a signal and can be performed with a small window w to leave the gross segment morphology, estimating the shape of the underlying signal with all of the noise and outliers removed. This is applied to the signal segment wrapping toroidally at the boundaries as seen in equation (8.6).

$$\tilde{S}(i_s, i_n, w) = \text{Median}(S(i_s, i_n), w) \quad (8.6)$$

Where $S(i_s, i_n)$ is the signal segment, $\tilde{S}(i_s, i_n, w)$ is the median filtered signal segment with a window of w , and $i_n \in 1..|S(i_s)|$ is the sample index in segment i_s .

If the median signal given above is assumed as an approximation to the signal, then one can find the noise level in the next section, by examining the characteristics of the residual $R(i_s, i)$ defined for the segment i_s , which is found by

$$R(i_s, i_n) = S(i_s, i_n) - \tilde{S}(i_s, i_n, w) \quad (8.7)$$

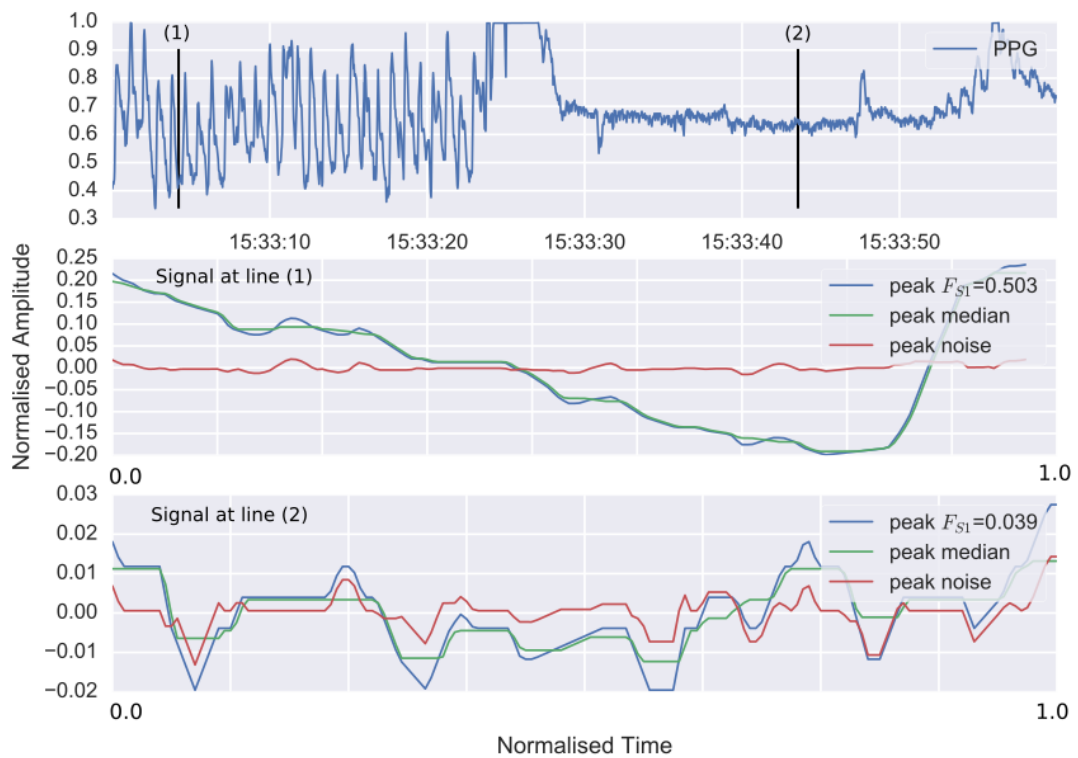
where, $R(i_s, i_n)$ is the residual signal, $S(i_s, i_n)$ is the signal segment, $\tilde{S}(i_s, i_n, w)$ is the median filtered signal segment with a window of w as defined above. i_s is the segment index, and i_n is the sample index from the segment.

Examples of the median filter signals $\tilde{S}(i_s, i_n)$ and the residual signal $R(i)$ are shown in figures 8.2a and 8.2b. These show the PPG and ECG signals respectively, along with the signal segment for two points. These graphs show how the median filter performs, splitting the segment into the signal and the residue as defined above. This acts as a simple low pass filter preserving the gross features of the signal. This seems effective as both the ECG and PPG signals have low-period, strong features. The PPG is a low frequency asymmetric oscillation with a prominent fast rise period and the ECG has a strong periodic QRS complex. The median filter's window can be set to preserve the main morphological features for the PPG and ECG segments. The segment normalisation re-samples the signal into 100 evenly spaced samples over the segments length normalising the time as in the previous chapter, A window of 12 samples was used for the PPG signal. The ECG signal used a window of 5 samples within the 100 sample segment.

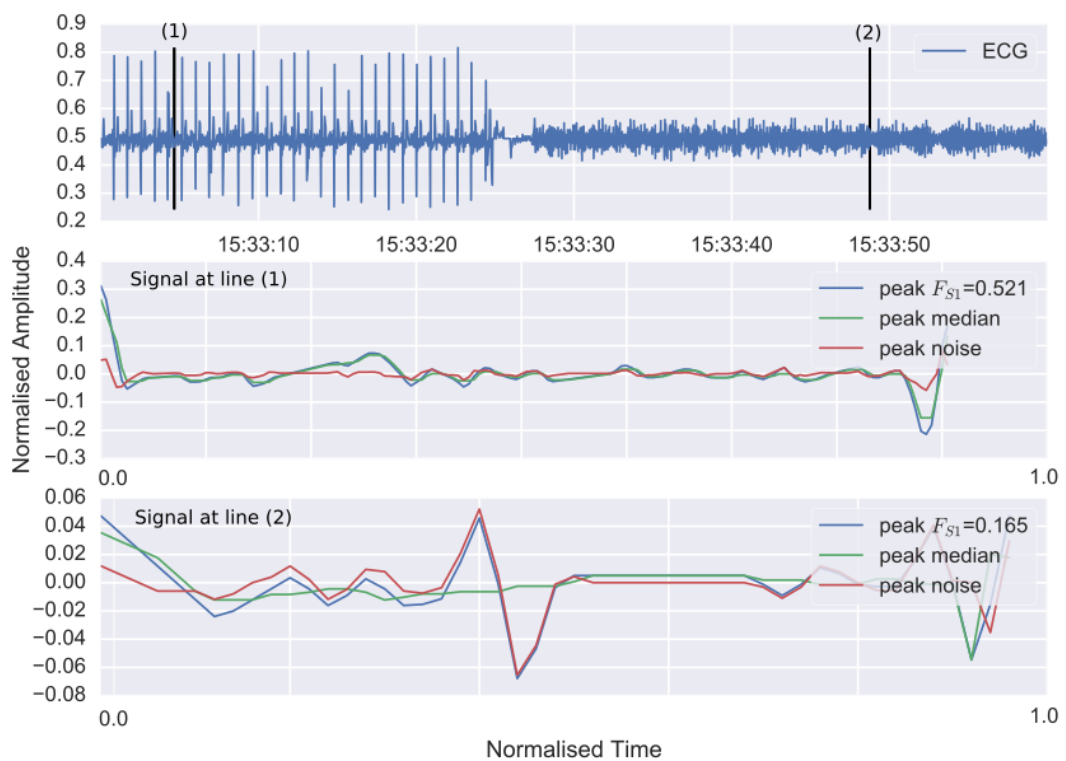
Given the median signal as described above, features can now be defined to transform the signal strength into a value. The features F_{S1} to F_{S4} are then normalised to the maximum of the full unfiltered signal $S(i_s, i)$ so that they represent a scale from 0-1. Feature F_{S5} is not normalised as this then measures the overall size of the signal, allowing the other features to decouple the relative proportion of the signal from the absolute size of the signal. F_{T_p} time span of the segment, to catch atypically short or long segment signals.

Signal characteristic measures:

- $F_{S1} :=$ Difference of the maximum to minimum amplitude of the signal (see equation (8.8)).
- $F_{S2} :=$ The signal peak normalised standard deviation of the signals amplitude (see equation (8.9)).
- $F_{S3} :=$ The signal peak normalised skew of the signals amplitude (see equation (8.10)).



(a) Graph showing example with main median and residual for the PPG signal.



(b) Graph showing example signal with main median and residual for the ECG signal.

Figure 8.2: Graphs showing an example signal with main median and residual for ECG and PPG sample segments. The top axis shows a 60s signal trace, the black vertical lines shows the locations along the trace of the lower two axes. The middle axis is the segment at the first line position (1), in a section of high apparent signal. The bottom axis, is the segment from the second black line position (2) on the right. This shows a segment in an area of low apparent signal. F_{S1} is the signal amplitude feature for the segment shown, which has been marked on each of the traces.

- $F_{S4} :=$ The signal peak normalised kurtosis for the signal (see equation (8.11)).
- $F_{S5} :=$ The maximum amplitude for the signal (see equation (8.12)).
- $F_{Tp} :=$ The time difference between the start and end of the segment (see equation (8.13)).

$$F_{S1}(i_s) = \frac{(\max(\tilde{S}(i_s, i_n)) - \min(\tilde{S}(i_s, i_n)))}{\max(S(i_s, i_n))} \quad (8.8)$$

$$F_{S2}(i_s) = \frac{\text{std}(\tilde{S}(i_s, i_n))}{\max(S(i_s, i_n))} \quad (8.9)$$

$$F_{S3}(i_s) = \frac{\text{skew}(\tilde{S}(i_s, i_n))}{\max(S(i_s, i_n))} \quad (8.10)$$

$$F_{S4}(i_s) = \frac{\text{kurtosis}(\tilde{S}(i_s, i_n))}{\max(S(i_s, i_n))} \quad (8.11)$$

$$F_{S5}(i_s) = \max(S(i_s, i_n)) \quad (8.12)$$

$$F_{Tp}(i_s) = T(S(i_s), i_s) - T(S(i_s), i_s - 1) \quad (8.13)$$

Where in the above equations, $\min(*_X)$ and $\max(*_X)$ are the minimum and maximum operation respectively, $\text{std}(*_X)$ is the standard deviation of the set $*_X$ and skew $\text{skew}(*_X)$ and kurtosis $\text{kurtosis}(*_X)$ are the 3rd and 4th order moment of the set $*_X$ and $T(*_X, *_y)$ is an operator to find the event time of the event set $*_X$ at segment $*_y$.

8.4.2.2 Noise Level

To explore possible objective measures of the noise in these signals, a number of characteristics can be taken into account. There are two main sources of noise in PPG and ECG signals, these are:

- Pathway noise - the noise associated with the source and the signal path. This can be further broken down into:
 - Source noise - originates from muscle tone in the vessels, and involuntary movement artefacts from the person.
 - Electrical noise which is electrical interference that has artefacts within the electronic signal path.

- Motion noise - the noise associated with movement sustained by differential velocity between the device and the user through deliberate movement.

In order to discover which measures to use to define these sources of noise, first the scope of what can be ascertained at the processing step needs to be examined. When looking at the data from just one signal, like the ECG for example, the signal noise has been mixed together through the signal input path. The motion artefacts have also been mixed into the resultant signal. The first task is to estimate the signal within a segment, then an estimate can be made about the noise level.

The residual $R(i_s, i_n)$ as discussed in section 8.4.2.1 could therefore contain the main outliers of the recorded signal and so can be taken for an estimate of the noise of the signal segment. Statistics can be used to map the residual signal into estimation of the noise components, describing this segment. The noise features F_{N1} to F_{N4} are normalised statistical measures that show relative noise within the overall signal similar to the signal features above apart from using the residual signal. F_{N5} to F_{N7} are simple measure of the mobility of the signal and are normalised to the number of samples within the segment. F_{N5} and F_{N6} look at the number of zero crossings the median and the residual signals have with respect to their mean. This give an simple approximation of the high frequency noise in the signals. F_{N7} counts the number of small samples jumps as they can be characteristic of a stuck signal, thus capturing low frequency situations. F_{N8} records the maximum residual signal to allow the relative noise values to be decoupled from the absolute measurement. Noise characteristic measures:

- $F_{N1} :=$ The sum of the residual (see equation (8.14)).
- $F_{N2} :=$ The standard deviation of the residual (see equation (8.15)).
- $F_{N3} :=$ The peak normalised skew of the residual (see equation (8.16)).
- $F_{N4} :=$ The peak normalised difference between the maximum and minimum of the residual (see equation (8.17)).
- $F_{N5} :=$ The peak normalised mean crossings of the median signal (see equation (8.18)).

- $F_{N6} :=$ The peak normalised mean crossings of the residual signal (see equation (8.19)).
- $F_{N7} :=$ The number of small sample differences of the raw signal (see equation (8.20)).
- $F_{N8} :=$ The maximum amplitude for the residual (see equation (8.21)).

$$F_{N1}(i_s) = \frac{E[R(i_s, i_n)]}{\max(S(i_s, i_n))} \quad (8.14)$$

$$F_{N2}(i_s) = \frac{\text{std}(R(i_s, i_n))}{\max(S(i_s, i_n))} \quad (8.15)$$

$$F_{N3}(i_s) = \frac{\text{skew}(R(i_s, i_n))}{\max(S(i_s, i_n))} \quad (8.16)$$

$$F_{N4}(i_s) = \frac{(\max(R(i_s, i_n)) - \min(R(i_s, i_n)))}{\max(S(i_s, i_n))} \quad (8.17)$$

$$F_{N5}(i_s) = \frac{(\text{zero_crossing}(\tilde{S}(i_s, i_n)))}{|\tilde{S}(i_s, i_n)|} \quad (8.18)$$

$$F_{N6}(i_s) = \frac{\text{zero_crossing}(R(i_s, i_n))}{|R(i_s, i_n)|} \quad (8.19)$$

$$F_{N7}(i_s) = \frac{|\{x : x \in S'(i_s, i_n), x < \eta\}|}{|S(i_s, i_n)|} \quad (8.20)$$

$$F_{N8}(i_s) = \max(R(i_s, i)) \quad (8.21)$$

Where $E[*_X]$ is the expected value operator, in most of these cases, the arithmetic mean is used. $|*_X|$ is the count or cardinality of the set $*_X$, $S'(i_s, i_n)$ is the first sample differential of the signal. η is a small threshold, was set $\eta = E[S'(i_s, i_n)] - \text{std}(S'(i_s, i_n))$. Finally $\text{zero_crossing}(*_X)$ is an operator to count the number of time the set $*_X$ crosses its own mean.

8.4.2.3 Morphology

The two features described above are estimates of the two main components of any waveform without reference to the shape or the look of that signal. Morphology defines dimensions to categorise the shape of this signal. The feature definition should be a continuous, monotonically increasing function. Ideally the function should be bounded, by being normalised against the signal segment, because the signal segment size is described by the signal component above, allowing the morphology measurements to be directly compared.

The following features form some sort of comparison to other segments. The first set look to the history, the second and third set compare with a ‘Good’ and ‘Bad’ Segment templates. These were computed from the signal dataset using the first session from each user 5, 6, 7, 8 and 14. This create a dataset of over 345 minutes of signal and the following number of segments:

- 20553 ‘Good’ ECG segments.
- 8901 ‘Bad’ ECG segments.
- 9725 ‘Good’ PPG segments.
- 17018 ‘Bad’ PPG segments.

These segment signal were statistically averaged to find the average segment signal for each of the above cases. These were then used for the templates in the equations below.

Morphology characteristic measures are comprised of:

- A comparison to the previous segment.

Comparing the current signal segment to the previous signal segment $i_s - i_l$ where i_l is the look back segment index ($i_l = 1$ is the previous segment). This forms a measure and definition of the consistency of the signal. The further back the segment is compared, the deeper the consistency check.

$$F_{M1}(i_s) = \text{correlation}(S(i_s, i_n), S(i_s - n, i_n)) \quad (8.22)$$

$$F_{M2}(i_s) = \frac{E[S(i_s, i_n) - S(i_s - n, i_n)]}{\max(S(i_s, i_n))} \quad (8.23)$$

$$F_{M3}(i_s) = \frac{\text{std}(S(i_s, i_l) - S(i_s - i_l, i_n))}{\max(S(i_s, i_n))} \quad (8.24)$$

- A comparison with known ‘Good’ morphology.

Comparing an extracted segment that has been picked a-priori to be one of ‘Good’ morphology, this then forms a dimension with respect to a known signal with the correct shape.

$$F_{M4}(i_s) = \text{correlation}(S(i_s, i_n), S_g(0)) \quad (8.25)$$

$$F_{M5}(i_s) = \frac{E[S(i_s, i_n) - S_g(0)]}{\max(S(i_s, i_n))} \quad (8.26)$$

$$F_{M6}(i_s) = \frac{\text{std}(S(i_s, i_n) - S_g(0))}{\max(S(i_s, i_n))} \quad (8.27)$$

- A comparison with known ‘Bad’ morphology.

Comparing an extracted segment that has been picked a-priori as one of ‘Bad’ morphology, this then forms a dimension of similarity with respect to a known signal without the correct shape.

$$F_{M7}(i_s) = \text{correlation}(S(i_s, i_n), S_b(0)) \quad (8.28)$$

$$F_{M8}(i_s) = \frac{E[S(i_s, i_n) - S_b(0)]}{\max(S(i_s, i_n))} \quad (8.29)$$

$$F_{M9}(i_s) = \frac{\text{std}(S(i_s, i_n) - S_b(0))}{\max(S(i_s, i_n))} \quad (8.30)$$

Where in all the above equations $E[*_X]$ is the expected value operator, in this case, the arithmetic mean is used, $\text{correlation}(*_X, *_Y)$ is the Pearson correlation coefficient between the sets $*_X, *_Y$. $S(i_s, i_n)$ is the signal segment from above, n is the sample index, i_s is the current segment index and $i_s - 1$ is the previous segment index, $S_g(0)$ and $S_b(0)$ is the first stored template of the ‘Good’ and ‘Bad’ signal respectively.

8.4.2.4 Entropy

To allow the probability distribution of the segment signal values to be represented and compared, the disorder of the signal was used by measuring the entropy of the signal. The hypothesis postulates that the more disordered the signal, the higher the entropy of the signal should be. Mutual information between the current segment was measured to gain a measure of the shared information between this and either the previous segment, or the ‘Good’ and ‘Bad’ segment templates as an additional measure to compare the probability density functions (PDFs) of received signal values.

Entropy characteristic measures are:

- $F_{E1} :=$ Current signal segment entropy (see equation (8.31)).

- $F_{E2} :=$ Difference in entropy between the current and previous segment (see equation (8.32)).
- $F_{E3} :=$ Mutual information between the current and previous segment (see equation (8.33)).
- $F_{E4} :=$ Conditional entropy between the current and previous segment (see equation (8.34)).
- $F_{E5} :=$ Correlation between the current and previous signal value PDFs (see equation (8.35)).
- $F_{E6} :=$ The L2 distance between the current and previous signal value PDFs (see equation (8.36)).

$$F_{E1}(i_s) = H(S(i_s)) \quad (8.31)$$

$$F_{E2}(i_s) = H(S(i_s)) - H(S(i_s - 1)) \quad (8.32)$$

$$F_{E3}(i_s) = H(S(i_s)) + H(S(i_s - 1)) - H(S(i_s), S(i_s - 1)) \quad (8.33)$$

$$F_{E4}(i_s) = H(S(i_s)|S(i_s - 1)) - H(S(i_s - 1)) \quad (8.34)$$

$$F_{E5}(i_s) = \text{correlation}(\text{pdf}(S(i_s - 1)), \text{pdf}(S(i_s - 1))) \quad (8.35)$$

$$F_{E6}(i_s) = \sqrt{\left\{ \sum_{i,j} [\text{pdf}(S(i_s), i) - \text{pdf}(S(i_s - 1), j)] \right\}} \quad (8.36)$$

Where $H(*_X)$ is the entropy of the signal set $*_X$ and $\text{pdf}(*_X, *_y)$ is the bin histogram containing the PDF of the signal set $*_X$ at bin $*_y$.

The next 8 measures are tailored to the signal type as measures F_{E7} to F_{E10} compare the signal against the ‘Good’ measures and F_{E11} to F_{E14} compare the signal to the ‘Bad’ signal templates from above. For both templates, they are defined as:

- The mutual information between the current and previous segment.

Using equation (8.33) for the definition. $F_{E7}(i_s)$, for the ‘Good’ template and $F_{11}(i_s)$ for the ‘Bad’ template.

- The conditional entropy between the current segment i_s and template segment.

Using equation (8.34) for the definition. $F_{E8}(i_s)$, for the ‘Good’ template and $F_{12}(i_s)$ for the ‘Bad’ template.

- The correlation between the current segment i_s and template signal value PDFs
Using equation (8.35) for the definition. $F_{E9}(i_s)$, for the ‘Good’ template and $F_{13}(i_s)$ for the ‘Bad’ template.
- The L2 distance between the current segment i_s and template signal value PDFs.

Using equation (8.36) for the definition. $F_{E10}(i_s)$, for the ‘Good’ template and $F_{14}(i_s)$ for the ‘Bad’ template.

The entropy features presented a problem when tested, since they rely on the PDFs from the two segments. When these can only take a known number of values the PDFs are simply calculated and the result of the PDF comparisons can be made on the same PDF basis. However, the signals segments being examined here, are real valued continuous signal samples. If the unique set of values are compared it is unlikely that two signals of this sort would have exactly the same values. Thus the numerical joint entropy would be low or even zero. Thus to rectify this, the segments were discretised using histograms. Experimentally using 10 bins calculated from the minimum to the maximum for the signals in question, worked the best for estimating signal quality. This ensures that there are always 10 possible values which span the best range of the segments being compared. For the joint PDFs the minimum and maximum over both segments was used. If more than 10 bins were used the entropy rose but it got more noisy, conversely if less bins were used the response was smoothed out but not very informative.

8.4.3 Discussion

The features presented above have been derived from those found in the literature, then expanded upon to create symmetries, in order to have a consistent set of features to explore and model the signal quality within the next study. The 37 features examined above do show reactions to the different sections of the signal

trace and form a 37 dimension description of each signal segment. These are difficult features to assess, like noise and entropy, because they are measuring the internal mathematical features of the signal. It is hard to subjectively gauge the amount of noise or entropy in a signal segment. However they do have subjective variations, so the best gauge is to allow the features to be objectively judged by allowing models to fit the features set to the actual signal quality and observe their performance at estimating the quality in practise. The single features have been assessed and described in section 8.5.3.3 along with the feature groups and overall modelling performance in the study described in the next section.

8.5 Case Study - Subjective Signal Quality With Objective Features

8.5.1 Overview

The previous case study in section 7.4, looks at the accuracy and consistency of annotations comparing human annotators and assessors with machines as a control. The current annotations are accepted as a standard for the signal quality given that the overall agreement of the assessors ranged from 87%, up to 92% for the clinical assessor. The manual annotations of the signal were shown to be very time consuming and open to interpretation as there are differences between assessors. An improvement could be to systematically discover features and models that could automate the annotations on a per-segment basis. The two main parts are: (1) to determine the best combinations of the features discussed above to use in the models and (2) to determine the type of model structure to gain accurate signal quality assessments. The framework for defining tests as described in section 8.3, will be used to explore the feature space.

8.5.2 Methodology

The annotated data to be used was the same random, 50, one-minute segments used to form the dataset in the previous study described in section 7.4, with the human given annotations assessed in the previous study. They were used as the quality standard for the test set. The training set was taken as the first period of the first session held for each person. The training and testing sets were then concatenated together to form the group training and testing set, for all of the users in the samples. This was used to train and test the models below. This gives the final number of classes for the training and test sets as:

- 64.5% (1278) ‘Good’ Training segments.
- 35.5% (2323) ‘Bad’ Training segments.
- 63.0% (1468) ‘Good’ Testing segments.
- 37.0% (2500) ‘Bad’ Testing segments.

These are not directly balanced classes as expected, and so the specificity and sensitivity will be taken into account in the analysis.

The system of signal segmentation and quality assignment described previously in section 7.3 was used to process the signal data from the study into a series of event segments with the predicted outcomes assigned. These events were then described by calculating those features above, including the context around each event. The resultant set of descriptions were then made available to have features selected and the models trained and assessed.

When choosing the learning model type to use to assess the features and to get the current model’s performance, SVM based models were used, utilising the:

- SVM Classifier (SVMc).
- Regressive SVM (SVMr).
- One-Class SVM (SVMoc).

This family of learning models was used since, as was mentioned above, SVMs use kernel methods to change the distance metric in the feature space. This can allow a flexible decision surface, that can be solved analytically, thus removing the dependence on the assessment of the models trained. The reason that there are three is that the SVMc allows the basic classification, SVMr allows for further exploration of the high dimensional kernel mapping and the SVMoc gives an indication whether training can be done with only one class of segment, known as an example of outlier detection. Each model was given 1,000,000 iterations to reach a model convergence. The models were named as per the test framework as $M_{i_{mc}, i_{mi}}$, where $i_{mc} \in a, b, c$ is the index for the class of the model (a is the SVMc, b is the SVMr and c is the SVMoc) and i_{mi} is the model index for the class i_{mc} . The parameters chosen for the grid were based on the parameters for the models:

- Kernel type := {'Radial basis function (RBF)', 'polynomial'}.
- Degree in the case of the polynomial kernels $d := 2, 3$.
- Gamma $\gamma := 0.5, 0.2, 0.1$.
- Cost function weighting $C := 0.5, 1.0$.
- Soft margin $\nu := 0.1, 0.2$.

These parameters were combined so that no models were duplicated and no parameters were added to a model which did not support them. The implementation for these models are based on those from the Scikit Learn Python library¹. This produced a population of 54 models because the degree is only used on the polynomial kernel, and gamma is used on both the RBF and linear kernels. 20 of these models were randomly selected, to allow a more manageable number of model sets to run the feature sets chosen below.

The feature space used consists of:

- 5 Signal measures F_S , plus 1 segment period F_{T_p} .

¹Version 0.18.1, used for analysis. For more information on the models and their implementation, please see <<http://scikit-learn.org/stable/modules/svm.html>>. Retrieved 2017/04/21

- 8 Noise measures F_N .
- 9 Morphology measures F_M .
- 14 Entropy measures F_E .

These were defined in section 8.4, and form a feature description set of 37 features to choose from. The possible feature space that this would supply would be 2^{37} combinations. To cut this down, a simple grid search strategy will be used.

The feature sets were selected using random combinations of the 37 features above to create groups of features, to create a fair selection of the feature combinations from such a large feature space. The selection was based on a system of statistical subset selections and combination. Each class of features provides combinations of anywhere from 0 (no feature of that class) to 5 features of that class to be combined with the other classes with 40 from each combination selected. The sets created were then randomly sampled and combined with a basic feature set consisting of the single features and the full features list to make a total list of feature combination sets. This system allows an even spread of feature combinations, taken from each subgroup of feature combination with the ability to include manual additions. As a part of this set, each single feature will be selected and tested on its own. Any other combination of features are then randomly chosen to make 15,000 feature sets named as $\phi_{i_{fs}}, i_{fs} \in (1, 15000)$ in total.

The tests were then created from the 20 models combined with the 15,000 feature sets to create 300,000 tests $\Psi_{i_{ii}} \in (1, 300000)$ as defined in section 8.3.

The 300,000 models now defined, were run using the analysis framework to load and manage the processes as described in section 3.4. The model and feature performances were evaluated by utilising the DFR, which in turn depends on the specificity and sensitivity. This was introduced in section 8.2.1.2.

8.5.3 Results

The next sections investigate the performance of the models run in terms of DFR for the assessment of signal quality on unseen test data for the ECG and PPG signals.

These have been investigated in the sections below to better illustrate the insights that can be found in the test performance population. First the relationship between the distance metric, DFR and the AUC will be investigated. Next the performance of the different feature classes and individual features will be examined to find the best feature class and best feature set. Lastly the overall performance of the models will be considered for the signals and the models, in order to find the best model and feature set to estimate signal quality.

8.5.3.1 DFR and AUC Relationship

For the analysis of the models within this chapter, the DFR was introduced. It would be useful to compare this metric against the more usual AUC.

Figure 8.3 illustrates this relationship between the distance metric as defined above and the more traditional AUC metric. Both show a positive correlation for the binary classifiers (SMVc and SMVoc). This was expected since, if using AUC on a binary class, only three points are defining the curve with the operating point as the defining point and the other two being the extremes of bias (all ‘Good’ or all ‘Bad’). The regression classifier shows a more variable relationship because the regression output was converted into a binary classifier by equation (7.27) and setting the threshold ϵ_c to 0. The graph follows a positive correlation with the distance metric, which interestingly, gets stronger as the performance rises. The AUC is higher for the regression models since they have a more complicated polygon to assess for the area rather than the two limits and the operating point. The AUC is higher overall as the maximum of the AUC metric is one, where the maximum for the DFR is $\frac{1}{\sqrt{2}} \approx 0.707$.

The distance metric can, therefore, be used just as well as the AUC for binary classifiers and could be a good approximation if only an operating point or the final classified instance totals are available for the regression models.

8.5.3.2 Interactions and Performance of the Feature Class Groups

The relative performance of the feature classes used on the SVMc model type is illustrated in figures 8.4a to 8.4d, separated by the number of grouped classes. Each

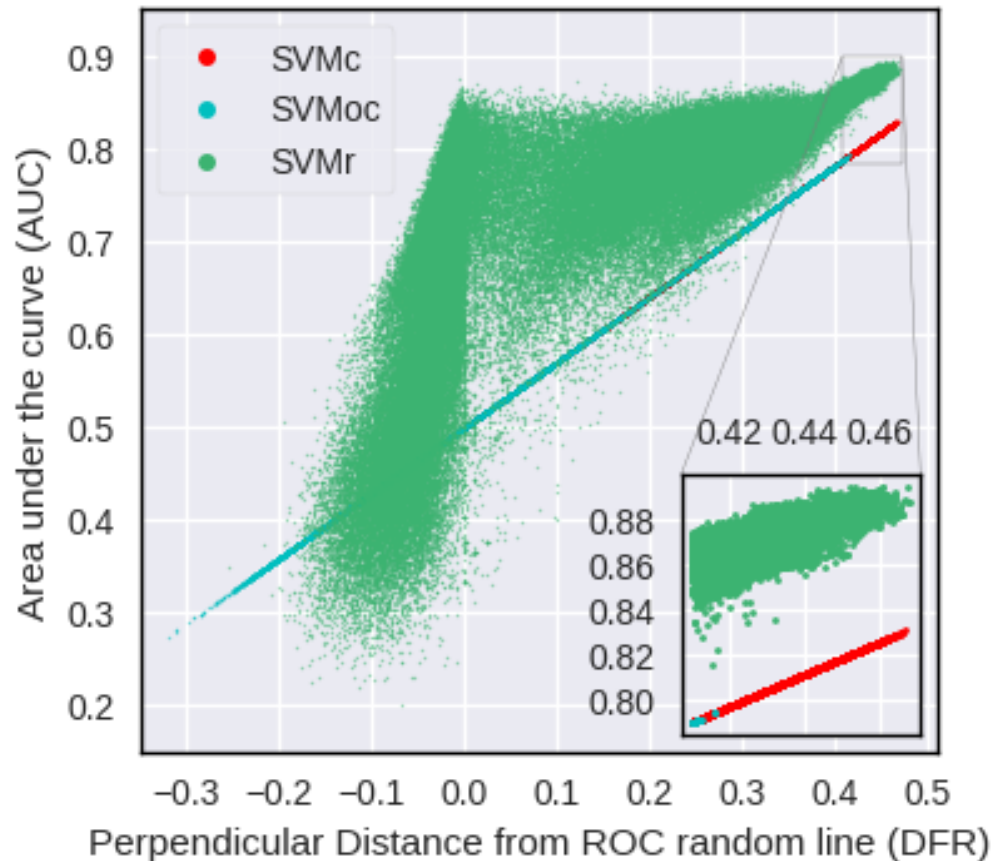
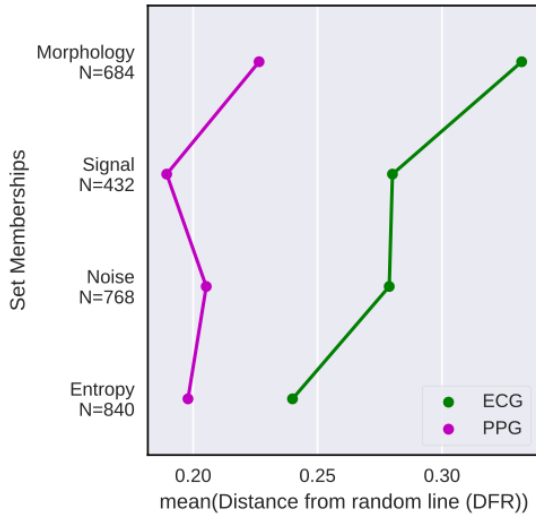


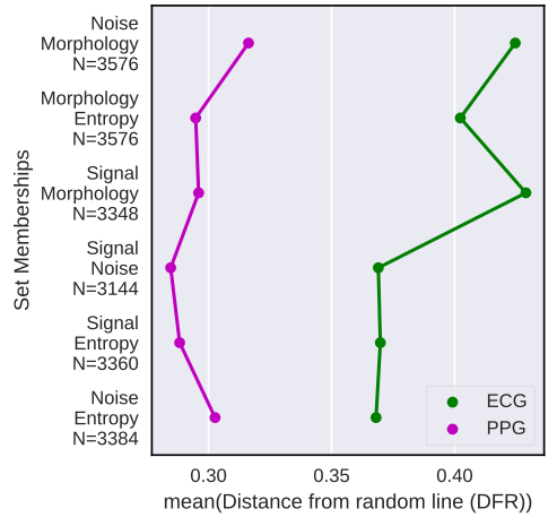
Figure 8.3: The relationship is shown between the area under the curve (AUC) and the perpendicular distance metric from the receiver operating characteristic (ROC) random line for the 300,000 tests. Three models are grouped together by the plotted colours to better show their relationships. Inset, A zoomed portion to better show the relationship at high performance.

group has one or more features of the feature classes shown on the Y axis categories. The graphs show the median of model populations, given the various feature group configurations.

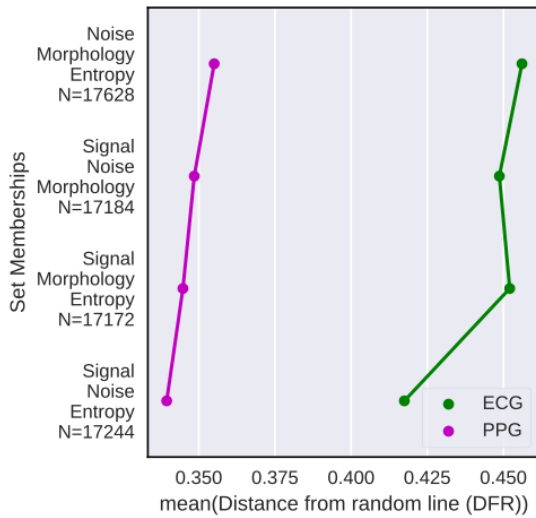
When using a single feature class as shown in figure 8.4a, the ‘morphological’ class of features creates the best models for both signals. The signals then diverge, with the ECG signal finding ‘signal’ features helpful and ‘entropy’ features being the least helpful. The PPG signal, however, found ‘noise’ features more useful and ‘signal’ features the least useful. This trend is followed when two feature classes are used as shown by figure 8.4b. The ‘morphological’ feature class can bolster any other feature class with the ‘entropy’ feature slipping into the background as both ‘signal’ and ‘noise’ features perform better for the ECG signal than in the single feature group case. The PPG signal follows the same trend as in the single feature



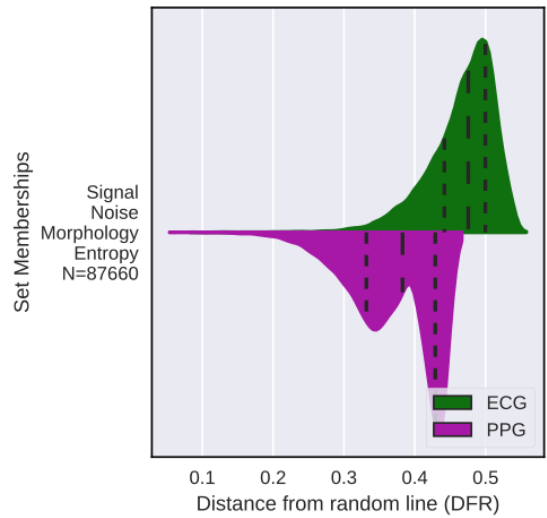
(a) Performance of tests using 1 class.



(b) Performance of tests using 2 classes.



(c) Performance of tests using 3 classes.



(d) Performance of tests using 4 classes.

Figure 8.4: The test performance using 1-3 feature classes has been summarised to the median point of the model group. They have been broken down by signal name to show the different model performance. N gives the population of models used for each feature type category, marking the median position. Tests that use all 4 feature classes in the bottom right shows the kernel density function of the distribution of the tests broken down by signal name. Performance improves to the right, increasing the distance from the random line (DFR). The median and inter-quartile range are shown by the dashed lines on the distribution.

class case with ‘morphology’ and ‘noise’ making the strongest sets. However, ‘noise’ and ‘entropy’ classes do surprisingly well for the PPG signal, suggesting that ‘noise’ features can help significantly. The picture changes when features from three classes are used as shown in figure 8.4c. This shows that the best models for the PPG signal are usually based on the ‘morphological’ features, but ‘noise’ features are a better companion than ‘entropy’. The ECG signal has the strongest tests using ‘morphological’ features, almost regardless of the other feature combinations chosen.

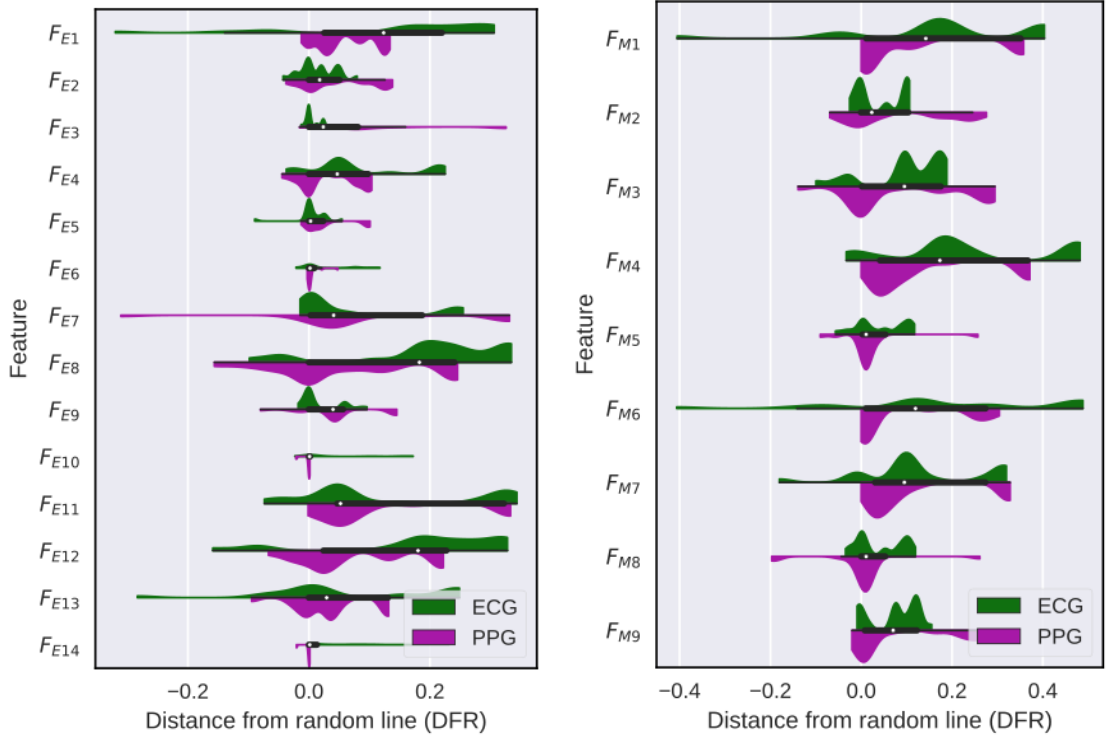
Figure 8.4d shows the estimated kernel density of the model’s performance when using all four feature classes. It is expected that these model’s performances would be the best on average since they can draw on combinations of all 37 features. The double peak is an interesting finding for the PPG tests as there could be a set of features which when removed could allow stronger features to be used as there is a population of feature tests which can not produce high performances.

Overall ‘morphological’ features are beneficial in the determination of signal quality, but the signals do favour different subclasses. The ECG signal does not seem to need many more features other than morphology suggesting that the other features share some overlap. However, the PPG signal finds ‘noise’ and partially ‘entropy’ to be good companion feature classes.

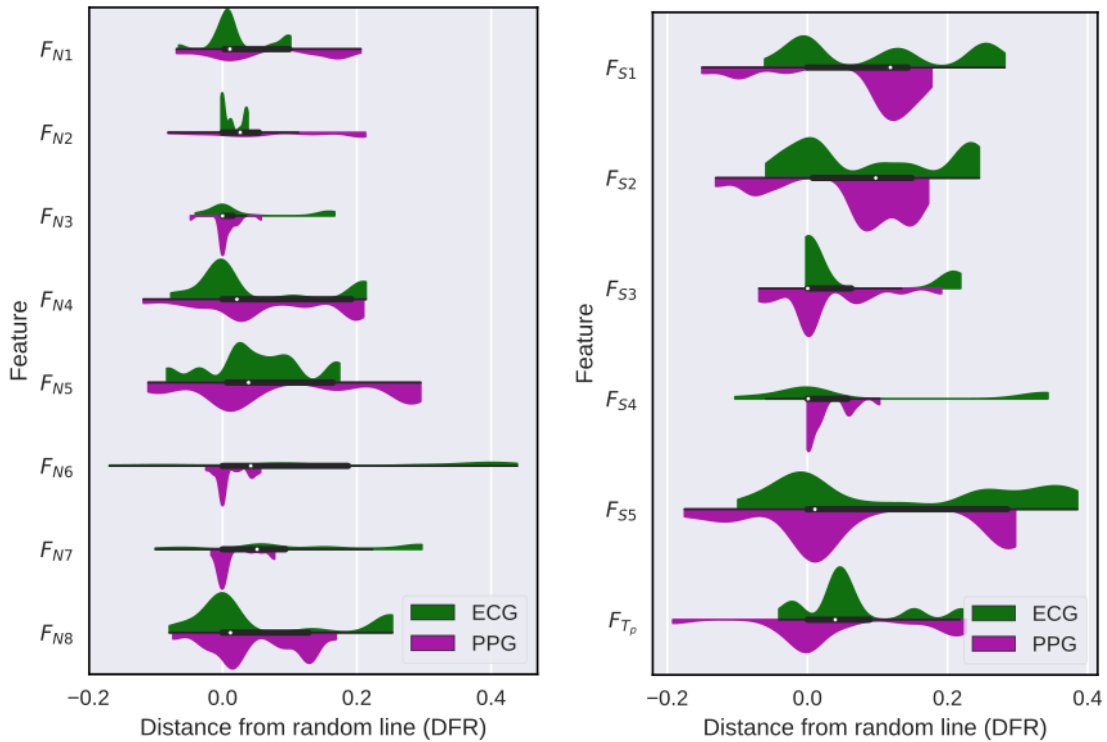
8.5.3.3 Single Feature Performance

Some conclusions can be drawn from the feature groups at this point: ‘morphological’ features, in general, form a strong SVM classifier. The ‘entropy’ feature generally can be seen to be weak when classifying the quality of a signal. The features tested in the model analysis allow the breakdown of the performance into individual features, to allow the comparison of the features within the class to identify particular winners.

This is shown in figures 8.5a to 8.5d, where each feature type has been broken down. The main feature type to be examined is the morphological feature class, as it has shown to be a strong candidate for future investigations. This is shown in figure 8.5b, out of these F_{M1} and F_{M4} were the strongest features for both signals,



(a) Tests using single features representing 'entropy', are split by single features. (b) Tests using single features representing 'morphology', are split by single features.



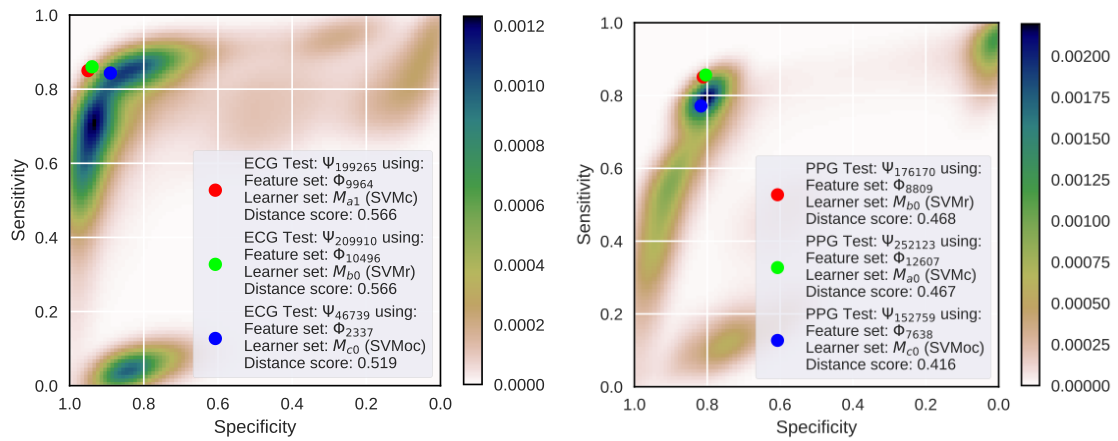
(c) Tests using single features representing 'noise', are split by single features. (d) Tests using single features representing 'signal', are split by single features.

Figure 8.5: The kernel density function of the models using features representing each feature class, each graph shows a single feature group. The centre of each distribution shows a box plot with the thin line marking the extents, the thicker line marking the inter-quartile range and a white mark for the median. N gives the number of models used for each distribution category. Performance improves to the right, increasing the distance from the random line (DFR).

showing that correlation to the previous segment and the ‘Good’ template were important for both signal types. F_{M6} was strong individually for the ECG signal, which translates to the standard deviation of the error between the current and ‘Good’ segment. F_{M4} defined above, was the correlation of the segment with the ‘Good’ template of the signal. As one might expect, the comparison to a ‘Good’ segment of the signal is important to both signal types. The signal consistency was also useful as shown by F_{M1} , where it is compared to the previous segment because if the correlation between segments is high, it suggests that there is consistency in the signal. The entropy feature has many strong measures for its class, but since they get combined with other features, they become less important as they may overlap with the other features. The correlation between the morphology was also significant, in the ‘entropy’ features since F_{E1} and F_{E3} along with F_{E7} and F_{E11} produce tests with high performances. These are the mutual information measures between the current signal and the previous signal in the first case, and the ‘Good’ and ‘Bad’ templates in the latter cases. F_{E8} and F_{E12} also do well since they measure the conditional entropy between the current and the ‘Good’ and ‘Bad’ templates respectively. These comparisons seem to be able to define the quality of the signals, which is partially unsurprising since these templates act like trained RBF kernel functions, with the templates defining the basis locations.

Overall features focused on the consistency of the segment with its immediate predecessor seem to be the best features in general. This can lead to two further directions to investigate. The first is to add to the features by comparing even older segments to look for improvement. Second is to investigate new models such as recurrent networks which would allow the system to train these parameters.

Next, the overall performance is considered to find the best models to estimate the signal quality for the two signals. This is a new dataset and there is currently no direct comparison, so the model performance surface allows for a visual representation of the data and facilitates estimation of where the models could be developed for further investigations in the future.



(a) Showing the kernel density for the ECG. (b) Showing the kernel density for the PPG. This shows that the majority of tests are in the upper left corner with a slight bias to the left edge, with very few along the top edge.

Figure 8.6: The kernel density function of the 300,000 test over a ROC field. The top models of each type determined by distance from the random (DFR) line are marked along with the learning and feature sets used.

8.5.3.4 Test Performance Using ROC Field Contours

Due to the 300,000 models and features combinations tested, the ROC graphs have been simplified by using kernel density contour plots to show the overall result visualisations.

Figures 8.6a and 8.6b show the ROC field graph and density contours with the top models for each type of model marked on the field. Figure 8.6a shows the performance surface for the ECG tests and the figure 8.6b shows the performance surface for the PPG tests. These contours are used to summarise the model tests performed and to give the best models, a context as to how well they perform based on the normalised densities of the other tests.

These graphs show that most models have a good operating point in the top left. However, there are differences between the signal types. The PPG signal shows a high concentration of tests near the best models marked. Some are shown down the left side of the field. The effect is more pronounced in the ECG signal where they are more spread out down the left side than along the top. The PPG tests seemly have trouble achieving a high sensitivity suggesting that they are having trouble with the ‘Good’ segments, as when it gets high the specificity drops low with the

Table 8.2: The top model performance for ECG based on the maximum distance from the random line (DFR).

Test	DFR	Sensitivity	Specificity	Learner	Feature Set
Ψ_{199265}	0.566	85.0%	95.0%	M_{a1}	Φ_{9964}
Ψ_{209910}	0.566	86.0%	94.0%	M_{b0}	Φ_{10496}
Ψ_{46739}	0.519	84.3%	89.0%	M_{c0}	Φ_{2337}

Table 8.3: The top model performance for PPG based on the maximum distance from the random line (DFR).

Test	DFR	Sensitivity	Specificity	Learner	Feature Set
Ψ_{176170}	0.468	85.1%	81.1%	M_{b0}	Φ_{8809}
Ψ_{252123}	0.467	85.6%	80.4%	M_{a0}	Φ_{12607}
Ψ_{152759}	0.416	77.1%	81.7%	M_{c0}	Φ_{7638}

high concentration in the top right corner. The problem with the ‘Good’ segments, then invites more work to be done to differentiate the ‘Good’ PPG signal segment. This is less marked in the ECG, as the tests down the left side show that ‘Bad’ classes can be found accurately for both signals.

8.5.3.5 Optimum Tests for the Signals

The top tests of each model type are grouped by the signal type. These are the best selected tests for each model type, as measured by the greatest DFR. The ECG models are shown in table 8.2 and the PPG models are shown in table 8.3. The tests are shown with the feature and learner settings in table 8.4 and table 8.5 respectively.

Tables 8.2 and 8.3 shows the best overall model’s performance on test data in terms of the DFR for the ECG and PPG signals. The best model overall is the SVMc with the SVMr slightly outperforming it for the PPG signal. The performance of these models are above 80.4% and 77.1% for specificity and sensitivity respectively based on the SVMoc. This is approximately 3-5% lower than for the other classifiers tested. A conservative estimate of the performance across the signal types is greater than 84% classification accuracy for the quality annotation of those signals.

Table 8.4: The top model performance based on the maximum distance from the random line (DFR).

Feature Set	Signal Name	Set Count	Features
Φ_{2337}	ECG	3	F_{N3}, F_{N6}, F_{M4}
Φ_{7638}	PPG	11	$F_{S4}, F_{Tp}, F_{S2}, F_{S5}, F_{S3}, F_{N5}, F_{N4}, F_{N1}, F_{M6}, F_{M2}, F_{M4}$
Φ_{8809}	PPG	15	$F_{S5}, F_{Tp}, F_{S3}, F_{N4}, F_{N7}, F_{N5}, F_{M7}, F_{M1}, F_{M6}, F_{M2}, F_{E11}, F_{E8}, F_{E7}, F_{E10}, F_{E13}$
Φ_{9964}	ECG	10	$F_{N6}, F_{N4}, F_{M6}, F_{M7}, F_{M4}, F_{M8}, F_{M1}, F_{E4}, F_{E6}, F_{E2}$
Φ_{10496}	ECG	8	$F_{N6}, F_{N4}, F_{M6}, F_{M7}, F_{M4}, F_{M8}, F_{M1}, F_{E4}$
Φ_{12607}	PPG	9	$F_{S4}, F_{Tp}, F_{N5}, F_{N4}, F_{M6}, F_{M2}, F_{M4}, F_{M1}, F_{M5}$

Table 8.5: The top model performance based on the maximum distance from the random line (DFR).

Learner	Signal Name	Model Type	Model Parameters
M_{a0}	PPG	SVMc	kernel=rbf, C=1.0, gamma=0.5
M_{a1}	ECG	SVMc	kernel=rbf, C=1.0, gamma=0.2
M_{b0}	ECG	SVMr	kernel=rbf, C=1.0, gamma=0.5
M_{c0}	ECG	SVMoc	kernel=rbf, nu=0.1, gamma=0.5

The one class models (SVMoc) which were tested do not have the highest performance, but they only require instances of the ‘Good’ segments. This allows them, for a performance loss of up to 7% for sensitivity and 5% for specificity and could allow easier training on the collection devices themselves, where a doctor could pick ‘Good’ segments they want from the first recording of a user and train the model on those.

8.5.4 Discussion

It has been shown with these experiments that the signal quality is something that can be predicted from the features and the signal morphology. The overall performance, for the set of models tested, was for ECG 86.0% and 95.0% for sensitivity and specificity respectively and similarly for PPG, the result was 85.6% and 81.7%.

The sensitivity performance on average between ECG and PPG signals over the models has a range of 2.50%. Within the signal, this drops to approximately 1.70% and 8.52%. This suggests that the model’s accuracy of detecting a ‘Good’ signal segment has reached a limit, as the varying features do not make a significant difference. The specificity, varies between signals over 11.6%. This suggests that the accuracy of the ‘Bad’ signal segments classification is lagging behind the classification accuracy for the ‘Good’ segments in the PPG signal. The PPG might have more trouble with the ‘Bad’ segments as the evidence above shows because there is a greater range of morphologies and failure types for the PPG than for the ECG. These being motion and the blood perfusion of the tissues in question and signal saturation. The ECG however, is mediated by an electrical potential and so is less affected by the movement. Finding better ways of categorising the ‘Bad’ PPG segments would be a good next step to improve the classification accuracy of the PPG signal. The ECG requires more work to improve its sensitivity and so work on categorising the ‘Good’ segment would be needed.

The features the final models always used were ‘morphological’ features with ‘noise’ and ‘signal’ features also included in the sets. ‘entropy’ features seem to do well on their own but struggle when grouped with others. Interestingly none of the

models use all of the features available, and they have settled on 8 to 15 features with the SVMoc for ECG, only using 3. F_{M4} has appeared in all the top 3 models, which shows that a comparison to the ‘Good’ segment template was always useful. F_{T_p} is used for all of the PPG signal models suggesting that the period of the signal is more helpful for the PPG signal than for the ECG signal.

Comparing this to the agreement scores from the case study where on average, for both signals, assessors scored 84% and 90% in section 7.4 for sensitivity and specificity respectively. Results in the SVM models here had a better agreement to the quality standard than some of the assessors when looking at the ECG signal. However, the PPG is lagging behind with specificity, suggesting that better features are needed to characterise the ‘Bad’ signal quality. Further work to improve this is addressed further in chapter 10.

8.6 Framework Data Flow Example

This analysis has been performed within the framework as designed in chapter 3. The framework runs the analysis programs and modules defined above. The processes ask for data and load modules during their program and the framework captures this as metadata and creates a data and program dependency graph of whole data process. The data flow can then be traced from the raw data through to the final presented results and report which makes up this chapter. Part of this meta-graph which the framework has created for this experiment can be seen in figure 8.7. This shows the data nodes in red clustering around the data preparation process node shown in green at the top. The processed result data is stored in the yellow nodes in the middle. These have further connections as they have been used for the current model analysis setup node in green called ‘Analysis Setup’. This builds the feature and learner sets and combines them into tests which are shown as the three yellow result nodes in the middle of the figure. The four green process nodes named ‘Analysis Workers’ are the workers which actually ran the models in parallel processes on the ProcessSR. The results are then pulled from these worker nodes in to the bottom process called ‘Concatenate Worker Results’. The final results are

processed by the 'Present Signal Quality Node' which produces a report including tables and graphs and are then presented here in the chapter. The framework allows any part of the system to be interrogated and reviewed. Improvements can be made and the framework now knows the influence of the programs and subsequently the utilisation of the result produced. Program components are shared between the processes so there should always be one implementation of any function or system.

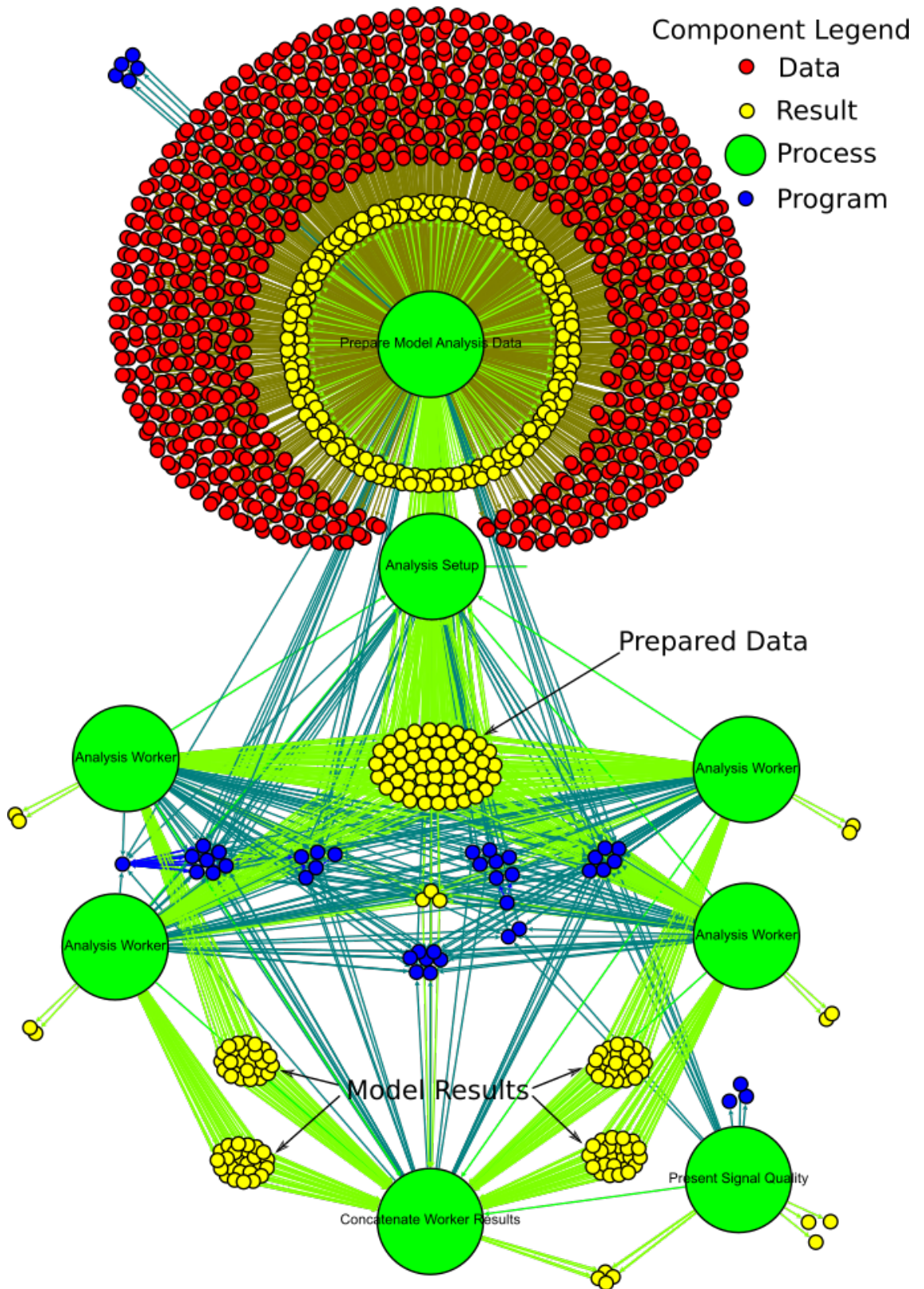


Figure 8.7: A partial signal quality framework dependency graph. The green nodes are the process components, yellow nodes mark result components, blue shows program components and red are the raw data nodes from the study. The lines show the linking between the node in the analysis structure. Due to the sheer number of nodes, only the process nodes are named.

8.7 Summary

This chapter has presented a system for determining and extracting potential features that could identify ‘Good’ signal streams and feature strategies. It also creates a description to assess signal quality automatically. The features extracted have been used in combination with models to explore the feature space to find the model and feature set which can bring the highest performance and accuracy when compared to human assessors. The performance given can be seen that the models performance was converging in the context of the other possible models shown for the current features. The next chapter now looks at using these models and the framework developed in the previous three chapters to investigate the modelling and estimation of blood pressure. This is a critical vital sign and provides an example of the further development possible within the framework as well as the usefulness of the signal quality assessment.

9 Blood Pressure Model Analysis

9.1 Overview

The previous work on determining the quality of bio-signals, can be used to better assess the measured variables offered by the signal data.

The estimation of blood pressure (BP) is a calculation which, in past work, relies on accurate measurements of the heart rate (HR) and pulse transit time (PTT) in order to estimate the users BP without the use of an inflation cuff. First a review of the literature on the past work on BP measurement and estimation will be presented. Following that, a study is presented using the signal quality models from the previous chapter to allow the quality of the measurements to be taken into account. This will then allow for the assessment of the known BP estimation models and features from the literature. Using the study data introduced and used in the previous chapters.

9.2 Review of Blood Pressure

BP measurements are used to determine the systolic and diastolic pressure of the blood. This can be measured at many points around the body, but the common point of recording is at the arteries close to the heart. The left arm is commonly used in practice as it is the closest to the aorta. The systolic pressure is the maximum pressure seen in the vessel when the heart is in contraction. The diastolic pressure is the minimum pressure seen in the vessel as the heart relaxes and the pressure is dissipated through the bodies arteries and capillary beds.

First, some background will be presented to illustrate the current state of BP measurement. Secondly, a discussion on the current methods of BP measurement

will be presented. Thirdly, the various methods of non-invasive BP measurement and modelling will be investigated.

9.2.1 Blood Pressure Measurement

BP has been used for many years to ascertain the activity level of the cardiovascular system in humans. The original method of measurement involved mechanically coupling an artery through a catheter to a column of moving liquid, then measuring the deflection that the pressure of the blood had to lift the liquid. Many liquids have been used, but mercury was found to be the best since it is the highest density liquid at room temperature and the deflection is smaller and more manageable for smaller devices. The measurement unit is mmHg which is the number of millimetres of deflection in a mercury column (Hg is the chemical symbol for mercury) and demonstrates the reflection of this history. The catheterised measurement is the gold standard and is still used in practise, using electronic pressure meters when people are under-going operations or diagnostic procedures, where real-time, and accurate measurements of their cardiovascular dynamics are important.

When a routine snapshot of the BP is sufficient, such as at a doctors examination, other methods of measurement are then employed. The main method used is the pressure cuff and stethoscope to measure the pressures at a point where the blood can be restricted in the vessels, usually at the elbow of the left arm in adults or the leg in paediatric care.

This has been automated by the use of the oscillometric method that uses a pump and valve to control the pressure applied to an arm or leg. The valve releases the pressure slowly while monitoring the change in pressure in the cuff. These methods examine this change, controlling the cuff pressure to find the maximum change in pulse pressure, seen in the pulsatile component of the cuff's bladder pressure. Once noted, the bladder has matched the mean arterial pressure (MAP) of the vessel and the minimum and maximum of the pulsatile signal can be used as an estimate of the systolic and diastolic pressures (Babbs, 2012).

9.2.2 Blood Pressure Estimation and Modelling

Work has been done to look for non-invasive methods of estimating BP without the use of a cuff. The estimation is made by using models of the cardiovascular system, commonly using the pulse transit time (PTT) defined as T_{PTT} and the instantaneous heart rate (HR) T_{HR} . Some mechanical properties of the vessel are also used as key parameters of the current models (Shriram *et al.*, 2010; Heard *et al.*, 2000; Cattivelli & Garudadri, 2009; McCombie *et al.*, 2006; Jadooei *et al.*, 2013; Shaltis *et al.*, 2006; Asif-Ul-Hoque *et al.*, 2011). The equation that is used as the common starting point, from the work that Isebree Moens (1878) carried out in their paper “Die Pulse Curve” with similar and independent work by D.J Korteweg, who derived the jointly named Moens-Korteweg’s equation, is described by Tijsseling & Anderson (2012). This established the basic wave speed formulae given in (9.1 Jadooei *et al.* (2013)).

$$V_{PWV} = PWV = \frac{L}{T_{PTT}} = \sqrt{\left(\frac{Eh}{\rho d}\right)} \quad (9.1)$$

Where, V_p = pulse wave velocity (PWV), L = length of the vessel, T_{PTT} = pulse transit time (PTT), ρ = Blood Density and d = inner radius of the vessel, h is the vessel wall thickness, E is the Youngs modulus of the vessel wall.

This has been used lately by the following groups Shriram *et al.* (2010); Heard *et al.* (2000); Cattivelli & Garudadri (2009); McCombie *et al.* (2006) and Jadooei *et al.* (2013). The models of estimation using PWV and the HR T_{HR} can be simplified into linear relationships to aid analysis, which are discussed in more detail below.

Another relationship between the BP and the PTT is given using the relationship by Jadooei *et al.* (2013) in dogs, where they found an exponential relationship between the Youngs modulus E at rest and the one observed when a given pressure was measured across the vessel’s wall. This is given by:

$$E = E_0 e^{\alpha P} \quad (9.2)$$

where, E is the Youngs modulus observed, E_0 is the Youngs Modulus at rest and the P is the pressure measured across the vessel wall and α is the exponential factor

between starting E_0 to E .

This can be introduced into equation (9.1) to give

$$P = -\frac{2}{\alpha} \ln(PTT) + \frac{1}{\alpha} \ln\left(\frac{L^2 \rho d}{g E_0 h}\right) \quad (9.3)$$

where g is the section length of the vessel. This was derived and used in work by Jadooei *et al.* (2013) and Wang *et al.* (2014).

These features and BP measurements are both linear relationships. The main model forms are listed below.

The relationship between the measured pressures for both the PWV V_{PWV} as shown in equation (9.1) or the PTT $\ln(T_{PTT})$ based on equation (9.3) and the HR T_{HR} as a linear sum, has been used for the estimation of both systolic P_s and diastolic P_d BP (Heard *et al.*, 2000; Jadooei *et al.*, 2013; Cattivelli & Garudadri, 2009). As there are two time measures, there are two formulations for the relationships give by

$$P_s(i_s) = AV_{PWV} + BT_{HR} + C \quad (9.4)$$

$$P_d(i_s) = AV_{PWV} + BT_{HR} + C \quad (9.5)$$

or

$$P_s(i_s) = A \ln(T_{PTT}) + BT_{HR} + C \quad (9.6)$$

$$P_d(i_s) = A \ln(T_{PTT}) + BT_{HR} + C \quad (9.7)$$

where, A, B and C are parameters to fit the models. Wang *et al.* (2014) have developed this further and suggested the inclusion of the previous BP measurement and free parameter D to give

$$P_s(i_s) = AV_{PWV} + BT_{HR} + CP_s(i_s - 1) + D \quad (9.8)$$

$$P_d(i_s) = AV_{PWV} + BT_{HR} + CP_s(i_s - 1) + D \quad (9.9)$$

or

$$P_s(i_s) = A \ln(T_{PTT}) + BT_{HR} + CP_s(i_s - 1) + D \quad (9.10)$$

$$P_d(i_s) = A \ln(T_{PTT}) + BT_{HR} + CP_s(i_s - 1) + D \quad (9.11)$$

Since there are two formulations of the relationship, both will be tested in the study to understand which feature V_{PWV} or $\ln(T_{PTT})$ can perform better. White

et al. (1993) and Takahashi *et al.* (2015) note that the error rate for a BP measurement as per the ISO81060-2:2009 standard should have a mean error of 5 mmHg and a standard deviation of less than 8 mmHg to be used as an accurate measurement device for personal use, and so this is the target that is ideally required in the following analysis.

9.3 Blood Pressure Estimation Structure

The BP and signal data from devices can be used to create a set of descriptions sufficient to train and test BP estimation models. The system utilises the signal quality assessment system as discussed in sections 7.3 and 8.3 to run BP estimation and modelling using the cleaned data. The principles from this are briefly extended below to handle the BP estimation.

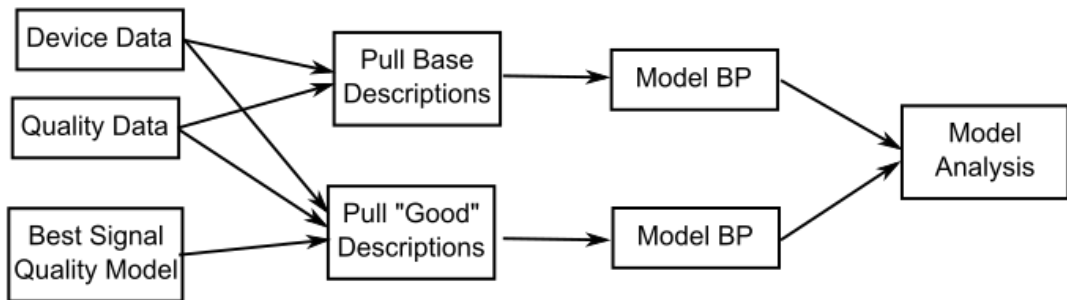


Figure 9.1: Diagram showing the basic architecture of the BP analysis framework.

9.3.1 Prepare and clean the data

The data to be used for the BP analysis needs to be gathered and checked to make sure all necessary parts of the data are found and matched up. Once the data is retrieved, contiguous sections are taken and processed using the signal quality framework to determine the signal quality features. Then using a pre-trained model from the previous chapter, estimates of the quality of each signal segment can be marked for feature extraction.

9.3.2 Extract the good features

Mapping the features from the fusion of the signals in the data set allows for bad data segments to be ignored. The signal events need to be processed to find the mapped signal features for BP which can be found from contiguous ‘Good’ quality signal segments, found in both the electrocardiogram (ECG) and photoplethysmogram (PPG) signals. Figure 9.2 shows diagrammatically the common features to be extracted, which include the:

- T_{PTT} := Pulse transit time (PTT) between the ECG and PPG signal
- T_{HR} := Instantaneous heart rate

These need to be extracted from the signal when a good section of signal data is found.

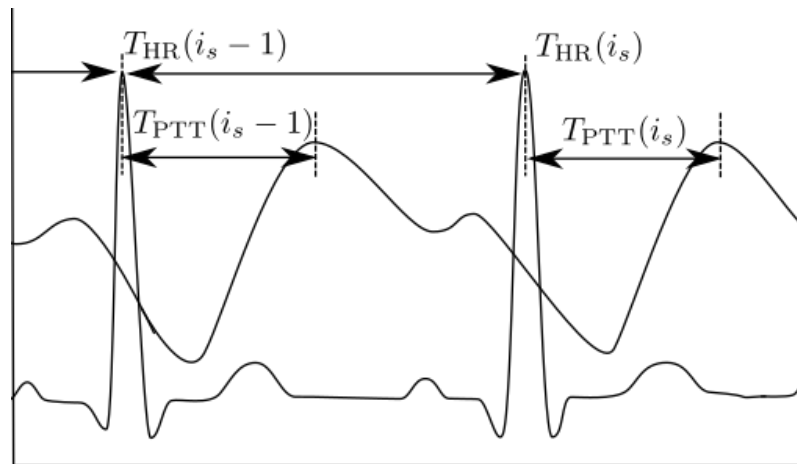


Figure 9.2: The pulse transit time (PTT) T_{PTT} and the instantaneous heart rate T_{HR} feature extraction diagram. The T_{PTT} is the time measured from the peak of the QRS on the ECG to the peak of the PPG signal. The T_{HR} is the time period measured from the previous peak to the current peak of the ECG signal because of the QRS it can have better temporal resolution.

When a data section is examined for features, the quality of the signal will be assessed. This can be recorded as the number of supporting events called the event support N_{support} . This is the number of ‘Good’ segments for the signals in question, which can be identified as ‘Good’ using the signal quality estimates above. This lends a meta-feature to the feature set to show the reliability of the other features extracted.

9.3.3 Model analysis

Once the features have been extracted from the ‘Good’ data, the dataset is then ready for model training and analysis as discussed in the previous chapters. Regression models are used in this case, such as the SVMr as used in the previous chapter or a linear or polynomial model can be substituted. The performance metrics are used here to assess two continuous values for agreement. Metrics will be used to describe both the performance of the models and also the final performance of the estimation expectation error. The R^2 metric will be used to describe the model and features, since it is bounded and does not rely on the units of the estimate. To assess the final model performance, two other metrics will be used: the mean, \bar{e} and the standard deviation, σ_e of the error e . These are standard metrics used for the assessment of regression models and estimates.

9.4 Case Study BP Feature Estimation Using Raw and Cleansed Data

9.4.1 Overview

The objective of this study is to utilise the signal quality assessment system as discussed in sections 7.3 and 8.3 to run BP feature comparisons and modelling using the raw and cleaned data from the quality assessment system with two aims. The first is to assess the signal quality system in practice. The second is to use the quality assessment to examine the features and their relationships to produce an improved BP estimate. The BP and signal data gathered in section 6.4 will be utilised for the experiments with the signal quality assessment structure described in sections 7.3 and 8.3. This will create a feature description for each BP measurement with the addition of the signal quality metric, which can then be used to investigate the feature relationships. The primary hypothesis is that the relationships can be made more prominent as the quality of the signals rises, by removing the weak measurements which are subject to noise. The measurement of the signal quality

will also help with the overall modelling errors by allowing minimums to be placed on the quality of data used for estimation and learning in the future.

9.4.2 Methodology

A process loads the data for the EIMO device and the Case-GE BP monitor from the framework. The BP measurements are spaced every 3 minutes, throughout the recorded data, and PTT and HR measurements need to be extracted from the data recorded from the EIMO device. Approximately 40 seconds before the BP measurement point, there is no signal. This gap in the signal is due to the user releasing their hold on the EIMO device to straighten out their arm so that the BP cuff could inflate and measure the user's BP. The lack of signal requires compensation by moving the feature measurement section further away from the BP measurement. However, the further away it gets from the BP measurement, the more feasible it is that the BP might have changed, potentially reducing the relationship.

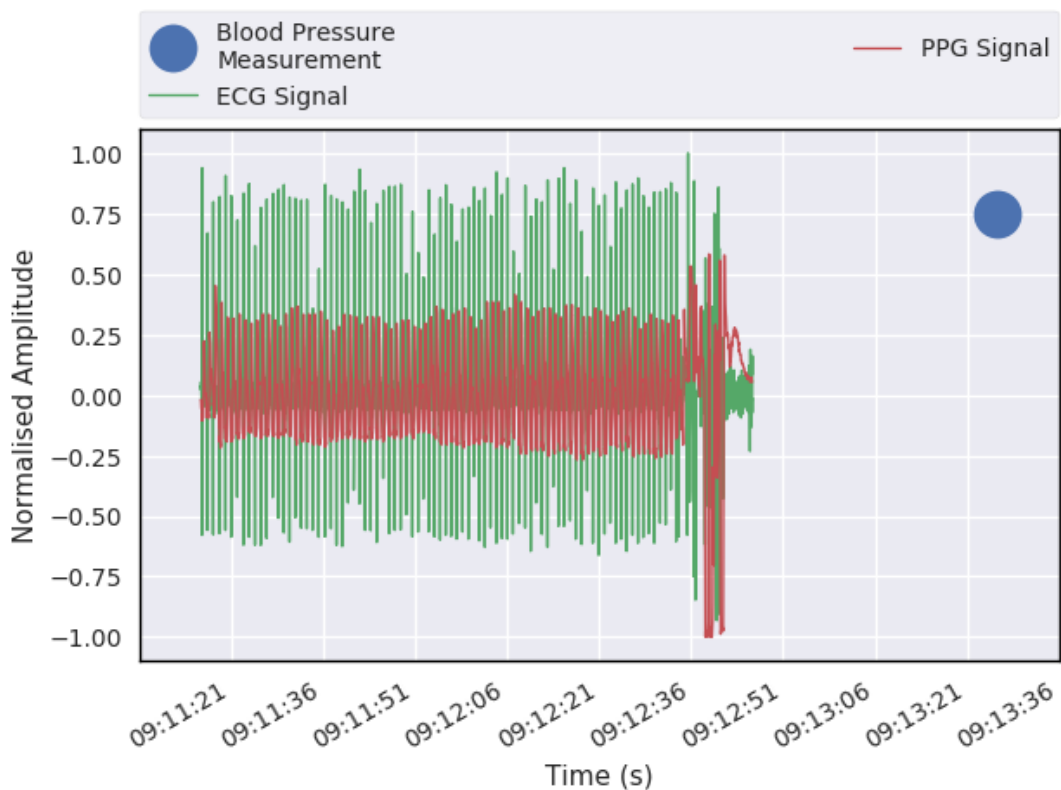


Figure 9.3: The graph shows the data section associated before the BP point marked in blue at 2015-04-24 09:13:26. The relationship between the BP measurement (marked in blue), and the data sections used to extract predictive features are shown. The ECG and PPG signals are drawn in green and red respectively.

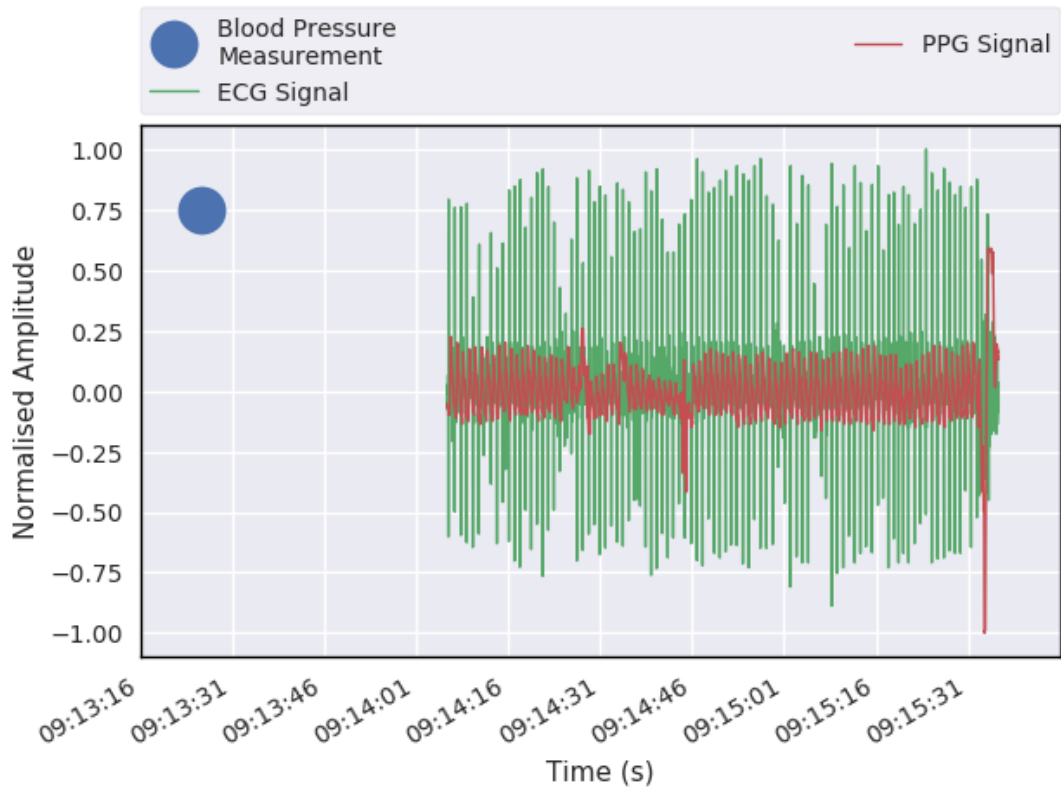


Figure 9.4: The graph shows the data section associated after the BP point marked in blue at 2015-04-24 09:13:26. The relationship between the BP measurement (marked in blue), and the data sections used to extract predictive features are shown. The ECG and PPG signals are drawn in green and red respectively.

Features have to be gathered to estimate the BP measurement. Each segment is measured independently between ECG and PPG cycles to gain the PTT and HR features as above. Windows are used to gather a group of signal segments to measure and average the result. The length and placement of the window sections were chosen to reflect a reasonable EIMO recording time of 90 seconds and will allow enough time to get a sufficient quantity of quality segments. The first measurement window was placed, ending at -40 seconds before the BP measurement to allow the window to be outside of the 40-second cuff measurement as mentioned above. A second feature measurement window was placed at the same time delay after the BP measurement for symmetry. The final section is brought forward, closer to the BP measurement. These window sections are used to allow for the loss of signal because of the cuff measurement. The sections oversample the BP measurement to make the most of a difficult measurement. Each window could be a legitimate section of the signals to capture. The three sections are summarised below. The times given for

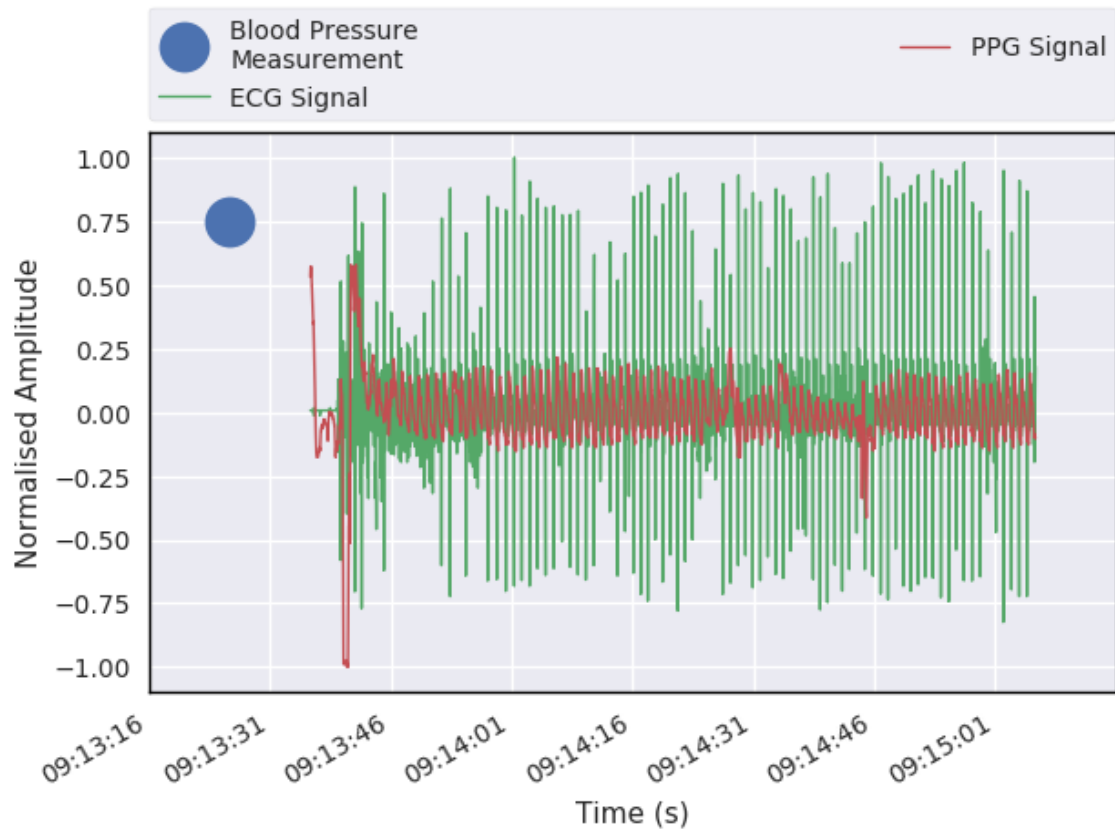


Figure 9.5: The graph shows the data section associated early after the BP point marked in blue at 2015-04-24 09:13:26. The relationship between the BP measurement (marked in blue), and the data sections used to extract predictive features are shown. The ECG and PPG signals are drawn in green and red respectively.

each section are relative to the BP measurement, with the negative values denoting a timestamp before the measurement and positive values after the measurement:

- Window one before the BP measurement defined as ‘Before’.

Starting at -130 seconds ending -40 seconds, shown as a sample figure 9.3.

- Window two after the BP measurement defined as ‘After’.

Starting at 40 seconds ending 130 seconds, shown as a sample figure 9.4.

- Window three just after the BP measurement defined as ‘Early’.

Starting at 10 seconds ending 100 seconds, shown as a sample figure 9.5.

The features measured from each window as defined above are to be considered.

The two features extracted are:

- The pulse transit time (PTT) T_{PTT} .

- The heart rate (HR) T_{HR} .

Due to the past research shown in section 9.2.2, the BP can be modelled with linear relationships to the PTT feature in section 9.2.2 to give the full features set to measure as:

- The heart rate (HR) T_{HR} .
- The pulse wave velocity (PWV) $V_{\text{PWV}} = \frac{1}{T_{\text{PTT}}}$.
- Logarithmic PTT $\ln(T_{\text{PTT}})$.

These features will be used for the assessment below. These are then compared to the BP measurements, which are the:

- Systolic BP P_s .
- Diastolic BP P_d .
- Pulse pressure or the difference between them $P_p = P_s - P_d$.

The features and measurements discussed above will be extracted from the previously identified sections and the number of supporting events will be recorded for each data section. The supporting events N_{support} for a section are defined here as the number of ‘Good’ quality event pairs of ECG and PPG segments, that were found after the signal quality assessment.

9.4.3 Results

After the data is assessed and classified, three feature measurements from the signal for each BP reading are recorded. These are then used to examine the relationship between the BP measurements (systolic P_s , diastolic P_d and the pulse pressure P_p) and the features discussed above.

First, the measurements extracted need to be cleaned of instances which have had no data or poor features as they are unusable or ill-defined. Table 9.1 shows a summary and description of the points gathered with the invalid sections removed.

Table 9.1: Summary of the data sections recorded and measured, showing how many sections were valid with enough data to extract the features and how many were invalid where they did not have enough data or held poor features, such as null values or heart rates below 20 bpm.

Data Section	Valid Segments	Invalid Segments	Total Segments
After	768	16	784
Before	783	1	784
Early	776	8	784

Table 9.2: Statistical description of the measurement instances of the data section windows extracted. This shows the statistics of the event support N_{support} the main feature mappings, along with the blood pressure (BP) measurements recorded for the study. T_{HR} is measured in beats per minute (bpm), P_s , P_d and P_p are measured in mmHg.

	N_{support}	$\ln(T_{\text{PTT}})$	V_{PWV}	T_{HR}	P_s	P_d	P_p
count	2327.000	2327.000	2327.000	2327.000	2327.000	2327.000	2327.000
mean	22.436	-1.098	3.065	79.190	128.548	67.225	61.323
std	28.867	0.199	0.881	15.157	16.605	10.538	16.801
min	0.000	-2.996	1.136	32.327	89.000	37.000	16.000
25%	0.000	-1.204	2.703	67.232	116.000	60.000	49.000
50%	7.000	-1.109	3.030	79.198	128.000	66.000	60.000
75%	39.000	-0.994	3.333	89.221	139.000	73.000	71.000
max	128.000	-0.128	20.000	134.831	190.000	104.000	119.000

This was performed to level the field before the signal quality was taken into account. These basic limits could be performed as the first step on any system. Table 9.2 shows a statistical summary of the valid feature measurements extracted. These tables show that most of the sections did have enough data to have features extracted. The average event support \bar{N}_{support} was 22.4 from the table. However, the event support per-measurement value varies between the users, which will be examined in more detail below.

The signal quality assessments allow for the ability to rank the measurements by the number of supporting segments N_{support} found for each section. The event support N_{support} can be used for the investigation of the BP estimation and modelling. Data can be selected based on the proportion of measurements with the highest number of supporting events N_{support} for example; one can select 75% of the top measurements. This kind of selection allows the most reliable data can illus-

trate groups and structured relationships with more clarity by removing the outliers, which also helps with the final accuracy of models using the cleaner data.

9.4.3.1 Distribution of Event Support Between Users

The event support metric provides a useful measure of the overall quality of the BP measurements recorded. This can be used to assess the dataset and the device to find out the average yield of quality data from the device. First, this requires an estimate of the average expected number of events that could occur within a section, which is the average number of beats. The average measured heart rate of 79.2 bpm over the 90 seconds section would give the average number of events to be 119 for a section.

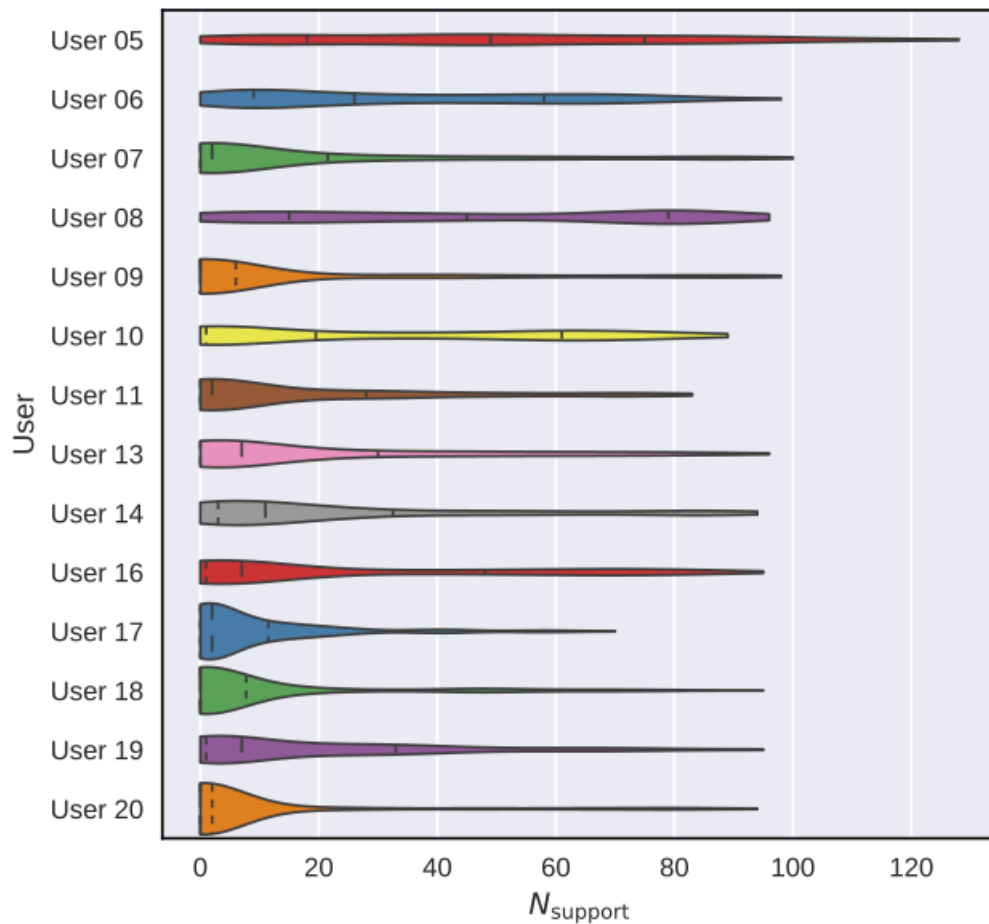


Figure 9.6: The distributions shown represent the densities of BP measurements with the given event support. The inter-quartile range is marked with the dashed lines to show the centres of mass and the spread of the measurements.

The quality of the signals recorded can be assessed across the users, as can be seen in figure 9.6. The graph shows the variation between the quality that the

users were able to record on the EIMO device, with User 5 exceeding the maximum value as their heart rate was higher than average and they have well-supported data sections as the distribution shows. Based on the expected average, approximately 19% of the 90-second window, or around 17 seconds on average was ‘Good’ helpful data. Some users were not able to record as much supported data as can be seen in the high densities between 0 and 20 supporting events ($0 \leq N_{\text{support}} \leq 20$) for users 11, 17, 18 and 20. Out of the BP measures that were taken 29% of the dataset lacked any ‘Good’ segments as estimated by the signal quality models from chapter 8. This allows a quantifiable assessment of the usability of this device in this context to gather data for estimating BP, where out of the 14 users assessed, the average support overall was 22 events, with a yield of 71% of the BP measurements were quality supported ($N_{\text{support}} > 0$).

9.4.3.2 Extracted Features and BP Measurement Relationships

The first comparison is used to examine the features and measurements looking for overall relationships from the whole dataset, because if accurate, a single model for all of the users would have the most predictive power. The BP measurements for the users have varying support based on the signal quality for each data section analysed. The features extracted need to be compared with the BP, while raising the amount of segment support for each of the measurements, to look for the trends as the data is cleaned.

Figure 9.7, figure 9.8 and figure 9.9, show where the measurements have been ranked and plotted against each of the features with varying support. The graphs on the left show 100% of all of the measurements taken from all three data sections. Each column raises the support bar, so only 50% and 25% of the most supported or strongest features are shown. The points are also sized and coloured using the normalised support for the group that they belong to, demonstrating that the larger and darker the marker is, the more support the feature has. Second-order polynomials have been fitted, in the best 50% and 25% supported graphs, to a randomly partitioned 50% of the data displayed. The model performance, as tested on the

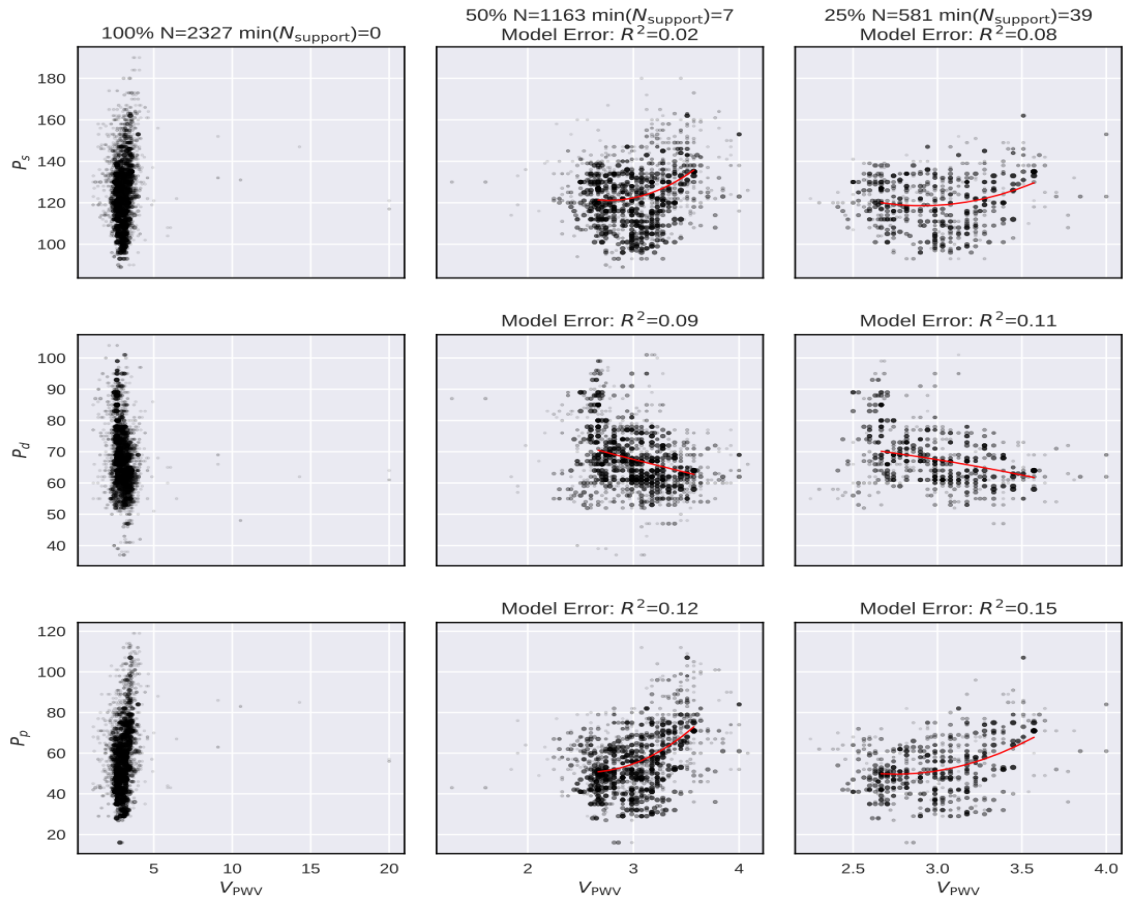


Figure 9.7: The axis columns show the effect of increasing the signal quality tolerance for the blood pressure (BP) measurements P_s , P_d , P_p versus pulse wave velocity V_{PWV} . The columns show 100%, 50% and 25% of the BP measurements with the column on the left, showing all of them and the column on the right showing 25% of them. The columns on the right show lines of best fit as calculated on 50% of the visible data, then these models are tested on the other 50% of the data. The marker size denotes the normalised number of supporting events for that measurement. N above the axis shows how many measurements are displayed on the axis. The size and opacity of the points are functions of the event support. The highly supported points are smaller and darker, showing there can be more certainty in the points position.

other 50% of the data displayed, is listed above the axis given as the R^2 score.

These graphs show that structures can be seen clearer within the dataset as the less supported measurements are removed. However, interestingly, the main relationships seen between the feature and measurements are not very strong with R^2 scores of a maximum of 0.09. There are structures within the clusters with darker linear paths through the cluster. Overall, this suggests that the modelling might be more effective on a per-user basis as well as on the whole dataset as discussed next.

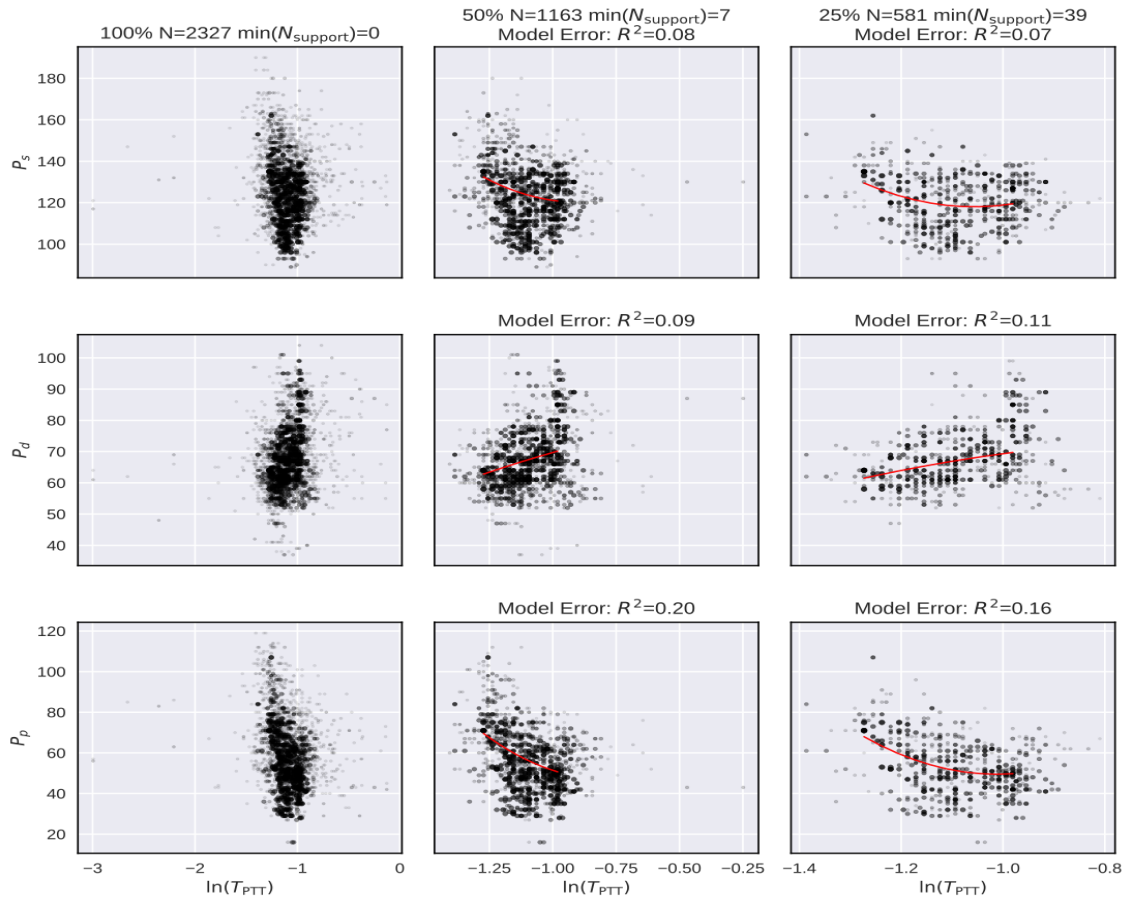


Figure 9.8: The axis columns show the effect of increasing the signal quality tolerance for the blood pressure (BP) measurements P_s , P_d , P_p versus logarithmic pulse transit time $\ln(T_{\text{PTT}})$. The columns show 100%, 50% and 25% of the BP measurements with the column on the left, showing all of them and the column on the right showing 25% of them. The columns on the right show lines of best fit as calculated on 50% of the visible data, then these models are tested on the other 50% of the data. The marker size denotes the normalised number of supporting events for that measurement. N above the axis shows how many measurements are displayed on the axis. The size and opacity of the points are functions of the event support. The highly supported points are smaller and darker, showing there can be more certainty in the points position.

9.4.3.3 Estimating The Blood Pressure

From the previous work above, it can be seen that we can select the features based on the relationships that are observable in the whole user dataset. There is very little difference between the two PTT-based features of $\ln(T_{\text{PTT}})$ and V_{PWV} , which have been derived and used in the literature separately as discussed in section 9.2.2. These features also need to be combined together as the models discussed above used a combination HR and the PTT features to estimate systolic and diastolic BP.

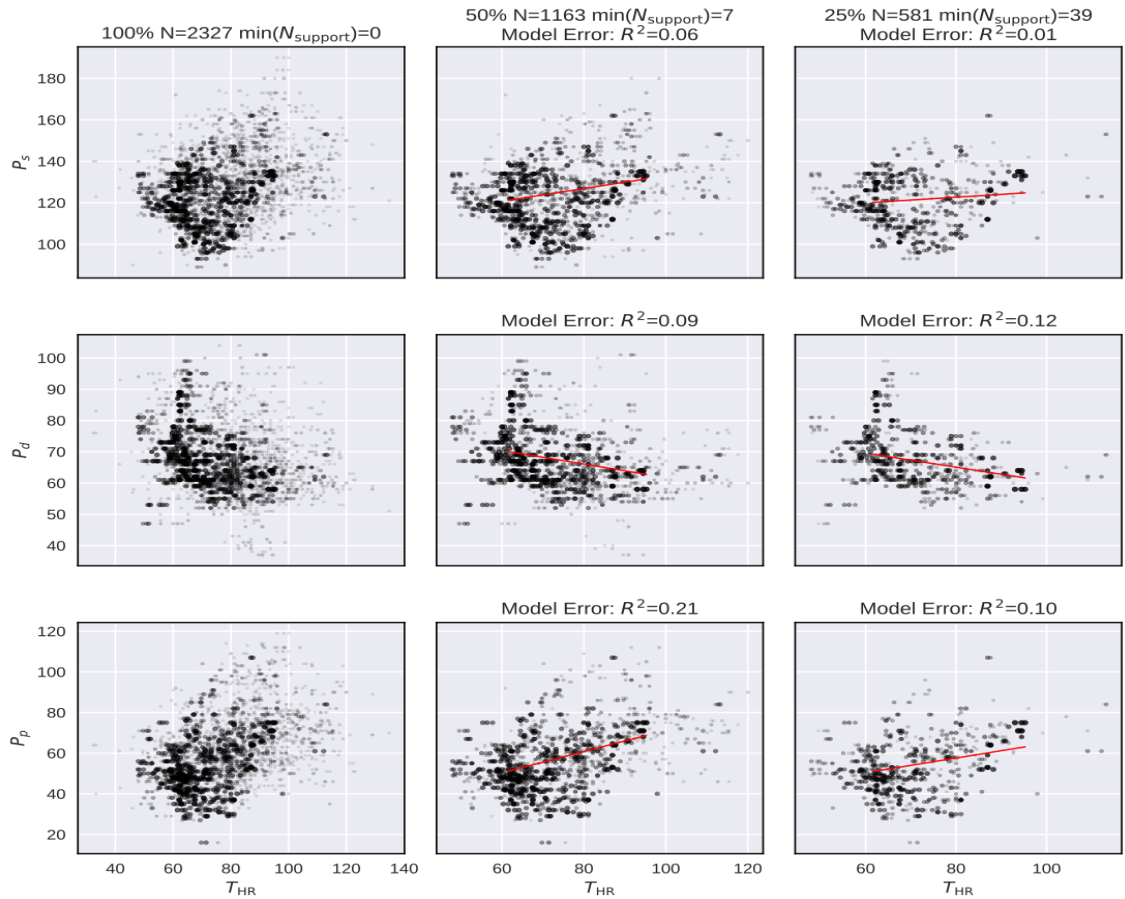


Figure 9.9: The axis columns show the effect of increasing the signal quality tolerance for the blood pressure (BP) measurements P_s , P_d , P_p versus heart rate T_{HR} . The columns show 100%, 50% and 25% of the BP measurements with the column on the left, showing all of them and the column on the right showing 25% of them. The columns on the right show lines of best fit as calculated on 50% of the visible data, then these models are tested on the other 50% of the data. The marker size denotes the normalised number of supporting events for that measurement. N above the axis shows how many measurements are displayed on the axis. The size and opacity of the points are functions of the event support. The highly supported points are smaller and darker, showing there can be more certainty in the points position.

Modelling analysis was performed in the testing framework described in sections 7.3 and 8.3 extending the system and analysis to regression models. The set of possible model relationships are listed below, using the test definitions from chapter 8 and equation (8.5). Defining the feature set is straight forward since there are only three features normally used in the literature. The tests Ψ_{iii} listed below in equations (9.12) to (9.15) are the common formulation of the features from the literature combined with simple second order linear regression to fit the data and allow for the curvature seen in the analysis above. This will be performed for the dataset as a whole to investigate a universal model and its performance and limitations.

Table 9.3: This table shows the model performance for example blood pressure (BP) estimate models, based on measurements with an event support $N_{\text{support}} \geq 0$ events. Number of measurements above threshold and used for analysis $N = 2327$.

Name	Features	Measures	R^2 Score	Mean	Std
Ψ_{a1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_s	-1.303	-0.929	21.418
Ψ_{a2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_s	-0.085	0.037	15.504
Ψ_{b1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_d	-0.866	0.641	13.561
Ψ_{b2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_d	0.028	0.078	10.260

Table 9.4: This table shows the model performance for example blood pressure (BP) estimate models, based on measurements with an event support $N_{\text{support}} \geq 22$ events. Number of measurements above threshold and used for analysis are $N = 803$.

Name	Features	Measures	R^2 Score	Mean	Std
Ψ_{a1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_s	0.024	-0.088	13.204
Ψ_{a2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_s	0.015	-0.090	13.282
Ψ_{b1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_d	0.218	0.006	7.793
Ψ_{b2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_d	0.225	-0.012	7.763

$$P_s = \Psi_{a1}(V_{\text{PWV}}, T_{\text{HR}}) \quad (9.12)$$

$$P_s = \Psi_{a2}(\ln(T_{\text{PTT}}), T_{\text{HR}}) \quad (9.13)$$

$$P_d = \Psi_{b1}(V_{\text{PWV}}, T_{\text{HR}}) \quad (9.14)$$

$$P_p = \Psi_{b2}(\ln(T_{\text{PTT}}), T_{\text{HR}})) \quad (9.15)$$

The model results shown in table 9.3 and table 9.4 display the mean and standard deviation of the error in the signal for the example models above. The models have been trained and tested on a four-way, K-fold cross-validation analysis using all of the raw measurement data for the first table and with a threshold of $N_{\text{support}} > 22$ events for the second. These tables show the performance of the cross-validated models. For the current dataset, it can be seen that as the event support rises, the standard deviation of the error σ_e measurement drops across all of the models. Also that Ψ_{a2} and Ψ_{b2} outperforms Ψ_{a1} and Ψ_{b1} for both systolic P_s and diastolic P_d BP respectively. This difference decreases for the models listed in table 9.4. To explore the relationship between the standard deviation of the error σ_e as the measurement event support increases. σ_e will be chosen to illustrate the difference for each model

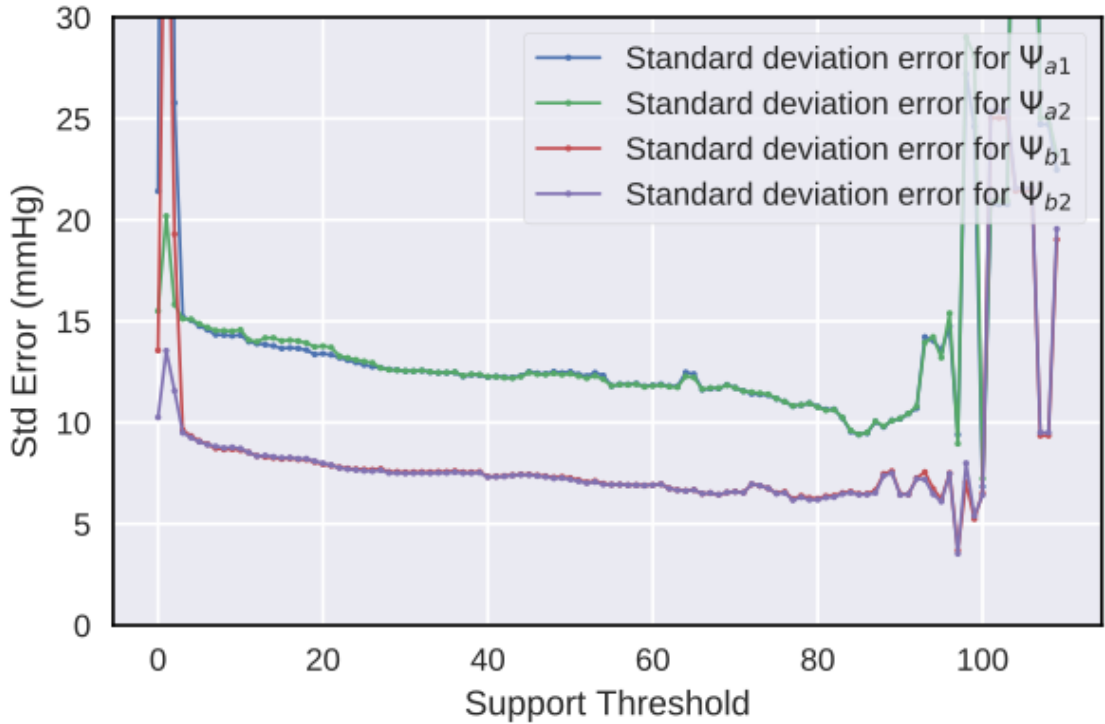


Figure 9.10: Modelling error verses support threshold for all users studied. The models were run on a K-fold validation while the event support threshold increases.

at each level of supporting events as the mean error \bar{e} is close to zero for all the models due to the optimisation algorithm. Figure 9.10 shows how the modelling error changes as the threshold increases. This illustrates that the error does indeed drop as the supporting events rise, but levels off (up to 85 events) where there are too few instances left to test the models, and performance of the cross-validation becomes erratic. A local optimum can be reached by removing noise (up to 20-30 events) but not going above 85 events so that the structure of the data is not lost.

This shows that when using one model the best standard deviation error one can gain with an average support level of 22 events is 13.204 mmHg and 7.76 mmHg for systolic and diastolic BP respectively. However, the error rates are very high. As discussed earlier the desired error rate to achieve are a mean of 5 mmHg and standard deviation of less than 8 mmHg, to reduce this error attention will now turn to the individual users.

Each users measure was taken and the same modelling structure was applied. Some users show an improved BP modelling result below the target threshold discussed above. The users with error rates under the target are shown in tables 9.5

Table 9.5: Model result performed on a 4 K-fold cross-validation scheme for User 6, on the full dataset and for a N_{support} of 22 which is the average for the dataset as a whole. N denotes the number of measure used for the modelling.

Support	Features	Measures	$N_{\text{support}} \geq 0$ N=161		$N_{\text{support}} \geq 22$ N=90	
Name			Mean	Std	Mean	Std
Ψ_{a1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_s	-1.532	9.415	-0.563	7.031
Ψ_{a2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_s	-2.097	11.357	-0.540	7.000
Ψ_{b1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_d	-1.591	8.470	0.451	5.086
Ψ_{b2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_d	-2.412	11.406	0.446	5.079

Table 9.6: Model result performed on a 4 K-fold cross-validation scheme for User 8, on the full dataset and for a N_{support} of 22 which is the average for the dataset as a whole. N denotes the number of measure used for the modelling.

Support	Features	Measures	$N_{\text{support}} \geq 0$ N=168		$N_{\text{support}} \geq 22$ N=114	
Name			Mean	Std	Mean	Std
Ψ_{a1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_s	18.871	73.589	-1.049	7.742
Ψ_{a2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_s	7.936	32.627	-1.113	7.829
Ψ_{b1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_d	-100.838	367.874	-0.684	4.950
Ψ_{b2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_d	-7.749	28.696	-0.712	4.967

to 9.8. These seem to favour a mean close to zero (maximum of 1.1 mmHg) and a standard deviation of 6.6-7.7 mmHg for systolic and 4.6-6.9 mmHg for diastolic BP. This puts the user's measurements under the target threshold set above, that could make EIMO an accurate personal BP measurement device based on these user's datasets.

Table 9.7: Model result performed on a 4 K-fold cross-validation scheme for User 10, on the full dataset and for a N_{support} of 22 which is the average for the dataset as a whole. N denotes the number of measure used for the modelling.

Support	Features	Measures	$N_{\text{support}} \geq 0$ N=170		$N_{\text{support}} \geq 22$ N=83	
Name			Mean	Std	Mean	Std
Ψ_{a1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_s	1.860	7.831	0.057	6.596
Ψ_{a2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_s	1.744	7.677	0.113	6.545
Ψ_{b1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_d	-3.397	11.926	0.074	6.944
Ψ_{b2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_d	-2.989	11.772	0.089	6.944

Table 9.8: Model result performed on a 4 K-fold cross-validation scheme for User 16, on the full dataset and for a N_{support} of 22 which is the average for the dataset as a whole.

Support Name	Features	Measures	$N_{\text{support}} \geq 0$ N=167		$N_{\text{support}} \geq 22$ N=57	
			Mean	Std	Mean	Std
Ψ_{a1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_s	0.516	10.858	-0.873	7.754
Ψ_{a2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_s	0.638	10.819	-0.828	7.770
Ψ_{b1}	$V_{\text{PWV}}, T_{\text{HR}}$	P_d	-0.428	7.381	-0.245	4.638
Ψ_{b2}	$\ln(T_{\text{PTT}}), T_{\text{HR}}$	P_d	-0.482	7.473	-0.243	4.641

9.4.4 Discussion

Determining the signal quality can play a valuable role in cleaning up the incoming data from the EIMO device. Allowing an informed reduction in spurious measurements has two effects. The first examines a need to investigate the underlying relationships between features and measurements more closely, while this allows the tools to do it. Secondly, once those measurements are established, they can be used as a benchmark for the future recordings of data to continuously reduce the ‘Bad’ features which can lead to ‘Bad’ estimation results. The nature of the optimal level in event clean-up illustrated above could be very useful for further investigations. Within most of the relationship graphs, there exists internal linear filaments and other structures. This leads on to the modelling of the BP estimation per user, discussed next.

Using simple models and structures, the error has been reduced from 15.5 mmHg to 13.1 mmHg for systolic BP and from 10.3 to 7.8 mmHg for diastolic by raising the event support to the average support for the dataset. By raising the threshold further, the error can be reduced further as per the graph above, where at the threshold of 82 events, the error drops to approximately 10 mmHg for systolic and 6 mmHg for diastolic BP. The two main models involving the logarithmic PTT mapping $\ln(T_{\text{PTT}})$ or the PWV V_{PWV} are seemingly very close to each other at high support levels, but the $\ln(T_{\text{PTT}})$ appears to be more stable when working with the full dataset as seen in table 9.3. The further analysis on the individual user models and the error estimation seen in tables 9.5 to 9.8 leads strongly to the conclusion

that users seem to have much clearer individual models with error rates within the targets set by White *et al.* (1993) and Takahashi *et al.* (2015) for personal devices. It would suggest that more work needs to be done to try to understand and predict the models a user might have, as this would improve the BP estimation while making it more convenient for the user.

9.5 Framework Data Flow Example

Like in the last chapter, this analysis was run within the data management framework. The graph created while the data was processed for the experiments here is illustrated in figure 9.11. This image shows a partial dependency graph that the framework builds as the blood pressure analysis was created and run. The green data preparation process components on the left start the data flow from the red raw signal, similarly for the signal quality analysis. The ‘Concatenate Worker Results’ that is seen in the signal quality data flow graph, seen in figure 8.7. This pulls the best models, identified from the signal quality analysis, so they can be reused and trained for the BP quality assessment and combined with the prepared data stored in the yellow nodes, seen in the centre cluster of the graph.

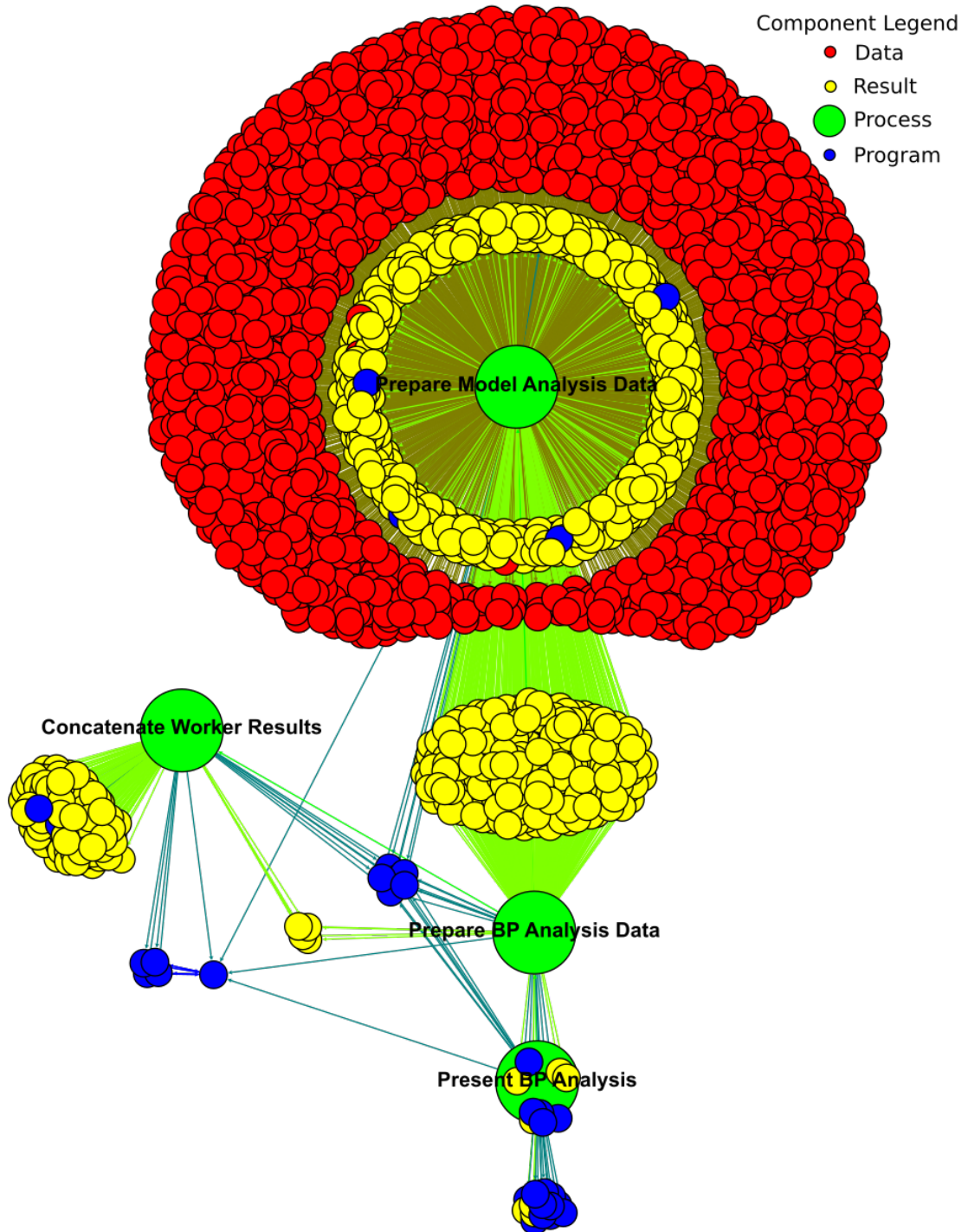


Figure 9.11: A partial blood pressure estimation framework dependency graph. The green nodes are the process components, yellow nodes mark result components, the blue shows program components and the red are the raw data nodes from the study. The lines show the linking between the nodes in the analysis structure. Due to the sheer number of nodes, only the process nodes are named.

9.6 Summary

The analysis above has shown that the the signal quality assessment can not only help to reduced the storage and computational load in a device by screening out the poor measurements but can also serve to clarify possible relationships. In the next chapter the overall results, limitations and suggestions for further work, from this and the previous chapters will be discussed.

10 Discussion and Future Work

10.1 Overview

The healthcare systems around us are under extreme pressure. New care models are being sought to find more efficient ways to better help more people. Patient-centric healthcare is now a possibility through the advances in electronics and mobile devices and artificial intelligence. There is now scope to be able to gather data conveniently and at volume. This data can be processed and used to train new, intelligent, machine-learning models that both use the data produced by the devices to learn, and use this information to produce estimates and recommendations for the specific user, taking their information and current conditions into consideration.

This thesis has shown that progress can be made on these aspects of patient-centric healthcare systems to contribute to making this form of care a reality. This has been addressed in two parts. (1) To develop a useful and straightforward data gathering device. (2) The data analysis using the data gathered had three tasks in mind:

- The first was to find ways to annotating the quality of the data recorded along with assessing those annotations seen in chapter 7.
- Second, to then be able to use the data management framework described in chapter 3 to leverage the annotations and analysis structure described in chapter 7 for the signal quality modelling as in chapter 8. These models are then used to further investigate blood pressure estimation, and to show that the signal quality can be used to improve the estimates and make the data and conclusions more accurate as seen in chapter 9.

- Lastly, both of the investigations not only serve as machine learning activities but also to illustrate the framework's operation in building up a picture of the data flow and keeping track of all the information and its connections as evidenced by section 8.6 and section 9.5.

These parts will be discussed next along with the current developments seen in each field, including limitations of the current work and ideas for further research.

10.2 Data Gathering

10.2.1 Discussion

EIMO has been developed to gather and record user data without the use of consumables and to make it straightforward to capture a snapshot of the users bio-signals for electrocardiograph (ECG) and photoplethysmography (PPG) as described in chapter 6. This device has since been CE mark approved and is being planned to be trialled in nursing homes and other care settings to investigate its impact and to gather information to further improve the device and its operation. For example, a measure of the user's respiration is being added to the system to facilitate the automation of the NEWS¹ monitoring system in hospitals. The current device has been compared to other monitoring systems in section 6.4.4. This has shown that the signal data recorded by the EIMO device is comparable, achieving correlations for ECG and PPG at over 0.43 and 0.93 respectively. The ECG recording is only a three lead recording device and is meant as a guide for the rate and rhythm of the heart. It can be seen that this device follows other devices closely in the types of measurements taken and yet is much easier and quicker to use and carry than others mentioned.

¹More information on the "National Early Warning Score" can be found at <<https://www.rcplondon.ac.uk/projects/outputs/national-early-warning-score-news>>. This is a system for the monitoring of patients that identify at-risk individuals for acute illness. Retrieved on: 2017/06/05

10.2.2 Current State

Other devices have been developed and released since EIMO was developed and tested, as interest has been growing in the realm of wearables. A recent paper by King *et al.* (2017) describe a wide range of devices that can be worn and used, to measure a range of information from vital sign monitors like EIMO through to sports and gait trackers. Challenges noted by the authors like data fusion and signal quality seem to be just as important today as they were when EIMO was started. The framework developed in this thesis, aims to make trialling new data fusion and processing techniques easy to try and benchmark by tracking improvements and differences between datasets and methods.

A Horizon 2020 funded project called Bitalino² are developing and selling sensor board platforms which can be rewired and redeveloped for wearable and custom sensing applications. Further platforms are offered for developers by a sister company called BioSignalsPlux³; both companies use the same underlying system but these are offered as a sensor package for researchers who want a finished platform. There are more sensors offered with the professional system with the addition of a PPG or blood volume pulse (BVP) sensor and a blood oxygen saturation (SPO2) sensor, which has recently been released. These platforms can log the sensor data off-line and also work with software they have developed called OpenSignals to run real-time and off-line visualisation and recording which run on the computer and mobile. These platforms all use standard sensing systems, with gel filled pads for the ECG and a full wrap around finger sensors for the PPG measurement. Neither of which are as convenient as the EIMO device as the ECG gel pads need to be replaced with every use, and wires get tangled. However the application software and sensors that they offer could be very useful for long term multi parameter monitoring as the device would have to stay attached to the user. EIMO was built to fill a separate niche where occasional monitoring is required as there are no pads or consumables, for an easy snapshot of a user with no fuss.

²More information can be seen at <<http://bitalino.com/en/>>. Retrieved on: 2017/06/06.

³More information on BioSignalsPlux can be found at <<http://biosignalsplux.com/en/>>. Retrieved on: 2017/06/06

10.2.3 Limitations and Future Work

The current device has the limitation, as mentioned earlier, that the final ECG morphology of recorded data is not that of a standard, wired, ECG monitor because the filters within the hardware and software remove more of the low frequency components in order to reduce the noise in the signal, than in other ECG monitors thus distorting the signal's shape. To correct this, a new analogue ECG front-end would need to be designed which was electrically shielded and has either a lower cut-off frequency for the signal filters or splits the low and high frequency components recording them separately, to then recombine the signal internally to rebuild the complete signal. The other limitation is the device's sensitivity to motion. This can be significant and impacts the quality of data recorded. The addition of an accelerometer would allow for motion compensation of those artefacts at the data source, therefore improving the recorded signal quality in adverse conditions.

10.3 Data Analysis

10.3.1 Discussion

Two forms of data analysis have been performed. The first was signal quality classification as described in chapters 7 and 8 and the second was blood pressure (BP) regression modelling, described in chapter 9. Both of these used the data gathering system and improved the quality and use-ability of this data.

The signal quality models were employed to remove the less reliable measurements from the BP modelling. This started by assessing the signal quality on individual sections of signal, then getting these assessments checked by clinicians and domain experts in order to peer-review the assessment. The data was then used to train machine learning models with various feature combinations to estimate the quality of the signal segments. The system for segmenting, describing and preparing the signal data was also defined, which was used for both the signal quality and BP because the system solved the general problems of parsing real-time signals into

event instances available for classification or regression machine-learning. While performing these assessments, interesting results have come to light for the stability of the PPG waveforms where the peak point of the signal can be both shown and verified to be more stable, in noisy PPG recordings.

The method of segmenting the signal and the limitation used for the features means that only four segments are required for analysis. These are the current and previous segments along with the examples of a ‘Good’ and ‘Bad’ signal segments. The features used allow the determination to be performed straight after the peak of the PPG signal has been received. This provides a low latency quality determination, which could be used to count the good signal segments received while recording the data on the device. This could then allow the system to spend a maximum of 90 seconds to gather a minimum of 20 good segments. This can provide more confidence in the final result of any estimates made, allowing the recording to stop early once 20 ‘Good’ segments have been acquired. Finally it could be used to identify people who are having trouble using the device by looking at the average quality of their recording as they are produced.

10.3.2 Limitations and Future Work

The different classification systems have different limitations which will be discussed separately.

10.3.2.1 Signal Quality Classification

The quality of the data was limited for the current model. The signal quality estimation could allow an estimate of the data to be computed. However, this could be employed at the device level to ensure higher quality data is recorded in the first place. This would lead to the stipulation of a minimum level of event support that an estimate would require before computing an estimated value.

The features used to provide the estimate were manually defined. To explore the extent to which the signal quality can be estimated, a deep-learning convolutional model might gain higher accuracies since they can find local and global features.

Other promising models are recurrent models, since the signal quality can be dependant on the segment sequence. The history, leading up to the loss of signal, could then be used to better classify the signal.

These classification scores have been achieved using only the raw signal data with no information from the fusion of the two signals. This was set up to allow the greatest applicability for the classification models presented. However, great improvement could be achieved for further work by looking at features arising from both of the signals that could be used to estimate the quality, missing portions or even regenerating the lost signals. This extension work would reduce the applicability as all the signals would need to be present for a particular scheme. However, it could greatly increase the accuracy of the estimates and possibilities for the future.

10.3.2.2 Blood Pressure Estimation

The BP measurement data gathered through the study certainly showed an embedded relationship between the features and the measurement, but there are internal structures visible in the graphs. The number of data points for each person are limited and they do not individually cover the whole relationship on their own, so a new study would be beneficial to take many measurements over a long period of time to gain measurements over the whole range of BP's in different circumstances. This could further reveal a deeper relationship and possibly enable better long-term modelling of a users cardiovascular system. Measuring other factors that influence their cardiovascular system, including their hydration, weight, fitness, stress levels etc, could also reveal other connections. The data from this study would mean that a more considered analysis could be undertaken using the same methodologies developed here but could find the full character of the relationship between the features and the BP measurements. Individual participants could also reveal some interesting results when compared side-by-side as per-user modelling improved the BP estimation in chapter 9 where it was suggested that determination of a relationship between these user models could make BP estimation easier and far more accurate, as there could be a way to estimate the model using the users characteristics such

as age, gender or weight.

10.3.2.3 Machine learning Schemes

The thesis has explored many schemes of machine learning. A current limitation is that the tests are manually prepared using the schemes in chapter 7, chapter 8 and chapter 9. The framework built has the scope to run many and varied processes and has been built with a view to run agent based or genetic evolutionary schemes for learners and features, where test sets could be built and analysed as a population of agents. This population could then be analysed for classification accuracy using some of the techniques used in the case study in chapter 8. If the analysis used regression performance, the techniques in chapter 9 could be reused. These libraries could be imported, or an analysis program could be triggered, using the process triggering mechanism. It is with this view that this architecture can allow self-organisation of machine learning schemes to achieve a deep learning type analysis using many other schemes other than pure neural networks only. Agent-led processes are envisioned for improvements to the framework itself. Model features, programs or other data processing agents could be evolved, within the framework designed.

10.4 Summary

The discussion of the two main elements in the second part of thesis have been explored, along with their limitations. From these, suggestions have been made for interesting further work to improve or extend the concepts that have been explored in this thesis. The next chapter concludes the thesis by comparing the discussions and the work with the aims of this thesis.

Part III

Conclusion and References

11 Conclusions

11.1 Overview

To review, the main issues addressed in this thesis discuss a need for better-integrated data gathering and management to support the tele-healthcare process. To deliver a patient-centred healthcare system successfully, there needs to be continuous improvement in data gathering devices, data management frameworks and in the tracking and management of the information through processing and analysis to inform this improvement. The main thesis contributions are explained in section 11.2. The thesis then finishes with the closing statement in section 11.3.

11.2 Thesis Contributions

The main contributions of this thesis are derived from the research aims as presented in chapter 1. The contributions towards each of the research aims will be presented next.

11.2.1 Aim 1 - Develop a Data Management and Analysis Framework.

This is developed in chapter 3 and is used for the experiments in the rest of the thesis. The main contributions made to this are outlined below.

1. A data management framework has been design and developed that can accept and process data from many different sources.

2. This framework can control and manage processes utilising them as data elements. The system automatically creates dependency and data flow graphs from running processes as they access data and programs within the graph database. The metadata from the process's data flow can be captured and interrogated by other processes in the architecture. This has been tested on the signal quality and blood pressure analyses as examples. This has greatly improved the scope for more complicated modelling architectures since all steps can be processed modularly and built upon accordingly, with full traceability of the data through to the results produced with a view to achieve self-organisation. Figure 8.7 shows the partial data flow of the signal quality analysis and figure 9.11 shows the blood pressure analysis data flow. These graphs show the generated data flows and overall ontologies of the data, program, process and result relationships, to illustrate the framework's operation.

11.2.2 Aim 2 - Application of the Framework for Data Capture and Analysis

This aim, using the framework developed in chapter 3 has captured the data recorded by a new portable device that measures medical vital signs described in chapter 6 and performed the analysis done in chapters 7 to 9. The contributions to this aim are below.

1. A simple vital sign capture device, by just holding the device with both hands, real time vital sign signals can be recorded similar to an ICU patient monitor. This requires no consumables such as ECG pads, wires or accessories. The device is flexibly powered and can be wirelessly connected to a tablet or phone to display the recordings directly to the user with low latency.
2. Architectures have been developed for the classification of signal quality from the independent signals themselves. An understanding of the stability of the signal cycles for the best placement of boundary events as described in sections 7.3 and 8.3.

3. Signal data has been annotated and checked to investigate how people assess the quality of signals and to establish a base line for signal quality as described in section 7.4.1. The average agreement was 84% and 90% on average for all of the assessors.
4. Four classes of object features totalling 37 features have been defined. These were used to examine the best feature groups and single features for categorising the signal quality of the ECG and PPG signals as described in section 8.4 with the analysis in section 8.5. The top models have a minimum performance of 85% and 81% sensitivity and specificity respectively for ECG and PPG. This is close to the same agreement that all of the assessors had with the original quality annotations for ‘Good’ segments and a little behind on ‘Bad’ segments for the PPG signal, so the signal quality models are comparable with the assessors, but require improvements for the performance on the PPG signal.
5. By using the cleaned signal data, blood pressure measurement models have been run with the accuracy being improved when utilising the signal quality system, as the event support metric as shown in section 9.4.3. The signal quality assessment can also be used to maintain a high level of quality data going into a model or by extension, allowing the collection of a requested amount of ‘Good’ quality data. For example, to improve data collection by collecting 20 ‘Good’ signal segments taking as long as required to achieve this, rather than a fixed 90 second recording, which may or may not have any usable data. Then basing the accuracy of the models around this level of ‘Good’ supporting events can have improved overall results.

11.3 Closing Statement

In concluding this thesis, it has been shown that the medical field is under pressure but there are options regarding optimal models for providing quality healthcare. The device presented here allows people to monitor their health in a proactive way by keeping records and making difficult recordings easier to achieve.

Automated data management can provide an enormous boost to streamline and standardise procedures, along with growing the capability for learning and implementation of new procedures and alleviating the pressures on the system. Data gathering and analysis has no panacea or cure all. However, the goal of this work is to highlight that change around and within a system should be embraced. The framework presented can aid change by allowing data flows and processing systems to be organised and new systems to interact and be used to automate new methodologies, systems or devices. This can help to improve and reduce the pressure on the health care system.

12 References

- Aboy, M., McNames, James, Thong, Tran, Tsunami, D., Ellenby, M.S., & Goldstein, B. 2005. An automatic beat detection algorithm for pressure signals. *Biomedical Engineering, IEEE Transactions on*, **52**(10), 1662–1670.
- Abt, Grant, Cheng, Yongqiang, Evans, Will, & Munnoch, Robert. 2015. The validity of EIMO: a cuffless blood pressure measurement device. Yorkshire Innovation Fund.
- Adeogun, O., Tiwari, A., & Alcock, J.R. 2011. Models of information exchange for UK telehealth systems. *International Journal of Medical Informatics*, **80**(5), 359 – 370.
- Al-Ali, A.R., Al-Rousan, M., & Al-Shaikh, M. 2003 (June). Embedded system-based mobile patient monitoring device. *Pages 355–360 of: Computer-Based Medical Systems, 2003. Proceedings. 16th IEEE Symposium.*
- Al Saiyd, N., Al Said, I., & Al Neaimi, A. 2009 (July). Towards an ontological concepts for domain-driven software design. *Pages 127–131 of: Networked Digital Technologies, 2009. NDT '09. First International Conference on.*
- Anliker, U., Ward, J.A., Lukowicz, P., Troster, G., Dolveck, F., Baer, M., Keita, F., Schenker, E.B., Catarsi, F., Coluccini, L., Belardinelli, A., Shklarski, D., Alon, M., Hirt, E., Schmid, R., & Vuskovic, M. 2004. AMON: a wearable multiparameter medical monitoring and alert system. *Information Technology in Biomedicine, IEEE Transactions on*, **8**(4), 415–427.

- Asif-Ul-Hoque, M., Ahsan, M. S., & Mohajan, B. 2011. Measurement of Blood Pressure Using Photoplethysmography. *Pages 32–35 of: Computer Modelling and Simulation (UKSim), 2011 UkSim 13th International Conference on.*
- Babbs, Charles F. 2012. Oscillometric measurement of systolic and diastolic blood pressures validated in a physiologic mathematical model. *BioMedical Engineering OnLine*, **11**(1), 56.
- Behar, J., Oster, J., Li, Q., & Clifford, G.D. 2012 (Sept). A single channel ECG quality metric. *Pages 381–384 of: Computing in Cardiology (CinC), 2012.*
- Behar, J., Oster, J., Li, Qiao, & Clifford, G.D. 2013. ECG Signal Quality During Arrhythmia and Its Application to False Alarm Reduction. *Biomedical Engineering, IEEE Transactions on*, **60**(6), 1660–1666.
- Berthold, Michael R., Cebron, Nicolas, Dill, Fabian, Gabriel, Thomas R., Kötter, Tobias, Meinl, Thorsten, Ohl, Peter, Sieb, Christoph, Thiel, Kilian, & Wiswedel, Bernd. 2007. KNIME: The Konstanz Information Miner. *In: Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*. Springer.
- Bhaskaran, P., Kaduskar, M., Saluja, P., Tallimani, S., Bhaumik, S., & Yoo, Sanghyun. 2012 (Oct). ForeTell: Facilitating doctor-patient conversation through interactive information visualization of risk prediction index. *Pages 1–4 of: Bioinformatics and Biomedicine (BIBM), 2012 IEEE International Conference on.*
- Boll, Friederike, & Brune, Philipp. 2015. User Interfaces with a Touch of Grey? Towards a Specific UI Design for People in the Transition Age. *Procedia Computer Science*, **63**, 511 – 516. The 6th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2015)/ The 5th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2015)/ Affiliated Workshops.
- Boser, Bernhard E., Guyon, Isabelle M., & Vapnik, Vladimir N. 1992. A Training Algorithm for Optimal Margin Classifiers. *Pages 144–152 of: Proceedings of the*

- Fifth Annual Workshop on Computational Learning Theory. COLT '92.* New York, NY, USA: ACM.
- Bradley, Andrew P. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, **30**(7), 1145 – 1159.
- Bramwell, J. Crighton, & Hill, A. V. 1922. The Velocity of the Pulse Wave in Man. *Proceedings of the Royal Society of London. Series B, Containing Papers of a Biological Character*, **93**(652), 298–306.
- Cattivelli, F. S., & Garudadri, H. 2009 (June). Noninvasive Cuffless Estimation of Blood Pressure from Pulse Arrival Time and Heart Rate with Adaptive Calibration. *Pages 114–119 of: 2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*.
- Chandola, Varun, Banerjee, Arindam, & Kumar, Vipin. 2007. *Outlier Detection: A Survey*. Tech. rept. Department of Computer Science and Engineering, University of Minnesota.
- Chandola, Varun, Banerjee, Arindam, & Kumar, Vipin. 2009. Anomaly Detection: A Survey. *ACM Comput. Surv.*, **41**(3), 15:1–15:58.
- Chang, Fay, Dean, Jeffrey, Ghemawat, Sanjay, Hsieh, Wilson C, Wallach, Deborah A, Burrows, Mike, Chandra, Tushar, Fikes, Andrew, & Gruber, Robert E. 2008. Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems (TOCS)*, **26**(2), 4.
- Chen, Z., Zhong, F., Yuan, X., & Hu, Y. 2016 (March). Framework of integrated big data: A review. *Pages 1–5 of: 2016 IEEE International Conference on Big Data Analysis (ICBDA)*.
- Chow, Li Sze, Rajagopal, Heshalini, & Paramesran, Raveendran. 2016. Correlation between subjective and objective assessment of magnetic resonance (MR) images. *Magnetic Resonance Imaging*, **34**(6), 820 – 831.

- Cichocki, Andrzej. 2014. Era of Big Data Processing: A New Approach via Tensor Networks and Tensor Decompositions. *CoRR*, **abs/1403.2048**.
- Clifford, G. D., Lopez, D., Li, Q., & Rezek, I. 2011 (Sept). Signal quality indices and data fusion for determining acceptability of electrocardiograms collected in noisy ambulatory environments. *Pages 285–288 of: 2011 Computing in Cardiology*.
- Cusack, Caitlin M, Pan, Eric, Hook, Julie M, Vincent, Adam, Kaelber, David C, & Middleton, Blackford. 2008. The value proposition in the widespread use of telehealth. *Journal of Telemedicine and Telecare*, **14**(4), 167–168.
- Dask Development Team. 2016. *Dask: Library for dynamic task scheduling*.
- DelliFraine, Jami L, & Dansky, Kathryn H. 2008. Home-based telehealth: a review and meta-analysis. *Journal of Telemedicine and Telecare*, **14**(2), 62–66.
- Deserno, Thomas M, Welter, Petra, & Horsch, Alexander. 2012. Towards a Repository for Standardized Medical Image and Signal Case Data Annotated with Ground Truth. *Journal of Digital Imaging*, **25**(2), 213–226.
- Deshpande, A, Khoja, S, McKibbin, A, & Jadad, A R. 2008. *Real-Time (Synchronous) Telehealth in Primary Care: Systematic Review of Systematic Reviews*. Tech. rept. Canadian Agency for Drugs and Technologies in Health.
- Ding, Xuemei, Li, Yuhua, Belatreche, Ammar, & Maguire, Liam P. 2014. An experimental evaluation of novelty detection methods. *Neurocomputing*, **135**(0), 313 – 327.
- Duan, L., Xiong, D., Lee, J., & Guo, F. 2006 (Oct). A Local Density Based Spatial Clustering Algorithm with Noise. *Pages 4061–4066 of: 2006 IEEE International Conference on Systems, Man and Cybernetics*, vol. 5.
- Duque, H., Montagnat, J., Pierson, J.M., Brunie, L., & Magninn, I.E. 2003 (May). DM2: a distributed medical data manager for grids. *Pages 606–611 of: Cluster Computing and the Grid, 2003. Proceedings. CCGrid 2003. 3rd IEEE/ACM International Symposium on*.

- Eijk, Martijn Van Der, Nijhuis, Frouke A.P., Faber, Marjan J., & Bloem, Bastiaan R. 2013. Moving from physician-centered care towards patient-centered care for Parkinson's disease patients. *Parkinsonism and Related Disorders*, **19**(11), 923 – 927.
- El-Sappagh, Shaker H., & El-Masri, Samir. 2014. A distributed clinical decision support system architecture. *Journal of King Saud University - Computer and Information Sciences*, **26**(1), 69 – 78.
- Elgendi, Mohamed. 2016. Optimal Signal Quality Index for Photoplethysmogram Signals. *Bioengineering*, **3**(4), 21.
- Epstein, Ronald M, & Street, Richard L. 2011. The Values and Value of Patient-Centered Care. *Annals of Family Medicine*, **9**(2), 100–103.
- Etemadi, M., Inan, O. T., Heller, J. A., Hersek, S., Klein, L., & Roy, S. 2016. A Wearable Patch to Enable Long-Term Monitoring of Environmental, Activity and Hemodynamics Variables. *IEEE Transactions on Biomedical Circuits and Systems*, **10**(2), 280–288.
- Farberow, Bonne, Hatton, Valerie, Leenknecht, Cindy, Goldberg, Lee R., Hornung, Carlton A., & Reyes, Bernardo. 2008. Caveat Emptor: The Need for Evidence, Regulation, and Certification of Home Telehealth Systems for the Management of Chronic Conditions. *American Journal of Medical Quality*, **23**(3), 208–214.
- Fawcett, Tom. 2006. An introduction to ROC analysis. *Pattern Recognition Letters*, **27**(8), 861 – 874. ROC Analysis in Pattern Recognition.
- Fielding, Roy Thomas. 2000. *Architectural Styles and the Design of Network-based Software Architectures*. Ph.D. thesis, University of California. AAI9980887 old code Fielding:2000:ASD:932295.
- Fox, John, Glasspool, David, Patkar, Vivek, Austin, Mark, Black, Liz, South, Matthew, Robertson, Dave, & Vincent, Charles. 2010. Delivering clinical decision support services: There is nothing as practical as a good theory. *Journal of Biomedical Informatics*, **43**(5), 831–843.

- Fu, Y., Bryan, H.S., Yang, I., & Lai, K. 2010 (Aug). Signal quality classification for an ambulatory monitoring system. *Pages 174–177 of: Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE.*
- Gambling, Tina, & Long, Andrew F. 2006. Exploring Patient Perceptions of Movement through the Stages of Change Model within a Diabetes Tele-care Intervention. *Journal of Health Psychology*, **11**(1), 117–128.
- Gambling, Tina, & Long, Andrew F. 2010. The realisation of patient-centred care during a 3-year proactive telephone counselling self-care intervention for diabetes. *Patient Education and Counseling*, **80**(2), 219 – 226.
- Gao, Junbin B., Gunn, Steve R., Harris, Chris J., & Brown, Martin. 2002. A probabilistic framework for SVM regression and error bar estimation. *Pages 71–89 of: Machine Learning.* MIT Press, Menlo Park.
- García-Lizana, Francisca, & Sarría-Santamera, Antonio. 2007. New technologies for chronic disease management and control: a systematic review. *Journal of Telemedicine and Telecare*, **13**(2), 62–68.
- Ghamari, Mohammad, Janko, Balazs, Sherratt, R. Simon, Harwin, William, Piechockic, Robert, & Soltanpur, Cinna. 2016. A Survey on Wireless Body Area Networks for eHealthcare Systems in Residential Environments. *Sensors*, **16**(6), 831.
- Goldberger, A. L., Amaral, L. A. N., Glass, L., Hausdorff, J. M., Ivanov, P. Ch., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., & Stanley, H. E. 2000. PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. *Circulation*, **101**(23), e215–e220. *Circulation Electronic Pages*: <http://circ.ahajournals.org/cgi/content/full/101/23/e215> PMID:1085218; doi: 10.1161/01.CIR.101.23.e215.
- Greenhalgh, Trisha, Wherton, Joe, Sugarhood, Paul, Hinder, Sue, Procter, Rob, & Stones, Rob. 2013. What matters to older people with assisted living needs?

- A phenomenological analysis of the use and non-use of telehealth and telecare. *Social Science and Medicine*, **93**(0), 86 – 94.
- Gómez-Verdejo, V., Arenas-Garcia, J., Lazaro-Gredilla, M., & Navia-Vazquez, A. 2011. Adaptive One-Class Support Vector Machine. *Signal Processing, IEEE Transactions on*, **59**(6), 2975–2981.
- Heard, Stephen O., Lisbon, Alan, Toth, Ildiko, & Ramasubramanian, Ramiah. 2000. An evaluation of a new continuous blood pressure monitoring system in critically ill patients. *Journal of Clinical Anesthesia*, **12**(7), 509–518.
- Hu, Han, Wen, Yonggang, Chua, Tat-Seng, & Li, Xuelong. 2014. Toward Scalable Systems for Big Data Analytics: A Technology Tutorial. *Access, IEEE*, **2**, 652–687.
- Huang, Guangxin, Chen, Huafu, Zhou, Zhongli, Yin, Feng, & Guo, Ke. 2011. Two-class support vector data description. *Pattern Recognition*, **44**(2), 320 – 329.
- Isebree Moens, A. (Adriaan). 1878. *Die Pulscurve*. Leiden, The Netherlands: Leiden : E.J. Brill. Scan in german.
- Isern, David, Moreno, Antonio, Sánchez, David, Hajnal, Ákos, Pedone, Gianfranco, & Varga, LászlóZ. 2011. Agent-based execution of personalised home care treatments. *Applied Intelligence*, **34**(2), 155–180.
- Jadooei, A., Zaderykhin, O., & Shulgin, V. I. 2013. Adaptive algorithm for continuous monitoring of blood pressure using a pulse transit time. *Pages 297–301 of: Electronics and Nanotechnology (ELNANO), 2013 IEEE XXXIII International Scientific Conference*.
- Jain, Anil K. 2010. Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, **31**(8), 651 – 666. Award winning papers from the 19th International Conference on Pattern Recognition (ICPR)19th International Conference in Pattern Recognition (ICPR).

- Jakoby, B., Vellekoop, M.J., Sardini, E., & Serpelloni, M. 2010. Eurosensor XXIV Conference Instrumented wearable belt for wireless health monitoring. *Procedia Engineering*, **5**, 580 – 583.
- Jiang, P., Winkley, J., Zhao, C., Munnoch, R., Min, G., & Yang, L. T. 2016. An Intelligent Information Forwarder for Healthcare Big Data Systems With Distributed Wearable Sensors. *IEEE Systems Journal*, **10**(3), 1147–1159.
- Karlen, W, Kobayashi, K, Ansermino, J M, & Dumont, G A. 2012. Photoplethysmogram signal quality estimation using repeated Gaussian filters and cross-correlation. *Physiological Measurement*, **33**(10), 1617.
- Karlen, W., Raman, S., Ansermino, J.M., & Dumont, G.A. 2013. Multiparameter Respiratory Rate Estimation From the Photoplethysmogram. *Biomedical Engineering, IEEE Transactions on*, **60**(7), 1946–1953.
- Kawamoto, Kensaku, Houlihan, Caitlin A, Balas, E Andrew, & Lobach, David F. 2005. Improving clinical practice using clinical decision support systems: a systematic review of trials to identify features critical to success. *BMJ*, **330**(7494), 765.
- King, Rachel C., Villeneuve, Emma, White, Ruth J., Sherratt, R. Simon, Holderbaum, William, & Harwin, William S. 2017. Application of data fusion techniques and technologies for wearable health monitoring. *Medical Engineering and Physics*, **42**, 1 – 12.
- Koch, Sabine. 2006. Home telehealth-Current state and future trends. *International Journal of Medical Informatics*, **75**(8), 565 – 576.
- Kononenko, Igor, & Kukar, Matjaz. 2007. *Machine Learning and Data Mining: Introduction to Principles and Algorithms*. Horwood Publishing Limited.
- Kumar, M., Singh, S., & Mahajan, S.C. 2012 (Dec). Computerized detection and classification of ECG signals. *Pages 126–130 of: Emerging Trends in Electrical Engineering and Energy Management (ICETEEEM), 2012 International Conference on*.

- Landau, L.D., Lifshitz, F.M., Sykes, J.B., & Reid, W.H. 1986. *Theory of elasticity*. 3rd edn. Oxford : Pergamon.
- Li, Q., & Clifford, G D. 2012. Dynamic time warping and machine learning for signal quality assessment of pulsatile signals. *Physiological Measurement*, **33**(9), 1491.
- Li, Qiao, Rajagopalan, Cadathur, & Clifford, Gari D. 2014. A machine learning approach to multi-level {ECG} signal quality classification. *Computer Methods and Programs in Biomedicine*, **117**(3), 435 – 447.
- Lu, Ssu-Hsuan, Lai, Kuan-Chou, Yang, Don-Lin, Tsai, Ming-Hsin, Li, Kuan-Ching, & Chung, Yeh-Ching. 2010 (July). Pervasive health service system: insights on the development of a grid-based personal health service system. *Pages 61–67 of: e-Health Networking Applications and Services (Healthcom), 2010 12th IEEE International Conference on*.
- Ma, Guoqiang, Liu, Juan, & Wei, Zhaoyu. 2010 (June). The Portable Personal Health Records: Storage on SD Card and Network, Only for One's Childhood. *Pages 4829–4833 of: Electrical and Control Engineering (ICECE), 2010 International Conference on*.
- Mahri, N., Beng, Gan Kok, & Ali, M.A.M. 2012 (Feb). Best pulse selection of photoplethysmography signal through comparative method. *Pages 389–392 of: Biomedical Engineering (ICoBE), 2012 International Conference on*.
- Martinez-Tabares, F.J., Espinosa-Oviedo, J., & Castellanos-Dominguez, G. 2012 (Aug). Improvement of ECG signal quality measurement using correlation and diversity-based approaches. *Pages 4295–4298 of: Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*.
- May, Carl, Mort, Maggie, Williams, Tracy, Mair, Frances, & Gask, Linda. 2003. Health technology assessment in its local contexts: studies of telehealthcare. *Social Science and Medicine*, **57**(4), 697 – 710.
- May, Carl, Finch, Tracy, Mair, Frances, & Mort, Maggie. 2005. Towards a wireless patient: Chronic illness, scarce care and technological innovation in the United

- Kingdom. *Social Science and Medicine*, **61**(7), 1485 – 1494. Building Trust and Value in Health Systems in Low- and Middle- Income Countries Building Trust and Value in Health Systems in Low- and Middle- Income Countries.
- May, Carl, Rapley, Tim, Moreira, Tiago, Finch, Tracy, & Heaven, Ben. 2006. Technogovernance: Evidence, subjectivity, and the clinical encounter in primary care medicine. *Social Science and Medicine*, **62**(4), 1022 – 1030.
- McCombie, D. B., Reisner, A. T., & Asada, H. H. 2006. Adaptive blood pressure estimation from wearable PPG sensors using peripheral artery pulse wave velocity measurements and multi-channel blind identification of local arterial dynamics. *Pages 3521–3524 of: Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE.*
- McWhinney, Ian R. 2003. *Patient-centered Medicine: Transforming the Clinical Method*. 2nd ed. edn. Patient-centered care series. Radcliffe Medical Press. Chap. The Evolution of Clinical Method, pages 17–30.
- Merat, Natasha, Lai, Frank, & Jamson, Samantha L. 2011. The comparative merits of expert observation, subjective and objective data in determining the effects of in-vehicle information systems on driving performance. *Safety Science*, **49**(2), 172 – 177.
- Moody, G.B., & Mark, R.G. 1996 (Sept). A database to support development and evaluation of intelligent intensive care monitoring. *Pages 657–660 of: Computers in Cardiology, 1996.*
- Moody, G.B., & Mark, R.G. 2001. The impact of the MIT-BIH Arrhythmia Database. *Engineering in Medicine and Biology Magazine, IEEE*, **20**(3), 45–50.
- Morgan, Stephanie, & Yoder, Linda H. 2012. A concept analysis of person-centered care. *J Holist Nurs*, **30**(1), 6–15.
- Munnoch, R., & Jiang, Ping. 2015 (July). A personal medical device for multi-sensor, remote vital signs collection in the elderly. *Pages 1122–1131 of: Science and Information Conference (SAI), 2015.*

- Neuman, M.R., Baura, G.D., Meldrum, S., Soykan, O., Valentinuzzi, M.E., Leder, R.S., Micera, S., & Zhang, Yuan-Ting. 2012. Advances in Medical Devices and Medical Electronics. *Proceedings of the IEEE*, **100**(Special Centennial Issue), 1537–1550.
- NHS. 2014 (November). *Telecare and telehealth technology*. Available Online at <<http://www.nhs.uk/Planners/Yourhealth/Pages/Telecare.aspx>> [Accessed: 25 May 2015].
- NICE. 2012 (February). *NICE quality standard for patient experience in adult NHS services*. Available Online at: <<https://www.nice.org.uk/guidance/qs15>> [Accessed: 24 May 2015].
- Nizami, S., Green, J.R., & McGregor, C. 2013. Implementation of Artifact Detection in Critical Care: A Methodological Review. *Biomedical Engineering, IEEE Reviews in*, **6**, 127–142.
- Ong, D., & Khaddaj, S. 2010 (June). Intelligent Framework for the Management of Distributed Architectures. *Pages 187–192 of: Software Engineering Artificial Intelligence Networking and Parallel/Distributed Computing (SNPD), 2010 11th ACIS International Conference on*.
- Orphanidou, C., Bonnici, T., Charlton, P., Clifton, D., Vallance, D., & Tarassenko, L. 2015. Signal-Quality Indices for the Electrocardiogram and Photoplethysmogram: Derivation and Applications to Wireless Monitoring. *Biomedical and Health Informatics, IEEE Journal of*, **19**(3), 832–838.
- Pantelopoulos, A., & Bourbakis, N.G. 2010. A Survey on Wearable Sensor-Based Systems for Health Monitoring and Prognosis. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, **40**(1), 1–12.
- Pimentel, Marco A.F., Clifton, David A., Clifton, Lei, & Tarassenko, Lionel. 2014. A review of novelty detection. *Signal Processing*, **99**(0), 215 – 249.

- Prasad, G.B., Seema, K., Shrikant, U.H., Garge, G.K., Anand, S.V.R., & Hegde, M. 2013 (Jan). SeaMo+: A virtual real-time multimedia service framework on handhelds to enable remote real-time patient monitoring for mobile doctors. *Pages 1–6 of: Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on.*
- Rocklin, Matthew. 2015. Dask: Parallel Computation with Blocked algorithms and Task Scheduling. *Pages 130 – 136 of: Huff, Kathryn, & Bergstra, James (eds), Proceedings of the 14th Python in Science Conference.*
- Roylance, David. 2001. *Pressure Vessels*. Ph.D. thesis, Department of Materials Science and Engineering, Massachusetts Institute of Technology, Cambridge.
- Saeed, Mohammed, Villarroel, Mauricio, Reisner, Andrew T, Clifford, Gari, Lehman, Li-Wei, Moody, George, Heldt, Thomas, Kyaw, Tin H, Moody, Benjamin, & Mark, Roger G. 2011. Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II): A public-access intensive care unit database. *Critical care medicine*, **39**(5), 952–960.
- Sagiroglu, S., & Sinanc, D. 2013 (May). Big data: A review. *Pages 42–47 of: Collaboration Technologies and Systems (CTS), 2013 International Conference on.*
- Shahid, N., Naqvi, I.H., & Bin Qaisar, S. 2012 (April). Quarter-Sphere SVM: Attribute and Spatio-Temporal correlations based Outlier & Event Detection in wireless sensor networks. *Pages 2048–2053 of: Wireless Communications and Networking Conference (WCNC), 2012 IEEE.*
- Shaltis, P. A., Reisner, A., & Asada, H. H. 2006. Wearable, Cuff-less PPG-Based Blood Pressure Monitor with Novel Height Sensor. *Pages 908–911 of: Engineering in Medicine and Biology Society, 2006. EMBS '06. 28th Annual International Conference of the IEEE.*

- Shin, Hang Sik, Lee, Chungkeun, & Lee, MyoungHo. 2009. Adaptive threshold method for the peak detection of photoplethysmographic waveform. *Computers in Biology and Medicine*, **39**(12), 1145–1152.
- Shriram, R., Wakankar, A., Daimiwal, Nivedita, & Ramdasi, D. 2010. Continuous cuffless blood pressure monitoring based on PTT. *Pages 51–55 of: Bioinformatics and Biomedical Technology (ICBBT), 2010 International Conference on.*
- Silva, I., Moody, G.B., & Celi, L. 2011 (Sept). Improving the quality of ECGs collected using mobile phones: The PhysioNet/Computing in Cardiology Challenge 2011. *Pages 273–276 of: Computing in Cardiology, 2011.*
- Silva, I., Lee, J., & Mark, R.G. 2012. Signal Quality Estimation With Multichannel Adaptive Filtering in Intensive Care Settings. *Biomedical Engineering, IEEE Transactions on*, **59**(9), 2476–2485.
- Sittig, Dean F., Wright, Adam, Osheroff, Jerome A., Middleton, Blackford, Teich, Jonathan M., Ash, Joan S., Campbell, Emily, & Bates, David W. 2008. Grand challenges in clinical decision support. *Journal of Biomedical Informatics*, **41**(2), 387–392.
- Smith, StevenM., & Vela, Edward. 2001. Environmental context-dependent memory: A review and meta-analysis. *Psychonomic Bulletin and Review*, **8**(2), 203–220.
- Smola, Alex J., & Schölkopf, Bernhard. 2004. A tutorial on support vector regression. *Statistics and Computing*, **14**(3), 199–222.
- Spyropoulos, V., Vasilakopoulou, C., & Kotidis, Y. 2016 (Dec). Digree: A middleware for a graph databases polystore. *Pages 2580–2589 of: 2016 IEEE International Conference on Big Data (Big Data).*
- Stewart, Moira, Brown, Judith Belle, W., Weston Wayne, McWhinney, Ian R., L., McWilliams Carol, & Freeman, Thomas R. 2003. *Patient-centered Medicine: Transforming the Clinical Method*. Patient-centered care series. Radcliffe Medical Press.

- Stoica, Ion, Morris, Robert, Karger, David, Kaashoek, M. Frans, & Balakrishnan, Hari. 2001. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. *SIGCOMM Comput. Commun. Rev.*, **31**(4), 149–160.
- Sukor, J Abdul, Redmond, S J, & Lovell, N H. 2011. Signal quality measures for pulse oximetry through waveform morphology analysis. *Physiological Measurement*, **32**(3), 369.
- Sun, Xuxue, Yang, Ping, & Zhang, Yuan-Ting. 2012 (Aug). Assessment of photoplethysmogram signal quality using morphology integrated with temporal information approach. *Pages 3456–3459 of: Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*.
- Tahat, A., Sacca, A., & Kheetan, Y. 2011 (Nov). Design of an integrated mobile system to measure blood pressure. *Pages 1–6 of: Communications and Vehicular Technology in the Benelux (SCVT), 2011 18th IEEE Symposium on*.
- Takahashi, Hakuo, Yoshika, Masamichi, & Yokoi, Toyohiko. 2015. Validation of two automatic devices for the self-measurement of blood pressure according to the ANSI/AAMI/ISO81060-2:2009 guidelines: the Omron BP765 (HEM-7311-ZSA) and the Omron BP760N (HEM-7320-Z). *Vascular Health and Risk Management*, **11**(jan), 49–53.
- Thung, K. H., & Raveendran, P. 2009 (Dec). A survey of image quality measures. *Pages 1–4 of: Technical Postgraduates (TECHPOS), 2009 International Conference for*.
- Tijsseling, A.S., & Anderson, A. 2012. *A. Isebreë Moens and D.J. Korteweg: on the speed of propagation of waves in elastic tubes / A.S. Tijsseling, A. Anderson*. Report.
- Varshney, Upkar. 2007. Pervasive Healthcare and Wireless Health Monitoring. *Mobile Networks and Applications*, **12**(2-3), 113–127.
- Verburgt, Lukas M. 2015. The objective and the subjective in mid-nineteenth-century British probability theory. *Historia Mathematica*, **42**(4), 468 – 487.

- Wang, Ruiping, Jia, Wenyan, Mao, Zhi-Hong, Sciabassi, Robert J, & Sun, Mingui. 2014. Cuff-Free Blood Pressure Estimation Using Pulse Transit Time and Heart Rate. *International conference on signal processing proceedings. International Conference on Signal Processing*, **2014**(oct), 115–118.
- Ward, Marcia M., Jaana, Mirou, & Natafqi, Nabil. 2015. Systematic review of telemedicine applications in emergency rooms. *International Journal of Medical Informatics*, –.
- White, W B, Berson, A S, Robbins, C, Jamieson, M J, Prisant, L M, Roccella, E, & Sheps, S G. 1993. National standard for measurement of resting and ambulatory blood pressures with automated sphygmomanometers. *Hypertension*, **21**(4), 504–9.
- Winkley, J., Jiang, P., & Jiang, W. 2012. Verity: an ambient assisted living platform. *Consumer Electronics, IEEE Transactions on*, **58**(2), 364–373.
- Witten, Ian H., Frank, Eibe, & Hall, Mark A. 2011. *Data mining: practical machine learning tools and techniques*. 3rd edn. Amsterdam; Boston, MA: Morgan Kaufman.
- Wu, F., Williams, M., Kazanzides, P., Brady, K., & Fackler, J. 2009 (April). A modular Clinical Decision Support System Clinical prototype extensible into multiple clinical settings. *Pages 1–4 of: Pervasive Computing Technologies for Healthcare, 2009. PervasiveHealth 2009. 3rd International Conference on*.
- Yang, Bin, Qian, Weining, & Zhou, Aoying. 2008. Using Wide Table to manage web data: a survey. *Frontiers of Computer Science in China*, **2**(3), 211–223.
- Zhang, Yatao, Wei, Shoushui, Long, Yutao, & Liu, Chengyu. 2015. Performance Analysis of Multiscale Entropy for the Assessment of ECG Signal Quality. *Journal of Electrical and Computer Engineering*, **2015**, 9.
- Zualkernan, I.A., & Shouman, M. 2008 (July). Towards Ontology-Driven Heuristic Assessment Generation for Software Design Patterns. *Pages 922–924 of: Advanced*

Learning Technologies, 2008. ICALT '08. Eighth IEEE International Conference on.

Part IV

Appendix

A Mathematical Notation

The tables A.1 to A.3 list useful mathematical notation from each chapter in the thesis, with descriptions for each term.

Table A.1: Useful mathematical notation used in Chapter 5.

Name	Description
TP	True Positives.
TN	True Negatives.
FP	False Positives.
FN	False Negatives.
U	Potential Energy.
\mathcal{K}	Elastic Constant (Hooks Constant).
E	Youngs Modulus.
Δ	Difference.
dr	Difference in the radius of the vessel.
r	Radius of the vessel.
A	Cross-sectional area of the vessel wall.
C	Circumference of the vessel (Length of the Spring).
dP	Difference in pressure.
\hat{P}	Peak pressure in the vessel.
\check{P}	Lowest pressure in the vessel.
SNR	Signal to noise ratio.
SNR_p	Peak signal to noise ratio.

Continued on next page

Table A.1 – continued from previous page

Name	Description
SNR_v	Valley signal to noise ratio.
U_n	Potential energy of the noise.
i_n	Current sample index.
i_s	Current segment index.
D_{i_s-i}	Decay constant at segment i_s , delayed by i segments.
τ	Slope rate.
$\hat{I}(i_s - 1)$	Peak amplitude at previous segment $i_s - 1$.
F_s	Sampling frequency.
std_{ppg}	The average standard deviation of the PPG signal.
$T_c(A, B)$	The instantaneous cycle time between segments A and B.
$T(\mathcal{E}(a), \mathcal{E}(b))$	The time between events at segment indexes a and b.
$I_{min}(i_s - 1)$	The estimated minimum amplitude for the previous event segment.
$P_G(i_s - 1)$	Amplitude on peak i_s from the good peak set.
$P_B(i_s - 1)$	Amplitude on peak i_s from the bad peak set.
H	Normalised decay height, the amount to drop by the next cycle time.
l	Look-back period. The number of index events to evaluate over.
$Th(n)$	Activation threshold at sample n , which a peak has to pass to become a good peak.
$\mathcal{E}(i_s)$	Set of event segments.
$S(i_s)$	Set of raw signal data segments.
$Q(e)$	Set of actual quality outcomes for the data segments.
e	Time to look for an quality assignment.

Continued on next page

Table A.1 – continued from previous page

Name	Description
h_d	The descriptive horizon offset moving assigned events into the past.
h_p	The predictive horizon offset moving assigned events into the future.
$D(i_s)$	Set of data descriptions for the segments.
$A(i_s)$	Set of actual truths values for the segments.
C_e	The final classed output set.
O_e	The continuous estimated output set.
ϵ_c	The threshold at which partition the continuous output of an estimate into two classes.

Table A.2: Useful mathematical notation used in Chapter 6.

Name	Description
$*_x, *_x$	Local variable to illustrate operators and functions.
$*X, *Y$	Local set to illustrate operators and functions.
W	Weight Vector for a linear decision function.
i_{fc}	Index for the feature classes.
i_{fi}	Integer index for the features with a class.
i_{mc}	Index for the model classes.
i_{mi}	Integer index for the models with a class.
i_{ti}	Integer index for the tests.
$F_{class}(i_{fc})$	The set of features in class type i_{fc} .
$f_{i_{fc}, i_{fi}}$	Function mapping the signal $S(i_s)$ into features i_{fi} belonging to the feature class i_{fc} .

Continued on next page

Table A.2 – continued from previous page

Name	Description
$F_{i_{fc}, i_{fi}}$	Feature set containing i_{fi} belonging to the feature class i_{fc} .
$\phi_{i_{fs}}(i_s)$	Is a feature set selection for feature set i_{fs} for the segment i_s .
$M_{i_{mc}, i_{mi}}$	A learner consisting of a modelling function of class i_{mc} indexed at i_{mi} including setup parameters.
$\Psi_{i_{ti}}$	Is a test combination of a feature set $\phi_{i_{fs}}(i_s)$ and a model $M_{i_{mc}, i_{mi}}$ at index i_{ti} .
i_s	Current event segment index.
i_l	Segment look back index.
$\mathcal{E}(i_s)$	Set of event segment descriptions indexed at segment i_s .
$S(i_s, i_n)$	Set of raw signal data segments for segment i_s at sample i_n .
$\tilde{S}(i_s, i_n)$	Set of medical signal data segments for segment i_s at sample i_n .
$R(i_s, i_n)$	Set of residual signal data segments for segment i_s at sample i_n .
$S'(i_s, i_n)$	First sample differential of the signal $S(i_s)$ at sample i_n .
$S_g(0)$	First 'Good' template stored.
$S_b(0)$	First 'Bad' template stored.
$P(*_x)$	A function for selecting features from the feature description $E(i_s)$ given a features $*_x$.
w	An integer for the window of the median function.
$Median(*_X, w)$	An operator to find the median of set $*_X$.
$max(*_X)$	An operator to find the maximum of a set $*_X$.
$min(*_X)$	An operator to find the minimum of a set $*_X$.
$std(*_X)$	An operator to find the standard deviation or 2nd order moment of the set $*_X$.

Continued on next page

Table A.2 – continued from previous page

Name	Description
$skew(*_X)$	An operator to find the skew or 3rd order moment of the set $*_X$.
$kurtosis(*_X)$	An operator to find the skew or 4th order moment of the set $*_X$.
$T(*_X, *_y)$	An operator to find the event time of the event set $*_X$ at segment $*_y$.
$E[*_X]$	The expected value operator (normally the arithmetic mean) of the event set $*_X$.
$correlation(*_X, *_Y)$	Pearson correlation coefficient between the sets $*_X, *_Y$.
$zero_crossing(*_X)$	An operator to count the number of times the set $*_X$ crosses its own mean.
$ *_X $	The count or cardinality of the set $*_X$.
$pdf(*_X, *_y)$	The bin histogram containing the probability density function (PDF) of the signal set $*_X$ at bin $*_y$.
$H(*_X)$	Is the entropy of the signal set $*_X$,

Table A.3: Useful mathematical notation used in Chapter 7.

Name	Description
E	Youngs Modulus.
E_0	Baseline Youngs Modulus in the vessel.
α	Exponential factor between starting E_0 to E
ρ	Blood density
L	Length of the vessel.
d	Diameter of the vessel.

Continued on next page

Table A.3 – continued from previous page

Name	Description
i_s	The integer index of the segment events.
T_{PTT}	Pulse transit time between the peak of the ECG and the peak of the PPG.
V_{PWV}	Pulse wave velocity the inverse of the T_{PTT} .
$\ln(T_{PWV})$	Pulse wave velocity the inverse of the T_{PTT} .
N_{support}	Number of 'Good' supporting events pairs of an ECG segment followed by PPG segment recorded for that measurement.
P	Pressure within the vessel.
P_s	Systolic pressure (maximum) within the vessel.
P_d	Diastolic pressure (minimum) within the vessel.
P_p	Pulse pressure (Pulse Pressure Height) from systolic to diastolic within the vessel.

B Device Logs

B.1 Data Logs for devices

The devices described are built to record and log data to the raw log files, which can be seen below along with the field information for these logs. The first is the EIMO device in appendix B.1.1, then the CM400 device in appendix B.1.2. The third device is the Case-GE which exports its own log file in XML format. The ECG signal information can be found in that file by using the XML labels in the file. The file is not included here as it is very large to include verbatim and the data is held in a compacted binary form. Following this in appendix B.1.3 are samples of the unified log files produced when the devices were processed by the unification software described in section 6.4.3.1.

The log files are organised into two parts, which can be seen in log file listings B.1 and B.2 and in the unified logs shown in log file listings B.3 to B.5. The top part of the file is a JSON formatted object containing the summaries of the data recorded and answers to any question asked by the application. The JSON format was chosen to allow easy reading and flexible data storage in this area as the notation allows for many kinds of field types and values. The bottom part is a CSV started with the column names. They can be split programatically by searching for the line with “Timestamp” which should always be the first column. The CSV format creates a human readable data log without using any more characters than necessary to separate the values. As data comes in, this can be simply appended to keep the processing overhead low.

B.1.1 EIMO Mobile Monitor

The EIMO device creates a log file shown in appendix B.1.1.1 which is internally split into two parts. The field ranges are given in appendix B.1.1.2.

B.1.1.1 Raw Log File Listing

The log file produced by the EIMO device is shown in log file listing B.1. Only the top of the file is included as the CSV part can run on for many samples.

```
{
  "Program" : "iPad",
  "Version" : "1.0.7",
  "DeviceAddress" : "A370B607-A0DE-6C26-B1E8-B9FC35CF877A",
  "Logstarttime" : "2015-05-01-09-36-28",
  "Name" : "YIF008_S1_2",
  "Sex" : "",
  "Age" : "",
  "SP02" : "91%",
  "Diastolic BP" : "40",
  "PTT" : "367",
  "Weight" : "",
  "Height" : "",
  "Activity" : "",
  "Systolic BP" : "96",
  "Hypertension" : "",
  "Heartrate" : "70",
  "Smoking" : "",
  "Notes" : "screen froze battery slipped"
}
Timestamp,ECG,PPG,SP02,PTT,HR,seq
2015-05-01-09-36-28.251,120,182,91,367,70,55060
2015-05-01-09-36-28.261,126,182,91,367,70,55061
2015-05-01-09-36-28.271,124,180,91,367,70,55062
...
```

Log File B.1: Log for the raw EIMO file

B.1.1.2 Field Descriptions

The log file above has a flexible JSON object for storing data and a more compact CSV area. The fields in these areas are explained in detail below. First the JSON object fields are defined in table B.1. Second, the CSV columns are defined in table B.2.

Table B.1: Field description for the log file and data produced by the EIMO device as event data and summaries.

Field	Type	Range	Notes
Program	string	{'ipad','pc'}	Value set by the software logging the data.
Version	string	'MM,mm,PP'	MM - Major revision, mm - Minor revision, PP - patch number respectively.
DeviceAddress	string	MAC or GUID	Unique address for the EIMO device to track the device that captured the recorded data.
Logstarttime	string	'YYYY-MM-DD-HH-mm-ss'	Full timestamp when the log button was pressed.
Name	string	{'Joe Blogs' or 'Blogs, Joe' or 'User001'}	Username or real name.
Sex	string	{'m','f'}	Male/Female selection.
Age	string	(0,120)	User entered age.
SPO2	string	{'0%','100%'}	Average blood oxygen saturation measurement over all recorded data in log (Percent).
Systolic BP	string	(0,250)	Average systolic blood pressure over all recorded data in log (mmHg).
Diastolic BP	string	(0,250)	Average diastolic blood pressure over all recorded data in log (mmHg).
PPT	string	(0,800)	Average pulse transmit time over all recorded data in log (Milliseconds).
Heartrate	string	(0,250)	Average heart rate over all recorded data in log (bpm).
Height	string	(0,300)	Recorded height (cm).
Weight	string	(0,300)	Recorded weight (Kg).
Activity	string	{'High','Med','Low'}	User entered apparent level of activity.
Hypertension	string	{'y','n'}	User entered answer, whether they have hypertension.
Smoking	string	{'y','n'}	User entered answer, whether they smoke.

Table B.2: Field description for the log file and data produced by the EIMO device as a time series of logged samples.

Field	Type	Range	Notes
Timestamp	DateTime	YYYY-MM-DD-HH-mm-ss.ssssss	Sample timestamp incrementing in 10mS as the frequency is 100Hz.
ECG	Integer	(0,255)	Sample ECG readings averaging 128 from the EPICs at 100Hz.
PPG	Integer	(0,255)	Sample PPG readings averaging 128 from the right hand PPG sensor at 100Hz.
SPO2	Integer	(0,100)	Current blood oxygen in percent concentration readings at 1Hz.
PTT	Integer	(0,1000)	Current pulse transmit time readings at 1Hz in milliseconds (mS).
HR	Integer	(0,220)	Current heart rate readings at 1Hz in beats-per-minute (bpm).
seq	Integer	(n)	Sequential sample counter.

B.1.2 CM400 Patient Monitor

The CM400 device creates a log file shown in appendix B.1.2.1 which is internally split into two parts. The field ranges are given in appendix B.1.2.2.

B.1.2.1 Raw Log File Listing

The log file produced by the CM400 interface device is shown in log file listing B.1. Only the top of the file is included as the CSV part can run on for many samples.

```
{
  "DeviceAddress": "CM400",
  "Version": "1.0.3",
  "Program": "CM400_Interface",
  "Name": "YIF008_S1_2",
  "Sex": "M",
  "Age": "",
  "HeartRate": "",
  "SP02": "",
  "PTT": "",
  "Systolic BP": "",
  "Diastolic BP": "",
  "Smoking": "",
  "Hypertension": "",
  "Weight": "81.7",
  "Height": "180.6",
  "Activity": "",
  "Notes": "second rest phase"
}
Timestamp,I,II,III,aVR,aVL,aVF,V,PPG,SP02,PTT,HR,seq
2015-05-01-09-36-24.610,0.096,0.176,0.079,-0.136,0.009,0.129,0.248,18.000,99.0,0.486,64.0,42880
2015-05-01-09-36-24.612,0.104,0.186,0.081,-0.144,0.012,0.134,0.265,17.000,99.0,0.486,64.0,42881
2015-05-01-09-36-24.614,0.106,0.201,0.094,-0.153,0.007,0.148,0.437,17.000,99.0,0.486,64.0,42882
...
```

Log File B.2: Log for the raw CM400 file

B.1.2.2 Field Descriptions

The log file above has a flexible JSON object for storing data and a more compact CSV area. The fields in these areas are explained in detail below. First the JSON object fields are defined in table B.3. Second, the CSV columns are defined in table B.4.

Table B.3: Field description for the log file and data produced by the CM400 patient monitor as event data and summaries.

Field	Type	Range	Notes
Program	string	{'pc'}	Value set by the software logging the data.
Version	string	'MM,mm,PP'	MM - Major revision, mm - Minor revision, PP - patch number respectively.
DeviceAddress	string	MAC or GUID	Unique address for the EIMO device to track the device that captured the recorded data.
Logstarttime	string	'YYYY-MM-DD-HH-mm-ss'	Full timestamp when the log button was pressed.
Name	string	{'Joe Blogs' or 'Blogs, Joe' or 'User001'}	Username or real name.
Sex	string	{'m', 'f'}	Male/Female selection.
Age	string	(0,120)	User entered age.
SPO2	string	{'0%', '100%'}	Average blood oxygen saturation measurement over all recorded data in log (Percent).
Systolic BP	string	(0,250)	Average systolic blood pressure over all recorded data in log (mmHg).
Diastolic BP	string	(0,250)	Average diastolic blood pressure over all recorded data in log (mmHg).
PPT	string	(0,800)	Average pulse transmit time over all recorded data in log (Milliseconds).
Heartrate	string	(0,250)	Average heart rate over all recorded data in log (bpm).
Height	string	(0,300)	Recorded height (cm).
Weight	string	(0,300)	Recorded weight (Kg).
Activity	string	{'High', 'Med', 'Low'}	User entered apparent level of activity.
Hypertension	string	{'y', 'n'}	User entered answer, whether they have hypertension.
Smoking	string	{'y', 'n'}	User entered answer, whether they smoke.

Table B.4: Field description for the log file and data produced by the CM400 patient monitor as a time series of logged samples.

Field	Type	Range	Notes
Timestamp	DateTime	YYYY-MM-DD-HH-mm-ss.ssssss	Sample timestamp incrementing in 2mS as the frequency is 500Hz.
I, II, III, aVR, aVL, aVF, V	Integer	(-32.767, 32.768)	Sample ECG readings averaging 0 from the EPICs at 500Hz.
PPG	Integer	(0,255)	Sample PPG readings averaging 128 from the right hand PPG sensor at 60Hz up sampled and synced with the ECG sample rate.
SPO2	Integer	(0,100)	Current blood oxygen in percent concentration readings at 1Hz.
PTT	Integer	(0,1000)	Current pulse transmit time readings at 1Hz in milliseconds (mS).
HR	Integer	(0,220)	Current heart rate readings at 1Hz in beats-per-minute (bpm).
seq	Integer	(n)	Sequential sample counter.

B.1.3 Unified Log File Listing

The unification files have the signal range and ontological signal information included in the JSON object. This allows programs reading this file better knowledge of the possible ranges of the signals included. Also, the signal summaries in the JSON object are calculated and filled in if not already given. This allows the unified log to be used by using the JSON summaries.

B.1.3.1 Unified Log File For The EIMO Mobile Monitor.

A sample unified log file for the EIMO device is shown in log file listing B.3. It shows the JSON object and CSV samples set. This layout is the full version of the same log file shown in log file listing B.1.

```

{
  "Activity": "",
  "Age": "",
  "DeviceAddress": "A370B607-A0DE-6C26-B1E8-B9FC35CF877A",
  "Diastolic BP": "40",
  "Heartrate": "70",
  "Height": "",
  "Hypertension": "",
  "Logstarttime": "2015-05-01-09-36-28.000000",
  "Name": "YIF008_S1_2",
  "Notes": "screen froze battery slipped",
  "PTT": "367",
  "Program": "iPad",
  "SP02": "91%",
  "Sex": "",
  "Signal_Limits": {
    "ECG": [ 0, 1 ],
    "HR": [ 0, 300 ],
    "PPG": [ 0, 1 ],
    "SP02": [ 0, 100 ]
  },
  "Signal_Ontology": {
    "ECG": [ "ECG" ],
    "PPG": [ "PPG" ]
  },
  "Smoking": "",
  "Systolic BP": "96",
  "Version": "1.0.7",
  "Weight": ""
}
Timestamp,ECG,PPG,SP02,PTT,HR,seq
2015-05-01-09-36-28.251000,0.471,0.714,91.0,0.367,70.0,55060
2015-05-01-09-36-28.261000,0.494,0.714,91.0,0.367,70.0,55061
2015-05-01-09-36-28.271000,0.486,0.706,91.0,0.367,70.0,55062
...

```

Log File B.3: Log for the unified EIMO mobile monitor file.

B.1.3.2 Unified Log File For The CM400 Patient Monitor.

A sample unified log file for the CM400 device is shown in log file listing B.4. It shows the JSON object and CSV samples set. This is the full version of the same log file shown in log file listing B.2.

```
{
  "Activity": "",
  "Age": "",
  "DeviceAddress": "CM400",
  "Diastolic BP": "",
  "HeartRate": "",
  "Height": "180.6",
  "Hypertension": "",
  "Logstarttime": "2015-05-01-09-36-24.610000",
  "Name": "YIF008_S1_2",
  "Notes": "second rest phase",
  "PTT": "",
  "Program": "CM400_Interface",
  "SP02": "",
  "Sex": "M",
  "Signal_Limits": {
    "ECG": [ -32.767, 32.768 ],
    "HR": [ 0, 300 ],
    "PPG": [ 0, 1 ],
    "SP02": [ 0, 100 ]
  },
  "Signal_Ontology": {
    "ECG": [ "I", "II", "III", "aVR", "aVL", "aVF", "V" ],
    "PPG": [ "PPG" ]
  },
  "Smoking": "",
  "Systolic BP": "",
  "Version": "1.0.3",
  "Weight": "81.7"
}
Timestamp,I,II,III,aVR,aVL,aVF,V,PPG,SP02,PTT,HR,seq
2015-05-01-09-36-24.610000,0.096,0.176,0.079,-0.136,0.009,0.129,0.248,0.142,99.0,0.486,64.0,42880
2015-05-01-09-36-24.612000,0.104,0.186,0.081,-0.144,0.012,0.134,0.265,0.134,99.0,0.486,64.0,42881
2015-05-01-09-36-24.614000,0.106,0.201,0.094,-0.153,0.007,0.148,0.437,0.134,99.0,0.486,64.0,42882
...
```

Log File B.4: Log for the unified CM400 patient monitor file.

B.1.3.3 Unified Log File For The Case GE Exercise Monitor.

A sample unified log file for the Case-GE exercise monitor is shown in log file listing B.5. It shows the JSON object and CSV samples set. The XML raw file has been processed by the unification software and converted into the same hybrid log file as above. The data is now in the CSV timeseries portion, just the top is sufficient to show the style of the log file produced. The signal ranges for the Case-GE device are included. The sample times have been calculated from the XML file structure as this information isn't directly available from the raw XML files.

```
{
  "Activity": "",
  "Age": "",
  "DeviceAddress": "CASE_GE",
  "Diastolic BP": "",
  "HeartRate": "",
  "Height": "",
  "Hypertension": "",
  "Logstarttime": "2015-05-01-09-36-13.000000",
  "Name": "",
  "Notes": "",
  "PTT": "",
  "Program": "CASE_GE",
  "SP02": "",
  "Sex": "F",
  "Signal_Limits": {
    "ECG": [ -255, 255 ]
  },
  "Signal_Ontology": {
    "ECG": [ "I", "II", "III", "aVR", "aVL", "aVF" ]
  },
  "Smoking": "No",
  "Systolic BP": "",
  "Version": "1.0.0",
  "Weight": ""
}
Timestamp,I,II,III,aVR,aVL,aVF
2015-05-01-09-36-13.000000,-39.000,-120.000,-81.000,80.000,21.000,-100.000
2015-05-01-09-36-13.005000,-36.000,-118.000,-80.000,77.000,23.000,-99.000
2015-05-01-09-36-13.010000,-36.000,-117.000,-82.000,77.000,23.000,-100.000
...
```

Log File B.5: Log for the unified Case-GE exercise monitor file.

C Study Protocol Information

C.1 Device Study

The study was originally funded by the Yorkshire Innovation Forward (YIF) program, which was completed in 2015. The study was designed and carried out by the Sport, Health and Exercise Science Group (SHES, University of Hull) with help from the author. The purpose was to compare and validate the data gathered by the EIMO device with two other medical data capture devices (Case-GE and CM400), more information can be found in the report produced by Abt *et al.* (2015). Department of Sport, Health and Exercise Science Ethics Committee (Application number: 1415218) approved the protocol and all experimental procedures conformed to the Declaration of Helsinki, as was originally described in the report for the YIF program (Abt *et al.* , 2015). Briefly the study protocol and information included:

- Volunteer participants that were required to be apparently healthy, non-smokers, aged between 18-50 years old.
- 20 participants (13 males and 7 females) started the study.
- 18 participants (12 males and 6 females) went on to complete the study.
- Each participant attended four sessions each separated by 24 hours.
 - Before each session, they were asked:
 - * To refrain from caffeine and alcohol consumption for 24 hours prior to testing.
 - * To refrain from un-habituated exercise for 24 hours prior to testing.
 - * To maintain normal hydration by drinking to thirst.

* To consume a light carbohydrate meal no less than 2 hours prior to testing

- The first session was for device familiarisation.
- The next three session consisted of:
 - 10 minute setup
 - two 12 minute recordings while resting,
 - one 45 minute recording while exercising on a recumbent bike at 25, 50, and 75 W effort levels with a 20W cool down power for 3 minutes.
- Three devices were used (EIMO, Case-GE, CM400).
- Blood pressure was taken via cuff on the right arm using a SunTech Tango automated monitor every 2 minutes through each recording session.

A summary of the participant measured characteristics are given in table C.1.

Table C.1: Statistical characteristics of the participants completing the study extracted from original report by Abt *et al.* (2015).

Characteristics	All Mean (SD) n = 18	Male Mean (SD) n = 12	Female Mean (SD) n= 6
Age (years)	29 (9)	30 (9)	28 (10)
Body mass (kg)	76.9 (9.0)	85.0 (13.8)	60.7 (5.5)
Height (cm)	174.0 (9.0)	178.0 (5.7)	165.9 (9.4)
BMI (kg/m^2)	25.3 (4.5)	26.9 (4.7)	22.1 (1.7)
Body fat (%)	18.7 (5.2)	18.2 (5.5)	19.5 (5.9)

C.2 Quality Assessment Study

This study was a continuation of the previous one, with ethical approval given as an extension of the the original ethical approval (Application number: 1415218).

First, human assessed annotations were added, using the above scheme, to previously captured data. Then samples of these annotations were produced, so human assessors could independently verify and comment on them. Annotated samples were produced by the machine learning system described above further described in appendix C.2.0.2 using the given annotations by the human annotator for the first session of User 5. They will be used for training the classifier. These two sample sets will then be sent to 4 assessors who volunteered to assess the samples, in a double-blind trial of the human assessed signal annotations. The first was a cardiologist and the three others were computer scientists with previous experience of the ECG and PPG signals and knowledge of signal processing. The control set of samples allows a base line comparison to be measured, to mitigate against unconscious bias, such as an assessor being led by the classifications already labelled on the sample or the types of signals that the assessor believes should be labelled ‘Good’ and ‘Bad’. Due to this reason, the placebo sample annotations should not be perfectly accurate. This was performed to examine the basic hypothesis that the assessor should agree more with the human annotated sample than with the placebo control set, as mistakes made by the control might be copied by the assessor into their appraisal. These placebo samples will be randomly chosen from the sample set and presented using the same graph layout so only when complete and under analysis, will the true origin of the sample annotations be known. The annotations given by the human annotator, and the opinions of the Assessors commenting on them will be compared.

The samples were the same as were used for the comparison in section 6.4.4, with 10 samples chosen from each of 5 users, to give 50 samples to assess. This was chosen to keep the assessment manageable by the volunteer assessors. Out of the 50 samples to be reviewed, a randomly chosen 25 of them were re-annotated

by a machine learning algorithm as explained in appendix C.2.0.2. The other half retained the annotations given by a human annotator who has had some experience of processing and classifying these signals.

C.2.0.1 Data Annotation

The first phase concentrated on gathering the annotations on the signals as a whole. The ECG and PPG signals were examined and the annotations recorded using the medicalDB annotation interface described in section 7.3.2.2. They were annotated, to classify the signal in terms of ‘Good’ and ‘Bad’ quality signals from the description and annotation discussions above in section 7.3.2.2. The session had three sections including, two 12 minute sections recording the user at rest and one 45 minute section recording the users during an exercise activity, which is explained in more detail in section 6.4.4. This data totalled 74 minutes of recorded data per person with five users annotated, mirroring the comparison in the previous chapter. All of the signal data from the first session was annotated before the samples were randomly selected. 20 second samples were drawn from this pool of annotated data, along with the 10 minutes from the first section to be used as training data for the model.

C.2.0.2 Production of the Control Annotation

The second phase is to produce machine learnt signals. The purpose of this is to provide a useful analogue to a signal placebo, as randomly setting the quality of the signal would be too easy to spot. A more informed method is to use a machine learning algorithm trained on the labelled annotations, but the final decision is made by classification of the features derived from the signal. For this purpose, any machine algorithm described above would be sufficient, given that the accuracy of this algorithm is not really under test. The machine led annotations allow the differences between assessors to be tested, similar to a placebo-controlled trial.

The choice of algorithm draws from the information and discussion in sections 7.2.2.1 and 7.3.4.1, where the decision of which algorithm to use in the framework above depended on the task at hand. For this problem, the task was to try and find the

best classification using a descriptive model. A two-class SVM model was chosen; this algorithm can be trained quickly due to its convex error surface and analytical optimisation. This model can then reliably to produce the optimal solution for the training data given to it. The annotated signal data was coupled with the recorded signal data for the session. This machine learning algorithm was able to use the first section's features for training and then was run on the features description segmented from the samples. This was completed for each user independently. The trained SVM model was then used to compute the control annotations for the 25 samples.

The features that can be used to describe the signal can be sourced from the work described above. However, the desired result of the model is not directly the accuracy of the classification, but that it follows the signal characteristics and looks in most instances, plausibly like, a set of annotations classified by a fairly well trained human. The number of features to be used are to be kept high so that the model with a soft margin may miss classify instances on the boundary, producing subtle differences in the final classification. The final features used are shown below:

- $F_{height}(i_s)$ - The segment i_s height from peak to valley between the current $\mathcal{E}(i_s)$ and last $\mathcal{E}(i_s - 1)$ events.
- $F_{period}(i_s)$ - The time difference or length of segment i_s between the current $\mathcal{E}(i_s)$ and the last $\mathcal{E}(i_s - 1)$ events.
- $F_{\bar{s}}(i_s)$ - Mean of the signal segment i_s .
- $F_{\sigma_s}(i_s)$ - The standard deviation of the signal segment i_s .

Followed by amplitude normalised samples from the signal segments divided up with $N = 50$ equally spaced signal taps to normalise the timing and number of sample in a segment:

- F_{last} - Correlation coefficient from current event using a 50 position tapped signal segment $S(i_s)$ to previous event signal segment $S(i_s - 1)$.
- V_{50} - 50 Tapped normalised signal values for segment $S(i_s)$.

- dV_{50} - First consecutive sample difference of V_{50} as a toroidal array for segment $S(i_s)$.
- ddV_{50} - Second consecutive sample difference of V_{50} as a toroidal array for segment $S(i_s)$.

Where $S(i_s)$ is the signal data for a segment bonded by event $\mathcal{E}(i_s)$ and $\mathcal{E}(i_s - 1)$, and i_s is the event index.

In total this creates a classification model with 155 feature parameters. This was trained on the first 10 minutes of the user's signal then tested on the samples randomly selected from that user. The training was repeated for each of the 5 users creating 5 models. These were then used to re-annotate placebo samples for that user with 25 computed in total.

C.2.0.3 Production of the Annotations Samples

The third phase of the study was to produce the samples. The system devised for this was to give each of the individual assessors printed documents with 20 seconds of the signal displayed, including the encoded quality annotations marked with coloured areas. The colours were chosen so that the annotations could be differentiated when printed in colour or grey-scale. The 20 second signal period was chosen to allow the morphology of the sample to be seen, while keeping a reasonable length visible on the sample. The assessors can then compare the signal against the given annotation and what they believe to be a 'Good' signal and 'Bad' signal and mark the preferred annotation transitions onto the sample should they not agree with the presented 'Good' and 'Bad' boundaries. Provision was made so that the assessor can optionally draw two symbols: an 'X' and an 'O', on a scale from 1-10 as to the agreement of the assessor to the given signal quality.

The sample annotations from the Assessors can be seen in figure C.1 and is explained in the caption. Where there are discrepancies, further clarification was obtained from the assessor. At the side of the agreement bar is space for them to add their initials and date. This is to give an indication of where and when the sample was assessed. Brief instructions are included on the sheet with more

complete instructions given in an e-mail or at a meeting with the assessor. The assessors were not told that a machine learning algorithm had annotated half of the samples and all of the samples, no matter what origin, are identical. All assessors and the person in the meeting were blind to the annotations origin while they were handed over and assessed. A scanned example of the presented data layout was given to the Assessors before the experiment to familiarise the Assessors visually and directions on how to fill them in is shown in figure C.2.

When the assessor was marking a ‘Good’ section of signal, they placed a horizontal line across the top of the printed signal. This horizontal line indicated the length of signal that the assessors felt was usable for clinical evaluation. When they deemed the signal to have been lost, or of poor quality, a vertical line was drawn to indicate the transition from ‘Good’ to ‘Bad’ or vice-versa. Another horizontal line would then be drawn from the vertical line forward, along the bottom of the signal trace to indicate the length of the ‘Bad’ quality signal, until the assessor was happy that a ‘Good’ signal was again observable. The vertical line would then be drawn at the transition point, back to the top of the trace and the process would repeat until they had categorised the whole portion (20 seconds) of signal.

C.2.0.4 Results Preparation

The final task was to collate the samples from the Assessors to examine the agreement for the various annotation origins. These results will be presented below, along with the machine learning performance and the annotation table. Comparisons were then made between the assessor of the human annotated signals and machine annotated samples using the agreement metric.

To calculate the metrics of agreement, the portions of the signals that were marked in grayscale were measured along with the assessors marks. The section boundaries were the original annotation boundaries in light and dark gray and the blue lines added by the assessors that were hand drawn and so not always vertical. To remove ambiguity, the point where the blue line crossed the signal line was used as the transition point denoted by a green vertical line. The lengths were

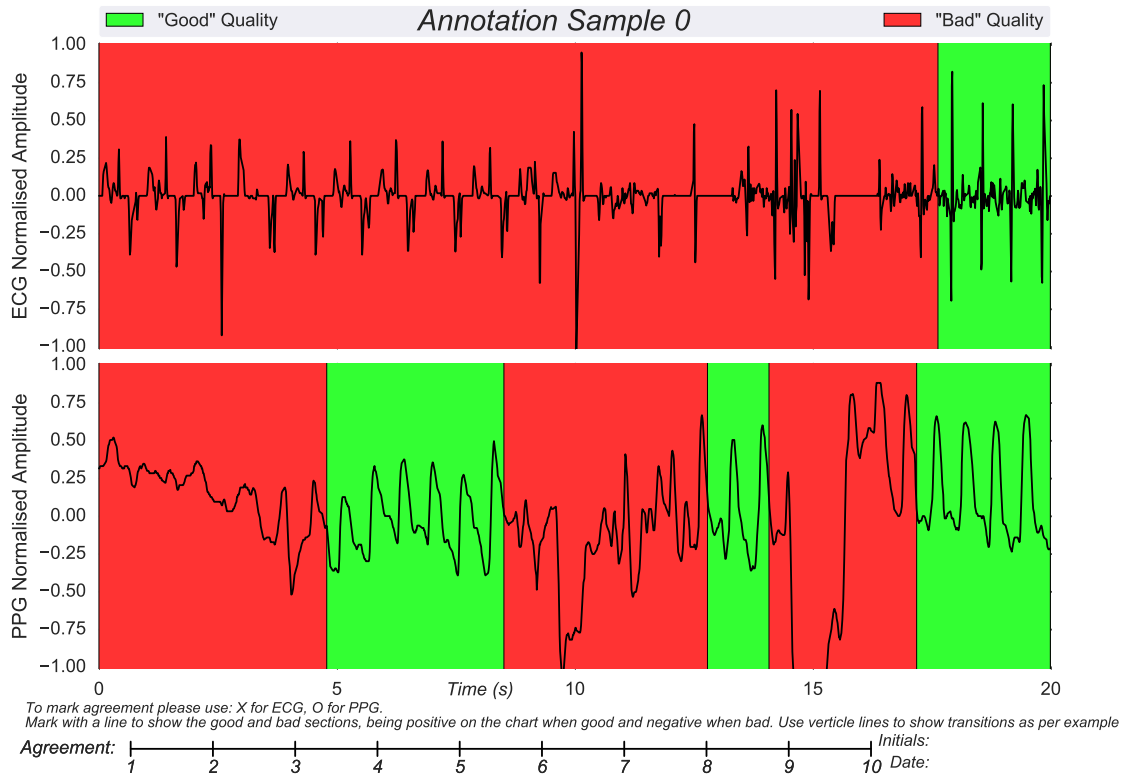


Figure C.1: An annotation assessment sample. The main area shows the ECG and PPG signals top and bottom respectively. Below these graphs is the agreement bar ranging from 1-10. This is used for both signals with an X for the ECG and an O for the PPG. To the right of this is space for the Assessors initials and date. Brief instructions are given on the sample to aid in keeping them easy to fill in. The graph is shown in colour with green for the 'Good' quality and red denoting the 'Bad' quality signal areas. This was annotated by the assessor using a highlighter. Using a line at the top of the graph for good quality, and a line which shows bad quality drawn at the bottom. These lines are connected using vertical lines forming a high-low trace. The vertical lines allow an estimation of the transition points in the signals and were matched where the highlighter crossed the signal.

then measured with vernier callipers, accurate to 10th of a millimetre from the vertical lines in green that marked the transitions between qualities to the grayscale boundaries that were either human or machine annotated. Also the overall section was measured so that the difference between print-outs could be normalised. These sections were then labelled with:

- GG - True Positive (presented annotation was 'Good' assessor marked it as 'Good' - agreement)
- GB - False Negative (presented annotation was 'Good' assessor marked it as 'Bad' - disagreement)

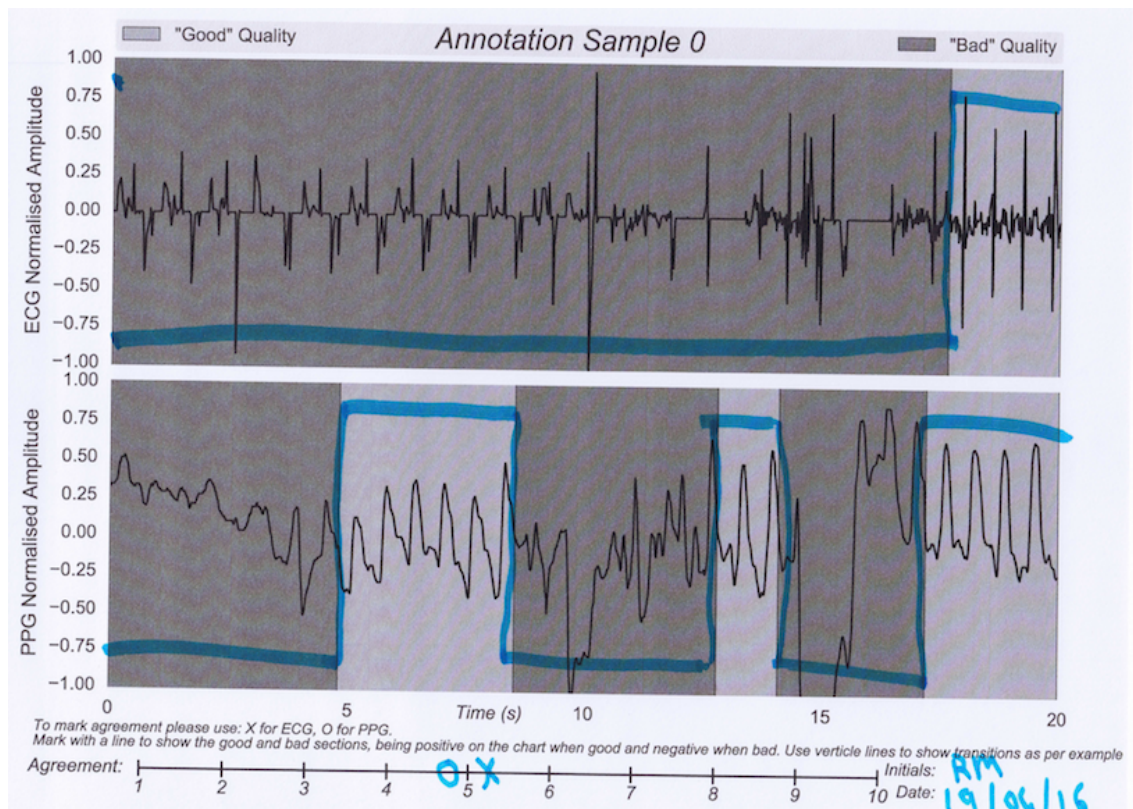


Figure C.2: This shows an example annotation assessment sample as it was scanned in after being assessed. This is the grayscale version of the graph with the light grey for the 'Good' signal and the darker grey the 'Bad' signal quality. The blue highlighter line follows the signal quality as the opinion of the assessor. A line at the top denoted 'Good' quality areas and a line along the bottom denotes 'Bad' cycles areas.

- BG - False Positive (presented annotation was 'Bad' assessor marked it as 'Good' - disagreement)
- BB - True Negative (presented annotation was 'Bad' assessor marked it as 'Bad' - agreement)

The metrics were measured directly from the annotated samples as seen in section 7.4.2 figure 7.11.

These lengths are then collected and analysed to produce three main measurements as described in section 7.2.2.4. The accuracy of the agreement defined in equation (7.1), or how much of the printed signal was annotated the same way. The agreement on 'Good' signals was measured by the sensitivity defined in equation (7.2), and the specificity of the agreement defined in equation (7.3) allows the

measurement of 'Bad' signal quality. This allows the agreements of the annotations to be compared on a receiver operating characteristic (ROC) graph as if the Assessors were behaving as signal quality predictors.

D Framework Implementation

D.1 Overview

The current system has implemented the responsibilities of the services defined in the framework described in section 3.2 so that it can be used to complete the data exploration and experiments described in this thesis. The system has been implemented in two main parts; the NodeSR storage system and a combined ProcessCR and an application specific service interface front-end called the MedicalDB since this implementation is to be applied to the medical study data discussed in the main body of the thesis. The ProcessCR service has been implemented together with the MedicalDB application specific front-end as they can then share web servers and API logic as this simplify the build. The services upon start up take all the components known by them and send them, using the REGISTER verb, to the NodeSR server they have been configured to use. The components defined by each search are described below. The NodeSR can save them locally and import them into its storage interface for native use. This allows the storage to be provided by one server and the processing, data investigation and annotations to be completed on separate machines, while allowing the monitoring of the process and data analysis to be carried out within the system. The implementation considerations of two services are described next starting with the NodeSR.

D.1.1 NodeSR

The implemented storage system has been designed to follow the prescription in section 3.2. The API endpoints are based on functionality as prescribed above for the graph representation. Although the modules within the system have not been fully

optimised, it is sufficiently fast enough that it has not been a problem. The graph search system is simple for the locality around a node and its immediate connections; it is sufficient for traversing the database for all of the experiments and data annotations that have been undertaken in this thesis. The process control system has been implemented and assembles process data flow graphs. A sample graph rendering from the database for a process node is shown in figure D.2. Other extracted graphs can be seen in Figure 8.7, which shows the partial data flow of the signal quality analysis program and figure 9.11, shows the blood pressure analysis program structure. Connections can be made through the storage system to MySQL and Neo4j resources and these can be used in almost any combination. MySQL includes the graph and bulk storage connections implemented with the object structure shown in tables D.1 to D.3. While Neo4j could be employed for both types, it is setup to only allow graph storage, because using Neo4j for bulk storage would not be useful as the large nodes would slow it down as stated in section 2.5. The management interface for the server has ‘search’ and ‘test’ facilities that allow the raw browsing and editing of nodes. The interface also has the facility to display the local graph for each node. The main web page interface shown in figure D.1 can allow users to see the current configuration and statistics of the databases connected. In the centre, the local graph is displayed for a node. This interaction allows nodes to be queried and more nodes to be added for visual examination of the graph in the system.

The framework keeps the data in a graph network. However the storage manager translates this into the raw internal structures used when saving the element of the graph. Appendix D.1.2 shows the structure of the node, bulk and link objects, which are used in the MySQL database backend.

D.1.2 NodeSR Internal Representational Structures

Tables D.1 to D.3 show an example of the internal structure of the basic resource objects as implemented for the MySQL resource connector.

Storage Manager Connection and Storage

Storage Manager Status's

status:	bulktype:	MYSQL
	graphtype:	MYSQL

Init DB
Empty DB

Storage Resource Status's

MYSQL

connection:	Obj:-mysql.connector.connection.MySQLConnection object at 0x10e571410>, IP:srv.local.com, PORT:6606 Server Stats:(u'Uptime': 245279, u'Open tables': 477, u'Queries per second avg': Decimal('0.361'), u'Slow queries': 0, u'Threads': 4, u'Questions': 88787, u'Flush tables': 1, u'Opens': 620)		
type:	MYSQL		
settings:	database: test	verbose_report: false	testing: false
	raise_on_warnings: false	host: srv.local.com	user: robert
	default_limit: 5	password: test	port: 6606
small:	{"bulk":{"count":570,"size":3776.02},"nodes":{"count":754,"size":2.28},"links":{"count":1013,"size":1.48}}		
test:	{"bulk":{"count":0,"size":0.02},"nodes":{"count":4,"size":0.02},"links":{"count":3,"size":0.02}}		
medical:	{"bulk":{"count":57209,"size":77034.64},"nodes":{"count":41899,"size":248.69},"links":{"count":192243,"size":86.83}}		
test1:	{"bulk":{"count":0,"size":0.02},"nodes":{"count":0,"size":0.02},"links":{"count":0,"size":0.02}}		
small_test:	{"bulk":{"count":4,"size":36.52},"nodes":{"count":11,"size":0.02},"links":{"count":0,"size":0.02}}		

MYSQL medical

medical

Node Search Link Search Testing Management Routes

Node Search Section

Search Query

be9bf9de-bd12-4d91-9e2e-572a4b227442

name

type

5

id,name,type

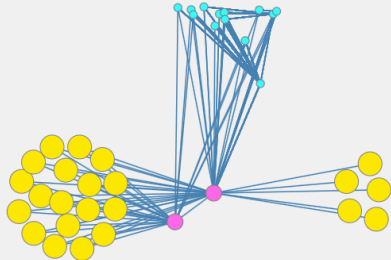
["limit":5,"columns":["id","name","type"],"id":"be9bf9de-bd12-4d91-9e2e-572a4b227442"]

Search manifest

Query Error - []

Query - ["limit":5,"columns":["id","name","type"],"id":"be9bf9de-bd12-4d91-9e2e-572a4b227442"]

Process present_Study1-2-3_model_analysis_results_roc_field MAIN 2017_03_14T15_15/c53fc0 Delete Unload Merge



Graph Show Main node Clicked: GET SELECTED

Currently Over Node: Results Process test_Study2-3a_ModelAnalysis_try1 MAIN 2016_10_19T23_50/b7251a Model Performance_29 2016_10_21T14_53/a9157e

Last clicked Node:

be9bf9de-bd12-4d91-9e2e-572a4b227442

Process present_Study1-2-3_model_analysis_results_roc_field MAIN 2017_03_14T15_15/c53fc0

Process

```
{
  "process_message": "Process Finished",
  "start_time": "2017-03-14T15:16:10.041134",
  "pid": "17231",
  "main_source": "bulknode//f5cd31e5-3846-4c11-afe6-943082aebdad",
  "source": "bulknode//65a34ee4-e3c6-46fa-acf5-c2b3dc93347d",
  "state": "finished",
  "end_time": "2017-03-14T15:25:24.518063"
}
```

Figure D.1: Screenshot of the main management page for the NodeSR service. At the top is automatically gathered information about the service including what resources are configured and tabbed statistics. Below the server status is the 'search' and 'test' controls.

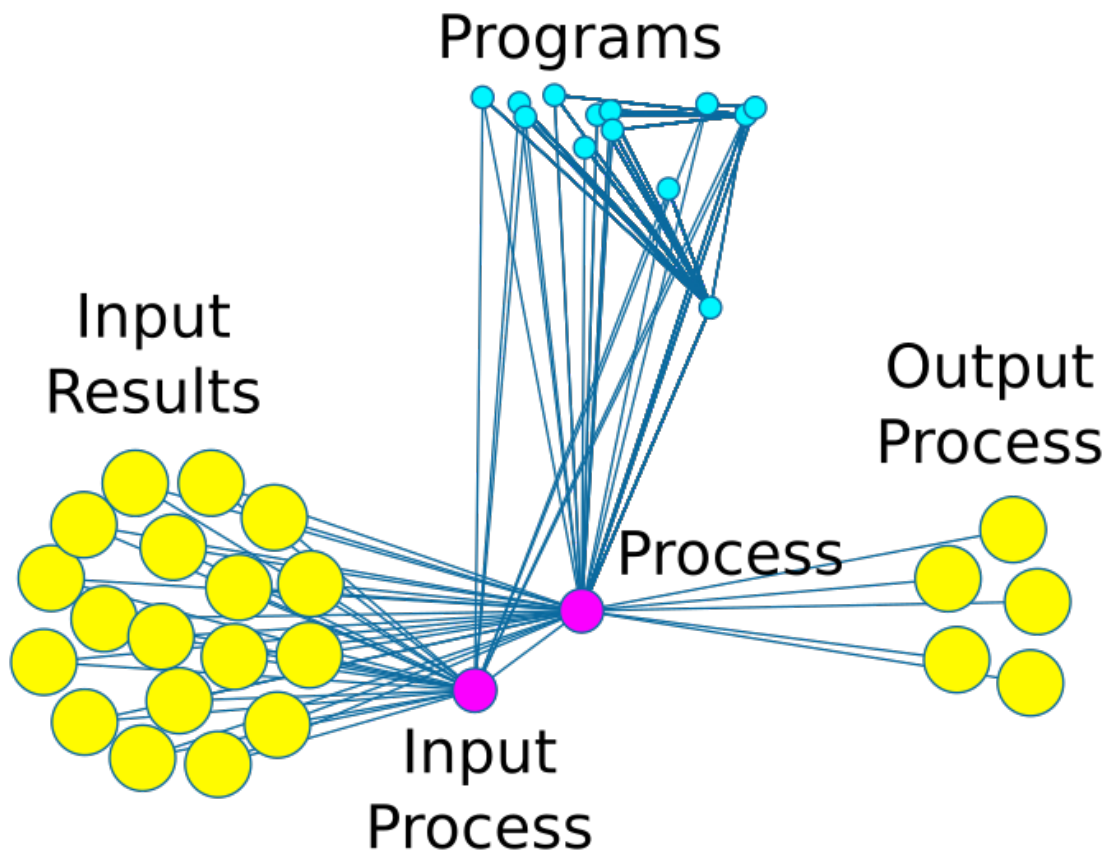


Figure D.2: A rendered capture from the website of a process node as it is represented within the real implementation. The yellow nodes are the result components, the left side nodes are the ones currently used as the data for the process. The right ones are the results from this process. The blue nodes are program components, which are run in the processSR and interfced through by the dark pink process components nodes. The connections are automatically created as the process was built then run.

Table D.1: Example field description for the node model within the NodeSR system when saving to MySQL.

Field	Type	Range	Notes
ID	string	(32)	Randomly generated and unique UUID.
name	string	(0,255)	Assigned name for the node.
type	string	(0,255)	Assigned ":" separated string types for the node.
attributes	BLOB	up to 4GB	The other attributes for this resource encoded as a JSON object. This can contain implicit links to bulk items.
nodehash	string	32	The node hash for all variables of this node not including the ID and the times.
created	DateTime	ISO 8601 Date and Time	The date the node was created.
modified	DateTime	ISO 8601 Date and Time	The date the node was modified.
accessed	DateTime	ISO 8601 Date and Time	The date the node was last accessed.

Table D.2: Example field description for the bulk model which is derived from the node model within the NodeSR system when saving to MySQL.

Field	Type	Range	Notes
ID	string	(32)	Randomly generated and unique UUID.
name	string	(0,255)	Assigned name for the node.
type	string	(0,255)	Assigned ":" separated string types for the node.
attributes	BLOB	up to 4GB	The other attributes for this resource encoded as a JSON object.
nodehash	string	32	The node hash for all variables of this node not including the ID and the times.
created	DateTime	ISO 8601 Date and Time	The date the bulk node was created.
modified	DateTime	ISO 8601 Date and Time	The date the bulk node was modified.
accessed	DateTime	ISO 8601 Date and Time	The date the bulk node was last accessed.

Table D.3: Example field description for the link model within the NodeSR system when saving to MySQL.

Field	Type	Range	Notes
ID	string	(32)	Randomly generated and unique UUID.
name	string	(0,255)	Assigned name for the link.
type	string	(0,255)	Assigned ":" separated string types for the link.
ida	string	(32)	Linked node ID.
idb	string	(32)	Linked node ID.
idc	string	(32)	Linked node ID.
attributes	BLOB	up to 4GB	The other attributes for this resource encoded as a JSON object.
linkhash	string	32	The link hash for all variable of this node not including the ID and the times.
created	DateTime	ISO 8601 Date and Time	The date the link was created.
modified	DateTime	ISO 8601 Date and Time	The date the link was modified.
accessed	DateTime	ISO 8601 Date and Time	The date the link was last accessed.

D.1.3 MedicalDB

The ProcessCR service and the application specific service interface (ASSI) components can share a number of resources. To keep the project implementation simple, the services defined above have been combined into one project called the MedicalDB, where they can share the web server configuration and be easier to maintain. This reduces the replicated, shared components and operations that would only require unnecessary communications, between the two systems for this application. This has allowed the system to be built, tested and be used to run all of the tasks this thesis has required. The internal architecture of the two services have been implemented as described in section 3.2, sharing common utilities. The merging of these services make sense since the computational load of the MedicalDB is minimal and so it can share a host with the ProcessCR services, as they can be separated in operation by their interfaces. How and where their logic is implemented is unimportant to the goals of this framework. Figure D.3 shows an abstract graph database and typed nodes, and how the linking structure connects the typed nodes within the storage system.

The attributes of the components used in the ProcessCR and in the MedicalDB service can be seen in tables D.4 and D.5 respectively. The data types information is stored as multipurpose internet mail extensions commonly called the MIME type information¹.

The style of these pages has been second to the functionality, but the layout and style is flexible. The look of the website is illustrated in figure D.4. The most important part of this system is the functionality and the ability to run the experiments which follow other exploratory tests used in the other chapter of this thesis. The styling of the page could be improved and is a distinct point of extension. The advantages of this system are that it is designed to allow these changes by separating the storage and compute logic away from the front-end ASSI interface. If other interfaces are required, different ASSI interfaces could be created with each

¹The MIME types used are listed in the specification at <http://www.iana.org/assignments/media-types/media-types.xhtml> retrieved: 2016/09/05

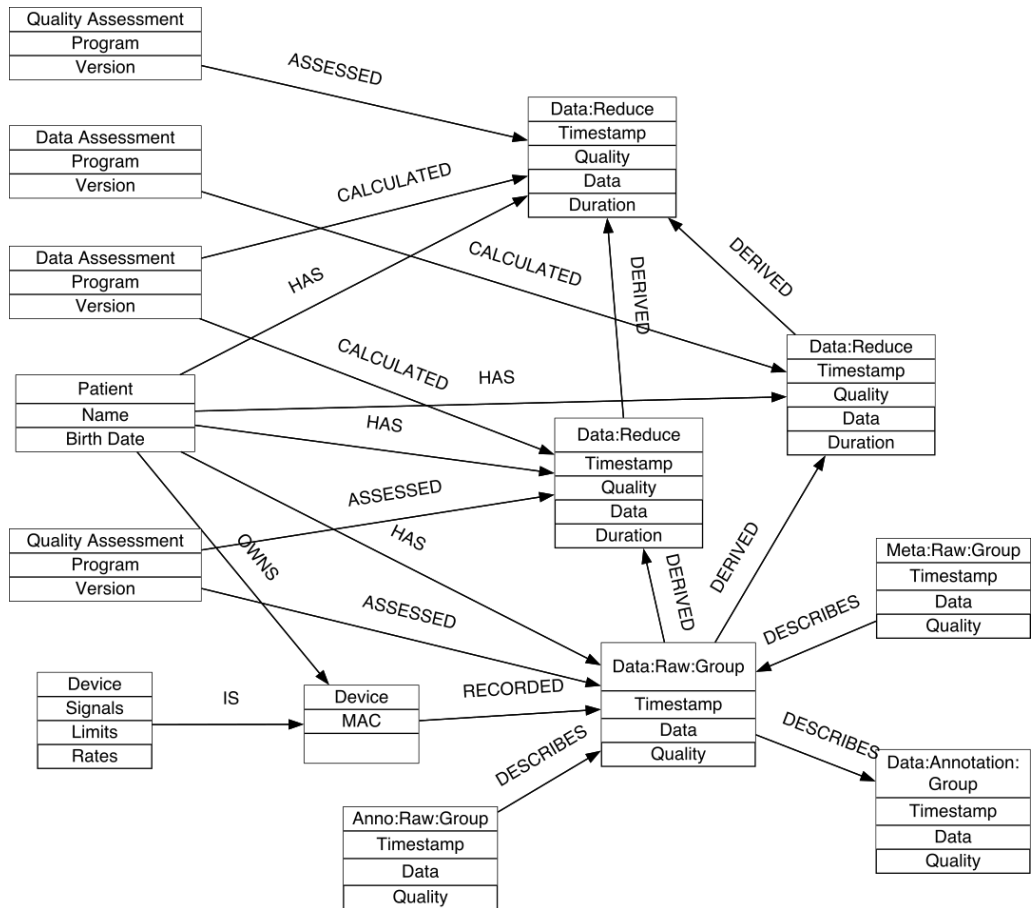


Figure D.3: Diagram showing the internal graph database representation of the medical database system. The data and groups of data are separated into data nodes with the user device and session meta information along with the ontology linked as part of the same structure.

Table D.4: A list of the components and attributes required by the ProcessCR service, with descriptions of what the component does and the properties it possesses.

Name	Attributes	Description
Program		Denotes a program node to use to store a program source.
	source_type	The program source MIME type e.g. for python: "text/x-python".
	source	The source code for the program to run.
	trigger	The source code for the trigger to run.
	trigger_type	The trigger source MIME type for the trigger program source e.g. for python: "text/x-python".
	trigger_duration	The time in seconds the trigger has to find a result.
Process		Denotes a process node to use to store information about the internal processing worker.
	PID	Worker UNIX process ID or equivalent
	status	The status of the process e.g. 'preparing', 'ready', 'running' etc.
	source_dict	A dictionary of the sources used in the program.
	memory	The memory load.
	memory_max	The maximum memory load observed.
	memory_avg	The average memory load observed.
	CPU	The CPU load.
	CPU_avg	The Average CPU load.
Result		Denotes a results node to store result files.
	report_elements	A list of objects consisting of a timestamp, and format type and text for reporting.
	file_list	A list of files saved within this results group.

Table D.5: A list of the components and attributes required by the MedicalDB application specific service interface (ASSI), with descriptions of what the component does and the properties it possesses.

Name	Attributes	Description
User		Denotes a user node component.
	first_name	First name of the user.
	last_name	Last name of the user.
	birthdate	The birthdate of the user.
	type	The type of user being a [User, Annotator, Assessor, Clinician].
Device		Denotes a device node component.
	signals	The signal types recorded by this device e.g. I,II,III,PPG etc.
	limits	The signal limits for the device to match with signal names above.
Meta		This holds the circumstantial information stored as a semi-structured dataset to tie other data nodes together.
Raw		Denote the data stored are raw signals from a device.
Data		For storage of data within the node so it should have a type and data attribute.
	data_type	the MIME data type for this node.
	data	The data for this node.
	start_time	The start of the recording.
	end_time	The end of the recording.
Annotation		For storage of event annotations coupled to data.
	annotation	The annotation stored in this node.

attached to the same NodeSR and ComputeCR service system.

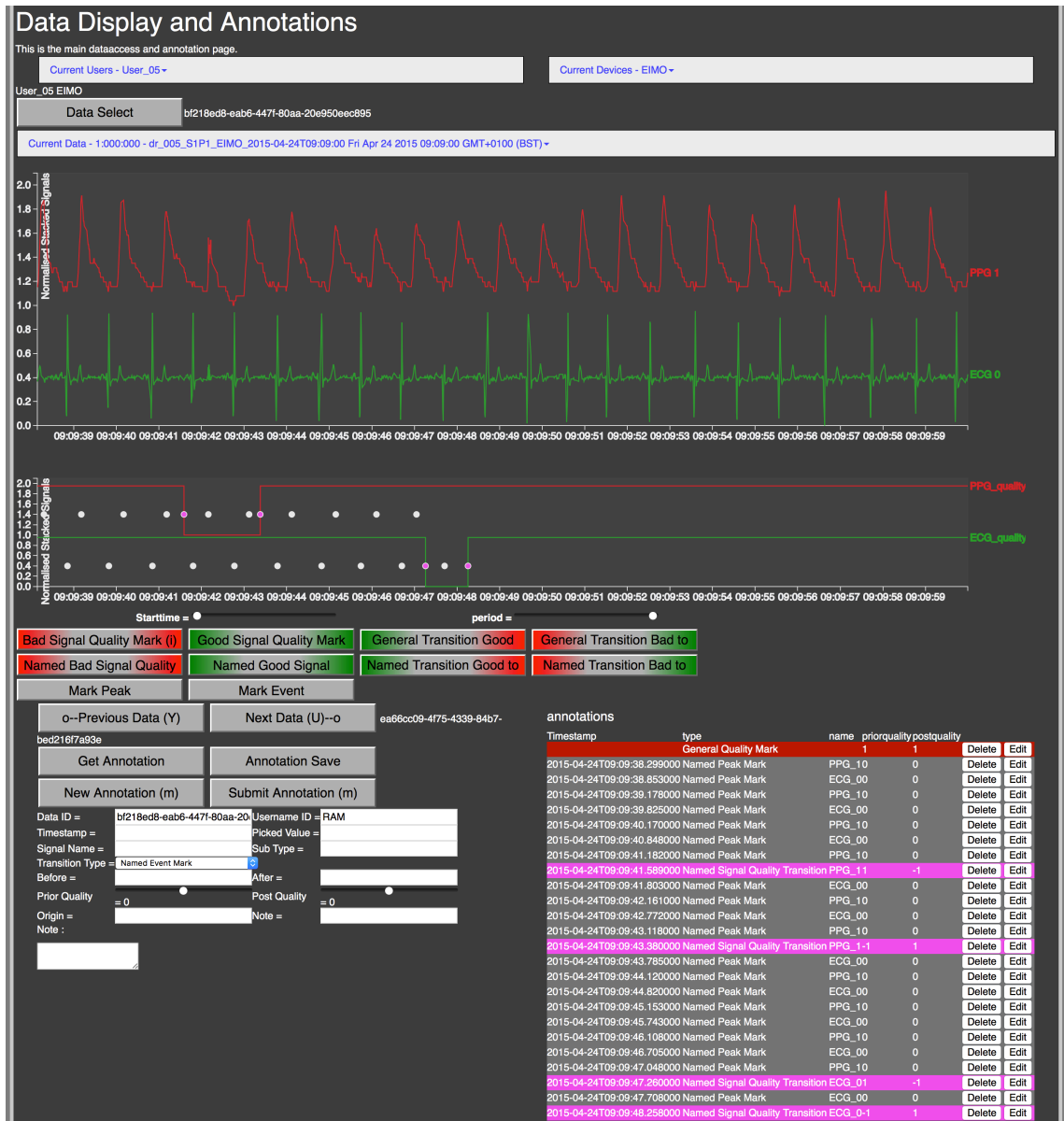


Figure D.4: A screenshot of the functional data controls for the MedicalDB service arranged on a web page for display. The user selection controls are located at the top, and the raw signal data is display below them, with the annotation control at the bottom, and the annotations displayed both on the data and in a table format. The flexible controls allow the design to be altered to best fit a particular application.

D.2 Medical Database Structures

This sections show the names definitions of the medical database as used for signal annotations and processing.

D.2.1 Annotation types

The annotations used in the database were drawn from the physionet system as shown in table D.6a, with some extra types added to aid with the more detailed study of signal quality as shown in table D.6.

Table D.6: A listing of the annotation types used in the MedicalDB.

(a) A listing of the annotation types used in the annotation system as taken from the the Physionet website (www.physionet.org) (Goldberger *et al.*, 2000).

Code	Description	Code	Description
"	Comment annotation	(Waveform onset
*	Systole	@	Link to external data
A	Atrial premature beat	E	Ventricular escape beat
Q	Unclassifiable beat	S	Supraventricular premature or ectopic beat (atrial or nodal)
a	Aberrated atrial premature beat	e	Atrial escape beat
s	ST segment change	u	Peak of U-wave
B	Bundle branch block beat (unspecified)	D	Diastole
F	Fusion of ventricular and normal beat	J	Nodal (junctional) premature beat
L	Left bundle branch block beat	N	Normal beat (displayed as "û" by the PhysioBank ATM, LightWAVE, pschart, and psfd)
R	Right bundle branch block beat	T	T-wave change
V	Premature ventricular contraction	f	Fusion of paced and normal beat
j	Nodal (junctional) escape beat	n	Supraventricular escape beat (atrial or nodal)
p	Peak of P-wave	r	R-on-T premature ventricular contraction
t	Peak of T-wave	x	Non-conducted P-wave (blocked APC)
[Start of ventricular flutter/fibrillation]	End of ventricular flutter/fibrillation
)	Waveform end	+	Rhythm change
/	Paced beat	=	Measurement annotation
?	Beat not classified during learning		Isolated QRS-like artifact
`	PQ junction	~	Change in signal quality
!	Ventricular flutter wave	'	J-point

(b) A listing of the extra annotation types used to extend the signal quality descriptions.

Code	Description
NT	Named Signal Quality Transition
NM	Named Signal Quality Mark
G	General Quality Transition
M	General Quality Mark